Une méthode de calcul de délais pire cas de bout en bout pour les réseaux SpaceWire

Thomas Ferrandiz, Fabrice Frances, Christian Fraboul

Une méthode de calcul de délais pire cas de bout en bout pour les réseaux SpaceWire

Thomas Ferrandiz* — Fabrice Francès* — Christian Fraboul**

**Université de Toulouse, ISAE*
10 avenue Edouard Belin
BP 54032, F-31055 Toulouse Cedex 4
{thomas.ferrandiz,fabrice.frances}@isae.fr

***Université de Toulouse, ENSEEIHT/IRIT*
2, rue Charles Camichel
BP 7122, 31071 Toulouse Cedex 7
christian.fraboul@enseeiht.fr

RÉSUMÉ. *SpaceWire est un standard de réseaux embarqués pour satellites choisi par l'Agence Spatiale Européenne comme base pour ses futures architectures de traitement de données. Cependant, les concepteurs réseaux ont besoin d'outils permettant de garantir qu'un réseau est capable de délivrer les messages critiques dans les délais requis. Les recherches menées jusqu'ici se sont seulement concentrées sur la détermination de délais de bout en bout probabilistes pour les réseaux de type Wormhole dont SpaceWire fait partie. Ces résultats fournissent des garanties insuffisantes pour le trafic critique. Dans cet article, nous proposons donc une méthode permettant de calculer une borne supérieure garantie sur le délai de bout en bout pire-cas d'un paquet traversant un réseau SpaceWire.*

ABSTRACT. *SpaceWire is a standard for on-board satellite networks chosen by the ESA as the basis for future data-handling architectures. However, network designers need tools to ensure that the network is able to deliver critical messages on time. Current research only seek to determine probabilistic results for end-to-end delays on Wormhole networks like SpaceWire. This does not provide sufficient guarantee for critical traffic. Thus, in this paper, we propose a method to compute a guaranteed upper-bound on the worst-case end-to-end delay of a packet in a SpaceWire network.*

MOTS-CLÉS : *délais pire cas, SpaceWire, réseaux embarqués.*

KEYWORDS: *worst-case delay, SpaceWire, embedded networks.*

1. Introduction

SpaceWire ((Parkes *et al.*, 2005),(ECSS, 2008)) est un réseau embarqué à haut débit pour satellite créé par l'ESA et l'Université de Dundee. Il a été conçu pour interconnecter les divers équipements d'un satellite comme les capteurs et les mémoires de masse à l'aide d'interfaces standard et simples. SpaceWire utilise des liens série, point-à-point, bidirectionnels et full-duplex fonctionnant à des vitesses allant de 2 à 200 Mbps et des routeurs simples permettant d'interconnecter les nœuds à l'aide de topologies arbitraires.

A l'avenir, l'ESA prévoit d'utiliser SpaceWire comme le réseau embarqué exclusif de ses satellites. Celui-ci est bien adapté à cette tâche dans la mesure où il s'agit d'un réseau rapide et consommant peu d'énergie. SpaceWire sera donc utilisé pour transporter à la fois le trafic scientifique de la charge utile et le trafic de contrôle du satellite, multiplexés sur les mêmes liens. Or ces deux types de trafics ont des contraintes très différentes. Le trafic de contrôle ne génère le plus souvent qu'une faible charge mais obéit à des contraintes temporelles très strictes. Au contraire, le trafic scientifique ne requiert pas de garanties temporelles strictes mais doit disposer d'un débit constant et élevé pour être transmis correctement.

Ces deux types de contraintes sont faciles à satisfaire sur des liens SpaceWire point-à-point dédiés. Cependant, pour pouvoir connecter tous les terminaux sur un unique réseau SpaceWire, les spécifications d'un routeur tenant compte des contraintes propres au domaine spatial ont été ajoutées au standard. En particulier, les routeurs doivent utiliser des mémoires résistantes aux radiations qui sont extrêmement coûteuses. Ces contraintes ont ainsi mené à l'utilisation du routage Wormhole (qui fonctionne avec une faible quantité de mémoire) dans les routeurs SpaceWire.

Toutefois, cette technique de routage pose un problème majeur. En effet, elle peut parfois provoquer d'importants délais lorsque des paquets se retrouvent bloqués dans un routeur comme nous le montrerons dans la section 2. Ces blocages font qu'il est très difficile de prédire les délais de bout en bout.

C'est pourquoi les précédentes méthodes d'évaluation des délais dans des réseaux Wormhole ont essentiellement porté sur l'obtention de résultats probabilistes qui ne fournissent pas suffisamment de garanties. En effet, nous avons besoin d'être certains que les paquets de contrôle sont toujours transmis dans des délais acceptables. De même, les simulations sont insuffisantes car elles ne peuvent couvrir tous les cas possibles. Il nous semble ainsi préférable de déterminer seulement une borne supérieure sur le délai pire-cas de bout en bout subi par un paquet.

Nous avons donc conçu une méthode de calcul qui nous permet de déterminer une telle borne. Nous présenterons d'abord cette méthode en section 3 puis une application de cette méthode sur un exemple de réseau en section 4. Enfin, nous conclurons et proposerons des pistes de recherches futures en section 5.

2. Une brève description de SpaceWire

2.1. *SpaceWire comme lien point-à-point*

Au départ, SpaceWire a été conçu uniquement dans le but de fournir des liens point-à-point entre les composants d'un satellite. Il est basé sur le standard IEEE 1355 mais utilise de nouveaux connecteurs et câbles adaptés aux contraintes spatiales.

SpaceWire utilise deux types de caractères comme unités de transmission : les caractères de données et les caractères de contrôle. Les caractères de contrôle sont utilisés pour signaler qu'un paquet est terminé (éventuellement de façon erronée) et pour le contrôle de flux.

Ce dernier permet de garantir qu'aucun nœud ne reçoit de données pouvant provoquer un dépassement de mémoire tampon. Sur chaque lien, l'émetteur ne peut pas envoyer de données à moins que le récepteur ne l'y ait autorisé par un caractère de contrôle (Flow Control Token ou FCT). Chaque FCT autorise l'émission de 8 caractères. Il suffit donc que le récepteur utilise un buffer de réception de taille supérieure (le maximum est de 64 octets) pour que le contrôle de flux ne cause pas de délais (tant que le récepteur ne sature pas).

Dans un réseau SpaceWire, les données sont organisées en paquets suivant un format très simple. Chaque paquet comprend trois champs : un champ d'adresse, un champ Cargo et un marqueur de fin de paquet. Les champs Adresse et Cargo peuvent être de longueur quelconque et il est même possible d'envoyer des paquets de taille infinie.

Ainsi, les liens point-à-point SpaceWire fournissent un moyen simple et rapide pour transmettre des données entre deux équipements terminaux. De plus, le délai de bout en bout est très simple à calculer. Si la source et la destination ne provoquent pas elles-mêmes de délais, le délai est simplement $\left\lceil \frac{T}{C} \right\rceil$ où T est la taille du paquet et C la capacité du lien.

2.2. *SpaceWire comme réseau d'interconnexion*

Avec l'augmentation des quantités de données échangées par les divers équipements du satellite, utiliser des liens point-à-point pour interconnecter l'ensemble des terminaux devient trop coûteux. De plus, pour réduire la complexité de l'architecture réseau, il est préférable d'utiliser un seul réseau pour transporter simultanément le trafic charge utile et le trafic commande/contrôle au lieu d'utiliser un bus dédié (typiquement le MIL-STD-1553B) pour le trafic commande/contrôle et des liens à haut débit pour le trafic charge utile.

Il a donc été nécessaire d'étendre le standard SpaceWire avec les spécifications d'un routeur. Ce routeur doit impérativement respecter deux contraintes. Premièrement, il doit rester compatible avec les liens point-à-point utilisés pour interconnecter

les terminaux. Deuxièmement, il doit utiliser le moins possible de mémoire car il est très difficile de fabriquer des mémoires résistantes aux radiations.

Par conséquent, la technique de routage Wormhole a été retenue pour transmettre les paquets à travers le réseau. Avec cette technique, le routeur n'a pas besoin de stocker un paquet en entier avant de le transmettre. Il lit seulement l'en-tête (le premier caractère) du paquet à la volée lorsque celui-ci arrive. Il utilise ensuite cette information pour déterminer le port de sortie approprié. Si le port de sortie est libre, le routeur transfère l'en-tête du paquet sans attendre le reste.

Lorsque le port de sortie est déjà utilisé par un paquet A, un second paquet B arrivant pendant le transfert de ce paquet doit alors attendre la fin du transfert de A (voir figure 1). Etant donné que le contrôle de flux est actif sur chaque lien, des fragments de B se retrouvent bloqués dans chacun des routeurs qu'il a traversé en attendant que l'en-tête du paquet ait accès au port de sortie. Ils bloquent ainsi aux autres paquets l'accès à tous les ports de sortie traversés par B dans les routeurs précédents. Quand le port de sortie est libéré, un des paquets en attente est sélectionné par un mécanisme de type Round-Robin et transféré.

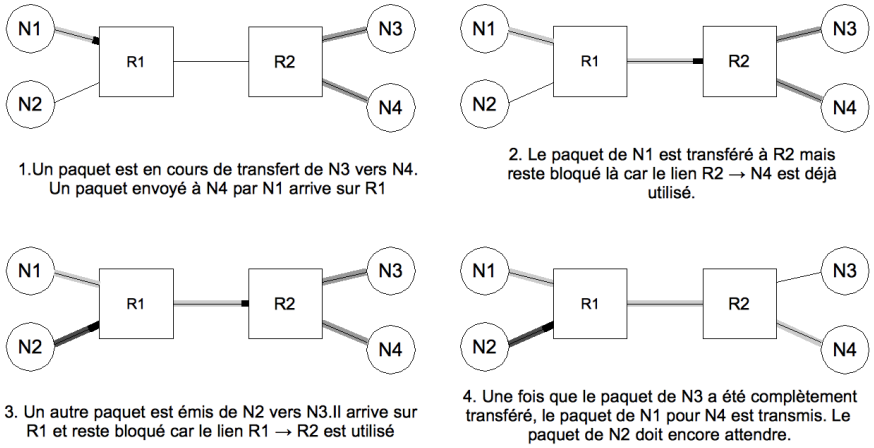

Figure 1. *Blocage des paquets dans un réseau SpaceWire*

L'utilisation du routage Wormhole permet donc de satisfaire les deux contraintes précédemment mentionnées. Les routeurs n'ont besoin que de 8 à 64 octets de mémoire pour les ports d'entrée et pas du tout pour les ports en sortie. Ils sont également compatibles avec les liens point-à-point SpaceWire puisqu'ils propagent le contrôle de flux sur chaque lien du chemin suivi par le paquet, du récepteur vers l'émetteur.

Cependant, le routage Wormhole souffre d'un défaut majeur. En effet, il peut arriver que des paquets subissent d'importants délais dans un ou plusieurs routeurs avant de pouvoir accéder au lien de sortie adéquat. Deux mécanismes ont donc été ajoutés à SpaceWire pour réduire le problème. Le premier est un mécanisme de priorités

statiques à deux niveaux mis en place dans les routeurs. Grâce à ce mécanisme, si plusieurs paquets attendent la libération d'un même port, les paquets envoyés vers une adresse de priorité haute seront toujours transmis avant les paquets envoyés vers une adresse de priorité basse.

Le second mécanisme a pour nom Group Adaptative Routing (GAR). Il consiste à placer au moins deux liens en parallèle entre deux routeurs et à les déclarer comme un groupe dans leurs tables de routage. Les routeurs sont alors capables d'équilibrer automatiquement la charge sur les liens. Cependant, ce mécanisme augmente le nombre de liens du réseau au lieu de partager les liens déjà en place et ne devrait donc être utilisé que lorsqu'il est vraiment impossible de transmettre tout le trafic sur un seul lien.

Ces deux mécanismes peuvent réduire les blocages mais pas les éliminer. Par conséquent, pour pouvoir évaluer les délais de bout en bout sur un réseau SpaceWire, il nous faut être capable de déterminer les durées des blocages subis par les paquets.

2.3. *Le problème de détermination des délais de bout en bout : état de l'art*

Dans le meilleur des cas, un paquet n'est bloqué dans aucun routeur et ne subit que le délai de commutation dans chaque routeur. Ce délai inclut le temps nécessaire pour lire l'en-tête du paquet, déterminer le port de sortie et copier l'en-tête vers ce port. Les spécifications du routeur SpaceWire imposent que ce délai soit inférieur à $0.5\ \mu s$ lorsque les liens fonctionnent à 200 Mbps. Par conséquent, dans le meilleur des cas, le délai de livraison d'un paquet à travers un réseau SpaceWire sera égal au délai sur un lien point-à-point plus $0.5\ \mu s$ pour chaque routeur traversé.

Cependant, des conflits d'accès peuvent se produire sur chaque lien emprunté par le paquet. De plus, un paquet bloquant un lien peut lui même se trouver bloqué plus loin dans le réseau. Cela rend très difficile la détermination analytique du délai de bout en bout pour un paquet donné.

De nombreux travaux ont été publiés portant sur l'estimation des délais de bout en bout dans les réseaux Wormhole, notamment dans les supercalculateurs. Toutefois, ces travaux ne cherchent qu'à déterminer des latences moyennes à l'aide de techniques comme la théorie des files d'attente (voir par exemple (Draper *et al.*, 1994) et (Dally, 1990)). Or, connaître les latences moyennes n'est pas suffisant ici car nous devons être certain que tous les messages sont délivrés dans un délai conforme aux spécifications.

Le calcul réseau (Boudec *et al.*, 2004) est une autre technique qui a été utilisée avec succès pour calculer une borne supérieure sur les délais pire-cas dans un réseau embarqué. Par exemple, dans (Mifdaoui *et al.*, 2007), le calcul réseau est utilisé pour analyser un réseau Ethernet commuté prévu pour remplacer une architecture à base de bus MIL-STD-1553B dans un réseau avionique militaire. Cependant, il ne semble pas que SpaceWire soit modélisable de façon simple en utilisant les éléments du calcul

réseau classique comme les multiplexers et les buffers. A notre connaissance, une telle modélisation n'existe pas à l'heure actuelle.

Par conséquent, nous proposons une approche plus spécifique qui nous permet de calculer une borne sur le délai pire-cas de chaque flux. Ainsi, bien que nous ne puissions pas déterminer le délai subi individuellement par chaque paquet, nous pouvons tout de même garantir que les paquets critiques sont livrés à temps.

3. Calcul d'une borne sur le délai pire-cas

3.1. *Notations et définitions*

Nous représentons un réseau SpaceWire par un graphe orienté connexe $\mathcal{G}(N, L)$. L'ensemble N des nœuds du graphe représente les terminaux et les routeurs SpaceWire. L'ensemble L des arcs représente les liens SpaceWire. Nous utilisons deux arcs de directions opposées pour représenter un lien SpaceWire bidirectionnel. Etant donné que deux routeurs peuvent être connectés par de multiples liens en parallèle, $\mathcal{G}(N, L)$ peut être un multigraphe. Par contre, il ne peut jamais y avoir plusieurs liens parallèles entre deux terminaux ou entre un terminal et un routeur. Le cas échéant, on remplacerait un terminal équipé de plusieurs interfaces par plusieurs terminaux.

Une communication entre une source et une destination est modélisée par un flux unidirectionnel f. Chaque flux f a une taille maximale de paquet notée T_f et passe par un ensemble de liens formant un chemin dans $\mathcal{G}(N, L)$. L'ensemble des flux est noté F. Nous n'avons pas besoin de connaître le débit de chaque flux mais nous faisons l'hypothèse que la capacité de chaque lien est suffisante pour transmettre toutes les données transitant par ce lien. Nous supposons également qu'aucun paquet ne peut être stocké intégralement dans les buffers qu'il traverse. Par conséquent, chaque paquet doit être de longueur supérieure à la somme des tailles des buffers de réceptions entre sa source et sa destination.

Notons $liens(f)$ la liste ordonnée des liens traversées par le flux f et $premier(f)$ la fonction qui renvoie le premier lien de $liens(f)$. Pour chaque groupe de liens, on considère qu'un lien du groupe est le lien canonique utilisé par tous les flux. Le routeur se charge ensuite de répartir dynamiquement les paquets sur les liens du groupe. Notons enfin $suiv(f, l)$ la fonction qui, étant donnés un flux f et un lien l de $liens(f)$ renvoie le lien suivant l dans $liens(f)$ et $prec(f, l)$ la fonction qui renvoie le lien précédent l dans $liens(f)$. Si l est le dernier lien de $liens(f)$, $suiv(f, l)$ renvoie $null$.

Nous faisons également les hypothèses suivantes :

– le routage est statique (c'est le cas sur le routeur SpaceWire actuel) ;

– tous les liens ont la même capacité C ;

– toutes les adresses sont de priorité basse ;

– tous les flux traversant un groupe utilisent le GAR avec tous les liens du groupe ;

– nous négligeons les délais causés par les Time-Codes et les Flow Control Token (les Times-Codes sont des caractères spéciaux utilisés par SpaceWire pour synchroniser tous les nœuds du réseau sur une horloge globale) ;

– les paquets sont traités dès qu'ils arrivent à destination (la destination ne cause donc aucun délai) ;

– lorsqu'une source a commencé à émettre un paquet, elle le transmet à la vitesse maximale (la source ne cause donc aucun délai).

3.2. *La méthode de calcul*

Notre méthode de calcul se base sur l'idée que, dans un réseau SpaceWire, la livraison d'un paquet a lieu en deux phases. Pendant la première phase, le premier caractère (l'en-tête) du paquet est routé jusqu'à sa destination et crée un "circuit virtuel" entre la source et la destination du paquet. Pendant la seconde phase, le corps du paquet est transféré sur ce "circuit virtuel" comme sur un lien point-à-point. Une fois le paquet transféré, les liens sont libérés et le circuit disparaît.

Calculons maintenant une borne supérieure sur le délai pire-cas subi par un paquet p du flux f cherchant à accéder à un lien l. Le cas le plus simple est celui où $l = null$. cela signifie que l'en-tête du paquet est arrivé à destination. Ainsi, il n'y pas de délais causés par un routeur. Le seul délai restant est le temps nécessaire à la transmission du corps du paquet sur le circuit virtuel créé par l'en-tête (seconde phase du transfert). Ce délai est tout simplement : $\left[d(f, null) = \frac{T_f}{C}\right]$.

Au contraire, si $l = premier(f)$, cela signifie que le premier nœud de l est la source de f. Il s'agit donc d'un terminal et il ne peut y avoir aucun flux entrant par d'autres ports et créant des conflits d'accès à l. Par contre, ce nœud peut être source d'autres flux que f. Leurs paquets peuvent alors utiliser l avant que p n'y ait accès. Le délai maximum qu'un paquet q peut causer à p est la durée nécessaire pour transférer q à partir de la source si q était seul, c'est-à-dire si q ne subissait aucun délai au démarrage. Cependant, q peut lui aussi être bloqué dans les routeurs qu'il va ensuite traverser. Il faut donc tenir compte du délai qu'il subira aux liens suivants. Si $l = premier(f)$, le délai est donc:

$$d(f, premier(f)) = \sum_{f_{in} \in S_F} d(f_{in}, suiv(f_{in}, l)) + d(f, suiv(f, l))$$

où $S_F = \{f_{in} \in F | f_{in} \neq f \, et \, premier(f_{in}) = premier(f)\}$ (i.e. S_F est l'ensemble des flux qui ont la même source que f).

Lorsque $l \neq null$ et $l \neq premier(f)$, étant donné que la source et la destination ne causent aucun délai au paquet, le seul délai subi par p est causé par le routeur traversé pour accéder à l. Ce délai a deux causes possibles. La première est le délai de commutation du routeur, que nous noterons d_C par la suite. Ce délai est identique pour tous les routeurs et tous les paquets.

Le second délai apparaît lorsque le port de sortie (ou le groupe de ports) est déjà utilisé par un autre paquet. Supposons dans un premier temps qu'il s'agit d'un lien simple. Avec le routage Wormhole, p doit attendre jusqu'à ce que le paquet bloquant le port ait été complètement transféré. Il est également possible que des paquets entrant par d'autres ports soient déjà en train d'attendre la libération du même port que p et passent avant lui. Notons que seuls les paquets sortant par le même port que p peuvent le retarder puisque les ports des routeurs SpaceWire sont indépendants.

Comme SpaceWire utilise un mécanisme Round Robin pour sélectionner le port d'entrée qui aura accès au port de sortie, même dans le pire des cas, au plus un paquet pour chaque port d'entrée autre que celui de p pourra passer avant p. De plus, pour chacun de ces ports d'entrée, plusieurs flux peuvent entrer dans le routeur. Il nous faut donc calculer une borne sur le délai maximal que chacun de ces flux pourraient produire et prendre le maximum de ces délais.

Le délai maximum qu'un paquet q peut causer à p est la durée nécessaire pour transférer q à partir du routeur courant si q était seul dans ce routeur, c'est-à-dire si q ne subissait aucun délai dans ce routeur. Cependant, q peut lui aussi être bloqué dans les routeurs qu'il va ensuite traverser. Enfin, une fois qu'il a obtenu l'accès au port de sortie, l'en-tête du paquet est transféré au routeur suivant et peut avoir à nouveau à attendre pour obtenir l'accès au port de sortie de ce routeur.

Nous pouvons donc maintenant énoncer une définition récursive d'une borne supérieure $d(f, l)$ sur le délai nécessaire pour transférer un paquet du flux f à partir du moment où il tente d'accéder à un lien l tel que $l \neq null$ et $l \neq premier(f)$:

$$d(f,l) = \sum_{l_{in} \in A_{f,l}} [\max_{f_{in} \in U_{l_{in}}} d(f_{in}, suiv(f_{in}, l)) + d_C] + d(f, suiv(f,l)) + d_C \quad [1]$$

où $A_{f,l} = \{l_{in} \in L,\ l_{in} \neq prec(f,l),\ pour\ lesquels\ \exists f_{in} \in F | suiv(f_{in}, l_{in}) = l\}$ (i.e. $A_{f,l}$ est l'ensemble des liens qui sont utilisés immédiatement avant l par au moins un flux f_{in} et qui ne sont pas utilisés par f) et $U_{l_{in}} = \{f_{in} \in F | l_{in} \in links(f_{in})\}$ (i.e. $U_{l_{in}}$ est l'ensemble des flux qui utilisent le lien l_{in}).

Supposons maintenant que le Group Adaptative Routing soit utilisé et que l fasse partie d'un groupe d'au moins deux liens. On notera n_l le nombre de liens du groupe auquel appartient le lien l, y compris l. Dans ce cas, les paquets entrant par des ports de $A_{f,l}$ ne seront pas transmis séquentiellement sur l mais répartis sur les liens du groupe. Ainsi, dès qu'un lien du groupe se libère, le routeur envoie immédiatement un des paquets en attente sur ce lien. Par conséquent, dans le pire des cas, un paquet du flux f est transmis seulement lorsqu'un lien est libre et qu'il ne reste plus aucun autre paquet en attente. Le délai subi est donc égal au temps d'occupation du lien le moins utilisé parmi les n_l liens du groupe. Ceci nous permet de calculer le délai pour une distribution donnée des paquets sur les n_l liens du groupe.

Notons $(A^k_{f,l})_{k\,\in\,\{1,\ldots n_l\}}$ une partition de $A_{f,l}$ en n_l parties. Si $prec(f,l)$ fait partie d'un groupe, $A_{f,l}$ contient tous les liens de ce groupe sauf celui emprunté par le paquet considéré. Le délai causé par les paquets traversant l avant un paquet du flux f est alors :

$$D((A^k_{f,l})) = \min_{k\in\{1,\ldots n_l\}} \sum_{l_{in}\in A^k_{f,l}} [\max_{f_{in}\in U_{l_{in}}} d(f_{in}, suiv(f_{in},l)) + d_C] \qquad [2]$$

Pour obtenir le délai pire-cas subi par un paquet de f, il ne nous reste plus qu'à calculer [2] pour chaque partitionnement possible de $A_{f,l}$ en n_l parties. Les partionnements tels que certaines parties soient vides sont valides. Au final, $d(f,l)$ s'exprimera donc ainsi :

$$d(f,l) = \begin{cases} \frac{T_f}{C} & \text{si } l = null \\ \sum_{f_{in}\in S_F} d(f_{in}, suiv(f_{in},l)) + d(f, suiv(f,l)) & \text{si } l{=}premier(f) \\ \max_{((A^k_{f,l})_k)} D((A^k_{f,l})) + d(f, suiv(f,l)) + d_C & \text{sinon} \end{cases}$$

[3]

Essayer tous les partitionnements possibles de $A_{f,l}$ nous oblige à prendre en compte un certain nombre de situations qui ne peuvent se produire en réalité. Cela n'est pas un problème car ces situations mènent à des délais meilleurs que la réalité. Elles n'ont donc pas d'influence sur le délai pire-cas puisque nous prenons le maximum des délais engendrés.

Notons que le troisième cas de [3] est valable également pour $n_l = 1$ et cohérent avec notre première formulation du délai : si $n_l = 1$, il n'y a qu'une seule partition possible de $A_{f,l}$ si bien que le troisième cas de [3] devient :

$$d(f,l) = \sum_{l_{in}\in A_{f,l}} [\max_{f_{in}\in Fl_{in}} [d(f_{in}, suiv(f_{in},l))] + d_C] + d(f, suiv(f,l)) + d_C$$

qui est bien notre première définition de $d(f,l)$.

La preuve de terminaison de la méthode est donnée en Appendice. (3) fournit ainsi une méthode de calcul d'une borne supérieure sur le délai pire-cas de bout en bout d'un flux quelconque f. Cette méthode peut ensuite être utilisée comme outil d'aide au dimensionnement de réseau SpaceWire en utilisant la métrique des délais pire-cas des différents flux comme critère de comparaison des réseaux obtenus en faisant varier la taille des paquets et la topologie comme nous l'illustrerons à la section suivante.

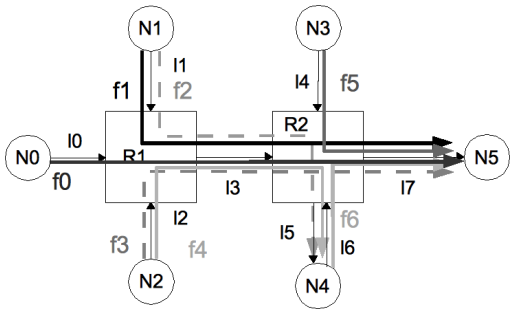

Figure 2. *Exemple de réseau SpaceWire (seuls les arcs utilisés par au moins un flux sont affichés)*

4. Exemple de calcul de délais

4.1. *Application de la méthode de calcul*

Nous allons maintenant présenter le fonctionnement de la méthode sur un exemple de réseau SpaceWire. Ce réseau est présenté sur la figure 2. Il comprend cinq nœuds et deux routeurs SpaceWire. Six flux se partagent le réseau. Il n'est pas nécessaire pour le calcul de connaître le débit de chaque flux, seulement la taille maximale des paquets que nous noterons $T_i,\ i \in \{1, \ldots, 6\}$ respectivement.

Nous devons d'abord déterminer le graphe de dépendance du réseau (voir l'Appendice pour la définition) et vérifier qu'il n'est pas cyclique. Comme on peut le voir sur la figure 3, ce graphe ne contient clairement aucun cycle et le calcul peut donc être effectué. En premier lieu, calculons une borne sur le délai pire-cas du flux 1 qui démarre en N_1 et arrive en N_5.

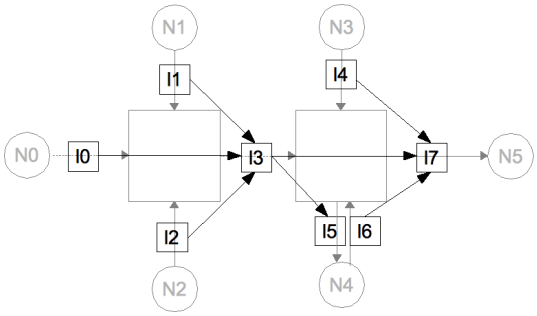

Figure 3. *Graphe de dépendance pour l'exemple de réseau (en gras)*

Nous avons d'abord :

$$\begin{aligned} d(f_1, first(f_1)) &= d(f_1, l_1) \\ &= d(f_2, suiv(f_2, l_1)) + d(f_1, suiv(f_1, l_1)) \\ &= d(f_2, l_3) + d(f_1, l_3) \end{aligned} \quad [4]$$

La deuxième moitié devient :

$$\begin{aligned} d(f_1, l_3) &= \max(d(f_3, l_7), d(f_4, l_5)) + d(f_1, l_7) + 2.d_C \quad \text{où} \\ d(f_1, l_7) &= d(f_5, null) + d(f_6, null) + d(f_1, null) + 3.d_C \\ &= T_1 + T_5 + T_6 + 3.d_C \\ d(f_3, l_7) &= d(f_5, null) + d(f_6, null) + d(f_3, null) + 3.d_C \\ &= T_3 + T_5 + T_6 + 3.d_C \\ d(f_4, l_5) &= d(f_4, null) + d_C = T_4 + d_C \end{aligned}$$

La première partie de [4] se développe en :

$$\begin{aligned} d(f_2, l_3) &= \max(d(f_3, l_7), d(f_4, l_5)) + d(f_2, l_5) + 2.d_C \\ d(f_2, l_5) &= d(f_2, null) + d_C = T_2 + d_C \end{aligned}$$

avec $d(f_3, l_7)$ et $d(f_4, l_5)$ qui ont déjà été calculé.

En substituant les développements dans [4] nous obtenons :

$$d(f_1, l_1) = \frac{T_1 + T_2 + T_5 + T_6}{C} + 2.\max(\frac{T_3 + T_5 + T_6}{C} + 2.d_C, \frac{T_4}{C}) + 9.d_C \quad [5]$$

De la même façon, pour f_4 on obtient :

$$d(f_4, l_2) = \frac{T_3 + T_4 + T_5 + T_6}{C} + 2.\max(\frac{T_1 + T_5 + T_6}{C} + 2.d_C, \frac{T_2}{C}) + 9.d_C \quad [6]$$

Pour f_5, on a d'abord :

$$\begin{aligned} d(f_5, l_4) &= d(f_5, suiv(f_5, l_4)) = d(f_5, l_7) \\ &= \max(d(f_1, null), d(f_3, null)) + d(f_6, null) + d(f_5, null) + 3.d_C \end{aligned}$$

ce qui nous donne :

$$d(f_5, l_4) = \frac{T_5 + T_6 + \max(T_1, T_3)}{C} + 3.d_C \quad [7]$$

4.2. *Quelques remarques sur ces résultats*

Nous pouvons faire plusieurs remarques sur ces résultats. Notons premièrement que [5] et [6] sont similaires. [6] peut être obtenu à partir de [5] en échangeant T_1 et T_3 d'une part, T_2 et T_4 d'autre part. Ceci est cohérent avec le fait que N_1 et N_2 ont des schémas de communication identiques vers N_4 et N_5.

On peut également voir dans [5] que f_5 et f_6 ont un impact plus important que f_2 et f_3 dans le délai subi par f_1. Par suite, à moins que T_4 ne soit plus grand que $T_3 + T_5 + T_6$, au pire, un paquet de f_1 sera retardé trois fois par un paquet de f_5 et un paquet de f_6, deux fois par un paquet de f_3 et une fois par un paquet de f_2.

Notons que ceci est également vrai pour f_4 bien que ce flux n'ait pas la même destination que f_5 et f_6. Ceci est dû au fait qu'un paquet de f_4 peut être bloqué derrière un paquet de f_3 ou f_1 qui peut, à son tour, être bloqué derrière un paquet de f_5 ou f_6. Cet exemple illustre bien le fait que les délais ne sont pas causés par des conflits d'accès à une destination commune mais par des conflits d'accès au lien partagé l_3. Par ailleurs, [7] montre qu'un paquet de f_5 ne peut être retardé qu'une fois par un paquet de f_4 et une fois par un paquet de f_1 ou f_3.

Ces observations suggèrent que lorsque plusieurs nœuds envoient des données à une même destination (une mémoire de masse par exemple) en partageant certains liens, les flux dont la source est proche de la destination ont un impact important sur les flux arrivant d'un nœud plus éloigné dans le réseau. Au contraire, les flux pour lesquels la source est éloignée de la destination ont un faible impact sur ceux partant d'un nœud proche de la destination. Par conséquent, il semble préférable de placer les nœuds émettant les plus longs paquets loin de la destination. Cela permet de réduire de façon importante le délai pire-cas pour les autres nœuds tout en n'augmentant que faiblement ce délai pour les paquets longs.

Toutefois, nous n'avons pris en compte ici que les délais pire-cas, pas les délais ou débits moyens. En effet, pour calculer un délai pire-cas, il nous faut supposer qu'à chaque lien, des paquets provenant des nœuds proches de la destination vont être entrelacés avec des paquets provenant de nœuds plus éloignés. Bien sûr, cela ne se produit pas forcément à chaque fois pour chaque paquet et chaque lien. Il se peut donc que placer les nœuds émettant des paquets longs loin de leur destination n'aboutisse qu'à bloquer tous les liens suivis par ces paquets pendant une durée plus importante. Cela peut également retarder d'autres flux adressés à d'autres destinations mais utilisant certains liens communs avec ces paquets de grande taille.

4.3. *Application numérique*

Afin de mieux comprendre la signification de ces formules, il peut être utile d'en considérer une application numérique. Nous utiliserons les valeurs suivantes pour les tailles maximales des différents paquets :

T_1, T_3 : 5120 octets ; T_2, T_4 : 50 octets ; T_5, T_6 : 1000 octets

Flux	sans segmentation		avec segmentation	
	Meilleur cas	Pire cas	Meilleur cas	Pire cas
f_1 (segment)	–	–	$3,5\ \mu s$	$346\ \mu s$
f_1 (paquet)	$257\ \mu s$	$1,08\ ms$	$71,2\ \mu s$	$6,9\ ms$
f_4	$3,5\ \mu s$	$1,08\ ms$	$3,5\ \mu s$	$346\ \mu s$
f_5	$50\ \mu s$	$357\ \mu s$	$50\ \mu s$	$113\ \mu s$

Tableau 1. *Délais meilleur et pire-cas pour l'exemple de réseau*

Ces valeurs correspondent à des tailles de paquets typiques si, par exemple, N_1 et N_2 sont des instruments d'observation, N_4 un processeur, N_5 une mémoire de masse et N_3 un équipement de monitoring. N_1 et N_2 envoient des données à haut débit et avec des tailles de paquet importantes vers N_5. Tous deux envoient également des paquets de monitoring vers N_4 qui en fait une synthèse puis transmet le résultat à N_5. N_3 envoie lui aussi ses données à N_5. Les flux f_2 et f_4 sont considérés comme critiques. Tous les liens ont une capacité $C = 200\ Mbps$ et le délai de commutation est $d_C = 0,5\ \mu s$ pour tous les routeurs. Nous pouvons maintenant calculer les délais pour f_1, f_4 et f_5 dans le meilleur et le pire des cas. Les résultats sont donnés dans la partie gauche du tableau 4.3.

Comme prévu d'après leurs formules, f_1 et f_4 ont le même délai pire-cas. En revanche, leurs délais dans le meilleur des cas sont très différents. Le rapport entre leurs délais dans le meilleur et le pire des cas est donc beaucoup plus faible pour f_1 (ratio de 4) que pour f_4 (ratio de 308). Le blocage récursif mène ainsi à de longs délais même pour les paquets de faible taille. Pour f_5, le rapport est de 7. Comme prévu, un flux dont la source est proche de la destination est peu influencé par les autres flux.

Nous pouvons ensuite essayer de segmenter les grands paquets de f_1 et f_3 en petits paquets de 256 octets pour réduire le délai pire-cas de f_4. Comme on peut le voir dans la partie droite du tableau 4.3, cela permet de réduire le délai pire-cas de f_4 à 0,346 ms soit environ un tiers de sa valeur précédente. Le délai pire-cas de f_5 est lui aussi réduit dans les mêmes proportions.

Cependant, la segmentation a un coût pour f_1. En effet, étant donné qu'il est maintenant nécessaire d'envoyer 20 segments pour transmettre la même quantité d'information qu'avant, le délai pire-cas pour un paquet complet est maintenant de 6,9 ms soit presque 7 fois plus que sans segmentation. La méthode de calcul peut ainsi aider à déterminer la taille optimale des paquets des différents flux en fonction des délais de bout en bout que l'on veut respecter pour chacun des flux.

5. Conclusion et perspectives

Nous avons présenté une méthode de calcul simple qui nous permet de calculer une borne supérieure sur le délai pire-cas de bout en bout des flux traversant un réseau SpaceWire. Cette borne est telle qu'elle ne peut pas être dépassée par un paquet tant que le réseau ne subit pas de pannes. Cette propriété est particulièrement importante pour le trafic de contrôle qui doit être transmis en temps et en heure pour assurer le

bon fonctionnement du satellite. La méthode utilise une définition récursive du délai subi par chaque paquet dans chacun des routeurs qu'il traverse.

De plus, l'exemple présenté nous a permis de mettre en évidence l'importance de l'écart entre les délais dans le meilleur et le pire des cas pour un flux donné. L'exemple a également illustré le fait que l'impact de chaque flux sur les délais subis par les autres flux dépend de la distance entre la source et la destination du flux. De plus, développer formellement les résultats de la méthode de calcul permet de déterminer le poids des différents flux dans le délai subi par chacun des autres. Enfin, nous avons montré sur l'exemple comment notre méthode pouvait être utilisée comme outil de dimensionnement de réseaux utilisant les délais pire-cas de bout en bout comme métrique.

Trois pistes s'offrent à nous pour étendre notre méthode de calcul. La première est de terminer la modélisation de SpaceWire en prenant en compte le mécanisme de priorités mis en jeu dans les routeurs. Ce mécanisme est important puisqu'il devrait permettre d'obtenir des délais pire-cas plus faibles pour les flux prioritaires.

Pour le moment, nous ne faisons pas d'hypothèses fortes sur le trafic en entrée du réseau. Cela permet d'avoir une méthode de calcul générale mais la rend également plus pessimiste. Nous prévoyons donc d'utiliser une modélisation plus fine du trafic en entrée pour obtenir des bornes plus serrées. Par exemple, nous pourrions utiliser une courbe d'arrivée pour décrire le trafic en entrée plutôt que de supposer que chaque flux essaie d'émettre le plus vite possible.

Une autre direction intéressante est de modéliser SpaceWire-RT en utilisant la méthode que nous avons mise au point pour SpaceWire. SpaceWire-RT (Parkes *et al.*, 2009) est une extension de SpaceWire conçue comme une couche de Qualité de Service entre le réseau SpaceWire et les applications. Cette extension est encore en cours d'élaboration mais elle devrait permettre de fournir de meilleurs garanties temporelles aux flux critiques. Grâce à notre méthode, nous devrions être capables de déterminer l'amplitude des progrès apportés par SpaceWire-RT.

Remerciements

Ces travaux ont été financés par une bourse de recherche du CNES et de TAS.

6. Bibliographie

Boudec J.-Y. L., Thiran P., « NETWORK CALCULUS A Theory of Deterministic Queuing Systems for the Internet », *Springer Verlag*, n° LNCS 2050, p. 1-265, Apr, 2004.

Dally W., « Performance analysis of k-ary n-cube interconnection networks », *Computers, IEEE Transactions on*, vol. 39, n° 6, p. 775 - 785, Jun, 1990.

Dally W., Seitz C., « Deadlock-Free Message Routing in Multiprocessor Interconnection Networks », *Computers, IEEE Transactions on*, vol. C-36, n° 5, p. 547 - 553, May, 1987.

Draper J. T., Ghosh J., « A Comprehensive Analytical Model for Wormhole Routing in Multicomputer Systems », *Journal of Parallel and Distributed Computing*p. 1-34, Jan, 1994.

ECSS, « SpaceWire – Links, nodes, routers and networks », p. 1-129, Aug, 2008.

Mifdaoui A., Frances F., Fraboul C., « Real-time characteristics of Switched Ethernet for "1553B"-Embedded Applications: Simulation and Analysis », *Industrial Embedded Systems, 2007. SIES '07. International Symposium on*p. 33 - 40, Jun, 2007.

Parkes S., Florit A. F., « SpaceWire-RT Initial Protocol Definition v2.1 », p. 1-120, Feb, 2009.

Parkes S. M., Armbruster P., « SpaceWire: A Spacecraft Onboard Network for Real-Time Communications », *IEEE-NPSS Real Time Conference*, n° 14, p. 1-5, Feb, 2005.

Annexe. Preuve de terminaison du calcul

Une définition préalable est nécessaire à la démonstration. On appelle *graphe de dépendance des liens* d'un réseau représenté par le graphe $\mathcal{G}(N, L)$, le graphe orienté $D = \mathcal{G}(L, E)$ dont les nœuds L sont les liens de $\mathcal{G}(N, L)$ et les arcs E sont les couples de liens utilisés successivement par au moins un flux :
$E = \{(l_i, l_j) \in L \; pour \; lesquels \; \exists f \in F | suiv(f, l_i) = l_j\}$

Théorème 1 *Pour tout flux f et tout lien l de $liens(f)$, le calcul de $d(f, l)$ termine si et seulement si D est acyclique.*

Preuve : $\Rightarrow$ Supposons qu'il existe un cycle dans D. Etant donné que $suiv(f, l)$ ne peut renvoyer l pour aucun l (un paquet n'emprunte pas deux fois de suite le même lien), la longueur est supérieure ou égale à 2. Vu la définition de d, cela veut dire qu'il est possible de trouver un flux f et un lien l tels que calculer $d(f, l)$ nécessite un appel de $d(f, l)$ pendant la récursion. Par suite, le calcul de $d(f, l)$ ne termine pas.

$\Leftarrow$ Supposons que D est acyclique. Il est alors possible d'utiliser le tri topologique pour assigner un ordre total aux sommets de D tel que si $(l_i, l_j) \in E$ alors $l_i > l_j$ (si D n'est pas connexe, il est possible d'ordonner chaque composant connexe et d'ordonner ceux-ci arbitrairement ensuite). D'après la définition de E et de $suiv(f, l)$, il vient que calculer $d(f, l)$ ne nécessite que des appels à d avec des liens inférieurs à l comme paramètres. Etant donné que F est fini et que $d(f, l)$ n'est appelé qu'une fois pour un f et un l donnés, il s'ensuit que le calcul de $d(f, l)$ termine toujours. ■

Notons qu'il est démontré dans (Dally *et al.*, 1987) que pour un réseau Wormhole sous des hypothèses semblables aux nôtres, il est équivalent de dire que le graphe de dépendance est acyclique et que le réseau ne contient pas d'interblocages. Un réseau sujet aux interblocages n'étant pas utilisable en pratique de toutes façons, le Théorème 1 montre que le calcul se termine dans tous les cas de figure pertinents. Ceci est également cohérent avec le fait que, lorsque le calcul ne finit pas, $d(f, l)$ tend vers l'infini ce qui est caractéristique d'un interblocage.