Cadre basé sur les réseaux de Petri pour l'analyse de la Qualité de Service

Slim Abdellatif et Guy Juanole †

Résumé

Ce papier présente un cadre basé sur les Réseaux de Petri Stochastiques (RdPS) pour l'analyse de la Qualité de Service (QdS) dans les réseaux à commutation de paquets. Nous y présentons les modèles RdPS des principaux algorithmes qu'implémentent les mécanismes de régulation de trafic, de gestion de buffer, et d'ordonnancement de paquets, ainsi qu'une méthode d'analyse permettant de mesurer les principales métriques de QdS, à savoir : le taux de pertes de paquets associé à un flux, le délai moyen de transfert des paquets d'un flux, des bornes supérieures et inférieures sur le délai de transfert des paquets d'un flux, l'utilisation de la bande passante d'une ligne, etc.

Mots clés

modélisation réseaux de Petri, mécanismes de fourniture de Qualité de Service, évaluation de performance

I. Introduction

Durant ces dernières années, la fourniture de Qualité de Service (QdS) dans les réseaux à commutation de paquets a fait l'objet d'un grand nombre de travaux de recherche. Ces travaux ont permis d'identifier un certain nombre de mécanismes clés pour la fourniture de services différenciés et de garanties de QdS. Ces mécanismes sont : l'ordonnancement de paquets, la gestion de buffer, la régulation de trafic, le contrôle d'admission, et le routage QdS ; ils constituent le squelette des architectures de QdS émergeantes.

En pratique, c'est la simulation à événements discrets qui est généralement utilisée pour l'analyse de tels mécanismes. Avec la simulation, il est possible de modéliser ces mécanismes, généralement complexes, sans prendre des hypothèses très contraignantes. En revanche, la simulation souffre de son coût qui augmente avec le degré de confiance que l'on souhaite avoir. De plus, à cause de sa non-exhaustivité, la simulation est inadaptée pour la mesure de paramètres de QdS, tels que les taux de perte de paquets qui ont des valeurs très faibles (problème des événements rares). Pour ces mêmes raisons, la simulation s'avère inadaptée pour des analyses nécessitant des garanties déterministes de QdS. Par exemple, si on s'intéresse à la détermination d'une borne supérieure sur le délai de transfert des paquets d'un flux, on ne peut pas affirmer que le plus grand délai mesuré corresponde à une borne. En effet, il se peut que le scénario amenant au délai le plus élevé n'ait pas été généré lors de la simulation.

D'un autre côté, la complexité des mécanismes de fourniture de QdS rend les études analytiques peu envisageables en pratique. A notre connaissance, le peu d'études qui ont été réalisées se sont focalisées sur un seul mécanisme, généralement le mécanisme d'ordonnancement de paquets. Elles se sont intéressées à la détermination de garanties déterministes de QdS, tels que le calcul d'une borne sur le délai de bout-en-bout d'un paquet d'un flux sous des hypothèses d'homogénéité du réseau, i.e. le réseau emploie le même algorithme d'ordonnancement dans tous les noeuds.

L'analyse formelle constitue une alternative à la simulation et aux approches analytiques, et le modèle Réseaux de Petri Stochastiques (RdPS) est un bon candidat. En effet,

- grâce à leur pouvoir d'expression, la plupart des mécanismes cités ci-dessus peuvent être modélisés ;
- le temps, ainsi que certains aspects probabilistes, peuvent être représentés par les RdPS, ce qui est primordial pour l'analyse de performances ;
- des techniques reconnues existent pour réaliser des analyses formelles à partir des RdPS.

Ce travail présente un cadre basé sur les RdPS permettant de réaliser des analyses de QdS dans les réseaux à commutation de paquets, et plus spécifiquement dans les réseaux ATM et IP. Dans ce cadre, nous présentons :

- une modélisation par RdPS des algorithmes les plus implémentés en pratique par les mécanismes qui agissent directement sur les paquets, à savoir : l'ordonnancement de paquets, la gestion de buffer, et la régulation de

†LAAS - CNRS. 7,Avenue du colonel Roche. 31077 Toulouse (FRANCE). e-mails : {slim,juanole}@laas.fr

trafic (dans la communauté IP, ces mécanismes sont appelés mécanismes du chemin de données (Data Path mechanisms)). Ces modèles permettent ensuite de construire le modèle RdPS d'un réseau (ou d'un bout de réseau) quelconque ; et

- une méthode d'analyse permettant de conduire des mesures de performances sur le modèle RdPS d'un réseau. Ces mesures concernent des métriques de QdS classiques telles que le délai de transfert moyen, le taux de perte de paquets, des bornes sur le délai de transfert, l'utilisation du réseau, etc.

Le plan de cet article se présente comme suit. La section II introduit les mécanismes pour la fourniture de QdS. La section III présente les principales caractéristiques des modèles RdPS considérés. La section IV est consacrée à la modélisation. Les modèles RdPS des algorithmes implémentés par les mécanismes du chemin des données y sont décrits. Faute de place, nous ne présenterons pas les modèles de tous les algorithmes que nous avons considérés dans ce cadre. Le lecteur intéressé pourra se référer à [1]. La cinquième section présente la partie analyse de performances du cadre. Un bref aperçu de la méthode employée pour calculer les métriques de QdS y est présenté. La section VI présente l'application du cadre à un exemple académique. La section VII décrit deux applications du cadre que nous avons dernièrement entreprises. Enfin, la septième section conclut l'article.

II. Mécanismes pour la fourniture de QdS

Nous utiliserons le terme *flux* pour dénoter une séquence de paquets qui suit une même route d'un point donné du réseau vers un autre point. Souvent, il sera utilisé pour dénoter l'ensemble des paquets d'une application qui sont véhiculés de la source (de l'application) vers la destination. Dans les réseaux ATM, cela correspond à une connexion ; dans les réseaux IP, son existence est temporaire et se termine lors d'un changement de route.

Il est actuellement admis que tout flux qui souhaite avoir des garanties de QdS doit caractériser le trafic qu'il va soumettre au réseau. Cette caractérisation est indispensable pour le réseau afin de pouvoir estimer les ressources nécessaires au flux et par conséquent décider s'il peut accepter le flux ou le rejeter, faute de ressources.

Plusieurs mécanismes de base ont été identifiés comme étant nécessaires pour la fourniture de QdS. Dans ce papier nous nous focalisons sur les mécanismes qui agissent directement sur les paquets en les retardant ou en les éliminant. Il s'agit des mécanismes de régulation de trafic, de gestion de buffer, et d'ordonnancement de paquets. Nous les introduisons ci-après.

A. la régulation de trafic

Les mécanismes de régulation de trafic regroupent deux types de mécanismes qui ont des comportements très similaires : les mécanismes de mise en vigueur de trafic et les mécanismes de mise en forme de trafic. Dans la suite, nous explicitons le rôle de ces deux mécanismes.

La mise en vigueur de trafic contrôle le trafic que soumet chaque flux au réseau afin de s'assurer qu'il est conforme à la caractérisation de trafic qu'avait faite le flux lors de son acceptation par le réseau (cette caractérisation constitue un contrat entre le réseau et le flux). Le but de la mise en vigueur de trafic est de protéger les flux bienveillants (ceux qui respectent leur contrat de trafic) des flux malveillants. En effet, la quantité de ressources réservée à un flux est calculée à partir de son contrat de trafic. S'il ne respecte pas son contrat, le flux risquerait de consommer des ressources qui ne lui sont pas destinées, ce qui pourrait nuire à d'autres flux. Lorsqu'un flux ne respecte pas son contrat, le mécanisme de mise en vigueur de trafic sanctionne les paquets non conformes, en les éliminant, en leur affectant une priorité faible (paquets qui seront éliminés en premier, en cas de congestion), ou en les retardant jusqu'à ce qu'ils deviennent conformes.

Comme pour la mise en vigueur de trafic, la mise en forme de trafic contrôle le trafic que génère une source. Mais, contrairement aux mécanismes de mise en vigueur qui sont employés par le réseau, c'est l'utilisateur final (côté émetteur) qui emploie les mécanismes de mise en forme de trafic. Ils ont pour rôle de déformer (si nécessaire) le trafic généré par la source afin de le rendre conforme au contrat de trafic que cette dernière a établi avec le réseau. La raison d'être de tels mécanismes vient du fait que la caractérisation de trafic n'est pas une tâche triviale et l'utilisateur final peut par méconnaissance déclarer un trafic qui ne corresponde pas exactement à la réalité. En mettant un mécanisme de mise en forme entre sa source de paquets (i.e. côté émetteur) et l'entrée du réseau, l'utilisateur évite à certains de ses paquets les sanctions drastiques de la mise en vigueur de trafic.

La fonction principale des mécanismes de régulation de trafic est de déterminer les paquets non conformes. Un paquet est dit non conforme lorsque, en le rajoutant au volume de trafic qui est déjà arrivé, le contrat de trafic n'est plus respecté. L'algorithme de détection de paquets non conformes dépend du modèle de trafic que le régulateur cherche à faire respecter. Pour un modèle de trafic donné, l'algorithme est le même pour la mise en forme et la mise en vigueur de trafic. En effet, l'unique différence entre ces deux types de mécanisme réside dans les actions correctrices qu'ils entreprennent à la détection d'un tel paquet. Le premier le retient jusqu'à ce qu'il devienne conforme. Suivant la politique adoptée par le réseau, les mécanismes de mise en vigueur du trafic peuvent soit retarder le paquet jusqu'à ce qu'il devienne conforme (même fonctionnement que la mise en forme du trafic), soit affecter au paquet une priorité moindre (e.g. champ CLP de l'en-tête d'une cellule ATM), soit simplement l'éliminer. Un paquet conforme sera, quant à lui, instantanément libéré par les deux mécanismes.

Plusieurs types de régulateurs ont été proposés dans la littérature. Les principaux sont : le régulateur à débit de crête, le seau à jeton, le seau percé, le régulateur $(\vec{\sigma}, \vec{\rho})$ [2], et le D-BIND [3]. Dans la suite, nous ne décrivons que le régulateur appelé *seau à jetons*. Nous nous focaliserons dans cette description sur la détection de paquets non conformes.

Le régulateur du *seau à jetons* est employé dans les réseaux IP et ATM. Des jetons sont générés avec un débit constant ρ et stockés dans un seau de taille σ. Si à l'arrivée d'un paquet, la quantité de jetons disponibles est supérieure à la taille du paquet, ce dernier est conforme. Il quitte alors le régulateur en enlevant du seau un nombre de jetons équivalent à sa taille. Dans le cas contraire, le paquet est non conforme.
Le régulateur du seau à jetons permet de garantir que le volume de trafic sortant sur n'importe quel intervalle de durée τ est inférieur à $\sigma + \rho.\tau$. Les paramètres ρ et σ peuvent respectivement être vus comme le débit moyen maximum au long terme et la taille maximale de rafale.

B. la gestion de buffer

Les mécanismes de gestion de buffer contrôlent l'accès à un buffer d'un noeud. Lorsque la charge du noeud devient élevée (i.e. buffer commence à se saturer), ils ont pour charge de choisir parmi les paquets qui arrivent, ceux à éliminer et les ceux à stocker. Plusieurs algorithmes de gestion de buffer ont été proposés dans la littérature pour la fourniture de QdS. Les principaux algorithmes sont : le Drop Tail (DT), Partial Packet Discard (PPD) pour ATM [4], Earlier Packet Discard (EPD) pour ATM [4], et Random Early Detect (RED) [5] et toutes ses variantes pour IP. Dans la suite, nous ne décrivons que l'algorithme dont le modèle va être décrit ci-dessous, à savoir l'algorithme DT.

L'algorithme *Drop Tail* est le plus simple algorithme de gestion de buffer. Il peut être utilisé dans les réseaux IP et ATM. Son fonctionnement est le suivant : un paquet qui arrive est éliminé seulement s'il n'y a pas de place disponible dan le buffer. Dans les autres cas, il est toujours accepté.

C. l'ordonnancement de paquets

Les mécanismes d'ordonnancement de paquets ont pour rôle d'arbitrer l'accès à une ligne de transmission en ordonnant la transmission des paquets mémorisés. Plus spécifiquement, lorsque la ligne devient libre, ils choisissent parmi les paquets qui sont stockés dans le buffer, celui qui va être transmis en premier. Plusieurs algorithmes d'ordonnancement ont été proposés dans la littérature pour la fourniture de QdS. Les principaux algorithmes sont : Premier Arrivé Premier Servi (PAPS), algorithme à priorités statiques (PS), Fair Queueing (FQ) [6] [7] et toutes ses variantes, le Weighted Round Robin (WRR), le Deficit Round Robin (DRR) [8], et la famille Rate-Controlled Service Discipline (RCSD) [9]. Dans la suite, nous ne décrivons que l'algorithme dont le modèle est décrit dans la section ci-après, à savoir l'algorithme PS.

Les algorithmes à priorités statiques sont une amélioration du PAPS puisqu'ils offrent plusieurs niveaux de service. Le mécanisme d'ordonnancement PS maintient une file par indice de priorité. Chaque flux est associé à une file, et les paquets d'une même file sont servis en PAPS lorsque les files de priorité supérieure sont vides.

III. Modèles RdPS considérés

Plusieurs modèles RdPS ont été développés durant la dernière décennie. Le modèle *Stochastic Petri Nets (SPN)* [10] [11], le modèle *Generalized Stochastic Petri Nets (GSPN)* [12], le modèle *Deterministic Stochastic Petri Nets (DSPN)* [13], et le modèle *Réseaux de Petri Temporisés Stochastiques (RdPTS)* [14] [15] développé au LAAS-CNRS, ne constituent qu'une partie d'une liste beaucoup plus longue. Afin que le cadre que nous présentons puisse être utilisé avec plusieurs modèles RdPS, nous présentons dans la suite une liste d'exigences que doit satisfaire tout modèle RdPS qui utilise le cadre. Ces exigences sont déduites des besoins rencontrés lors de la modélisation des mécanismes pour la fourniture de QdS et sont classées en deux groupes : celles relatives au pouvoir d'expression et celles relatives au pouvoir d'analyse.

A. Exigences sur le pouvoir d'expression du modèle RdPS considéré

Les modèles RdPS que nous considérons sont une extension temporisée des réseaux de Petri place/transition avec arcs inhibiteurs. Une variable aléatoire est associée à chaque transition pour décrire sa durée de sensibilisation (franchissement), i.e. le temps qui s'écoule entre l'instant où la transition devient sensibilisée et l'instant où elle se tire. La variable aléatoire est spécifiée par une densité de probabilité, appelée *densité de tir*, définie sur un intervalle de temps qui délimite les instants de tirs possibles. La borne inférieure de l'intervalle représente le temps minimal que la transition doit attendre depuis l'instant de sensibilisation pour pouvoir être tirée, alors que la borne supérieure représente le temps maximal qu'une transition sensibilisée peut attendre avant d'être tirée.

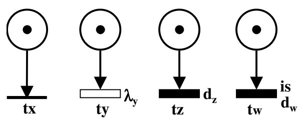

Fig. 1. Les symboles dans les RdPS

Les modèles RdPS doivent :
1. supporter les densités de tir suivantes :
• $\delta(x), x \in [0, \infty[$. Les transitions avec ce type de distribution sont tirées en un temps nul. Elles sont appelées transitions *immédiates* et sont représentées graphiquement sous la forme d'une barre mince (voir la transition **tx** de la figure 1) ;
• $\lambda \exp(-\lambda x), x \in [0, \infty[$ où le paramètre λ peut dépendre d'un marquage. Une transition avec ce type de distribution est appelée transition *exponentielle*. Elle est représentée graphiquement sous la forme de rectangle vide à côté duquel la valeur de λ est inscrite (voir la transition **ty** sur la figure 1) ;
• $\delta(x - d), x \in [0, \infty[$ avec $d \neq 0$. La durée de sensibilisation d'une telle transition dure exactement d unités de temps. Les transitions de ce type sont appelées transitions *déterministes*. Le paramètre d peut même dépendre du marquage [16]. Une transition déterministe est représentée graphiquement sous la forme d'une barre épaisse à côté de laquelle la valeur de d est inscrite (voir la transition **tz** à la figure 1) ;
2. pouvoir affecter des indices de priorité aux transitions afin d'ordonner leurs tirs. Les transitions temporisées (exponentielles ou déterministes) ont la plus faible priorité, i.e. l'indice de priorité 0 leur est affecté automatiquement. Les transitions immédiates peuvent, quant à elles, prendre plusieurs niveaux de priorités. Leur indice de priorité peut varier de 1 à l'infini. Plus l'indice est élevé, plus la priorité de tir de la transition est grande ;
3. être capable de lever le non-déterminisme engendré par deux transitions immédiates en conflit, *stochastiquement*. Ceci est assuré en affectant des poids (dont la valeur par défaut est 1 et qui peuvent dépendre du marquage), dits *poids de tir*, aux transitions immédiates. Lorsque plusieurs transitions immédiates se retrouvent en conflit, leur poids sont utilisés pour calculer la probabilité avec laquelle chaque transition est tirée. Plus précisément, supposons que le modèle RdPS se trouve dans un marquage M où un ensemble $Sc(M)$ de transitions immédiates sont en conflit, la probabilité que la transition i de $Sc(M)$ se tire en M est

$$P_i(M) = \frac{poids\ de\ i\ en\ M}{\sum_{j \in Sc(M)} poids\ de\ j\ en\ M} \tag{1}$$

4. supporter la politique **de course** comme politique de tir (*race policy*), i.e. la transition qui a la plus petite durée de sensibilisation est tirée en premier ;
5. supporter comme politique mémoire : *la mémoire de la dernière sensibilisation*(*enabling memory*). Une transition adoptant une telle politique a une mémoire qui remonte jusqu'au dernier instant où elle s'est sensibilisée. En d'autres termes, tant qu'elle reste sensibilisée, à chaque tir de transition, la durée de temps qui s'est écoulée depuis sa sensibilisation est pris en compte pour calculer sa nouvelle densité de tir. Une fois désensibilisée, à la prochaine resensibilisation elle reprend sa densité de tir initiale.;
6. supporter aussi bien la sémantique mono-serveur que infini-serveur (i.e. à nombre infini de serveurs). Considérons un marquage M où une transition t est multi-sensibilisée [12]. Si t est infini-serveur, il y a plusieurs (autant que le nombre de sensibilisations) instances de t qui s'exécutent en concurrence. Dans le cas mono-serveur, il n'y a qu'une instance de t à la fois. Une transition infini-serveur est graphiquement identifiée par l'étiquette **i.s** qui est placée juste à côté de la transition (voir la transition **tw** sur la figure 1). Par défaut, toute transition a la sémantique mono-serveur.

B. Exigences sur le pouvoir d'analyse du modèle RdPS considéré

Nous nous intéressons à des analyses en régime stationnaire. Les seules exigences sur le pouvoir d'analyse du modèle RdPS est qu'il soit capable de calculer :
- la fréquence moyenne de tir de transitions en régime stationnaire ;
- le marquage moyen de places en régime stationnaire.

Nous notons par $R(t)$ la fréquence moyenne de tir d'une transition t, et par $E[P]$ le marquage moyen d'une place P.

Les exigences que nous venons d'énumérer sont toutes supportées par les modèles RdPS suivants : le modèle STPN, le modèle DSPN, et le modèle GSPN si l'on approche une distribution déterministe à l'aide d'une distribution erlangienne. Des techniques reconnues pour l'analyse en régime permanent sont associées à chacun de ces modèles. Ces techniques sont implantées dans les outils suivants : STPN [15] pour le modèle RdPTS, DSPN-express [17] pour le modèle DSPN, et GreatSPN [18] pour le modèle GSPN.

IV. Modélisation

A. Principes, hypothèses, et notations

Les analyses que nous visons se basent sur une exploitation des métriques de QdS associées aux flux de paquets. Ainsi, notre principal objectif est d'être capable de mesurer ces métriques. D'un point de vue modélisation, cela implique que la vue du réseau que nous devons avoir doit se confondre avec la vue qu'ont les paquets du réseau. En d'autres termes toutes les actions dictées par les mécanismes pour la fourniture de QdS et que subissent les paquets, doivent être modélisées explicitement. Pour ce faire, nous proposons de modéliser les paquets par des *jetons*(réseaux de Petri). Les actions des mécanismes pour la fourniture de QdS sont modélisées par des transitions qui retardent ou éliminent les jetons.

Ce choix de modélisation nous oblige à supposer que les paquets sont tous de même taille. En effet, à notre connaissance, il n'existe pas de méthodes et d'outils qui permettraient à partir de réseaux de Petri colorés stochastiques de calculer formellement, sans hypothèse sur leur structure, la fréquence moyenne de tir des transitions et le marquage moyen des places. Dans les réseaux ATM, cette hypothèse est valide puisque les paquets, appelés cellules, ont tous une taille de 53 octets. Dans les réseaux IP, cela n'est pas le cas et cette hypothèse s'avère contraignante. Dans la suite, nous prenons la taille d'un paquet comme unité de mesure pour exprimer des paramètres tels que la taille buffer, le débit moyen, la taille d'une rafale, etc.

Une autre hypothèse concerne l'architecture des noeuds qui sont à mémorisation en sortie [19], c'est-à-dire les buffers sont situés au niveau des interfaces de sortie (une par ligne de sortie), et non-bloquants, c'est-à-dire qu'un paquet ne peut pas être perdu au niveau de la fabrique de commutation. Avec une telle architecture, seuls les flux d'une même interface de sortie interagissent entre eux. Ainsi, au lieu de modéliser des noeuds entiers, il suffit de modéliser seulement les interfaces de sortie qui sont traversées par les flux qui nous intéressent. Cette hypothèse réduit sensiblement la taille du RdPS à analyser. Elle est aussi réaliste puisqu'une grande proportion de commutateurs ATM et de routeurs commerciaux ont une telle architecture.

Enfin, nous négligeons les délais de propagation ainsi que les délais de commutation / de routage (délais que connaît un paquet au niveau d'un noeud pour atteindre l'interface de sortie).

En adoptant toutes les hypothèses citées jusqu'ici, nous présentons sur la figure 2 un exemple typique des mécanismes traversés par un flux f.

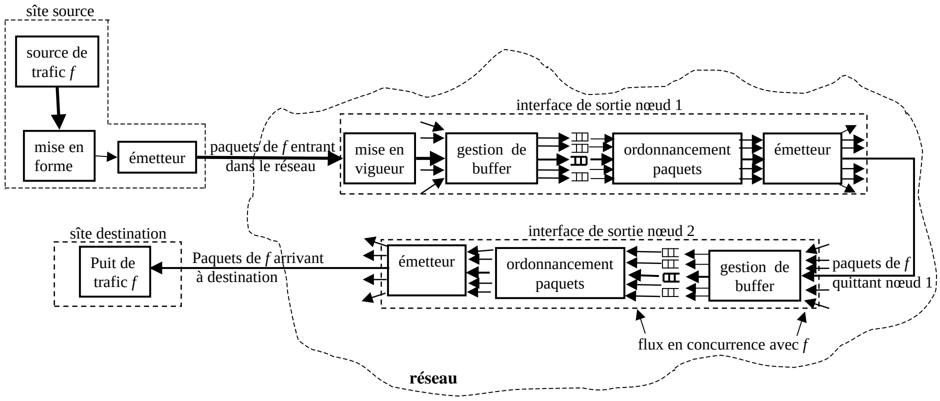

Fig. 2. Mécanismes traversés par un flux f

Nous adoptons une approche modulaire pour développer le modèle RdPS d'un réseau. Chaque mécanisme est modélisé par un module RdPS qui décrit son comportement vis-à-vis des flux qui le traversent. Chaque module comprend deux places pour chaque flux, une modélisant les paquets du flux qui arrivent à l'entrée du mécanisme, et l'autre modélisant les paquets du flux qui en sortent. Similairement à la technique de composition présentée dans [20], ces places sont respectivement qualifiées de *places port d'entrée* et *places port de sortie* d'un module. Comme nous allons le voir dans la section IV-G, ces places jouent un rôle primordial lors de l'interconnexion des modules pour construire le modèle RdPS global du réseau. Elles seront représentées graphiquement par des cercles hachurés.
Nous présentons, à partir de la section suivante, des modules RdPS des principaux algorithmes que implémentent les mécanismes pour la fourniture de QdS. Contrairement aux mécanismes de mise en vigueur et de mise en forme du trafic qui régulent le trafic appartenant à un seul flux, les mécanismes d'ordonnancement de paquets et de gestion de buffer sont traversés par plusieurs flux en même temps. Le nombre et les caractéristiques de ces flux dépendants du réseau à analyser, il n'est donc pas possible de développer un seul RdPS qui décrive le comportement de ces mécanismes dans toutes les situations. Puisque les mécanismes de gestion de buffer et d'ordonnancement ont un comportement générique similaire vis-à-vis de tous les flux qui les traversent, nous ne présenterons que le modèle RdPS qui décrit leur comportement vis-à-vis d'un flux donné. Le modèle complet du mécanisme est ensuite obtenu en dupliquant attentivement le modèle du comportement générique autant de fois que le nombre de flux traversant le mécanisme.

Dans la suite, nous appellerons f le flux impliqué dans la description des modules des différents mécanismes. Pour les mécanismes de gestion de buffer et d'ordonnancement, nous noterons par F, $F = \{f_1, f_2, \dots f_{|F|}\}$, l'ensemble des flux qui les traversent. f peut être n'importe quel flux de F.

B. Modèles RdPS des mécanismes de mise en forme et de mise en vigueur du trafic

Nous ne considérons dans ce travail que les régulateurs de trafic qui retardent ou éliminent les paquets non conformes. Ceux qui sanctionnent les paquets non-conformes, en réduisant leur priorité, ne peuvent être modélisés avec les modèles RdPS que nous considérons. En effet, leur modélisation nécessite l'utilisation de jetons colorés, la couleur représentant la priorité du paquet. Nous appellerons les régulateurs qui Retardent les paquets non conformes, *régulateurs de type R*, et ceux qui Eliminent, *régulateurs de type E*.

Les modules RdPS des régulateurs de trafic possèdent une place port d'entrée et une place port de sortie. La première place s'appelle *paquets de f à réguler* et modélise les paquets qui rentrent dans le régulateur. La seconde place s'appelle *paquets de f régulés* et modélise les paquets qui quittent le régulateur. Nous avons modélisé l'ensemble des régulateurs que nous avons cités dans la section II et qui, d'ailleurs, constituent les principaux régulateurs que l'on rencontre en pratique. Faute de place, nous ne décrivons dans la suite que les modèles du régulateur du seau à jetons.

Les modèles des régulateurs du seau à jetons sont présentés à la figure 3. Le seau est modélisé par la place *Seau*. Puisque les paquets sont de taille constante, un jeton dans la place *seau* représente une permission pour un paquet pour quitter le régulateur. Le seau à jetons est rempli avec un débit ρ tant que le seau n'a pas atteint sa taille maximale de σ jetons (arc inhibiteur de multiplicité σ qui relie la place *seau* à la transition *remplir seau*).

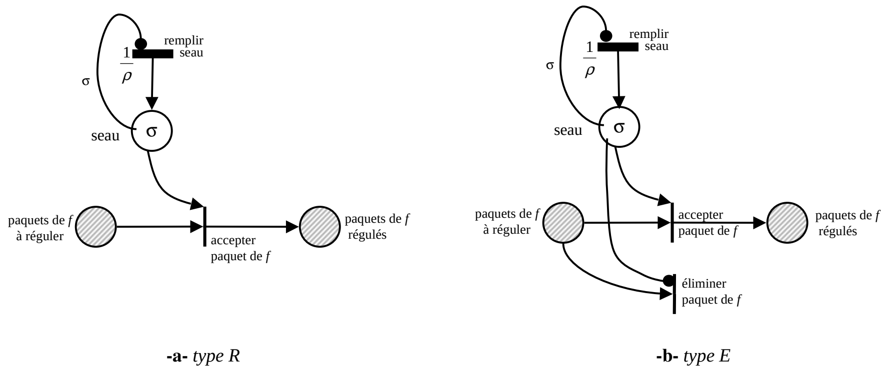

Fig. 3. Modèles RdPS des régulateurs du seau à jetons

Un paquet qui arrive alors que le seau est vide est un paquet non conforme. Le *régulateur de type R* le retient jusqu'à ce que de nouvelles permissions soient générées (place *seau* devient marquée, sur la figure 3-a) alors qu'un *régulateur de type E* l'élimine (tir de la transition *éliminer paquets de* f, sur la figure 3-b).

C. Modèles RdPS des mécanismes de gestion de buffer

Les places port d'entrée et port de sortie, relatifs à un flux f, du module RdPS d'un algorithme de gestion de buffer sont respectivement appelées *arrivée paquet de* f et *paquets de* f *mémorisés*. Elles modélisent l'arrivée d'un paquet de f à l'interface de sortie et les paquets de f qui sont acceptés pour être mémorisés dans le buffer.

Les mécanismes de gestion de buffer mesurent la charge d'un noeud à partir de l'occupation du buffer. Les modules de gestion de buffer utilisent deux places complémentaires pour modéliser l'occupation du buffer :

- *espace buffer disponible* qui modélise le nombre de places buffer libres ;
- *paquets mémorisés* qui modélise le nombre de places déjà consommées par les paquets.

Les places *paquets mémorisés* et *espace buffer disponible* sont respectivement initialisées à 0 et à la taille de buffer TB. A chaque fois qu'un paquet est accepté pour être mémorisé dans le buffer, le marquage de la première est incrémenté alors que celui de la deuxième est décrémenté. En revanche, à chaque fin de transmission de paquet, c'est l'opération inverse qui est effectuée. Nous verrons plus loin que ces opérations sont effectuées par le module de transmission qui partage ces places avec le module de gestion de buffer.

Nous avons modélisé l'ensemble des algorithmes de gestion de buffer que nous avons énuméré ci-dessus. Faute de place, nous ne décrirons que le modèle du plus simple algorithme, à savoir le Drop Tail.

Le modèle du *drop tail* vis-à-vis d'un flux f est présenté à la figure 4. La sémantique de chaque place du modèle a déjà été présentée ci-dessus. Un paquet qui arrive à l'interface de sortie est mémorisé dans le buffer seulement s'il y a de l'espace disponible dans le buffer. Si c'est le cas, la transition *mémoriser paquet de* f est tirée, sinon la transition *éliminer paquet de* f est tirée.

Le modèle complet d'un drop tail est construit en dupliquant la partie supérieure (celle relative à f) du modèle de la figure 4 $|F|$ fois. Toutes ces parties se partagent la place *espace buffer disponible* et la place *paquets mémorisés*. Comme expliqué dans la section III-A (point 3), le conflit entre les transitions *mémoriser paquet de* f_i des différents flux de F est résolu par un choix équiprobable (par défaut, ces transitions ont un poids de tir égal à 1).

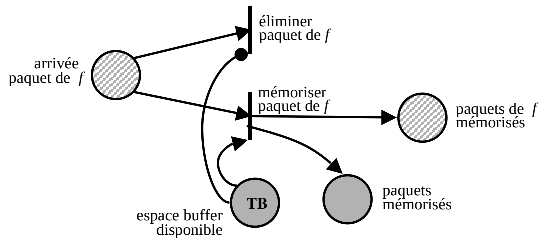

Fig. 4. Modèle RdPS du Drop Tail

D. Modèles RdPS des mécanismes d'ordonnancement de paquets

Le module d'ordonnancement possède une place port d'entrée et une place port de sortie par flux. Pour un flux donné f, la première s'appelle *paquets de f mémorisés* et modélise les paquets de f qui sont mémorisés dans le buffer. La place port de sortie s'appelle *paquets de f à transmettre* et modélise le fait qu'un paquet de f a été choisi pour être transmis.

A chaque fin de transmission d'un paquet, l'ordonnanceur doit choisir le prochain paquet à transmettre. Une place appelée *ligne libre* modélisant l'état de la ligne de sortie signale la fin de transmission des paquets. Le module d'ordonnancement de paquet partage cette place avec le module de transmission qui se charge de la marquer à chaque fin d'une transmission d'un paquet.

Nous avons modélisé l'ensemble des algorithmes d'ordonnancement que nous avons énumérés dans la section II. Faute de place, nous ne décrirons que le plus simple modèle, à savoir l'ordonnancement à priorités statiques.

Le modèle décrivant le comportement d'un ordonnanceur à priorités statiques vis-à-vis d'un flux f est présenté à la figure 5. Les paquets de f interagissent différemment avec trois groupes de paquets :

• les paquets des flux qui ont une priorité supérieure aux paquets de f. Ces paquets sont appelés paquets H.P et sont modélisés par des jetons dans la place *paquets H.P mémorisés.* Ce regroupment de flux H.P est possible puisque nous nous focalisons, pour le moment, sur le comportement vis-à-vis de f. Evidemment, le modèle décrivant le comportement de l'ordonnancement à priorités statiques vis-à-vis de tous les flux de F nécessiterait de faire apparaître une place par niveau de priorité ;

• Les paquets qui appartiennent à des flux de Même Priorité que f. Ces paquets sont appelés paquets M.P et sont modélisés par des jetons dans la place *paquets M.P mémorisés* ;

• Les paquets qui appartiennent à des flux ayant une priorité inférieure à f. Ces paquets sont appelés paquets F.P et sont modélisés par des jetons dans la place *paquets F.P mémorisés.* Même s'ils ont une priorité plus faible que les paquets de f, les paquets F.P peuvent affecter les performances des paquets de f à cause du caractère non-préemptif des algorithmes d'ordonnancement de paquets. Cela arrive typiquement lorsqu'un paquet de f arrive durant la transmission d'un paquet F.P(la ligne n'est pas libre à ce moment là). Même s'il a une priorité supérieure, le paquet de f doit attendre la fin de transmission du paquet F.P. C'est pour prendre en compte ce type d'interaction que le groupe de paquets F.P doit être pris en compte dans la modélisation.

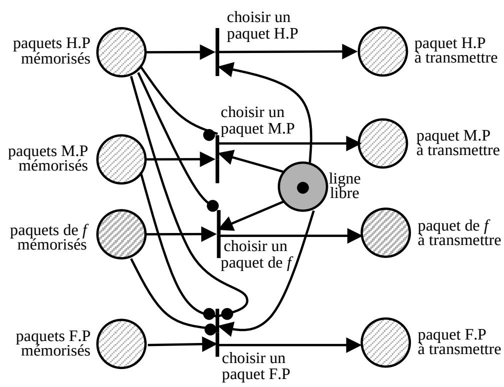

Fig. 5. Modèle RdPS d'un ordonnanceur à priorités statiques

Nous utilisons des arcs inhibiteurs pour modéliser la priorité entre les groupes de paquets. La transmission des paquets de flux de même priorité est réglée par un ordonnancement PAPS. Le modèle PAPS que nous proposons est une approximation du comportement réel de PAPS qui a été proposée dans [21] et qui consiste à choisir le paquet à transmettre aléatoirement parmi tous les paquets mémorisés dans le buffer. Ce choix aléatoire est modélisé en affectant aux transitions en conflit *choisir un paquet de* f et *choisir un paquet* $M.P$ un poids égal au marquage de leur place d'entrée (se référer au point 3 de la section III-A). Ainsi, la probabilité que la transition *choisir un paquet de* f soit tirée est égal à

$$P(\textit{choisir un paquet de } f) = \frac{\text{Marquage}(\textit{paquets de } f \textit{ mémorisés})}{\text{Marquage}(\textit{paquets } M.P \textit{ mémorisés}) + \text{Marquage}(\textit{paquets de } f \textit{ mémorisés})}$$

E. Modèle RdPS du mécanisme de transmission

Le mécanisme de transmission est chargé de transmettre les paquets qui ont été choisis par le mécanisme d'ordonnancement de paquets. Les places port d'entrée et port de sortie, associées à f, du module de transmission sont respectivement appelées *paquet de* f *en cours de transmission* et *paquet de* f *transmis*. Le modèle des opérations du mécanisme vis-à-vis des paquets de f est présenté à la figure 6. La transmission d'un paquet est modélisée par la transition déterministe *transmettre paquet de* f dont le délai est égal à la durée de transmission d'un paquet de f sur la ligne de sortie, i.e. l'unité de temps.

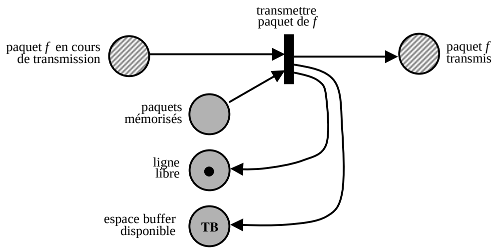

Fig. 6. Modèle RdPS du module de transmission

A chaque fin de transmission, le module de transmission met à jour l'occupation du buffer, respectivement, en incrémentant et décrémentant le marquage des places *espace buffer disponible* et *paquets mémorisés*, qu'il partage avec le module de gestion de buffer. Il marque également la place *ligne libre* qu'il partage avec le module d'ordonnancement de paquets.

F. Modèles RdPS des sources de trafic

Les sources de trafic permettent de décrire le processus de génération des paquets d'un flux et sont donc indispensables pour l'évaluation de performances. Le module d'une source de trafic possède une place port de sortie appelée *paquet de* f *généré* et qui modélise une génération d'un paquet de f.

Dans ce cadre, nous avons modélisé les sources suivantes : la source de poisson, la source périodique, la source de Bernoulli, les sources On/Off, et les sources modulées par processus markovien. Faute de place, nous ne présenterons, dans la suite, qu'un seul modèle, à savoir la source de Bernoulli.

La source de Bernoulli est la version temporelle discrète de la source de Poisson. Le temps est découpé en Tranches de Temps (TT) de durée fixe. A chaque fin de TT, la source génère un paquet avec une probabilité P et ce indépendamment des arrivées dans les tranches précédentes.

Le modèle RdPS d'une source de Bernoulli est présenté à la figure 7. La durée d'une tranche de temps est modélisée par la transition déterministe *tranche de temps*. Chaque fois qu'elle est tirée, un choix probabiliste décide si un paquet a été généré durant la TT qui vient de s'écouler, ou pas. Ce choix est modélisé par deux transitions immédiates en conflit, *générer paquet de* f et *ne pas générer paquet de* f qui ont respectivement un poids de tir de P et $1 - P$. Ces derniers assurent qu'un paquet de f est, à chaque fin de tranche de temps, généré avec une probabilité de P.

G. Modèle du réseau

Dans cette section, nous montrons comment interconnecter les modules que nous venons de présenter, pour construire le modèle RdPS d'un réseau. Nous procédons étape par étape. Chaque étape se focalise sur un

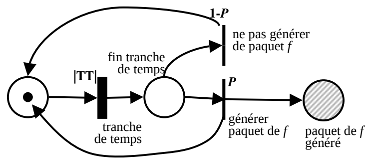

Fig. 7. Modèles de la source de Bernoulli

flux et établit les connexions entre les modules des mécanismes qu'il traverse. Lorsque tous les flux ont été considérés, le modèle du réseau est construit. Dans la suite, nous décrivons une étape, celle relative au flux f.

Rappelons que (1) chaque module comprend deux places particulières par flux f, une qui modélise les paquets de f qui arrivent au mécanisme qu'il décrit (place port d'entrée) et l'autre qui modélise les paquets qui quittent le mécanisme (place port de sortie) ; (2) comme le montre la figure 2, il existe un ordre de passage des mécanismes d'un même noeud. En effet, un paquet qui arrive à un noeud doit commencer par traverser un mécanisme de mise en vigueur. Ensuite, il est soumis au mécanisme de gestion de buffer qui décide, selon la charge du noeud, de l'accepter ou de l'éliminer. Une fois mémorisé dans le buffer, c'est le mécanisme d'ordonnancement qui planifie son instant de début de transmission. Quand ce moment arrive, le mécanisme de transmission se charge de le transmettre au noeud suivant.

Les modules des mécanismes que traverse f dans un noeud peuvent être inter-connectés en fusionnant la place port de sortie associée à f de chaque module du noeud avec la place port d'entrée associée à f du module suivant dans l'ordre explicité au paragraphe précédent. Rappelons que le module de transmission partage les places *espace buffer disponible* et *paquets mémorisés* avec le module gestion de buffer, et la place *ligne libre* avec le module d'ordonnancement de paquet.

Similairement, puisque les délais de propagation sont supposés nuls, il est facile de voir que les places port d'entrée/sortie de deux noeuds consécutifs dans la route de f doivent également être fusionnées.

V. Calcul des métriques de QdS

Nous présentons dans ce qui suit comment exploiter les modèles RdPS présentés dans la section précédente pour calculer des métriques de QdS. Ces métriques sont décomposées en deux groupes :

- **métriques de QdS associées à un flux :** Elles incluent le taux de perte de paquets, le délai de transfert moyen des paquets, des bornes inférieures et supérieures sur le délai de transfert des paquets, une borne supérieure sur la gigue, et le débit d'un flux. Ces métriques permettent d'évaluer le niveau de service que reçoit un flux (e.g. une application) du réseau. D'ailleurs, les paramètres classiques de QdS qu'un réseau peut s'engager à fournir à un flux, peuvent être vérifiés ou estimés par l'une des métriques citées ci-dessus ;
- **métriques de QdS associées au réseau :** Ces métriques sont associées à un noeud du réseau ou au réseau en entier. Il s'agit de métriques telles que le délai moyen de transfert et le taux de perte de paquets, tous flux inclus, dans une interface de sortie. Il s'agit, aussi, de l'utilisation de buffer et de l'utilisation de la bande passante dans une interface de sortie ou dans le réseau en entier. Ces métriques sont typiquement utilisées pour estimer la charge d'un noeud ou du réseau, et évaluer l'efficacité avec laquelle les ressources réseau (buffer + bande passante) sont utilisées.

Dans la suite, nous donnons un aperçu sur la méthode de calcul des métriques deQdS que nous venons de citer.

A. Métriques de QdS associées à un flux

Considérons un flux f et supposons qu'il traverse F_f noeuds numérotés de 1 à F_f.

A.1 Métriques en relation avec la perte de paquets de f :

Sous les hypothèses de la section IV-A, il existe deux sources de perte de paquets dans le réseau : les mécanismes de mise en vigueur du trafic *de type E* et les mécanismes de gestion de buffer. Dans la suite, nous montrons comment calculer le taux de perte de paquets de f au niveau d'un régulateur r_i *de type E* et d'un mécanisme de gestion de buffer employés dans un noeud i. Nous expliquons, ensuite, comment déduire le taux de perte de bout-en-bout.

Le taux de perte de paquets dans un élément du réseau est calculé comme le quotient du nombre de paquets de f qui ont été éliminés dans l'élément considéré, par le nombre de paquets qui sont arrivés à l'élément durant toute la durée de vie de f. Dans la plupart des calculs, nous exploiterons le fait que le nombre de paquets qui sont arrivés à l'élément du réseau, est égal à la somme des paquets qui ont été acceptés et des paquets qui ont été éliminés. Enfin, l'acceptation et l'élimination de paquet étant modélisées par des transitions, les taux de perte de paquets sont exprimés en fonction de la fréquence moyenne de tir de ces transitions.

Taux de perte de paquets de f au niveau du régulateur r_i : TPP_REG(f, r_i)

Dans un régulateur r_i une élimination et une acceptation de paquets dans r_i sont respectivement modélisées par les transitions *éliminer paquet de f, en r_i* et *accepter paquet de f, en r_i*. Ainsi, l'expression de TPP_REG(f, r_i) est

$$TPP\_REG(f,r_i) = \frac{R(\textit{éliminer paquet de } f, \textit{ en } r_i)}{R(\textit{éliminer paquet de } f, \textit{ en } r_i)+R(\textit{accepter paquet de } f, \textit{ en } r_i)} \quad (2)$$

Taux de perte de paquets de f au niveau du mécanisme de gestion de buffer du noeud i : TPP_GB(f,i)

Contrairement aux modèles des régulateurs de trafic, plusieurs transitions peuvent modéliser une élimination de paquet dans les modèles de gestion de buffer. Nous notons par $R_e(f,i)$ la somme des fréquences de tir des transitions qui éliminent les paquets, et par $R_m(f,i)$ la somme des fréquences de tir des transitions qui mémorisent les paquets dans le buffer. Les transitions qui modélisent les éliminations et les mémorisations de paquets varient d'un algorithme de gestion de buffer à un autre. Par conséquent, nous donnons une expression générique de TPP_GB(f,i) en fonction de $R_e(f,i)$ et $R_m(f,i)$. Ces derniers seront, ensuite, exprimés pour chaque algorithme.

$$TPP\_GB(f,i) = \frac{R_e(f,i)}{R_e(f,i)+R_m(f,i)} \quad (3)$$

Pour l'algorithme Drop Tail qui est le seul algorithme qui a été présenté dans ce papier,

$$R_e(f,i) = R(\textit{éliminer paquet de } f, \textit{ en } i) \quad (4)$$
$$R_m(f,i) = R(\textit{mémoriser paquet de } f, \textit{ en } i) \quad (5)$$

Taux de perte de paquets de f au niveau du noeud i : TPP(f,i)

Deux cas sont à considérer. Le cas où f ne traverse pas de régulateur de type E dans le noeud i, et le cas où il en traverse (ce régulateur est alors appelé r_i). Dans le premier cas, TPP(f,i) se confond avec TPP_GB(f,i). Dans le second cas, TPP(f,i) peut être

- soit calculé directement :

$$TPP(f,i) = \frac{R(\textit{éliminer paquets de } f, \textit{en } r_i) + R_e(f,i)}{R(\textit{transmettre paquet de } f \textit{ en } i-1)} \quad (6)$$

où *transmettre paquet de f en $i-1$* modélise la transmission d'un paquet de f dans le noeud $i-1$. Son tir implique une arrivée au noeud i ;

- soit, déduit de TPP_GB(f,i) et TPP_REG(f,r_i) comme suit :

$$TPP(f,i) = TPP\_REG(f,r_i) + (1 - TPP\_REG(f,r_i)).TPP\_GB(f,i) \quad (7)$$

Ce résultat peut être déduit de l'équation 6 après quelques développements algébriques, en remarquant que :

$$TPP\_REG(f,r_i) = \frac{R(\textit{éliminer paquet de } f, \textit{ en } r_i)}{R(\textit{transmettre un paquet de } f, \textit{ en } i-1)}$$

$$\begin{aligned} TPP\_GB(f,i) &= \frac{R_e(f,i)}{R(\textit{accepter paquet de } f, \textit{ en } r_i)} \\ &= \frac{R_e(f,i)}{(1 - TPP\_REG(f,r_i)) * R(\textit{transmettre un paquet de } f, \textit{ en } i-1)} \end{aligned}$$

Taux de perte de paquets de f de bout-en-bout : TPP(f)
Le taux de perte des paquets de f de bout-en-bout est égal à la somme des fréquences moyennes de tirs de toutes les transitions, qui modélisent une élimination d'un paquet de f dans les différents noeuds traversés par f, sur la fréquence moyenne de tir de la transition qui modélise le départ d'un paquet de f de son site source.

$$TPP(f) = \frac{\sum_{i=1}^{F_f} (R(\textit{éliminer un paquet de } f, \textit{ en } r_i) + R_e(f,i))}{R(\textit{transmettre un paquet de la source de } f)} \tag{8}$$

A.2 Débit du flux f : Deb(f)

Le débit moyen de f, mesuré à la destination, est une métrique de QdS très importante. Il peut facilement être déduit du modèle RdPS du réseau.

$$Deb(f) = R(\textit{transmettre un paquet de } f \textit{ au noeud } F_f) \tag{9}$$

A.3 Métriques en relation avec le délai de transfert des paquets de f

Sous les hypothèses de délais de propagation et de commutation nuls, le délai que subit un paquet dans le réseau est la somme des délais d'attente dans les régulateurs *de type R* et dans les buffers, et des temps de transmission dans chaque noeud. Plusieurs métriques relatives au délai de transfert des paquets de f peuvent être utiles pour l'analyse de QdS : le délai moyen de transfert au niveau d'un noeud ou de bout-en-bout moyen, une borne supérieure ou inférieure sur le délai de bout-en-bout ou dans un noeud, ou une borne supérieure sur la gigue. Faute de place, nous nous focaliserons dans ce papier sur le calcul des délais moyens ; le lecteur intéressé peut se référer à [1] pour le calcul des bornes sur le délai de transfert et la gigue.

Les délais moyens de transfert sont calculés à partir de **la loi de Little**. Le délai moyen d'attente des paquets de f dans le régulateur *type R* du noeud i (noté r_i) est obtenu en appliquant la Loi de LITTLE au modèle RdPS constitué de la place *paquets de f à réguler*.

$$\overline{TD}(f, r_i) = \frac{E[\textit{paquets de } f \textit{ à réguler}]}{\textit{accepter paquet de } f} \tag{10}$$

Similairement, le délai moyen de transfert des paquets de f dans l'interface de sortie IS_i de i ($\overline{TD}(f, IS_i)$) est obtenu en appliquant la Loi de LITTLE au module RdPS composé des places *paquets de f mémorisés dans i* et *paquet de f en cours de transmission, en i*. En effet,

$$\overline{TD}(f, IS_i) = \frac{E[\textit{paquets de } f \textit{ mémorisés dans } i] + E[\textit{paquet de } f \textit{ en transmission, en } i]}{R(\textit{transmettre paquet de } f, \textit{ en } i)} \tag{11}$$

Le délai moyen de transfert des paquets de f en i, noté $\overline{TD}(f, i)$ est égal à

$$\overline{TD}(f,i) \quad = \quad \overline{TD}(f, r_i) + \overline{TD}(f, IS_i) \tag{12}$$

et le délai de bout-en-bout, $\overline{TD}(f)$ est égal à

$$\overline{TD}(f) = \sum_{i=1}^{F_f} \overline{TD}(f,i) \tag{13}$$

B. Métriques de QdS associées au réseau

Nous nous intéressons aux métriques de QdS qui sont associées à une interface de sortie d'un noeud, ou au réseau en entier. Nous notons par N le réseau à étudier. Notons respectivement par $O(N)$ et $|O(N)|$ l'ensemble des interfaces de sortie de N et le cardinal de cet ensemble. Soit i une interface de sortie du réseau ($i \in O(N)$). Notons par l_i et B_i, respectivement la ligne et le buffer (de taille TB_i) associés à l'interface i. Enfin, notons par FL_i l'ensemble de flux qui traversent l'interface i.

B.1 Délai moyen de transfert au niveau d'une interface de sortie i : $\overline{TD}(i)$

Le délai moyen de transfert en régime stationnaire au niveau de l'interface i, est obtenu en appliquant le théorème de Little au module RdPS, composé du module d'ordonnancement et de transmission du noeud i.

$$\overline{TD}(i) = \frac{\sum_{f \in FL_i} (E[\textit{paquets de } f \textit{ mémorisés, en } i] + E[\textit{paquet de } f \textit{ en transmission, en } i])}{\sum_{f \in FL_i} R(\textit{transmettre un paquet de } f\textit{, en } i)} \tag{14}$$

B.2 Taux de perte de paquets à l'interface de sortie i : PLR(i)

PLR(i) mesure la portion de paquets qui est perdue au niveau d'un noeud i. Cette métrique est également obtenue en divisant la somme de la fréquence de tir des transitions qui éliminent un paquet en i par la somme des transitions qui modélisent une arrivée de paquet en i.

B.3 Utilisation de la bande passante de l_i : $UBp[l_i]$

L'utilisation de la bande passante d'une ligne mesure la portion de la bande passante qui est utilisée en moyenne. L'utilisation de la bande passante de la ligne l_i est égale à :

$$UBp[l_i] \quad = \quad 1 - E[\textit{ligne } l_i \textit{ libre}] \tag{15}$$

B.4 Utilisation du buffer B_i : $UBu[B_i]$

L'utilisation du buffer mesure la portion de buffer qui est utilisée en moyenne. L'utilisation du buffer B_i est égale à :

$$UBu[B_i] \quad = \quad \frac{TB_i - E[\textit{espace buffer disponible en } B_i]}{TB_i} \tag{16}$$

Il est possible d'affiner $UBu[B_i]$ et calculer la portion du buffer B_i qui est utilisée, en moyenne, par un flux f ($UBu[B_i](f)$). En effet, à chaque flux f qui traverse i correspond une place appelée *paquets de f mémorisés, en i* et qui modélise les paquets de f en attente de service en i. $UBu[B_i](f)$ est égale à :

$$UBu[B_i](f) \quad = \quad \frac{\mathrm{E}[\textit{paquets de } f \textit{ mémorisés, en } i]}{TB_i} \tag{17}$$

Il est également possible de calculer l'occupation maximale de B_i ainsi que l'occupation maximale d'un flux f en B_i. Ces métriques sont respectivement égales à TB_i- marquage minimal de la place *espace buffer disponible, en i* et marquage maximal de la place *paquets de f mémorisés, en i*.

B.5 Utilisation de la bande passante du réseau N : $UBp[N]$

L'utilisation de la bande passante de N est calculée en effectuant la moyenne sur l'utilisation de la bande passante de toutes ses lignes.

$$UBp[N] \quad = \quad \frac{\sum_{i \in O(N)} UBp[l_i]}{\mid O(N) \mid} \tag{18}$$

B.6 Utilisation de buffer dans le réseau N : $UBu[N]$

L'utilisation de buffer dans le réseau N est calculée en divisant la somme de l'occupation moyenne des buffers au niveau de chaque interface de sortie du réseau par la taille totale de buffer de N, i.e.

$$UBu[N] \quad = \quad \frac{\sum_{i \in O(N)} (TB_i - \mathrm{E}[\textit{espace buffer disponible, en } i])}{\sum_{i \in O(N)} TB_i} \tag{19}$$

VI. Application à un exemple académique

Considérons le flux f de la figure 8 qui traverse un commutateur ATM appelé *Cm*. Le commutateur emploie un régulateur (σ_f, ρ_f) *de type E* pour mettre en vigueur le trafic que soumet f au réseau. L'interface de sortie que traverse f est appelée *Sw*. Elle emploie un drop tail pour contrôler l'accès à un buffer de taille TB_{Sw}, et un ordonnancement à priorités statiques pour contrôler l'accès à la ligne de sortie de *Sw*, notée l_{Sw}, qui a un débit de C cellules/UT. f partage *Sw* avec $f_1, \ldots f_4$. f_1 est prioritaire sur f qui est à son tour prioritaire sur f_2, f_3 et f_4. Le trafic de f qui arrive sur le commutateur *Cm* est un modèle de Bernoulli de paramètres $|TT|_f$ et P_f. L'objectif de la modélisation est de mesurer les paramètres de QdS classiques associés à la connexion f. Nous présentons dans ce qui suit la modélisation du réseau vue par f (notre étude se focalise sur la connexion f).

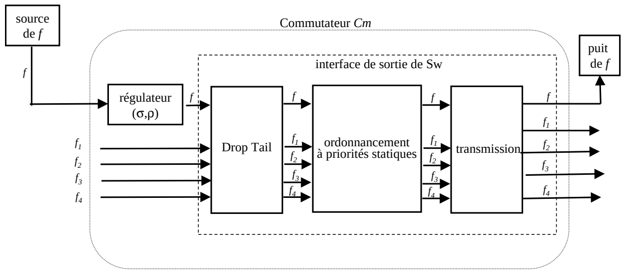

Fig. 8. Exemple étudié

Un aspect crucial qui doit être pris en compte lors de la conception d'un modèle RdPS est le problème de faisabilité (*tractability*). Deux points doivent être abordés :

• Comment modéliser le trafic des flux $f_1 \ldots f_4$ qui arrivent à Sw? Si nous voulons être très précis, nous sommes amenés à modéliser tous les mécanismes que chacun de ces flux traverse depuis leur source jusqu'à *Sw*. Bien évidemment, une telle approche engendre des modèles RdPS non faisables. L'autre alternative est de modéliser le trafic de ces flux à l'entrée de *Sw*. Cette modélisation n'est pas évidente, elle doit prendre en compte le modèle de leur source de trafic et l'effet des noeuds que les flux traversent. Nous adoptons la deuxième approche pour construire le modèle RdPS du réseau. Toutefois, nous omettons volontairement la discussion relative au choix d'une caractérisation du trafic des flux $f_1, \ldots f_4$ à l'entrée de *Sw* et supposons qu'ils sont tous conformes à un modèle de Bernoulli de paramètres $|TT|$ et P.

• comment agréger les flux ensemble? Le grand avantage de l'agrégation de flux est de réduire significativement la taille et la complexité du modèle RdPS, ce qui a un impact direct sur sa faisabilité. Malheureusement, l'agrégation de flux n'est pas toujours possible puisque les flux peuvent avoir des caractéristiques incompatibles ou ne pas avoir la même destination. Plus encore, le comportement de certains mécanismes vis-à-vis d'un agrégat de flux peut ne pas coïncider avec l'agrégat des comportements de chaque flux considéré comme une entité à part entière. L'agrégation de flux est une technique très puissante pour améliorer la faisabilité des modèles du réseau. Malheureusement, la définition d'agrégats n'est pas une tâche facile, elle peut dépendre de plusieurs paramètres et en particulier de l'objectif de la modélisation. Cette tâche est laissée à la charge du concepteur qui définit les agrégats qui correspondent à ses besoins.

Commençons par déterminer des agrégations de flux pour l'exemple de la figure 8. Puisque l'objectif de notre modélisation est de mesurer les métriques de QdS de f, nous pouvant déduire, premièrement, que f ne doit être regroupé avec aucun autre flux et, deuxièmement, que le rôle des flux $f_1 \ldots f_4$ est de partager les ressources de *Sw* avec f. Comme conséquence, il est tentant de vouloir agréger ces flux ensemble. Malheureusement, cela n'est pas possible puisque f_1 est plus prioritaire que f alors que $f_2 \ldots f_4$ sont moins prioritaires que f. En revanche, il est possible de regrouper les flux f_2,f_3, et f_4 puisque (1) ils sont tous, comparés à f, de moindre priorité ; et (2) il est possible de caractériser puis modéliser l'agrégat de trois sources de Bernoulli. Le flux agrégé est appelé f_{2-4}.

Maintenant que les agrégats de flux sont définis, il est possible de construire les modules RdPS du drop tail et l'ordonnancement à priorités statiques de l'exemple considéré. Le modèle complet du réseau tel que vu par f est obtenu en connectant ces modules, comme expliqué dans la section IV-G. Il est présenté à la figure 9.

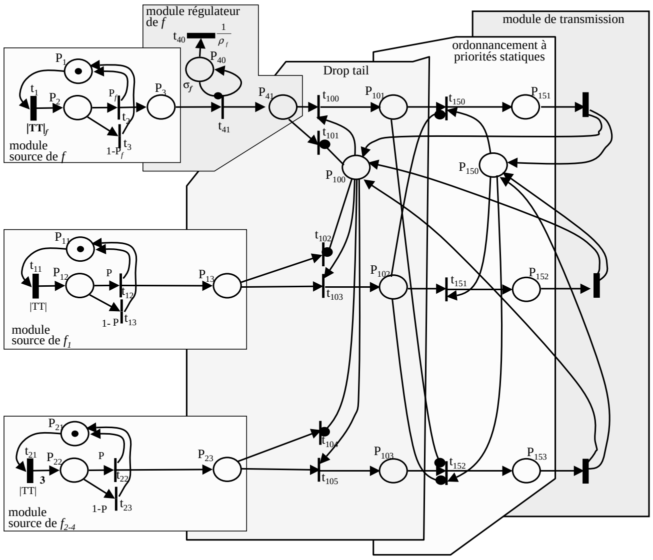

Fig. 9. Modèle RdPS du réseau, vu par f

Nous présentons dans le tableau I quelques mesures que nous avons effectuées à partir du modèle du réseau de la figure 9. Les paramètres de la source de f, $|TT|_f$ et P_f, sont respectivement égaux à $1UT$ et 0.25. Les paramètres du σ_f et ρ_f du régulateur sont égaux à 3 cellules et 0.25 cellules/UT. La taille du buffer de l'interface de sortie Sw est fixée à 15 cellules. Le paramètre $|TT|$ des sources des flux f_1, f_2, f_3 et f_4 est fixé à 1 alors que la probabilité P varie de 0.05 à 0.25 ce qui permet de faire varier la charge de Cm. Les métriques de QdS présentées ont toutes été calculées à partir des expressions de la section précédente. La fréquence moyenne de tir des transitions et le marquage moyen des places ont été calculés en utilisant l'outil associé au modèle Réseau de Petri Temporisés Stochastiques (RdPTS), développé au LAAS-CNRS.

P	0.05	0.1	0.15	0.2	0.25
$TPP\_REG(f,Cm)$	0.134	0.134	0.134	0.134	0.134
$TPP\_GB(f,Cm)$	0	0	$3.\ 10^{-5}$	0.027	0.161
$TPP(f,Cm)$	0.134	0.134	0.1344	0.15	0.273
$TPP(Sw)$	0	0	$4.4\ 10^{-5}$	0.0321	0.152
$\overline{TD}(f,Cm)$ en UT	1.18	1.394	1.756	2.2	2.23
$\overline{TD}(Sw)$ en UT	1.22	1.575	2.59	8.65	10.4
Throu(f) en cellules/UT	0.2164	0.2164	0.2164	0.2105	0.182
$UBu[B_{Sw}]$	3.6%	6.7%	14.7%	58.5 %	91%
$UBp[l_{Sw}]$	41.6%	61.6%	82%	98.4%	99.9%
$UBu[B_{Sw}](f)$	0.2%	0.6 %	1.1%	1.7%	1.5%

TABLE I

VII. Sur des applications du cadre défini

Le cadre que nous avons présenté possède un large éventail d'applications. Il peut, par exemple, être utilisé pour l'étude de performances de mécanismes de fourniture de QdS ou pour évaluer le niveau de service reçu par une application distribuée.

Jusqu'au moment de l'écriture de ce papier, le cadre a été appliqué à deux études de performances. La première concerne l'étude des performances en régime stationnaire de deux mises en oeuvre de la famille Assured Forwarding (AF) de l'architecture Differentiated Services. Ces mises en oeuvre se basent sur l'algorithme de marquage de paquets proposé par le RFC2698 de l'IETF, sur un ordonnancement à priorités statiques (à chaque classe de trafic AF est associé un indice de priorité), et un algorithme de gestion de buffer dit à seuils par classe, i.e. qui prend les décisions d'acceptation et d'élimination de paquets par simple comparaison à des seuils pré-définis. L'étude a permis de vérifier que les mises en oeuvre que nous proposons vérifient les recommandations de l'IETF sur la famille AF.

La deuxième concerne l'étude d'une relation producteur/consommateur périodique distribuée au travers d'un commutateur ATM. Plus précisément, nous avons montré comment exprimer et mesurer les paramètres de QdS en régime stationnaire du service qu'offre le réseau à la relation producteur/consommateur (e.g. délai moyen de transfert et taux de perte de cellules), et du service qu'offre la relation Producteur/Consommateur aux applications de plus haut niveau (e.g. fréquences moyennes de consommation, de lecture de mémoire vide, et d'écrasement). De telles mesures permettent de clarifier et d'entrer dans le détail des relations complexes qui existent entre les mécanismes utilisés par le réseau (e.g. concentrateur, gestion de buffer et ordonnancement), l'architecture adoptée par le consommateur (guidée par le temps ou les événements) et le service reçu par les applications de haut niveau ; elles peuvent donc aider à la conception d'applications distribuées (dans un réseau ATM) qui utilisent la relation producteur/consommateur comme modèle de coopération.

En vue de nos premières expérimentations, nous avons été confrontés à un problème d'explosion combinatoire du graphe d'états généré à partir du modèle RdPS du réseau et à partir duquel la fréquence moyenne de tir des transitions et le marquage moyen des places sont calculés. En effet, dès que le nombre de noeuds atteint 2 et le nombre de flux atteint la dizaine, le graphe d'états généré possède une taille de quelque centaines de milliers d'états ce qui le rend non analysable à l'aide de machines conventionnelles. En d'autres termes, l'applicabilité du cadre est limitée à des réseaux avec une configuration simple.

VIII. Conclusion

Nous avons présenté un cadre basé sur les réseaux de Petri stochastiques pour l'analyse de la qualité de service dans les réseaux à commutation de paquets. Dans ce cadre, nous avons premièrement décrit, à l'aide de modèles RdPS, les principaux algorithmes qui sont implémentés en pratique par les mécanismes de mise en forme de trafic, de mise en vigueur de trafic, de gestion de buffer et d'ordonnancement de paquets. Ensuite, nous avons montré comment exploiter ces modèles pour conduire des mesures de performances permettant d'évaluer les métriques classiques de QdS, e.g. taux de perte de paquets, délai moyen de transfert de paquets, utilisation de la bande passante du réseau, etc.

Le cadre que nous proposons peut être utilisé à l'aide de plusieurs modèles formels, définis dans la littérature : RdPTS, DSPN, et GSPN. Cela a été possible en adoptant un modèle RdPS avec un pouvoir d'expression classique et en exprimant toutes les mesures de performances à partir de la fréquence moyenne de tir des transitions et le marquage moyen des places. Ces derniers peuvent être calculés par tous les outils d'analyse associés aux modèles cités ci-dessus.

Ce cadre présente, par ailleurs, la particularité d'être ouvert dans le sens où les modèles RdPS d'autres algorithmes peuvent être introduits et venir l'enrichir. Il faut juste respecter certaines règles dictées par la méthode de modélisation.

Comme nous l'avons signalé dans la section précédente, ce cadre fait apparaître des limites quant à son applicabilité à des réseaux avec un nombre élevé de noeuds. La principale perspective à ce travail est de rechercher un moyen pour contourner le problème d'explosion combinatoire auquel nous sommes confrontés. Pour ce faire, nous nous proposons d'approcher les métriques de QdS à partir d'une analyse structurelle des modèles RdPS évitant, par ce fait, la génération du graphe d'états. Cette approche se base sur les travaux suivants [22] [23] [24] qui calculent des bornes supérieures et inférieures sur la fréquence moyenne de tir d'une transition et le marquage moyen d'une place en résolvant un problème de programmation linéaire déduit de la structure d'un réseau de Petri. Puisque toutes nos métriques de QdS sont déduites simplement à partir des deux indices précédents, des bornes inférieures et supérieures peuvent être calculées. Le point à étudier est la précision des bornes obtenues.

Références

[1] S. Abdellatif, *Contributions à la modélisation et à l'analyse de la Qualité de servie dans les réseaux à commutation de paquets*, Ph.D. thesis, université Toulouse III, janvier 2002.

[2] Rene L. Cruz, "A calculus for network delay, part ii: network analysis," *IEEE transactions on information theory*, vol. 37, no. 1, 1991.

[3] E.W. Knightly, *Traffic models and admission control for integrated services networks*, Ph.D. thesis, University of California - Berkeley, May 1996.

[4] S. Floyd et A. Romanov, "Dynamics of tcp traffic over atm networks," *IEEE journal on selected areas in communications*, vol. 13, no. 4, pp. 633–641, 1995.

[5] S. Floyd et V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM transactions on networking*, vol. 1, no. 4, pp. 397–413, 1993.

[6] S. Keshav et S. Shenkar A. Demers, "Analysis and simulation of a fair queueing algorithme," *Internet Research and Experiments*, vol. 1, 1990.

[7] A.K. Parekh et R.G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single node case," *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, june 1993.

[8] M. Shreedhar et George Varghese, "Efficient fair queuing using deficit round robin," *IEEE/ACM transactions on networking*, vol. 4, no. 3, june 1996.

[9] H.Zhang et D.Ferrari, "Rate-controlled static-priority queueing," in *IEEE INFOCOM'93*, 1993.

[10] M. Molloy, *On the integration of delay et throughput measures in distributed processing models*, Ph.D. thesis, university of california, 1981.

[11] M. K. Molloy, "Performance analysis using stochastic petri nets," *IEEE trans. on comp.*, vol. C-31, pp. 913–917, 1982.

[12] M.A. Ajmone Marsan et al., *Modelling with Generalized stochastic Petri nets*, John Wiley & sons, 1995.

[13] C. Lindemann, *Performance modelling with deterministic and stocahstic Petri nets*, John Wiley & sons, 1998.

[14] Y. Atamna, *Réseaux de Petri Temporisés Stochastiques Classiques et Bien Formés : Définition, Analyse et Application aux Systèmes Distribués Temps Réel*, Ph.D. thesis, Université Paul Sabatier de Toulouse (FRANCE), 1994.

[15] L. Gallon, *Le Modèle Réseaux de Petri Temporisés Stochastiques:Extensions et Applications*, Ph.D. thesis, Université Paul Sabatier de Toulouse (FRANCE), 1997.

[16] G.Juanole S.Abdellatif L.Gallon et M.Zayenne, "Nouveaux concepts sur le modèle rdps : priorités dynamiques et attributs temporels dynamiques," in *Conférence Internationale Francophone d'Automatique*, juillet 2000.

[17] DSPNexpress, ," http://ls4-www.cs.uni-dortmund.de/~Lindemann/software/DSPNexpress/mainE.html.

[18] http://www.di.unito.it/ greatspn/, "Graphical editor and analyzer for timed and stochastic petri nets," .

[19] R.Y. Awdeh et H.T. Moufteh, "Survey of atm switch architectures," *Computer Networks and ISDN Systems*, 1994.

[20] J. Wang, *Timed Petri nets: theory and application*, Kluwer Academics publishers, 1998.

[21] B.R.Haverkort H.P. Idzenga et B.G. Kim, "Performance evaluation of treshold-based atm cell scheduling policies under markov-modulated poisson traffic using stochastic petri nets," in *Performance Modelling and Evaluation of ATM Networks. 1995*. 1995, Chapman and Hall.

[22] G. Chiola et al., "Operational analysis of timed petri nets and applications to the computation of performance bounds," in *proceedings of the 5th IEEE international workshop on Petri Nets and Performance Models*, Toulouse (FRANCE), october 19-22 1993, pp. 128–137.

[23] Zhen Liu, "Performance analysis of stochastic timed Petri nets using linear programming approach," *IEEE Transactions on Software Engineering*, vol. 24, no. 11, pp. 1014–1030, Nov. 1998.

[24] Javier Campos, *Performance models for discrete event systems with synchronizations: formalisms and analysis techniques*, vol. 1, chapter 17- performance bounds, pp. 587–636, KRONOS, 1998.