Gestion du Trafic

Modèles pour les Réseaux de Télécommunications

Gérard Hébuterne
Département RST

Table des matières

Gestion du Trafic et
Modèles pour les Réseaux de Télécommunications

L'organisation des réseaux de télécommunications vise à donner la maîtrise des phénomènes qui s'y produisent, à l'occasion du traitement des communications, et plus généralement des "services" qu'ils offrent. Ces phénomènes sont gouvernés par le hasard de l'apparition des requêtes, et s'étudient indépendamment des choix de technologie mis en œuvre. Ils sont justiciables du formalisme du calcul des probabilités, et donnent naissance à la notion de *trafic*, qui va jouer un rôle central dans leur appréhension. Ainsi, les phénomènes de trafic conditionnent-ils pour une large part la structure effective des réseaux modernes.

Ce cours présente les éléments de la théorie du trafic, en en montrant l'impact sur les architectures des réseaux et des systèmes, et en introduisant les grands problèmes que doit affronter l'Ingénieur chargé du trafic.

Les notions développées ici trouvent leur application dans les domaines de la *conception* des réseaux, du *dimensionnement* des équipements, de la caractérisation et de la mesure de la *Qualité de Service*, et dans les techniques de la *planification* des réseaux.

Chapitre 1

Présentation du cours

1.1 Panorama général

On étudie ici le réseau indépendamment des solutions techniques adoptées pour son implémentation. Il s'agit donc simplement d'un graphe, reliant des noeuds (commutateurs) et desservant des terminaux. Le réseau a comme fonction de relier les terminaux, de leur permettre des échanges d'information, ce qu'on appelle le *trafic*[1]. Première tâche, définir en quoi consiste le trafic. Afin d'expliciter cette notion, il sera nécessaire d'examiner plus en détail le type de réseau et surtout le service d'interconnexion qu'on envisage.

La Qualité de Service (QS) est une autre notion importante. Il faudra tout d'abord la définir, se donner les moyens de la chiffrer (cela dépend aussi du service). Les processus se déroulant dans le réseau sont de nature aléatoire. Il en résulte que la QS met en jeu des notions de "probabilité" (probabilité de rejet, etc.).

Par conséquent, la définition de la QS et la vérification des contraintes qu'elle fait peser sur le réseau feront appel aux notions du calcul des probabilités, et plus spécialement de la *théorie des files d'attentes*. Cette discipline aura deux fonctions complémentaires: d'une part évaluer le niveau de performance d'un réseau donné; d'autre part, estimer les ressources nécessaires au respect de critères de QS donnés (on parle de *dimensionnement*).

En général, les éléments de la théorie seront insuffisants à traiter entièrement les problèmes de dimensionnement. On aura alors recours à la *simulation*, qu'il faut considérer comme un complément à l'approche mathématique, les deux méthodes se confortant mutuellement. L'usage conjoint de ces deux approches amènera un bon niveau de sécurité dans les prévisions.

Il ne suffit pas d'estimer le niveau de ressources nécessaire; cette estimation s'intègre, en fait, dans une démarche beaucoup plus générale de *planification*. Le responsable du réseau doit en effet commencer par estimer la demande (son volume, notamment) et prévoir son évolution, dimensionner les ressources en conséquence, et organiser leur mise en service. La structure du réseau, et pas seulement sa taille, sont des sujets d'étude pour la planification (les problèmes et les méthodes de planification ne sont pas traités dans ce cours).

1. On écrit *traffic* en anglais

Le cours abordera les points suivants:

- définition du trafic;
- rappels et compléments en files d'attentes; application à la modélisation;
- initiation à la simulation par événements, au travers de modélisations-types.

Les notions développées dans ce cours concernent tous les types de réseaux, qu'ils soient à commutation de circuit ou de paquets, qu'ils soient fixes ou mobiles. En pratique, les exemples seront pris dans chacune des catégories, sans idée préconçue.

Elles sont applicables aussi bien pour de grands réseaux publics (réseaux d'opérateurs, typiquement) que dans des réseaux privés (réseaux locaux, interconnexion de réseaux locaux constituant des réseaux privés virtuels).

1.2 La Qualité de Service

La notion de Qualité de Service veut rendre compte, de façon chiffrée, du niveau de performances que l'utilisateur attend du réseau. Remarque: dans une architecture en couches, l'utilisateur de la couche de niveau n est un processus de couche $n+1$. La notion de QS, son chiffrage, sa vérification sont des tendances modernes – c'est à dire que l'apparition des mécanismes de la "dérégulation" leur donne une importance croissante.

Le contenu de la notion de Qualité de Service (QS) dépend évidemment du *service* envisagé. Ainsi le service a-t-il des exigences de temps de réponse; quelle est sa sensibilité aux erreurs de transmission; etc. Une définition complète se référera souvent au mode de transport de l'information – circuit ou paquet, bien que la solution adoptée par le réseau pour rendre le service doive rester transparente à l'utilisateur.

1.2.1 Réseau à Commutation de Circuit

Le réseau téléphonique est le type même de réseau à commutation de circuits. Une demande acceptée se voit offrir à son usage exclusif un "circuit" (circuit électrique continu, ou intervalle de temps d'une trame MIC) pour toute la durée de la connexion. La QS est définie en premier lieu par la possibilité de ne pas obtenir de circuit lors de la demande. On la mesure par la *probabilité d'échec*.

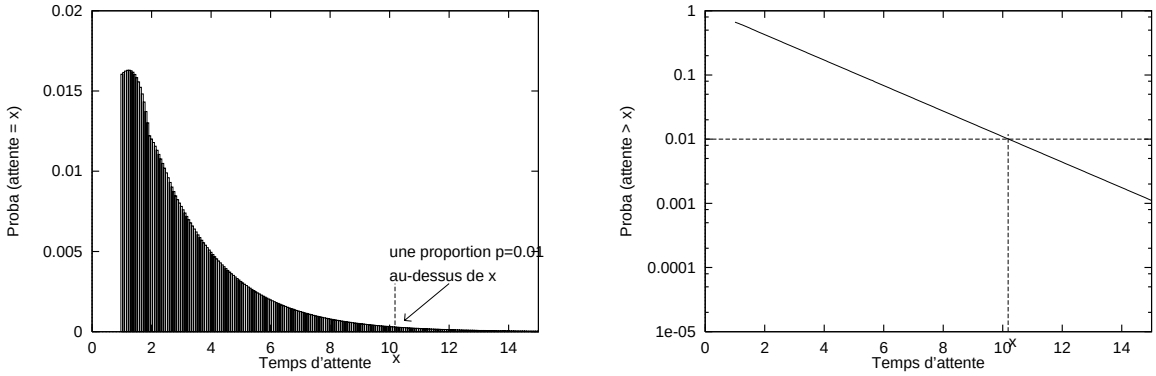
D'autres critères, liés aux technologies, interviendront dans la QS. Ils ne nous intéressent pas ici, ce sont les critères liés à la qualité de la liaison (atténuation du signal, distorsions).

1.2.2 Réseau à Commutation par Paquets

Dans un réseau offrant le service de commutation de paquets, l'information des sources est fragmentée en blocs élémentaires qui voyagent dans le réseau indépendamment les uns des autres. Ces blocs ne possèdent aucune ressource en propre (comme dans le cas du circuit). Les paquets d'une connexion se retrouvent en compétition avec d'autres pour accéder aux mémoires ou aux lignes de transmission.

Il en résulte un critère supplémentaire de Qualité de Service, lié au sort individuel de chaque paquet, qui subira un retard variable, et pourra même être perdu, par exemple si une des mémoires qu'il doit traverser est saturée. On mesurera la QS par la probabilité de perte de paquet et par le délai, qu'on spécifiera par sa moyenne ou par un *quantile*.

Pour introduire cette notion, supposons qu'on ait pu mesurer les temps d'attente des paquets qui traversent un commutateur. On trace l'*histogramme* qui donne les fréquences empiriques d'observation d'un temps compris entre x et $x + \Delta x$ (Figure 1.1.a). Ou, mieux encore, la distribution cumulée empirique (figure 1.1.b) – en fait ici la distribution complémentaire, qui donne la proportion des mesures supérieures à x.



(a) Histogramme des temps (b) Distribution cumulee

FIG. 1.1 – *Observation des temps d'attente*

Une lecture directe du graphique donne le quantile cherché. Dans une campagne de mesure, ou à l'issue d'une simulation, il suffit d'estimer la valeur x dépassée par une proportion p des mesures recueillies. En termes mathématiques, le quantile est défini à partir de la fonction de répartition, ou de la densité. Si on note F et f les fonctions correspondantes pour une variable aléatoire X (le délai du paquet, ici):

$$\begin{aligned} F(x) &= P\{X \leq x\} \\ f(x) &= dF/dx \end{aligned} \tag{1.1}$$

le quantile pour la valeur p (le p-quantile) est la valeur x_p telle que:

$$F(x_p) = 1 - p \tag{1.2}$$

$$\int_{x_p}^{\infty} f(x)\mathrm{d}x = p \tag{1.3}$$

L'importance du quantile provient de la remarque suivante: les "utilisateurs" d'un réseau – plus généralement de tout "serveur" – ne sont pas sensibles à la moyenne, grandeur théorique que personne ne ressent. Ils sont en réalité insensibles aux délais courts, dont ils n'auront pas conscience; à l'inverse ils seront sensibles à des grandes valeurs du délai, qu'ils perçoivent. Par exemple, dans le cas où un paquet subit un retard trop important, une temporisation (qui teste le retour d'un acquittement) va expirer, provoquant sa réémission (l'émetteur interprétant le retard comme une perte). La temporisation joue le rôle d'un seuil,

et seule la probabilité que le retard dépasse ce seuil est importante. Le quantile mesure la valeur du délai que le réseau assure dans $1-p$ des cas.

Ainsi, le temps qui s'écoule entre le décrochage du combiné téléphonique et la réception de la tonalité est de l'ordre de 0.4 secondes. Il devient "intolérable" au delà de 3 secondes (c'est à dire que s'il dépasse cette valeur, on voit apparaître des comportements d'impatience - qui dans ce cas peuvent mettre en cause le bon fonctionnement de l'ensemble). Les étages d'entrée des commutateurs doivent assurer que $t_p \sim 3s$ pour $p = 0.01$, en situation de trafic de surcharge.

D'autres critères définissent la QS. Notamment, les notions liées à la disponibilité du service sont extrèmement importantes. En gros, il s'agira de garantir la présence permanente du service, et l'absence de ruptures intempestives de connexions.

Dans un certain nombre de cas, les notions de réseau à commutation de circuit et à commutation de paquet se superposent. La technique ATM par exemple, commute des paquets (les "cellules") dans un réseau de "circuits virtuels". Les problématiques des deux grandes catégories se trouvent ainsi mêlées dans l'étude de ce réseau.

1.3 La Modélisation

Un réseau, un commutateur, sont des objets physiques complexes qu'il est difficile d'appréhender dans leur totalité. D'autre part, on sent bien l'inutilité de tenir compte de la plupart de leurs caractéristiques pour traiter un problème de trafic. La *modélisation* est l'approche adaptée à ces questions.

Faire un modèle, c'est en partant d'une description complète du système réel, construire une abstraction simplifiée qui conserve les particularités des phénomènes à étudier – et seulement celles-ci. La modélisation est une technique délicate et empirique, un artisanat. Nous essaierons d'en montrer la puissance et l'intérêt, au travers d'exemples-types pris dans la pratique de développement et de l'exploitation des réseaux de télécommunications modernes.

Chapitre 2

Notions élémentaires de Trafic, Modèles de base

Les mécanismes d'occupation des ressources constituent le phénomène fondamentalement responsable de la dégradation possible de QS; en effet, le réseau offre des "ressources" en commun à ses utilisateurs, que ceux-ci doivent se partager. Le terme de trafic rend compte de la quantité de "travail" présente dans un serveur, sous forme d'occupation de ce serveur. Le terme désigne à la fois la notion de travail qu'un réseau va traiter (on parle ainsi du trafic routier, par exemple) et la grandeur numérique qui mesure ce travail.

2.1 Réseau fonctionnant sans attente

C'est le cas normal d'un réseau à commutation de circuits. Observons un groupe de circuits (faisceau reliant deux commutateurs du réseau téléphonique, par exemple). Le nombre de circuits occupés concerne directement la source (susceptible de réclamer un circuit et d'essuyer un échec). Ce nombre fluctue, et l'on peut en définir une valeur moyenne, qu'on appellera *trafic écoulé*. En termes mathématiques, si on note $N(t)$ le nombre des circuits occupés à l'instant t, le trafic écoulé A_e sera l'espérance mathématique de $N(t)$ (supposant les bonnes conditions de stationnarité).

$$A_e \equiv E(N) = \lim_{T\to\infty} \frac{1}{T} \int_0^T N(t)dt \tag{2.1}$$

Comment estimer expérimentalement cette grandeur? Supposons qu'on observe un groupe de R circuits (ceci est valable, bien sûr, pour tout système de "serveurs" recevant des "clients", selon la terminologie de la Théorie des Files d'attentes). L'observation dure un temps T, suffisamment long; des connexions ont lieu, dont on enregistre les durées: $d_1, d_2, \cdots d_k$. Le plus souvent, des connexions ont déjà commencé et sont en cours à l'instant 0, d'autres ne s'achèveront qu'après T. On incorpore alors au comptage la fraction incluse dans l'intervalle. On estimera le trafic écoulé par:

$$A_e \approx \frac{\sum d_i}{T}$$

(en toute rigueur, il faudrait augmenter T indéfiniment, ce qui éliminera tout "effet de bord": la quantité ci dessus est un *estimateur* du trafic écoulé). La preuve de l'identité entre les deux expressions de A_e est dans le mode de calcul "expérimental" de la moyenne: on estime la surface en comptant les "pavés" de la figure 2.1.

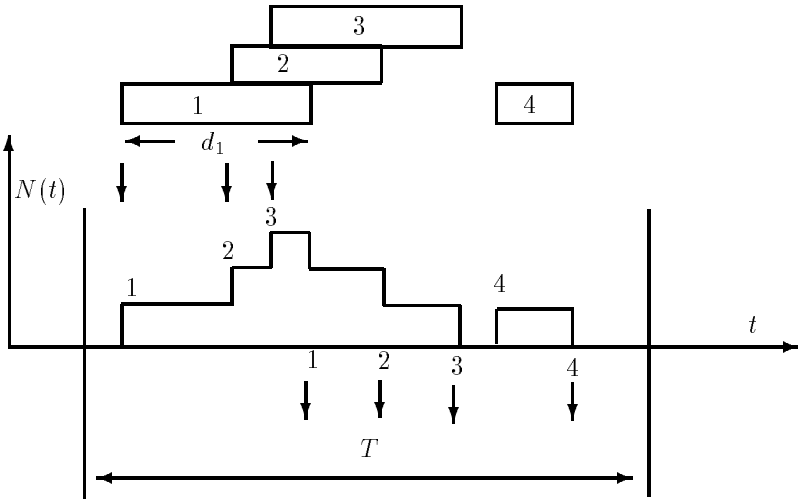


FIG. 2.1 – *Calcul de l'intégrale donnant* A_e

Le trafic écoulé peut s'interpréter d'une autre façon. Soit d la durée moyenne de service (supposant, encore une fois, l'existence de toutes les limites nécessaires). Soit n le nombre des "clients" arrivant dans la période de mesure $(0, T)$. La formule définissant le trafic peut s'écrire:

$$A_e = \frac{\sum d_i}{T} = \frac{\sum d_i}{n} \times \frac{n}{T}$$

Le premier terme représente la durée moyenne de service (là encore, si T, et donc n, tendent vers l'infini). Le second terme représente le taux des arrivées acceptées (nombre par unité de temps). Il faut faire la distinction entre les clients arrivés et ceux qui sont acceptés, puisque des rejets peuvent s'observer.

Le produit ci-dessus admet lui-même une autre formulation. Le produit d'un taux d'entrée par une durée donne évidemment le nombre entrant pendant cette durée: A_e est la moyenne du nombre des clients qui entrent pendant la durée du service.

En résumé, le **trafic écoulé** est, à la fois (on retrouve là un avatar de la formule de Little):

- Le nombre des clients entrant dans le système pendant la durée moyenne de service.
- Le produit du taux d'arrivée par la durée moyenne du service.
- Le nombre moyen des serveurs occupés.

Le trafic écoulé est une grandeur sans dimension. On le compte en erlangs (du nom du mathématicien danois, père de la théorie (1878-1929).

On définit aussi le **trafic offert**. C'est le nombre A des clients arrivant pendant la durée de service (dont certains seront rejetés).

Par construction, le trafic A_e écoulé par un groupe de R serveurs vérifiera évidemment l'inégalité $0 \leq A_e \leq R$. Pour le trafic offert, il n'a pas de limite. Le taux de rejet B, ou la probabilité de perte, s'écrira comme le quotient du nombre de demandes rejetées au nombre de demandes présentées n (n_e est le nombre accepté) dans tout intervalle de temps, par exemple la durée moyenne de communications. Et donc,

$$B = \frac{n - n_e}{n} = \frac{A - A_e}{A} \tag{2.2}$$

2.2 Réseau fonctionnant avec attente

On est, ici, dans la configuration du réseau à commutation de paquets. Les mêmes notions, les mêmes définitions s'appliquent encore. Le cas très fréquent d'un serveur unique offre la possibilité d'enrichir les notions présentées. Considérons, dans un réseau de paquets, la ligne de transmission sortante, reliant deux noeuds du réseau. Les paquets doivent attendre dans la file de sortie pour être émis l'un après l'autre. Cette fois-ci, l'état de la ligne est encore décrit par $N(t)$, $0 \leq N \leq R$, mais avec $R = 1$ (c'est à dire, le serveur est libre ou occupé).

Si l'on suppose une file d'attente de capacité infinie, aucun rejet ne se produit, et les notions de trafic offert et de trafic écoulé se confondent. Puisqu'il n'y a qu'un seul serveur, le nombre moyen de serveurs occupés se réduit à la probabilité d'observer le serveur occupé (moyenne de l'indicatrice de l'événement {serveur occupé}). Les définitions équivalentes mentionnées plus haut conduisent à la relation classique:

$$\rho = \lambda E(s) = 1 - P_0 \tag{2.3}$$

Dans cette expression, λ désigne le taux des arrivées (c'est la notation traditionnelle), $E(s)$ le temps moyen de service, ρ est le trafic offert (c'est aussi une notation classique). P_0 représente la probabilité stationnaire de trouver le serveur libre (proportion du temps d'inactivité. Les clients qui arrivent peuvent mesurer une probabilité d'inactivité différente, on y reviendra).

Les systèmes à attente sont cependant, de façon inévitable, à *capacité limitée*, et la distinction offert/écoulé reprend alors son sens.

2.3 Les processus d'arrivées

Pour aller plus avant dans l'étude des propriétés du trafic, il va être nécessaire de passer en revue les deux composantes dont il dépend. A savoir, les arrivées de "clients", et leur "service".

On observe les arrivées de clients à l'entrée du système. Pour décrire le phénomène, la première idée venant à l'esprit sera d'utiliser l'intervalle du temps entre arrivées successives, ou bien le nombre des arrivées dans un intervalle donné.

Pendant la durée T, $n(T)$ arrivées se produisent. On chiffre le volume du flux arrivant

par le taux d'arrivée dont la définition intuitive est:

$$\lambda \equiv \lim_{T \to \infty} \frac{n(T)}{T} \tag{2.4}$$

On pourra aussi estimer l'intervalle entre arrivées: c'est l'inverse de la quantité précédente.

2.3.1 Processus de Renouvellement

Dans le cas le plus favorable, la description statistique du temps entre les arrivées est très simple. Supposons que la situation puisse être décrite de la façon suivante:

Chaque fois qu'une arrivée se produit, je tire au sort, selon une loi donnée, l'intervalle jusqu'à la prochaine arrivée, de telle sorte que les intervalles successifs soient indépendants.

On est alors dans le cas très particulier d'un *processus de renouvellement*. Il faut comprendre l'intérêt d'une telle notion, mais aussi ce qu'elle a de "rare". Soit par exemple un processus d'arrivée qui résulterait de la superposition de deux processus de renouvellement, indépendants.

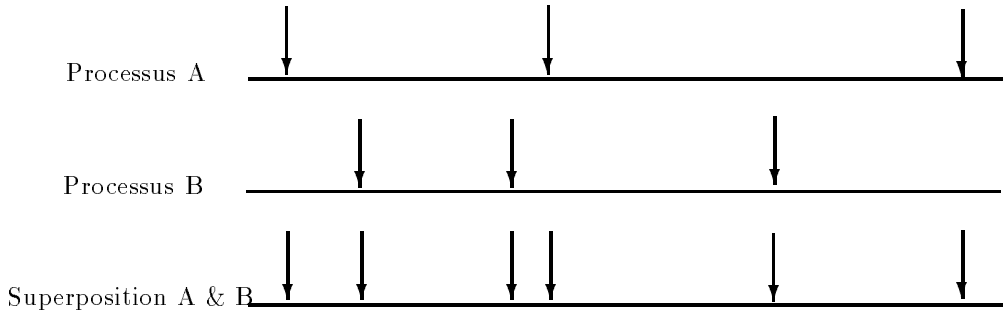


FIG. 2.2 – *La superposition de A et B n'est pas un renouvellement*

Inutile de remarquer combien cette situation est commune! Le processus de superposition *n'est pas* un renouvellement. En effet, pour prévoir l'instant d'arrivée du 3ème client, il faut se référer au 2ème, par exemple en tirant le temps interarrivées. Mais pour le 4ème, son arrivée ne peut être prévue que par référence au 1er, et non au 3ème.

2.3.2 Processus de Poisson

Supposons que le processus des arrivées obéisse aux règles suivantes:

- la probabilité d'une arrivée dans un intervalle $[t, t + \Delta t[$ ne dépend pas de ce qui s'est passé avant l'instant t. C'est la propriété dite "sans mémoire".
- la probabilité d'apparition d'un client est proportionnelle à Δt, la probabilité de plus d'un événement étant "négligeable" (infiniment petit d'ordre supérieur). Le coefficient de proportionnalité est noté λ (intensité du processus).

Ce sont là les hypothèses classiques qui conduisent au processus de Poisson. La probabilité d'observer k arrivées dans un intervalle de longueur t vaut:

$$P_k(t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t} \tag{2.5}$$

La loi de probabilité de l'intervalle entre deux arrivées successives est

$$A(t) = 1 - e^{-\lambda t} \tag{2.6}$$

La moyenne de cet intervalle vaut $1/\lambda$ (voir cours files d'attentes pour une justification). Noter que le nombre moyen d'arrivées observé dans tout intervalle de longueur t est λt. Noter aussi que le processus de Poisson est un processus de renouvellement.

2.3.3 Où rencontre-t-on les processus de Poisson ?

Quand on "ne sait rien" d'un processus d'arrivées, il est assez tentant de lui attribuer les deux propriétés énoncées ci-dessus, et qui conduisent aux formules du processus de Poisson, tant elles semblent générales et raisonnables pour de nombreux systèmes. En réalité, il n'en est rien – et la vérification de ces hypothèses n'ira généralement pas de soi. Le plus souvent, la justification du recours à l'hypothèse Poissonnienne repose sur le résultat suivant.

Théorème [2, 4]: Soient k processus de renouvellement indépendants, non nécessairement poissonniens de taux d'arrivées $(\lambda_i, i = 1, \cdots, k)$. On note $\lambda = \sum \lambda_i$ le taux d'arrivée global du processus résultant de leur superposition. Si la somme précédente admet une limite $\lambda*$ lorsque k augmente indéfiniment, alors le processus superposition tend vers un processus de Poisson de taux $\lambda*$.

(Remarque: l'emploi de "λ" et de la notion de taux d'arrivée ne préjugent absolument pas d'une hypothèse Poissonnienne).

2.4 Le temps de service

Le processus de service pourra être d'une complexité extrême, mais on se borne le plus souvent à supposer que chaque durée de service est indépendante des autres, et qu'elles obéissent toutes à une même loi de distribution: on parle de variables indépendantes et identiquement distribuées (i.i.d.). On décrira cette loi par sa distribution de probabilité :

$$B(x) = P\{\text{temps de service } \leq x\}$$

2.4.1 Le temps de service résiduel

Une question qui revient souvent: j'observe un service qui a déjà atteint "l'âge" y. Je m'interroge sur la distribution de la durée restant à courir jusqu'à la fin de ce service (notons X la variable aléatoire correspondante).

La réponse est très simple: dire que $X > x$ secondes restent à accomplir revient à dire que la durée totale sera supérieure à $y + x$, et savoir que $Y = y$ revient à calculer la probabilité

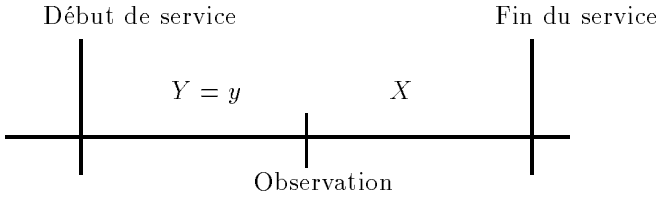


FIG. 2.3 – *Calcul de la distribution du temps résiduel*

"sachant que" la durée totale est supérieure à y: c'est une probabilité conditionnelle. On a alors pour la distribution de X:

$$\begin{aligned} P(X > x | Y = y) &= P(\text{service} > x + y | Y = y) \\ &= \frac{P(\text{service} > x + y)}{P(\text{service} > y)} \\ &= \frac{1 - B(x + y)}{1 - B(y)} \end{aligned} \tag{2.7}$$

De façon équivalente, la densité sera

$$P(X \in [x, x + dx[|Y) = \frac{b(x + y)dx}{1 - B(y)} \tag{2.8}$$

2.4.2 La loi exponentielle

La loi de service la plus populaire est la loi exponentielle, qu'il est traditionnel d'écrire en utilisant comme " taux de service " la lettre μ:

$$B(x) = P(\text{service} \leq x) = 1 - e^{-\mu x} \tag{2.9}$$

la densité correspondante est $b(x) = \mu e^{-\mu x}$

La loi exponentielle doit une bonne partie de son prestige à la propriété "sans mémoire", qui pourrait s'énoncer ainsi: savoir que le service a déjà duré 1 heure n'apporte aucun renseignement sur sa fin prochaine. On remarquera en effet que, pour la loi exponentielle, la densité calculée à l'équation (2.8) vaut:

$$b(x|y) = \frac{\mu e^{-\mu(x+y)}}{e^{-\mu y}} = \mu e^{-\mu x} = b(x)$$

indépendamment de y. L'application de cette "absence de mémoire" montre que la probabilité d'une fin de service dans l'instant qui vient (dans l'intervalle $[t, t + dt[$) est μdt, quel que soit l'âge du service.

La loi exponentielle se caractérise par ses moments:

- Moyenne de la variable : $1/\mu$.
- Variance de la variable : $1/(\mu^2)$

On introduit le *coefficient de variation*, rapport de l'écart-type[1] du temps de service à sa moyenne. Ici, on voit que le coefficient de variation vaut 1.

1. Est-il besoin de le rappeler – l'écart-type est la racine carrée de la variance?

2.4.3 Lois d'Erlang

Supposons que le processus de service soit composé d'une cascade de k serveurs élémentaires exponentiels, identiques (c'est à dire, de même paramètre μ), et indépendants les uns des autres. Le temps du service est la somme des temps passés dans chaque serveur.

Supposons $k = 2$. Notons X le temps total, X_1 et X_2 les durées des deux temps services; la distribution B de X est donnée par la *convolution* de B_1 et B_2:

$$P\{X \leq x\} = P\{X_1 + X_2 \leq x\} = \int_{u=0}^{x} B_1(x-u)dB_2(u)$$

Evidemment, B_1 et B_2 sont identiques, et correspondent à l'exponentielle.

$$\begin{aligned} P\{X \leq x\} &= \int_{u=0}^{x} \left[1 - e^{-\mu(x-u)}\right] \mu e^{-\mu x} du \\ &= 1 - e^{-\mu x} - \mu x e^{-\mu x} \end{aligned}$$

Plus généralement, on montre que la mise en cascade de k serveurs conduit à une distribution (voir Appendice):

$$B(x) = P\{X_1 + X_2 + \cdots + X_k \leq x\} = 1 - \sum_{j=0}^{k} \frac{(\mu x)^j}{j!} e^{-\mu x} \tag{2.10}$$

On appelle cette distribution la *distribution d'Erlang*-k, et la loi de probabilité 2.10 est la "loi d'Erlang-k". Puisqu'il s'agit d'une somme de variables aléatoires indépendantes, moyenne et variance s'obtiennent facilement:

- Moyenne de la variable : k/μ.
- Variance de la variable : $k/(\mu^2)$
- Le coefficient de variation est $1/\sqrt{k}$.

2.5 Le modèle d'Erlang

C'est le modèle de système le plus simple, et le premier à avoir été étudié. Considérons un groupe de serveurs, exploités "en pool", c'est à dire que chacun des serveurs peut servir indifféremment les clients qui se présentent. Les clients arrivent devant le groupe des serveurs selon un processus de Poisson, de taux λ. A leur arrivée, les clients sont servis immédiatement tant qu'un serveur au moins est libre; si les serveurs sont tous occupés, le client qui arrive est rejeté, et est supposé disparaître définitivement.

On note R le nombre des serveurs. La loi de service est supposée exponentielle, de paramètre μ. Cette hypothèse n'est pas nécessaire, les résultats ci-après restant vrais pour une loi quelconque (de moyenne $1/\mu$). L'hypothèse exponentielle permet des calculs simples.

Selon la terminologie de Kendall, nous avons affaire au système M/M/R/R.

Grâce aux hypothèses exponentielles (et Poisson), le système est justiciable d'une analyse markovienne. En fait, l'état du système est représenté par n, le nombre de clients en cours de

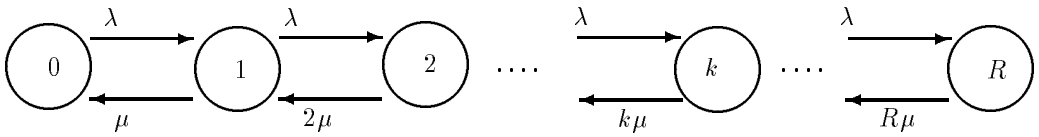


FIG. 2.4 – *Diagramme d'état du modèle d'Erlang*

service (nombre de serveurs occupés), et l'évolution de n obéit à un processus de Naissance et de Mort, conforme au diagramme de la Figure 2.4.

On notera particulièrement les valuations des transitions. Lorsque le système est dans l'état n, les n services en cours se déroulent en parallèle, indépendamment les uns des autres, la probabilité que chaque service s'interrompe dans l'instant dt est μdt; la probabilité d'un passage de n à $n-1$ résulte de la fin d'un des n services, soit $n\mu dt$.

De ce diagramme, on déduit la probabilité d'observer le système dans l'état n:

$$P(n) = \frac{A^n/n!}{\sum_{j \leq R} A^j/j!} \tag{2.11}$$

Dans cette expression, $A = \lambda/\mu$ désigne le trafic offert. La probabilité de rejet est notée traditionnellement $E(A,R)$, c'est la probabilité stationnaire d'observer $n = R$:

$$E(A,R) = \frac{A^R/R!}{\sum_{j \leq R} A^j/j!} \tag{2.12}$$

Son importance est capitale: elle permet le dimensionnement des organes des réseaux, notamment dans le réseau téléphonique [2].

Le calcul du trafic écoulé permet de vérifier la concordance des définitions:

$$A_e = \sum jP(j) = A[1 - E(A,R)] \tag{2.13}$$

Le calcul effectif de la formule 2.12 pose problème sous la forme ci-dessus: il serait vain d'évaluer une factorielle par une méthode directe! Il vaut mieux utiliser une récurrence:

```
X := 1
pour j de 1 a R faire
   X := 1 + X*j/A
fin pour
E(A,R) := 1/X
```

On pourra aussi souvent utiliser une approximation (remplacer la somme au dénominateur par l'exponentielle, et utiliser la formule de Stirling pour les factorielles):

$$E(A,R) \approx \left(\frac{Ae}{R}\right)^R e^{-A}/\sqrt{2\pi R} \tag{2.14}$$

2. Pour d'obscures raisons, les anglo-saxons la nomment "formule d'Erlang-B"

2.6 Les modèles à files d'attentes

Nous rappelons très brièvement les principaux résultats: se reporter à un cours élémentaire de files d'attentes.

2.6.1 File M/M/1

C'est le modèle à file d'attentes le plus célèbre – à juste titre, certainement. Il permet en effet d'illustrer les concepts fondamentaux liés à l'attente devant un serveur, tels que les présentent les quelques exercices (classiques) proposés en annexe.

Le système est décrit par le processus des arrivées, Poissonnien de taux λ, la loi exponentielle de service (taux μ), la file d'attente de capacité infinie. Les clients pourront être supposés servis dans l'ordre de leur arrivée si cela peut aider l'intuition – mais cette hypothèse n'est en fait pas nécessaire.

L'état est encore décrit par n, qui représente cette fois le nombre total des clients dans le système. Il évolue selon un processus de Naissance et de Mort très simple (figure 2.5).

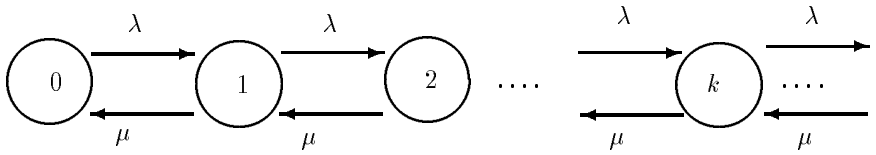


FIG. 2.5 – *Evolution de l'état de la file M/M/1*

Du diagramme, on déduit les résultats classiques:

$$\begin{aligned} P_n &= \rho^n(1-\rho) \qquad n \geq 0 \\ P_W &= \rho \\ E(n) &= \frac{\rho}{1-\rho} \\ E(n_W) &= \frac{\rho^2}{1-\rho} \end{aligned} \tag{2.15}$$

P_W est la probabilité qu'un client ait à attendre. $E(n)$ donne le nombre moyen de clients dans le système, et $E(n_W)$ le nombre moyen en attente. On en déduit, *via* la formule de Little, les temps moyens d'attente $\frac{\rho/\mu}{1-\rho}$ et de séjour $\frac{1/\mu}{1-\rho}$.

Les expressions ci-dessus font jouer à ρ un rôle important. Tout spécialement, elles imposent une limite: ρ doit être strictement inférieur à 1, sinon les quotients n'ont plus de sens. On montre que la condition $\rho < 1$ est une condition nécessaire et suffisante d'existence des probabilités stationnaires, ce qui est conforme à l'intuition.

2.6.2 La file M/M/c

Mêmes hypothèses et notations que ci-dessus, avec un nombre c de serveurs identiques en pool. Le taux de service vaut $k\mu$ tant que $k < c$ et $c\mu$ au-delà. Le trafic offert est noté A

et le trafic par serveur est $\rho = A/c$. La condition de stabilité reste $\rho < 1$ – soit $A < c$.

On trouve:

$$P(k) = P(0)\frac{A^k}{k!} \quad k \leq c \tag{2.16}$$

$$P(k) = P(0)\frac{A^c}{c!}(\rho)^{k-c} \quad k > c \tag{2.17}$$

avec

$$P(0) = \left[\sum_0^{c-1} \frac{A^k}{k!} + \frac{A^c}{c!}\frac{1}{1-\rho}\right]^{-1} \tag{2.18}$$

La probabilité d'attendre est

$$P_W = \frac{A^c}{c!}\frac{1}{1-\rho}P(0) = C(A,c) \tag{2.19}$$

On connait cette formule sous le nom de "Formule d'Erlang avec attente" ou "Formule Erlang-C".

Le temps moyen d'attente se déduit de ce qui précède, au prix de quelques calculs:

$$E(W) = \frac{P_W}{c(1-\rho)}E(s) \tag{2.20}$$

2.6.3 Modèles à capacité limitée

Le modèle M/M/1/K correspond au cas d'une capacité de K clients. Attention, l'usage veut que K représente le nombre total de clients dans le système, c'est à dire en attente ou en service.

La troncature du diagramme M/M/1 donne facilement le résultat:

$$P(n) = \frac{\rho^n(1-\rho)}{1-\rho^{K+1}} \tag{2.21}$$

Le critère de QS est bien sûr la probabilité de rejet:

$$\Pi = \frac{\rho^K(1-\rho)}{1-\rho^{K+1}} \tag{2.22}$$

Evidemment, ici on peut lever la condition $\rho < 1$. On remarque que $\rho = 1$ conduit à une difficulté sur la formule. C'est qu'en réalité elle est "arrangée", et il faudrait lire par exemple

$$\Pi = \frac{\rho^K}{1+\rho+\rho^2+\cdots+\rho^K}$$

Compléments et Exercices

Les courbes d'Erlang

Voici, à titre d'illustration (Figure 2.6), l'allure générale des courbes donnant la *perte d'Erlang*, c'est à dire la perte calculée selon les principes de la Section 2.5 : formule (2.12).

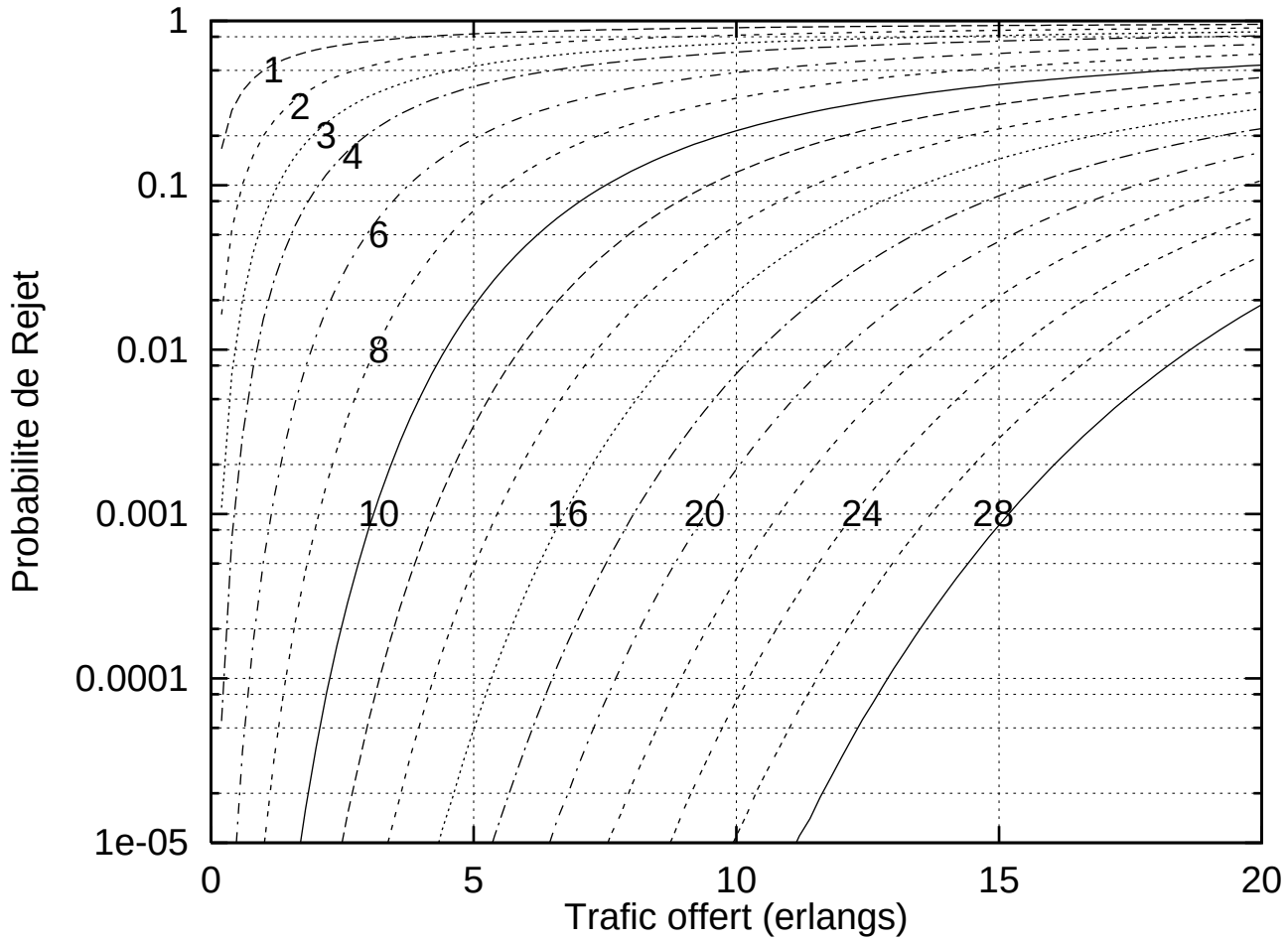


FIG. 2.6 – *Perte d'Erlang*

La somme de n exponentielles

Soient $X_1, \cdots, X_n$ des variables exponentielles i.i.d. (indépendantes et identiquement distribuées). Quelle est la distribution de probabilité de la somme $X_1 + \cdots + X_n$? Cette question est avantageusement abordée via les méthodes de transformées. La *transformée de Laplace* de la densité $f(x)$ est donnée par:

$$\tilde{F}(s) = \int_0^\infty e^{-sx} f(x) dx$$

On se reportera à son cours de maths pour les propriétés de ces fonctions.

La transformée de Laplace de la distribution commune des X_i est $\mu/(s+\mu)$. La transformée de la somme des variables indépendantes est le produit des transformées, soit en notant $F_n(t)$ la distribution:

$$\mathcal{L}(F_n) = \left[\frac{\mu}{\mu + s}\right]^n$$

On trouve, en consultant une table de transformées, la densité de probabilité correspondante:

$$f_n(t) = \mu \frac{(\mu t)^{n-1}}{(n-1)!} e^{-\mu t}$$

Exprimé sous forme de distribution:

$$F_n(t) = P\{X_1 + \cdots + X_n \leq t\} = \sum_{k=0}^{n} \frac{(\mu t)^k}{k!} e^{-\mu t} \tag{2.23}$$

Repères bibliographiques

L'introduction du Calcul des Probabilités et donc du "Hasard", est inévitable. Il faut donc au lecteur de bonnes notions de probas. Voir par exemple l'ouvrage classique [4]. Il faut aussi de bonnes notions en théorie des files d'attentes – nous avons seulement rappelé ici les résultats directement nécessaires. Pour une étude plus approfondie, on lira, outre les polycopiés du cours FA, les références [3, 6, 7, 11] – la liste n'est pas limitative.

Pour changer de registre, la lecture de [14] est très enrichissante. Il développe des considérations stimulantes sur la façon dont le hasard s'introduit dans la physique – considérations applicables au trafic.

Exercices

2.1 - Un groupe de circuits téléphoniques reçoit des appels, selon le schéma suivant: arrivées poissonniennes, 150 appels par heure, duré moyenne des appels 4 minutes.

a) Calculer le trafic offert.

b) Pour un taux de perte de 0.01, combien faut-il installer de circuits? Même question pour un taux de 0.001 .

2.2 - Sur le groupe de circuits ci-dessus, des mesures, après quelques temps d'exploitation, donnent un flux *observé* de 250 appels / heure. Est-il alors possible de calculer le trafic offert? (indice: il s'agit de calculer A dans la formule d'Erlang (2.12), connaissant A_e et R: il faut procéder par approximations successives).

2.3 - Exprimer le temps moyen d'attente (ou de séjour) des files M/M/1 et M/M/c en unités du temps de service (c'est toujours l'unité "naturelle" dans laquelle les résultats doivent s'exprimer, parce qu'elle donne l'échelle de référence des clients). Vérifier qu'alors les résultats ne dépendent pas de λ et μ séparément, mais seulement de leur quotient.

Même remarque pour les probabilités de rejet et d'attente.

2.4 - Donner une expression permettant de calculer $C(A, c)$, à partir de la formule d'Erlang. Application: algoritme de calcul par récurrence de $C(A, c)$.

2.5 - A la Banque Générale, les clients se font servir par deux employés, l'un traitant les virements, l'autre les retraits. Les taux d'arrivée des clients sont identiques pour chaque guichet: 12 clients par heure. On les suppose arriver selon un processus de Poisson.

Les temps de service sont identiques aux deux guichets, distribués exponentiellement de moyenne 4 minutes.

a) Quelle est la probabilité d'attente, et le temps moyen d'attente?

b) Le directeur de l'agence, ancien élève de l'INT, décide de regrouper les deux services, en demandant à chaque employé d'exécuter indifféremment chacune des opérations, les clients voyant ainsi une file unique. Le Directeur a-t-il bien assimilé ses cours de trafic (c'est à dire, les temps d'attente, etc, ont-ils diminué)?

2.6 - Un Réparateur

Un atelier comporte N machines (produisant des vis à tête molle), travaillant indépendamment. Ces machines peuvent tomber en panne. Une machine est soit opérationnelle (débit nominal) soit en panne (débit nul). Le processus de panne est "aléatoire": outil qui casse, vis qui se coince, défaut d'alimentation, etc. On adopte un modèle poissonnien, la machine en ordre à t a une probabilité $\lambda.dt + o(t)$ de tomber en panne entre t et $t + dt$.

Un réparateur tente de faire fonctionner tout ça: dès qu'une machine s'arrête, il la répare, ce qui lui prend un temps distribué exponentiellement (paramètre μ). Naturellement, si deux machines sont arrêtées, il répare l'une, puis l'autre.

a) Montrer qu'on peut représenter le fonctionnement de l'atelier par une file d'attente ou par un processus de naissance et de mort [exemple: l'état = nombre de machines en pannes]. Préciser les coefficients de transition.

b) Ecrire - et résoudre - les équations donnant les probabilités d'état.

c) Décrire les paramètres de performance du système, et les relier aux probabilités d'état.

d) L'efficacité semble insuffisante: la direction embauche un second réparateur. Que deviennent les coefficients, etc.

Application numérique: N=10, λ=1/heure, $1/\mu$=10 minutes.

2.7 - Les joyeux magasiniers

Dans un atelier, des magasiniers alimentent les ouvriers en outils, pièces détachées, etc. Le responsable se lamente devant le temps perdu en attentes au magasin, mais ne veut pas payer trop de magasiniers! Pingre mais avisé, il confie une étude d'ingénierie à un ancien élève de l'INT. D'un comptage préliminaire, il ressort que:

- le temps de service d'une demande au magasin est distribué exponentiellement, avec une moyenne b=5 minutes;
- les demandes arrivent selon un flux qu'on peut supposer poissonnien, une demande toutes les 3 minutes.

Les salaires horaires des magasiniers et des ouvriers sont identiques. Combien faut-il de magasiniers pour minimiser la dépense? Indications:

a) Ecrire le comportement de la file d'attente (taux d'arrivées, taux de départs - fonctions peut-être de l'état). Ecrire le diagramme du processus de naissance et de mort.

b) La résolution fournit le nombre de clients; en déduire le nombre moyen. Ecrire le temps d'attente moyen en fonction de c (le nombre de magasiniers);

c) en déduire le temps total passé en attente (par jour, par exemple);

d) en déduire le coût total de la solution à c magasiniers, et faire varier c (que remarque-t-on, quant au rendement optimum?).

Chapitre 3

Systèmes à population limitée

L'étude des systèmes à population limitée offre un moyen commode d'illustrer quelques conséquences de la présence de trafics *non Poissonniens*, et spécialement la distinction entre les probabilités stationnaires et les probabilités observées par les clients.

3.1 Le cas de système avec attente

3.1.1 Un exemple d'application

Considérons une installation informatique de type "temps partagé". Des terminaux sont reliés à un organe (calculateur central), qui les sert selon un mode de travail interactif. Chaque "client" fonctionne selon un cycle d'activité régulier: réflexion du programmeur devant sa console, suivie de l'émission d'une requête à destination du calculateur, attente de la réponse, puis à nouveau réflexion ("repos", vu du calculateur - l'usager élabore la requête suivante). Il y a N usagers au total: la population est de taille limitée.

On cherche ici un modèle global du comportement de l'ensemble, laissant momentanément de côté les détails du fonctionnement du calculateur (un modèle fin ferait intervenir les unités d'entrée/sortie, les disques, etc; voir pour cela une prochaine leçon). Il suffit, à ce stade, de préciser que l'étude fine permet de justifier le modèle global qu'on utilise ici.

Le comportement de chacun des usagers se caractérise par le diagramme suivant (Figure 3.1). Le service sera supposé exponentiel, de paramètre μ, par souci de simplicité. Le temps de réflexion est aussi exponentiel, de paramètre λ. Le service est supposé FIFO.

La Figure 3.2 schématise le fonctionnement du système complet et du mouvement des clients. C'est le cas le plus simple de *réseau de files d'attentes fermé*.

3.1.2 Modélisation et résolution

On construit aisément le modèle du Processus de Naissance et de Mort correspondant: soit n le nombre des clients en attente ou en service - et $N - n$ le nombre des clients "au repos". Le taux de passage de n à $n + 1$ est fonction de n. En effet, une source libre à t a

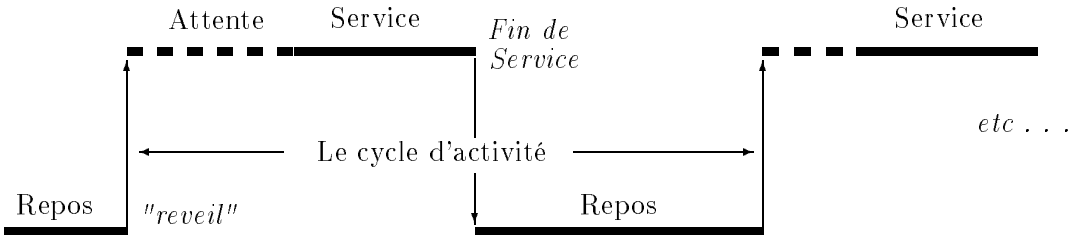


FIG. 3.1 – *Comportement cyclique de la source*

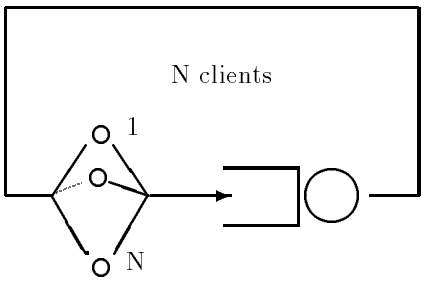


FIG. 3.2 – *Modèle fermé d'une installation "temps partagé"*

une probabilité $\lambda dt + O(dt)$ d'émettre une demande entre t et $t + dt$. En d'autres termes, si n clients sont actifs, seules $N - n$ sources sont susceptibles d'émettre, soit un taux de "naissance" égal à: $\lambda(n) = (N - n)\lambda$.

Remarque: Le lecteur est encouragé à vérifier que les hypothèses qu'on a posées autorisent l'usage du modèle de Naissance et de Mort, c'est à dire que la seule donnée de n à l'instant t permet de prévoir (statistiquement) l'état à tout instant $t' > t$.

On note P_n les probabilités stationnaires d'observer n clients dans le serveur ou la file. Pour mémoire, les équations (que le lecteur retrouvera aisément):

$$\begin{cases} (N-n)\lambda P_n &= \mu P_{n+1} \qquad n = 0, \ldots, N-1 \\ \displaystyle\sum_{n=0}^{N} P_n &= 1 \end{cases} \tag{3.1}$$

Et leur solution:

$$\begin{aligned} P_n &= \frac{N!}{(N-n)!}\alpha^n P_0 \qquad \alpha = \lambda/\mu \\ P_0 &= \left[\sum_{n=0}^{N} \frac{N!}{(N-n)!}\alpha^n\right]^{-1} \end{aligned} \tag{3.2}$$

Le paramètre important de ce modèle est T, le **temps de réponse**. C'est le temps passé dans les états d'attente et de service. On peut le calculer rapidement, au moyen du raisonnement astucieux suivant. Notons C le temps moyen d'un cycle d'un client (temps moyen entre deux fins de services successives d'un même client). Comme on le vérifie sur la Figure 3.1, $C = T + 1/\lambda$. Le débit du serveur sera N/C: chacun des clients sort, en moyenne,

une fois toutes les C secondes, soit N sorties en C secondes. Mais le serveur a un débit égal à $\mu(1-P_0)$ (débit μ, sauf s'il est au repos). D'où:

$$\begin{array}{rl} \frac{N}{C} & = \mu(1-P_0) \\ T & = \frac{N}{\mu(1-P_0)} - \frac{1}{\lambda} \end{array} \tag{3.3}$$

De ces solutions, on peut définir des grandeurs globales. Ainsi, le taux d'arrivée dans l'état (k) est $(N-k)\lambda P_k$; le taux d'arrivée moyen est donc:

$$\Lambda = \sum \lambda(N-n)P_n = [N - E(n)].\lambda$$

où $E(n)$ désigne le nombre moyen de clients dans le système (serveur ou file d'attente).

3.1.3 L'application pratique

Observons ce qui se passe lorsque on augmente le nombre des utilisateurs du système. Le graphique de la Figure (3.3) représente le temps de réponse normalisé, c'est à dire le produit μT, qui exprime le temps en unités du temps de service en fonction de N, pour plusieurs valeurs de α.[1] On remarque l'existence de deux régimes de fonctionnement, symbolisés par les deux comportements-limites:

- Quand $N \to 0$, le temps de réponse normalisé tend vers 1 (l'attente est nulle).
- Quand N augmente, le temps de réponse croît linéairement. En effet, $P_0 \to 0$, et $T \to N/\mu - 1/\lambda$.

Les courbes suggèrent l'existence d'une valeur critique N^*, point d'intersection des deux asymptotes, et qui représente le changement de régime. On voit que $N^* = (\alpha+1)/\alpha$. Sur le dessin, $N^* = 11$, pour $\alpha = 0.1$.

Remarque: C'est le *modèle de Scherr*, qui date de 1967, et qui fait preuve d'une grande "robustesse" - c'est à dire qu'il a été vérifié dans des configurations où les hypothèses faites ici n'étaient pas du tout vérifiées (on se reportera à un cours sur les réseaux de files pour comprendre les raisons de cette robustesse – et notamment au "théorème BCMP).

3.1.4 L'Etat aux instants d'arrivées

La solution exhibée ci-dessus pour les P_n ne satisfait pas les espérances de l'utilisateur. Par chance, on en a déduit le temps de réponse. Mais on ne saurait en déduire mieux – et par exemple la distribution de l'attente, ou toute autre quantité liée à l'état vu par les arrivants.

En effet, les P_n donnent la probabilité stationnaire d'observer 1, 2, ... clients dans le système. Pour en donner une signification expérimentale, les P_n représentent la proportion du temps pendant lequel le système contient 1, 2, .., n clients. Le système, par exemple,

1. Ce genre de représentation, faisant intervenir une quantité sans dimension, est à rechercher: le graphique acquiert un caractère très général, il ne dépend que de 2 paramètres, N et α.

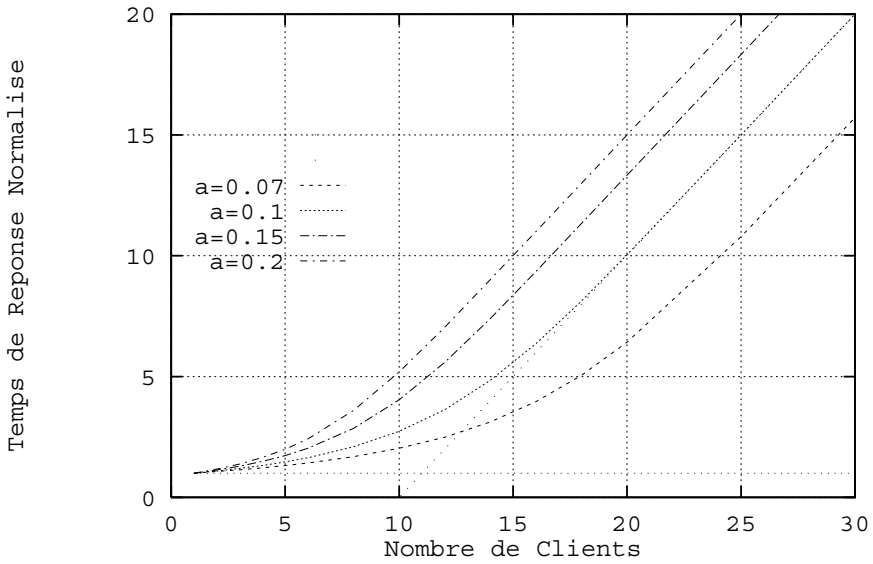


FIG. 3.3 – *Temps de réponse du réseau fermé*

séjourne un temps positif dans l'état $n = N$, mais *aucun client arrivant ne peut voir cet état*, puisqu'alors aucun client n'est inactif et ne peut se réveiller !

Inévitablement, la modélisation par processus de Naissance et de Mort conduit aux probabilités *stationnaires* d'état. Les probabilités effectivement recherchées sont différentes, elles concernent l'état constaté par les clients qui arrivent. On les obtient, pour notre système (qu'il faut noter M(n)/M/1, selon Kendall), par le raisonnement suivant.

Ecrivons la probabilité qu'une arrivée voit l'état k, c'est à dire la proportion des arrivants qui observent cet état. On notera Q_k le jeu des probabilités vues par les arrivants. C'est la probabilité que le système soit dans un état k à t, **sachant que** une arrivée se produit à cet instant. En d'autres termes, et peut-être plus intuitivement, on écrit Q_k comme le quotient du taux des arrivées dans l'état k au taux moyen des arrivées Λ (si l'on préfère, c'est le quotient du nombre moyen d'événements "arrivées dans l'état k" au nombre "arrivées" dans un intervalle quelconque Δt).

$$Q_k = \frac{(N-k)\lambda P_k}{\sum (N-n)\lambda P_n} = \frac{N-k}{N-E(n)}.P_k \tag{3.4}$$

3.2 La Propriété "PASTA"

Que déduit-on de ce calcul? Si nous reprenions le même raisonnement pour la file M/M/1, il nous donnerait $P_k = Q_k$. On a mis là le doigt sur une propriété essentielle des flux Poissonniens:

Un flux de clients poissonnien observe l'état stationnaire du système (c'est à dire que la proportion des clients qui voient une configuration donnée est égale

à la proportion du temps où cette configuration sera vue par un observateur externe).

Aussi banale qu'elle paraisse, cette propriété est remarquable. Sa démonstration complète dépasse le cadre de cet exposé. On la connait sous le nom de *Propriété PASTA* (Poisson Arrivals See Time Averages).

La file M/M/1 obéit évidemment à PASTA. Notre système "n'obéit pas à PASTA". Pourquoi cela? C'est qu'après le repos exponentiel la source est en attente, puis active pendant un certain temps, pendant lequel elle n'intervient plus dans les arrivées. On dit parfois *"une source active ne génère pas d'appels"* - pour le dire autrement, si k sources sont actives, le taux d'émission instantané est dû aux $N-k$ libres. Il en résulte une corrélation entre l'état interne du système et le processus des arrivées (l'hypothèse de Poisson voudrait que ce taux soit indépendant du passé ou de l'état présent).

Jusqu'à ce point, on a fait un usage immodéré du "processus de Poisson" pour décrire les arrivées des clients. Il existe fort heureusement un grand nombre de cas où ce choix est légitime. Le plus souvent, ce sont des théorèmes-limites qui motivent cette hypothèse. Par exemple, le théorème cité au 2.3.3, et donnant le processus de Poisson comme la limite d'une superposition d'un nombre croissant de processus *indépendants*.

3.3 Système sans attente

3.3.1 La mise en équations

Examinons un autre cas, celui d'un système à R serveurs sans attente; les sources ont toujours le même comportement statistique, avec cette différence qu'une source devenant active et trouvant tous les serveurs occupés n'attend pas, et retombe immédiatement dans l'état de repos. Elle aura subi un échec, et le problème principal sera de calculer la probabilité de cet événement. C'est le système M(n)/M/R/R.

On récrit les équations. Cette fois, les événements élémentaires sont:

- L'arrivée d'un nouveau client, qui fait passer de n en $n+1$ si $n < R$. Le taux correspondant est $\lambda_n = \lambda(N-n)$: il reste $N-n$ sources au repos susceptibles d'engendrer un client.

- La fin d'un service, avec le taux $n\mu$, qui fait passer de n en $n-1$.

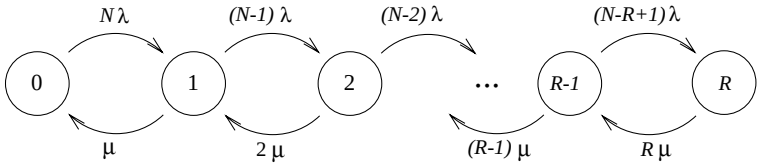


FIG. 3.4 – *Diagramme du processus du système M(n)/M/R/R*

Le jeu des équations donnant la probabilité stationnaire d'observer n serveurs occupés

s'écrit:

$$\begin{cases} (N-n)\lambda P_n &= (n+1)\mu P_{n+1} \qquad n = 0, \ldots, N-1 \\ \sum_{n=0}^{N} P_n &= 1 \end{cases} \tag{3.5}$$

La solution s'écrit:

$$P_n = \frac{\alpha^n \begin{pmatrix} N \\ n \end{pmatrix}}{\sum_{j \le R} \alpha^j \begin{pmatrix} N \\ j \end{pmatrix}} \qquad \text{Rappel} : \begin{pmatrix} N \\ n \end{pmatrix} = \frac{N!}{n!(N-n)!} \tag{3.6}$$

3.3.2 La Probabilité de Rejet

On retrouve, sous une forme presqu'identique, le phénomène de la Section 1: P_R **n'est pas la probabilité de rejet !** Méditant sur la construction du diagramme et sur sa signification, on se rend compte, à nouveau, que cette quantité représente la proportion du temps où tous les serveurs sont occupés. C'est la probabilité d'occupation que mesurerait par échantillonnage un observateur extérieur, c'est à dire indépendant du système (de façon générale, P_k est la proportion du temps où le système réside dans l'état k). **Mais les clients qui arrivent ne le font pas indépendamment de l'état du système**, toujours à cause de la forme des λ_n. Ils ne se font donc pas rejeter indépendamment de l'état.

C'est donc encore une fois, la propriété PASTA qui fait défaut. Comment alors calculer la probabilité de rejet?

La manière la plus simple de l'obtenir est de l'écrire comme le quotient du taux d'arrivée dans l'état de rejet $\{n = c\}$ par le taux moyen d'arrivée Λ (c'est à dire exactement la proportion de clients perdus).

$$B \equiv P(\text{Rejet}) = \frac{\lambda(N-R)P_R}{\Lambda}$$

Au prix d'une gymnastique algébrique modérée, la formule s'exprime en fonction des quantités déja connues:

$$B = \frac{\alpha^n \begin{pmatrix} N-1 \\ R \end{pmatrix}}{\sum_{j \le R} \alpha^j \begin{pmatrix} N-1 \\ j \end{pmatrix}} \tag{3.7}$$

On constate que $B = P_R^{(N-1)}$, c'est à dire la probabilité d'occupation totale pour un système comptant 1 source en moins. Signalons, en commentaire, que la formule (3.7) ci-dessus s'applique même si la loi de service n'est pas exponentielle. Cette formule est connue sous le nom de "formule d'Engset".

3.3.3 Dimensionnement d'un Etage d'Abonnés

A titre d'exemple, considérons un étage d'abonnés, concentrant le trafic de N usagers sur un nombre restreint R de lignes vers le commutateur. Compte tenu des usages téléphoniques des abonnés, il faut calculer R pour offrir une Qualité de Service donnée.

La figure suivante montre ce que donne la formule, selon les valeurs de N et de α, pour $R = 10$.

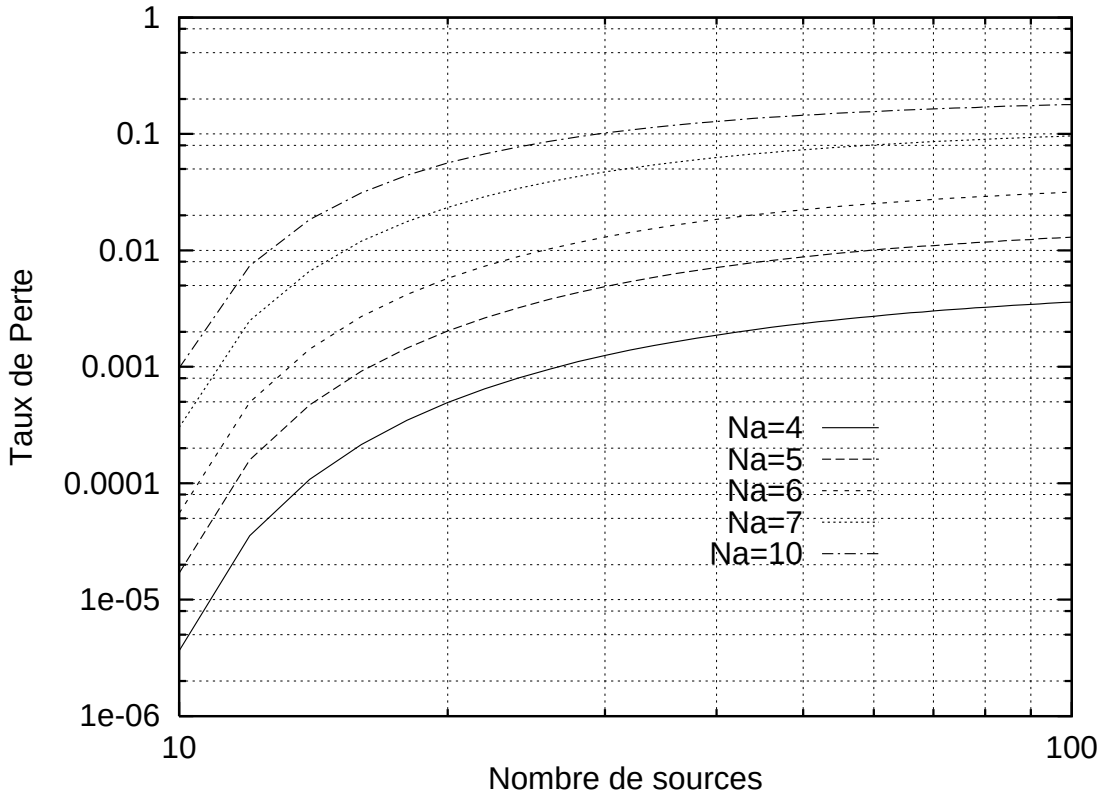


FIG. 3.5 – *Valeur de la probabilité de rejet en fonction du nombre des sources, pour $R = 10$ serveurs*

On voit que quand N grandit, (pour un produit $N\alpha$ constant), le taux de perte augmente. On vérifierait que la limite (quand $N \to \infty$) est celle que donne la "formule d'Erlang" (file M/M/R/R). On voit aussi sur la courbe l'effet "bénéfique" de la population limitée.

A titre d'exemple, un trafic poissonnien de 4 erlangs offert à 10 circuits subit une perte de l'ordre de 5.10^{-3}; si le trafic est issu d'un groupe de 20 sources, le rejet sera 10 fois plus faible (5.10^{-4}).

Compléments et Exercices

Repères bibliographiques

Les considérations développées dans ce chapitre peuvent, sous l'aspect stochastique, être vues comme des variations sur les modèles markoviens. Ceux-ci sont analysés en détail dans les références [3, 6, 7, 11] Ceux que l'aspect technique de la Section 3 intéresse consulteront [8].

Exercices

3.1 - Montrer que la formule (3.2) peut se réduire à la "formule d'Erlang". Aurait-on pu le deviner sur le diagramme des états? (astuce: renuméroter les états, en faisant $j = N - n$).

3.2 - A partir des équations (3.1) et (3.2), montrer que:

$$E(n) = N - \frac{1 - P_0}{\alpha}$$

[Conseil: repartir du système (3.1), pour éviter tout calcul fastidieux]. Vérifier la formule (3.3) à partir des Q_k du Paragraphe 3.4.

3.3 - (Difficile) La formule (3.7) admet-elle une interprétation intuitive, permettant de la déduire sans calcul de (3.6) (et laquelle)? Trouver une récurrence permettant de calculer la "perte d'Engset" - formule (3.7). (Suggestion: adapter la récurrence de la formule d'Erlang).

3.4 - Que deviennent les formules (3.6) lorsque $N = R$. Que vaut B dans ce cas? (On parle de "trafic de Bernoulli). En donner une preuve directe.

3.5 - Expliquer (et démontrer) la tendance de (3.7) vers le cas M/M/R/R. Faire un modèle d'Erlang (M/M/R/R) de l'étage d'abonnés, supposant un trafic Poisson $N\lambda$ offert à R serveurs, et une population infinie. Estimer le gain en nombre de serveurs que permet la prise en compte de la taille de la population.

Chapitre 4

Calcul des Temps d'Attente

La terminologie est mal fixée. On distinguera l'*attente* proprement dite, du *séjour* (attente et service). Les anglo-saxons parlent de *queueing delay*, de *waiting time*, de *sojourn time*, mais ces mots n'ont pas le même sens chez différents auteurs. Pour notre part, nous essayerons d'adopter la terminologie suivante:

- Le **temps d'attente** est le temps qui s'écoule entre l'arrivée du client dans la file et l'instant où commence son service. On le notera w ou W.
- Le **temps de séjour** est le temps qui s'écoule entre l'arrivée dans la file et le départ après la fin du service. On le notera T : $T_k = w_k + s_k$ (séjour = attente + service du kième client). On symbolisera cette différence par le schéma:

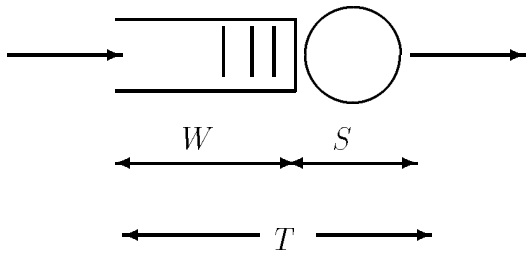


FIG. 4.1 – *Les composantes du temps de séjour*

Les "méthodes markoviennes" étudiées jusque là amènent aux probabilités d'état. Comment en déduire les temps d'attente?

4.1 La formule de Little

C'est sûrement la formule la plus célèbre de la théorie des files d'attente. On la rappelle ici, à cause de son importance. Elle relie les moyennes des temps aux moyennes des occupations.

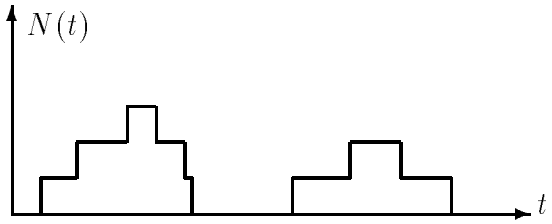


FIG. 4.2 – *Trace de l'occupation d'une file d'attente*

Considérons un "système", où entrent des clients. On peut tracer une courbe donnant le nombre de clients présents en fonction du temps (Figure 4.2).

Les segments ascendants représentent l'arrivée de clients, les segments descendants des départs. L'aire de la courbe (aire emprisonnée entre la courbe et l'axe des temps) admet deux méthodes d'estimation. Le calcul fait ici reprend l'approche du Chapitre 2.

- La "moyenne empirique" sur une longueur T est définie par la surface:

$$\begin{aligned} E_T(N) &= \frac{1}{T}\int_0^T N(t).dt \\ E(N) &= \lim_{T\to\infty} E_T(N) = \lim_{T\to\infty} \frac{1}{T}\int_0^T N(t).dt \end{aligned}$$

  avec $N(T)$ le nombre des clients arrivant entre 0 et T. L'intégrale n'est autre que $S(T)$, surface entre les abscisses 0 et T.

- La surface peut également être vue comme un empilement de briques; la brique i correspond à la contribution du client C_i: elle a une longueur x_i égale à cette contribution (selon le cas, il s'agira du "séjour, de "l'attente", du "service", voir remarques plus bas). D'où

$$S(T) = \sum_1^{N(T)} x_i$$

- Enfin, remarquons que par définition, le taux d'arrivées moyen $\lambda(T)$ dans la période $[0, T]$ est le quotient $N(T)/T$.

On rapproche ces trois formules, et

$$E_T(N) = \frac{S(T)}{T} = \sum_1^{N(T)} \frac{x_i}{N(T)} \times \frac{N(T)}{T}$$

Un passage à la limite s'impose. Dans les conditions "normales", chacun des termes possède une limite, égale respectivement à $E(N)$, λ et $E(x)$. Alors:

$$E(N) = \lambda E(x) \tag{4.1}$$

C'est la **formule de Little**. On notera sa généralité: on n'a rien imposé au "système", ni mode de fonctionnement, ni même une définition particulière, ni loi spécifique des variables aléatoires. Montrons sur quelques exemples la signification de la formule.

Le serveur d'une file Prenons comme "système" le serveur d'une file GI/G/1. $E(x)$ s'interprète comme le temps moyen de service. Le membre de droite est donc ρ. E(N) désigne le nombre moyen de clients dans le serveur, c'est à dire le taux d'occupation - dont on redécouvre la signification: $\rho = \lambda E(s)$.

Temps moyen d'attente A l'habitude, le système est la file d'attente. La formule relie le nombre moyen de clients en attente avec le temps moyen d'attente:

$$E(N_{\text{att}}) = \lambda W$$

On peut aussi adopter comme système l'ensemble (file+serveur). Alors la formule relie le temps de séjour avec l'occupation du système:

$$E(N_{\text{syst}}) = \lambda T$$

Définition plus complexe du système Le "système" peut être défini de façon moins simple: imaginons une file avec des priorités, la formule s'appliquera pour chaque flux séparément: le nombre moyen de client de type ℓ est le produit de leur taux d'arrivée par leur temps de séjour.

Application: M/M/1

- Le nombre moyen de clients dans le système est donné par la méthode classique.

$$E(N_{\text{att}}) = \sum n(1-\rho)\rho^n = \frac{\rho}{1-\rho}$$

  On en déduit le temps de séjour moyen

$$E(T) = \frac{1/\mu}{1-\rho}$$

- Pour l'attente, il faut connaître l'occupation de la file, qu'on n'a pas directement. Il est plus simple de déduire l'attente du séjour:

$$E(W) = E(T) - \frac{1}{\mu} = \frac{1}{\mu}.\frac{\rho}{1-\rho}$$

4.2 Distribution de l'attente: la File M/M/1

On se placera dans le cas simple de la file M/M/1. Les probabilités d'état sont connues:

$$P_n = (1-\rho)\rho^n \tag{4.2}$$

On gardera bien présent à l'esprit le fait suivant: *Les probabilités P décrivent l'état stationnaire, c'est à dire l'état que verrait un observateur externe, ou un processus d'échantillonnage indépendant. En d'autres termes, ce sont les proportions du temps que le système passe dans chacun des états.*

Heureusement pour le modélisateur, le trafic offert obéit au modèle de Poisson. On invoquera alors la propriété *PASTA*: Poisson Arrivals See Time Averages. Les clients arrivant (et entrant dans la file) selon un processus de Poisson observent l'état stationnaire du système, c'est-à-dire ici les P_n. On a déja rencontré cette propriété dans un cours précédent.

Le client qui arrive et voit n clients dans le système va attendre pendant n services successifs. Noter le cas particulier du client en service: le service est exponentiel, la durée restante a une distribution identique à l'exponentielle d'origine. L'attente est la somme de n exponentielles de même paramètre μ.

On a déja rencontré dans le Chapitre consacré aux rappels la distribution dite "Erlang-k", correspondant à une somme d'exponentielles. Soient $X_1, \cdots, X_n$ des variables exponentielles. La transformée de la somme des variables indépendantes est le produit des transformées, soit en notant $F_n(t)$ la distribution:

$$\tilde{F}_n(s) = \mathcal{L}(F_n) = \left[\frac{\mu}{\mu + s}\right]^n$$

On note $W(X)$ la transformée de Laplace de la distribution $W(t)$ du temps d'attente:

$$\begin{aligned} W(s) &= \sum_n F_n(s).(1-\rho)\rho^n \\ &= \sum_n \left(\frac{\mu}{\mu+s}\right)^n (1-\rho)\rho^n \\ &= \frac{(1-\rho)(\mu+s)}{\mu-\lambda+s} \end{aligned}$$

La consultation d'une table de transformées inverses, ou bien un traitement approprié, montre que:

$$W(t) = 1 - \rho e^{(\mu-\lambda)t} \qquad t \geq 0 \tag{4.3}$$

Cette approche n'a rien d'universel. Pour pouvoir l'appliquer, il faut connaitre la distribution des occupations aux instants des arrivées, et que le client en service ait la même distribution de service restant - ce qui est faux pour le système M/G/1.

4.3 Les formules de Pollaczek-Kintchine

4.3.1 Préliminaire: calcul du temps de service restant

Observons le serveur d'un système M/G/1, à un instant quelconque. Quel est le temps restant jusqu'à la fin du service en cours (temps nul si le serveur est au repos)?

Notons la différence avec le calcul du Chapitre 2, où il s'agissait de calculer le temps restant sachant que y secondes s'étaient déja écoulées. Ici, l'observation du serveur se fait indépendamment de celui-ci. Nous cherchons donc la moyenne temporelle du temps de service restant – en ignorant quand il a débuté.

Nous observons le système pendant une durée T. Des clients arrivent, sont servis, de telle sorte que le temps restant évolue de façon analogue au graphique de la Figure 4.3.

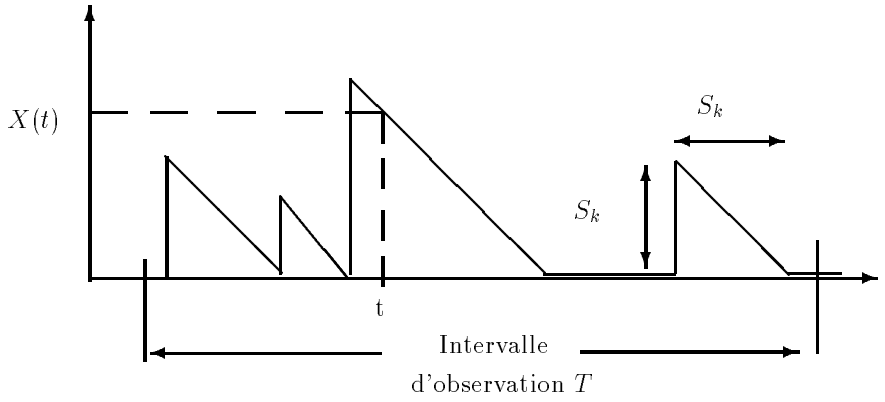


FIG. 4.3 – *Variation du Temps de service restant de la M/G/1*

Notons $X(t)$ le temps restant. Le graphe est composé de sauts initiaux (au début de service, le temps restant est le temps de service total), puis de pentes à 45°. On peut calculer la moyenne temporelle:

$$\frac{1}{T}\int_0^T X(t)\,dt = \frac{1}{T}\sum_1^{N(T)} \frac{1}{2}s_i^2$$

où $N(T)$ représente le nombre de clients servis dans l'intervalle, et s_i la suite des temps de service. Tous les clients sont statistiquement identiques, le passage à la limite donnera:

$$W_0 \equiv E(X) = \lim_{T\to\infty} \frac{1}{T}\sum_1^{N(T)} \frac{1}{2}s_i^2 \tag{4.4}$$

$$= \frac{1}{2}\lim_{T\to\infty} \frac{N(T)}{T}E(s_1^2) \tag{4.5}$$

$$= \frac{\lambda E(s^2)}{2} \tag{4.6}$$

Suivant Kleinrock [11], on note W_0 cette quantité.

4.3.2 La formule de Pollaczek-Khintchine

Nous nous plaçons dans le cadre de la file M/G/1. Un client qui arrive selon le processus de Poisson, et donc indépendamment de l'état de la file, va observer l'état stationnaire de celle-ci. Il trouve peut-être un client en service (ou bien le serveur inactif). Il observe le temps de service restant x correspondant. Il observe aussi n autres clients dans la file, qui seront servis avant lui. Son attente sera donc:

$$w = x + \sum_1^n s_i$$

Prenons les moyennes. Celle de x a été calculée ci-dessus. La moyenne de la somme sera $E(n)$ fois le temps moyen de service. Mais $E(n)$, nombre moyen de clients en attente, est

relié à $E(W)$, par la formule de Little ! D'où:

$$\begin{aligned} E(W) &= W_0 + E\left(\sum_1^n s_i\right) && (4.7)\\ &= W_0 + E(N)E(s) && (4.8)\\ &= W_0 + \lambda E(W)E(s) && (4.9) \end{aligned}$$

Et finalement la formule recherchée:

$$\begin{aligned} E(W) &= \frac{W_0}{1-\rho} && (4.10)\\ &= \frac{\rho E(s)}{1-\rho} \times \frac{1+C_s^2}{2} && (4.11) \end{aligned}$$

où on a introduit le *coefficient de variation* du temps de service:

$$C_s^2 = \frac{\sigma_s^2}{[E(s)]^2} = \frac{E(s^2) - [E(s)]^2}{[E(s)]^2} \qquad (4.12)$$

C'est la formule de Pollaczek-Kintchine.

Exemple: la file M/M/1 Pour l'exponentielle, on montre aisément que $E(s^2) = \frac{2}{\mu^2}$, soit $C_s^2 = 1$, et

$$E(W)_{M/M/1} = \frac{\rho E(s)}{1-\rho} \qquad \text{Rq : } E(s) = 1/\mu$$

Exemple 2: la file M/D/1 Evidemment, la variance du temps de service est nulle! Donc, $C_s^2 = 0$, et:

$$E(W)_{M/D/1} = \frac{\rho E(s)}{2(1-\rho)} = \frac{1}{2}E(W)_{M/M/1}$$

4.4 Les Approximations

On ne connait pas de résultat exact pour la file la plus générale; aussi les formules d'approximations sont-elles les bienvenues. On ne connait de bonnes approximations que pour les moyennes: variances ou quantiles n'en bénéficient pas !

4.4.1 La formule d'Allen

Pour ces approximations, de nombreuses approches ont été tentées: hypothèse "fort trafic", approximation "fluide", diffusions, etc. A l'heure actuelle, l'une des formules parmi les meilleures et les plus universelles est la **formule d'Allen-Cunnen**. Elle est valable pour un système GI/G/c. Posons:

- C_s^2 carré du coefficient de variation du temps de service;

- C_a^2 carré du coefficient de variation du temps inter-arrivées;
- $A = \lambda/\mu$ le trafic offert, et $\rho = A/c$ le taux d'occupation de chaque serveur;
- $C(A,c)$ est la probabilité d'attente de la M/M/c:

$$C(A,c) = \frac{\frac{A^c}{c!}}{\frac{A^c}{c!} + (1-\rho)\sum_{n<c} A^n/n!}$$

Alors,

$$\frac{E(W)}{E(s)} = \frac{C(A,c)}{c(1-\rho)} \times \frac{C_s^2 + C_a^2}{2}$$

C'est en fait la formule de la M/M/c qu'on corrige d'un facteur prenant en compte les coefficients de variation des lois d'arrivée et de service. On notera:

1. que la formule est **exacte** pour la file M/M/c;
2. qu'elle est **exacte** pour la file M/G/1.

4.4.2 Et pour un réseau?

L'exemple qui suit illustre le cas général dans lequel le problème est posé.

Soit à calculer le délai de bout-en-bout dans un réseau de communication. La connexion observée traverse une cascade de N multiplexeurs, où elle côtoie momentanément d'autres flux: Figure 4.4.

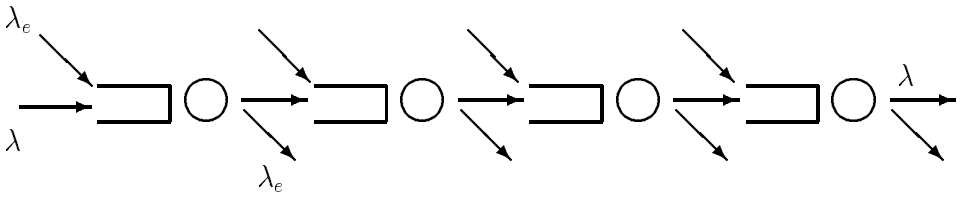


FIG. 4.4 – *Mise en cascade de serveurs représentant la traversée d'un réseau*

A chaque étage, on considère que le flux observé se mélange avec un flux incident indépendant de lui-même et des flux déjà rencontrés. L'approximation usuelle consiste:

- A considérer que chaque étage est "identique" au précédent, et en particulier que le flux pisté, d'intensité λ, conserve les mêmes propriétés statistiques (par exemple, Poisson, ou géométrique). Ceci est évidemment faux, le flux de sortie étant perturbé.
- A calculer le retard (moyenne, variance) de chaque étage (par un modèle M/G/1 approprié, par exemple).
- A estimer la moyenne et la variance de bout en bout par la somme des moyennes et variances individuelles.

Si l'addition des moyennes est légitime, celle des variances réutilise l'hypothèse contestable d'indépendance. En fait, il semblerait qu'on soit toujours pessimiste en utilisant cette approche.

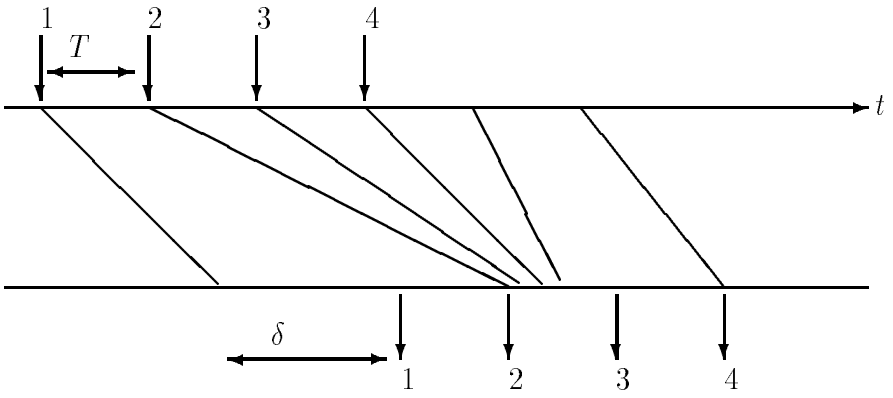
4.5 Calcul des Quantiles

C'est un grand mot pour qualifier une notion essentielle: les clients d'un système à attente sont sensibles non pas aux attentes moyennes, mais plutôt aux attentes "inadmissibles". On a déjà introduit à ce propos la notion de *quantile* au début du Cours.

4.5.1 Un exemple d'application

Soit à transmettre une communication *synchrone* sur un réseau de paquets. On prend le cas du réseau ATM comme exemple.

Les cellules synchrones sont émises aux instants T, $2T$, .., nT, et reçues avec un retard (retard W_k pour la cellule de rang k). Pour reconstruire le flux original périodique, on retarde d'une quantité δ la première cellule reçue, puis on lit les suivantes avec la période T:



Comment choisir δ? La cellule de rang k est émise au temps $(k-1)T$, reçue à $(k-1)T+W_k$, et sera "consommée" au temps $W_1 + \delta + (k-1)T$. Pour être assuré de sa présence, on devra faire:

$$(k-1)T + W_k \leq W_1 + \delta + (k-1)T$$

soit:

$$\delta \geq W_k - W_1$$

On ignore évidemment la valeur de W_1: le cas le plus défavorable consiste alors à faire $W_1 = 0$. Mais W_k peut être très grand, et la condition ne peut pas être remplie avec certitude. On demandera qu'elle soit "presque toujours" satisfaite, en la reformulant:

$$P\{W_k \geq \delta\} < \epsilon$$

avec pour ϵ une valeur très faible, 10^{-9} par exemple. Cela revient à choisir pour δ un quantile à 10^{-9} de la distribution de W.

4.5.2 Estimation approchée

Le calcul des quantiles est délicat, la distribution étant rarement connue explicitement, exception faite des systèmes M/M/c. On utilise assez souvent une formule empirique, connue

sous le nom de *Formule de Martin*:

$$\begin{cases} t_{90} = E(T) + 1.3\sigma_T \\ t_{95} = E(T) + 2\sigma_T \end{cases} \tag{4.13}$$

La formule est en fait très générale: elle s'appuie sur la ressemblance existant inévitablement entre toute "bonne" distribution et une *loi Gamma* (excluant les distributions non unimodales, ou non continues). On l'appliquera de préférence au temps de séjour, qui n'a pas de discontinuité à l'origine. Voici, pour l'exemple, le cas du quantile à 90% du temps de séjour de la file M/D/1 (c'est à dire le temps ayant 1 chance sur 10 d'être atteint ou dépassé):

Charge	Valeur Exacte	Approximation par la formule (4.13)
0.3	1.85	1.73
0.5	2.5	2.35
0.7	4.1	3.6
0.8	6.0	5.1
0.9	11.8	9.6

Compléments et Exercices

Généralisation de P-K : Formule de Takacs

Takacs a montré la formule suivante, valable pour la file M/G/1. Les moments successifs du temps de service y apparaissent.

$$\begin{cases} E(W^0) &= 1 \\ E(W^k) &= \dfrac{\lambda}{1-\rho} \displaystyle\sum_{i=1}^{k} \binom{k}{i} \frac{E(s^{i+1})}{i+1} E(W^{k-i}) \end{cases} \tag{4.14}$$

Par exemple, pour le temps moyen, $k = 1$:

$$E(W) = \frac{\lambda}{1-\rho} \frac{E(s^2)}{2}$$

Etc. Pour le séjour, on en déduit la forme suivante:

$$E(T^k) = \sum_{i=0}^{k} \binom{k}{i} E(s^i) E(W^{k-i})$$

(Par exemple, en $k = 1$: $E(T) = E(s) + E(W)$).

Repères bibliographiques

L'exemple de compensation des variations de délai de transport des paquets, emprunté à l'ATM, est développé dans les cours correspondants, et notamment à l'occasion du traitement de l'AAL type 1.

Pour les formules d'approximation, on se reportera à l'ouvrage de A.E. Allen.

Exercices

4.1 - Retrouver directement la formule (4.3) d'après la distribution (4.2). Calculer la moyenne et la variance du temps d'attente d'après la transformée de Laplace. Quelle est la probabilité d'une attente nulle (toujours d'après la transformée).

4.2 - Appliquer la formule de Little au cas du système M/M/c, en calculant le nombre moyen à partir des probabilités d'occupation.

4.3 - Calculer le temps moyen d'attente d'une file M/H2/1 (temps de service hyperexponentiel), puis M/Ek/1 (temps de service distribué selon une Erlang-k).

4.4 - Ecrire un programme de simulation permettant de mesurer le quantile d'un système M/D/1. Vérifier alors les formules de Martin.

Chapitre 5

Modèles de Programmes et de Processeurs

Un système informatique moderne est une organisation complexe, à la fois de par l'architecture matérielle et de par le type de programmes qui y sont traités. Même "centralisé", il comporte de nombreuses entités distinctes (contrôleurs, disques, coprocesseurs, etc), qui coopèrent.

L'agencement harmonieux et efficace de tous ces éléments impose des mécanismes d'ordonnancement des tâches. La modélisation doit impérativement prendre en compte ces particularités de fonctionnement, et doit aider à leur mise au point.

5.1 Un exemple de système temps-réel

5.1.1 Un serveur de messagerie

Considérons une machine, utilisée par les services de messagerie vocale (un *serveur vocal*): répondeurs, boite aux lettres, jeux, annnonces, etc. Fonctions du serveur: stocker et gérer les messages, les restituer sur demande. Le dialogue usager-service est vocal ou utilise les touches du combiné.

La figure 5.1 donne un exemple d'architecture d'une telle machine.

Chaque communication se conforme approximativement au schéma suivant:

- sur détection d'un appel du réseau, affectation d'un processeur;
- dialogue avec l'usager (envoi d'un message, écoute de la réponse; action suivante; etc);
- le "service" désigne un processeur particulier chargé du pilotage du dialogue; c'est lui qui contient le service effectif, on parle de *dérouleur de script*;
- consultation de la BD où sont stockés les messages vocaux de l'usager, pour les délivrer;
- etc.

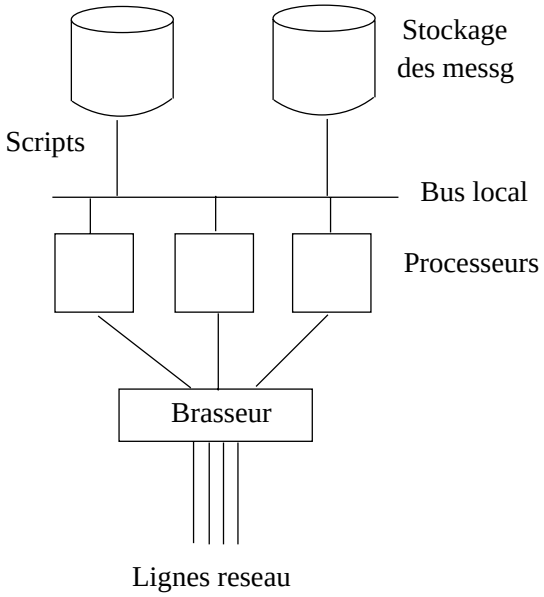


FIG. 5.1 – *Architecture d'un serveur vocal*

Chaque communication est décrite par un "script", dans lequel chaque action correspond à un travail d'une des entités du serveur. Le script contient de nombreux branchements, correspondant aux décisions de l'usager ou aux fausses manœuvres.

On pourrait expliciter encore le schéma ci-dessus, en y faisant apparaître les actions élémentaires des processeurs, les lectures et écritures en mémoire, par exemple. On voit combien le déroulement d'une communication est complexe, et ressemble en fait au traitement des programmes d'un système informatique multiprogrammé. Les questions que cette description amène concernent en premier lieu les possibilités et les performances de la machine – combien de communications peut-elle accepter, quel est son temps de réponse. Elles concernent aussi les choix de conception et de dimensionnement – nombre de processeurs, taille de mémoires, organisation du traitement des requêtes, etc.

Pour y répondre, on bâtira un modèle du fonctionnement de la machine. Ce sera un *"réseau de files d'attentes"*, chaque serveur y représente une ressource active (processeur) devant lequel s'accumuleront les clients; chaque communication est représentée par un client qui chemine entre les ressources au gré des différentes étapes du script.

A ce stade, c'est la question de *l'ordonnancement* des tâches présentes devant les processeurs qui va nous occuper.

5.1.2 Caractéristiques des tâches

L'examen de l'exemple précédent met en valeur les caractéristiques du fonctionnement des systèmes informatiques temps-réel.

- Plusieurs "usagers" sont actifs simultanément - même si un seul reçoit à un instant le service du processeur. Il y a ainsi meilleure utilisation du temps processeur (celui-ci

pouvant se consacrer à d'autres tâches). On parle de systèmes **multiprogrammés**.

- Plusieurs tâches sont généralement en attente devant chaque serveur (processeurs, disques, etc). Plutôt que de laisser le hasard seul décider de l'ordre des services, on préfère définir des mécanismes d'ordonnancement – ce chapitre y est consacré.
- Notamment, on utilisera des **mécanismes d'interruptions**, pour accélérer la réponse de périphériques lents ou saisir des événements fugaces, du **polling**, pour partager équitablement la puissance, etc.

Remarque: La mémoire rapide est partagée entre les usagers actifs. On introduit la notion de **mémoire paginée**. Le plus souvent, les données utilisées occupent des positions proches les unes des autres, et la lecture d'un élément de donnée se fait en mémoire centrale. Plus rarement, il faut aller lire sur disque un bloc de données. La fréquence de ces *défauts de pages* augmente quand la mémoire allouée diminue. Ce point limite le nombre d'usagers simultanés (degré de multiprogrammation).

5.2 Les mécanismes d'ordonnancement

Considérons la file d'attente simple (M/G/1, si on veut). A la fin d'un service, comment le serveur choisit-il son prochain client? Les possibilités qu'on décrit s'appliquent à un "vrai" système à file et serveur, aussi bien qu'à un modèle conceptuel de fonctionnement complexe. Description des disciplines habituellement répertoriées:

- FIFO ("First-In, First-Out") c'est à dire "premier entré, premier servi" - le mécanisme habituellement considéré, qu'on a toujours supposé jusqu'ici.
- LIFO ("Last-In, First-Out"): le dernier arrivé sera servi le premier. Intérêt: diminue l'effet pervers des impatiences pendant le service, cf. ESS4; utilisé aussi en "gestion des stocks" (même motif: effet du vieillissement);
- au hasard: cas où on ne peut garder trace de l'ordre d'arrivée;
- Round Robin: on définit un "quantum de temps de service"; le serveur traite à tour de rôle chaque client pour une portion du temps égale au quantum; il accomplit ainsi une sorte de cycle sur les clients en attente. Dès qu'un client a reçu, en plusieurs fois, le service qu'il réclame, il quitte le serveur;
- Processor Sharing (PS): version asymptotique du "round robin": le quantum tend vers 0; si n clients sont présents entre t et $t+\Delta t$, chacun d'eux reçoit un service élémentaire égal à $\Delta t/n$ pendant ce temps (avec $\Delta t \to 0$).

Ces mécanismes ordonnancent les choix sans discrimination; pour les suivants, on attache à chaque client une *classe*, et le choix s'appuie sur cette classe. C'est une véritable discrimination, qui permet de tirer parti des caractéristiques des clients.

- priorité simple HOL ("Head of the Line") les clients de classe 1 viennent se ranger en file devant ceux de classe 2 (extension immédiate à N classes); moyen usuel de favoriser un flux;

- Serveur Cyclique ("Polling"): dans ce mécanisme, chaque classe se voit allouer une file particulière, que le serveur explore de façon cyclique;
- Tout autre algorithme peut être imaginé. Par exemple, choix basé sur le temps de service demandé (les temps courts d'abord).

On parle de "classe" pour différencier les flux: selon leur priorité, selon la distribution des temps de service, etc. On distingue aussi les priorités **préemptives**. Dans ce cas, le client prioritaire interrompt le service en cours pour être servi sans attente. Le client interrompu reprendra son service après celui du client interrompant. La prise en compte des événements fugaces justifie cette particularité (coûteuse en temps de calcul, le plus souvent).

5.3 Notion de Système Conservatif

On conçoit que dans certains cas le choix du client soit indifférent au serveur, c'est à dire que ce choix ne change pas le travail total qu'il devra traiter. Conséquence: à chaque instant, le serveur est capable de comptabiliser la quantité de travail qui lui est offerte (on parle de son "travail restant", "unfinished work" en anglais): temps de service restant à fournir au client en cours, augmenté de la somme des services des clients en attente à cet instant. On peut (presque) toujours évaluer cette quantité à un instant fixé, mais elle n'a de sens que lorsque le travail restant ne dépend pas de la décision prochaine! Pour donner une autre signification à cette quantité: soit τ le travail restant à l'instant t. Stoppons à ce moment le processus d'arrivées, et observons le comportement aléatoire du système: le serveur va travailler continûment jusqu'à $t+\tau$, quelles que soient les décisions d'ordonnancement prises.

On appelle système conservatif (work conserving) un tel système où le travail restant ne dépend pas de la discipline de choix que pratique le serveur.

Contre exemples: système multi-queues avec temps de basculement; système avec impatience; système où le temps de service est fonction de l'attente (vieillissement), etc. Pour la plupart de ces systèmes, on ne peut même pas calculer ce travail à un instant donné. Le lecteur est encouragé à expliciter, dans chacun de ces cas, les raisons rendant le serveur non-conservatif.

Formule de la somme pondérée

Il va de soi que, si le travail restant ne dépend pas de l'ordonnancement, le temps d'attente, lui, en dépendra. La propriété de conservation marche "vue du serveur". Pour les clients, on va voir de quoi il retourne.

Remarque préliminaire:
Une file d'attente reçoit des flux différents, on note λ_j l'intensité du flux de type j et W_j l'attente moyenne qu'il subit. On note λ le flux total. Il est possible de mesurer le temps d'attente sans tenir compte des classes des clients. Notons $\bar{W}$ la quantité correspondante. C'est une somme pondéré, la pondération faisant intervenir la proportion de chaque flux:

$$\bar{W} = \sum \frac{\lambda_j}{\lambda} W_j$$

L'importance de la notion de temps moyen pondéré réside dans le fait que peut ne pas savoir mesurer une autre quantité – supposons par exemple un processus de mesure incapable de

différencier entre les classes.

Théorème. Imaginons un serveur devant une file d'attente traitée selon un mécanisme de priorité quelconque, conservatif et non préemptif. On a la loi de conservation suivante:

$$\sum \rho_i W_i = \frac{\rho}{1-\rho} W_0 = Cste \tag{5.1}$$

L'indice i court sur les classes de clients; ρ_i représente la charge de la classe i: $\rho_i = \lambda_i E(s)_i$, W_i le temps moyen d'attente qu'elle subit; $\rho = \sum \rho_i$ la charge totale; et W_0 désigne la moyenne du temps de service restant (on l'a déja rencontré au Chapitre consacré à la formule de Pollaczek):

$$W_0 = \sum_i \lambda_i \frac{E(s_i^2)}{2} = \sum_i \rho_i \frac{E(s_i^2)}{2E(s_i)} \tag{5.2}$$

La formule signifie que l'expression ne dépend pas de la discipline; bien noter que la somme des temps d'attente est pondérée par les ρ_i et non par les λ_i comme pour $\bar{W}$.

Exemples: Cas particuliers

- Rappel: loi exponentielle: $E(s^2) = 2E(s)^2$, d'où $W_0 = \sum \rho_i Es_i$; durée constante: $W_0 = \sum \rho_i Es_i/2$;

- Supposons une seule classe de clients (pas de priorité, etc). Vérifier alors que la loi est identique à la formule de Pollaczek-Khintchine.

- Supposons 2 flux, de temps de service différents, mais sans priorité. Leur temps d'attente est identique. D'où

$$W_1 = W_2 = \frac{1}{1-\rho} \sum \lambda_i \frac{E(s_i^2)}{2}$$

  on redémontre dans ce cas la formule PK.

- Supposons une priorité, mais des *temps moyens* identiques. La formule se réécrit, en faisant sortir le temps de service moyen, de telle façon que le temps moyen pondéré y apparaît:

$$\overline{W} = \sum \frac{\lambda_i}{\lambda} W_i = \frac{\rho}{1-\rho} \sum \frac{\lambda_i}{\lambda} \frac{E(s_i^2)}{2E(s_i)}$$

  (à vérifier).

Preuve de la loi de conservation: on observe le système, à t on trouve $n(j)$ clients de classe j (=1,...P); le client en cours de service réclame encore x_0. Le travail restant est donc:

$$U(t) = x_0 + \sum_j \sum_{i=1}^{n(j)} x_{i,j}$$

On a noté $x_{i,j}$ le temps demandé par le client i de la classe j. En prenant la moyenne,

$$E(U) = W_0 + \sum_j E[\sum_i x_{i,j}]$$

la moyenne restant est le nombre moyen de clients de classe j fois le temps moyen de service. Grâce à Little, on peut relier le nombre moyen au temps d'attente, et donc:

$$E(U) = W_0 + \sum_j \rho_j W_j$$

Maintenant, le travail restant est indépendant du mécanisme d'ordonnancement (système conservatif). On prend la valeur pour FIFO: dans ce cas, tous les W_j sont égaux et $E(U) = W_0/(1-\rho)$

Priorité entre clients identiques

On suppose maintenant des clients aux caractéristiques identiques: même distribution du temps d'attente (non plus seulement même moyenne). Quel que soit le mécanisme de choix, pourvu que le système soit conservatif, le choix du serveur n'a pas d'influence sur l'occupation totale de la file; en d'autres termes, *la distribution du nombre total de clients en file ne dépend pas de la discipline.*

En particulier, le nombre moyen de clients est indépendant de la discipline.

Donc (Little) le temps moyen d'attente ne dépend pas de la discipline (mais la distribution de l'attente en dépend !).

Remarque: vérifier que ce résultat est bien en accord avec la relation de conservation (1). Exemple: FIFO, LIFO, Hasard donnent le même temps moyen d'attente.

5.4 La discipline HOL

Dans toute la suite, "priorité 1" plus forte que "priorité 2". On reprend le cas général (services différents), serveur de type conservatif, N classes.

On note σ_k la somme des charges des classes 1 à k (inclus); rappelons que la classe 1 est prioritaire sur 2, ..., $N-1$ est prioritaire sur N. σ_k est la charge que "voit" la classe k (qui double k+1, k+2,.. et ne les voit pas). Remarque: la classe 1 voit quand même la classe 2, en effet un client 2 peut géner un client 1 qui arrive pendant son service. On montre que:

$$W_k = \frac{W_0}{(1-\sigma_k)(1-\sigma_{k-1})} \qquad (5.3)$$

$$\text{avec} \qquad \sigma_k = \sum_{i=1}^{k} \rho_i$$

où W_0 a la même expression que plus haut. Exemple: 2 classes. Vérifier que la loi de conservation est encore valide. (Discussion: caractère sécurisant de la discipline pour la classe 1, même si $\rho_1 + \rho_2 > 1$).

Preuve: pour une discipline non préemptive,

$$W_j = W_0 + \sum_{i<j} Es_i(\overline{N}_{ij} + \overline{M}_{ij})$$

W_j est le temps moyen de la classe j, l'indice i court sur les classes ($i = 1, ...N$); $\overline{N}_{ij}$ représente la moyenne du nombre de clients de la classe i déja présents à l'arrivée du client de type j **et qui seront servis avant lui**; $\overline{M}_{ij}$ représente le nombre moyen de clients de la classe i arrivant pendant l'attente du client de type j **et qui seront servis avant lui**. Pour notre discipline,

- $\overline{N}_{ij} = \lambda_i W_i$, pour i=1,...,j et nul pour i=j+1,.. N (notre client test double les clients des classes moins prioritaires);
- $\overline{M}_{ij} = \lambda_i W_j$, pour i=1,...,j-1 et nul pour i=j,...,N (seules les classes de rang strictement plus prioritaires vont le doubler).

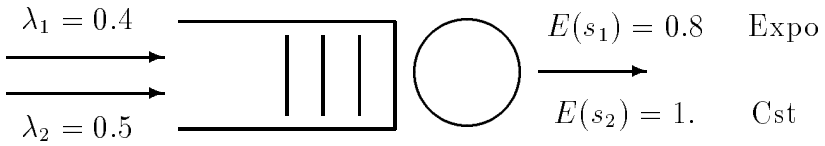
Finalement:

$$W_j = W_0 + \sum_{i=1}^{j} \rho_i W_i + \sum_{i=1}^{j-1} \rho_i W_j$$

soit
$$W_j(1 - \sigma_{j-1}) = W_0 + \sum_{i=1}^{j} \rho_i W_i$$

A partir de cette dernière formule, on calcule d'abord W_1, puis W_2, etc.

Un exemple numérique



On a $\rho_1 = 0.32$, $\rho_2 = 0.5$ (charge totale 0.82). Le paramètre $W_0 = 0.506$

- sans priorité, on a $\overline{W} = W_1 = W_2 = \frac{W_0}{1-\rho} = 2.8$
- priorité au flux 1, on a $W_1 = \frac{0.506}{1-0.32} = 0.74$, et $W_2 = \frac{0.506}{(1-0.32)(1-0.82)} = 4.13$ soit $\overline{W} = 2.35$

(Rappel: $\overline{W}$ représente la moyenne que l'on mesurerait sur l'ensemble des clients sans distinction de classe).

5.5 Le Serveur Cyclique ("Polling")

5.5.1 Introduction

Modèle de serveur très utilisé, sûrement l'un des plus importants par l'étendue de ses usages. Description: N tampons d'attente sont visités périodiquement par un serveur. Celui-ci accomplit un itinéraire fixé: il visite la file 1, puis la file 2, etc, jusqu'à la file N, puis retourne vers la 1, et ainsi de suite.

Le passage d'une file à la suivante coûte un temps non nul (le "temps de basculement" - "changeover time"). Lors de la visite à la file i, le serveur y passe un temps nul si elle est vide. Sinon, il peut vider totalement la file avant de passer à la suivante (on parle de *service exhaustif*, le serveur ne quitte la file que lorsqu'elle ne contient plus aucun client); ou bien il sert un nombre maximum l_k lors de la visite à la file k (*service limité*, le cas particulier le plus intéressant étant celui où $l_k = 1$); ou encore il sert uniquement les clients déjà présents à son arrivée (*service à porte*, on parle plus souvent de *gated service*); etc.

Pour montrer l'intérêt du modèle et la variété des situations, et aussi pour illustrer les variantes du mécanisme, voici une liste non limitative d'applications possibles.

- Liaison des périphériques au processeur central d'un calculateur. L'UC va scanner périodiquement l'état des périphériques. Le cas échéant, elle déclenche une opération d'E/S. Autre possibilité: interruptions. Chaque méthode a ses avantages (à l'avantage de l'interrogation cyclique: elle évite les complications liées à la nécessité de sauvegarder le contexte des tâches interrompues, et qu'elle simplifie la logique du périphérique).

- Liaison multipoint vers des terminaux. Exemple: une ligne tph louée, des terminaux de données, et un frontal; celui-ci envoie l'ordre de parole "roll-call polling", ou bien le droit circule: "hub polling" ou "token passing".

  Remarques: les topologies physiques sont très différentes, mais équivalentes.

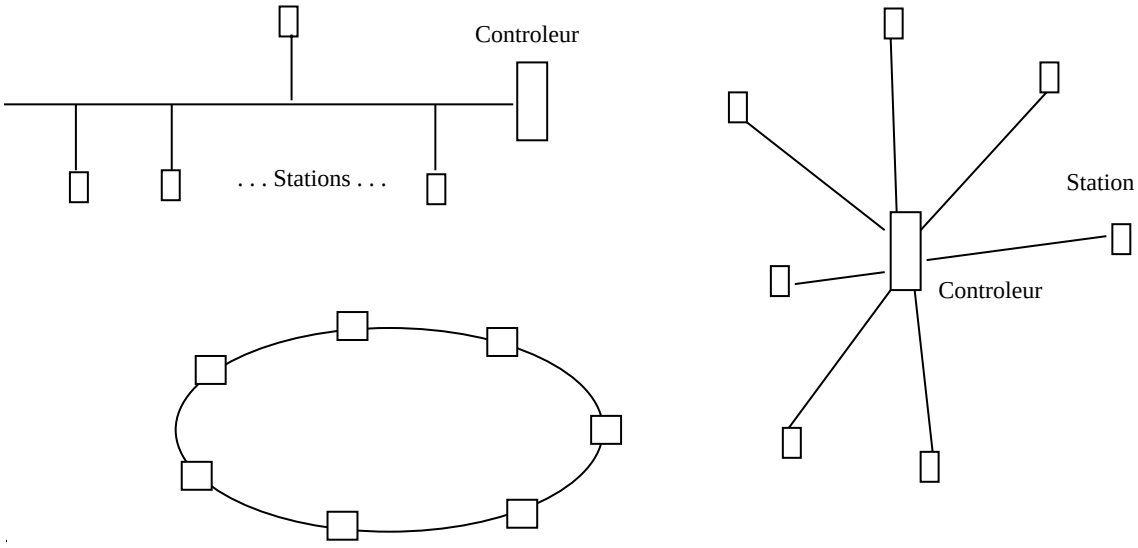


FIG. 5.2 – *Des topologies équivalentes de "polling"*

HDLC est bien adapté à ce type d'usage (utilisation du bit P/F).

- L'anneau à jeton IEEE802.5 "token passing": c'est l'exemple parfait de serveur cyclique.

- Inspection de machines par un opérateur itinérant. Des machines sont en fonctionnement continu sur des sites séparés, un opérateur visite continûment ces sites. S'il trouve des machines en panne, il les répare avant de passer au site suivant.

Analyse mathématique

On notera 1, 2, ... N les files servies, λ_i l'intensité du flux (poissonnien) offert à chacune, θ_i le temps de basculement (passage de la file i à la file $i+1$); s_i le temps de service à la file i - de moyenne $E(s_i)$; on adopte la convention $N+1=1$ (!). On notera les trafics: $\rho_i = \lambda_i E(s_i)$, et $\rho = \sum \rho_i$.

Le temps de cycle

Le temps de cycle (intervalle entre deux passages successifs à une même file) est le paramètre-clé. On le note C. On détermine sa valeur par le raisonnement suivant: en un temps de cycle il arrive en moyenne $\lambda_i C$ clients à la file i. Pour qu'un régime permanent existe, il faut qu'il en soit servi autant (en moyenne). C'est à dire que le cycle C doit avoir une durée telle qu'elle permette au serveur de servir (en moyenne) autant de clients qu'il en arrive. Le temps de cycle est la somme des temps de basculement et du temps passé dans chaque file - donné par la remarque précédente. Donc:

$$C = \sum \theta_i + \sum \lambda_i E(s_i) C$$

$$C = \frac{\sum \theta_i}{1 - \sum \rho_i} = \frac{\sum \theta_i}{1 - \rho} \tag{5.4}$$

L'équation n'est valable que pour un régime stable (aucune des files d'attente n'est saturée).

Le temps de basculement

Que se passerait-il si on faisait θ_i=0? On voit qu'alors $C = 0$. En réalité, pour que le système soit stable, il faut qu'il existe des instants où les files sont vides. Mais alors si le temps de basculement est nul, en un temps infinitésimal le serveur effectuera une infinité de cycles: cela correspond bien à un cycle moyen nul, mais fait perdre tout sens physique au modèle. Il existe des exemples où θ_i peut être difficile à estimer. Néanmoins on le prendra non nul, pour éviter les complications.

Dans les exemples donnés ci-dessus, θ_i pourra être le temps d'interrogation d'un terminal au repos (interrogation + réponse vide).

Le temps de réponse

Le calcul du temps de réponse vu d'une file est beaucoup plus complexe. Il n'existe pas de formule exacte; la complexité provient de la corrélation qui s'installe inévitablement entre toutes les files. On fait appel à des approximations.

La plus simple consiste à représenter tout le système par une file M/G/1, comme si toutes les files étaient confondues. Mais il faut tenir compte du caractère cyclique, qui oblige à attendre le serveur; celui-ci peut être dans l'une quelconque des files à l'arrivée du client test. Des analyses plus ou moins convaincantes mais très bien validées par la simulation conduisent à la formule suivante pour le délai moyen W, dans le cas d'un système symétrique (tous les ρ_i égaux), et d'un service exhaustif ou "gated":

$$W_{\text{Exhaustif}} = \frac{C}{2}.(1 - \frac{\rho}{N}) + \frac{\lambda E(s^2)}{2(1-\rho)} \tag{5.5}$$

$$W_{\text{Gated}} = \frac{C}{2}.(1 + \frac{\rho}{N}) + \frac{\lambda E(s^2)}{2(1-\rho)} \tag{5.6}$$

Le terme λ est le taux d'arrivée total ($\lambda = \sum \lambda_i$). Le teme C représente le temps de cycle, donné par (5.4). La formule fait apparaître un premier terme de comportement cyclique. Le client qui arrive doit attendre que le serveur vienne servir sa file (en moyenne, cela correspond à un "demi-tour" $C/2$, sauf s'il s'y trouve déjà, d'où le terme correctif). le serveur doit accomplir un "demi-tour" avant de rejoindre la file, sauf s'il s'y trouve déjà, ce qui augmente d'autant l'attente). Le second terme est simplement celui d'une file M/G/1 équivalente au système complet.

5.5.2 Un exemple numérique

Une compagnie de réservation aérienne gère un ensemble de terminaux distants, reliés à un serveur central par une liaison de "polling". On prendra l'application numérique suivante:

- Il y a $N = 100$ terminaux.
- La vitesse de propagation du signal: 80 000 km/s (ce chiffre tient compte de la présence de répéteurs, commutateurs, etc).
- Distance moyenne entre les stations et le site central 400 km. Il faut 5 ms pour atteindre chaque station.
- Débit de la ligne 14400 b/s (débit qui peut sembler relativement faible aujourd'hui, mais au moins "conservatif").
- Paquets d'interrogation de longueur fixe 100 bits; le temps d'émission, à 14 kb/s, est 7 ms.
- Paquets de réponse active exponentiels, longueur moyenne 10 000 bits, on supposera pour simplifier q'une réponse nulle prend un temps nul. On calcule immédiatement $E(s) = 700$ millisecondes.

(Remarque: on écarte les problèmes posés par une transmission "réelle", tels que les temps de synchronisation). Supposons un polling classique: la station-maître envoie à chaque

terminal à tour de rôle un signal de polling: il faut émettre le paquet (temps de 7 ms), puis lui laisser le temps d'arriver à la station, soit 5 ms: le temps de basculement est $\theta_i = 12$ ms. On calculera les cas du service exhaustif et à porte: le terme correctif est d'importance négligeable dans notre application. D'où l'application numérique suivante:

ρ= 0	Cycle = 1.2 s	E(délai) = 0.6 s
ρ= 0.4	Cycle = 2. s	E(délai) = 1.47 s
ρ= 0.8	Cycle = 6. s	E(délai) = 6.8 s

5.5.3 D'autres politiques

On a supposé un polling où le serveur vide chaque file avant de passer à la suivante, ou bien sert seulement les clients déjà arrivés – ici, ces deux politiques sont équivalentes. L'inconvénient majeur de ces mécanismes réside dans leur mauvaise résistance aux surcharges: il suffit qu'une file soit saturée pour paralyser toutes les autres. On peut utiliser d'autres mécanismes, par exemple limitation à l_k du nombre servi à chaque passage dans la file k. Le cas le plus simple est celui où chaque passage du serveur ne prélève qu'un client au maximum. Alors, même si une file sature ($\rho_i > 1$), les autres files voient toujours revenir le serveur au bout d'un temps fini. Inconvénient: l'importance du temps de basculement augmente, puisqu'un basculement précède toujours un service (si $l_k = 1$).

Condition de stabilité

Rappelons que le temps de cycle C est, là encore, donné par la formule (5.4). Une analyse exacte, mais assez complexe, donne le résultat suivant [1].

Notons λ_j, S_j, ρ_j les paramètres (taux d'arrivée, temps moyen de service et charge de la file j). Notons encore $\rho = \sum \rho_i$ la charge totale et $\Theta = \sum \theta_i$ le temps de cycle à vide. Nous nous intéressons au fonctionnement d'une file, de rang j. Supposons toutes les autres files $k \neq j$ stables. La file j sera stable (c'est à dire, par exemple, son remplissage n'augmentera pas indéfiniment) si et seulement si la condition suivante est respectée:

$$\lambda_j < \lambda_j^{\max} = \frac{l_j(1-\rho+\rho_j)}{\Theta + l_j S_j} \tag{5.7}$$

La condition de stabilité globale sera que pour chacune des files, l'inégalité ci-dessus soit respectée. Imaginons par exemple un système entièrement symétrique (tous les ρ_j, etc. égaux). On réécrit cette condition sous la forme

$$\rho < \frac{N l_j S_j}{\Theta + N l_j S_j}$$

Dans le cas où une ou plusieurs des files sont instables, les conditions ci-dessus ne s'appliquent pas directement.

Dans le cas général, la discussion de ces conditions se révèle délicate, la stabilité d'un flux dépendant de l'ensemble des paramètres qui définissent le système. Dans le cas particulier de deux flux, on peut donner une analyse plus complète, illustrée par le graphique Figure

1. Voir P.J. Kuehn, Multiqueue systems with nonexhaustive cyclic service, *Bell System Technical Journal*, vol. 58, N.3, mars 1979

5.3. Chacun des flux $i = 1, 2$ se voit définir une limite, qui dépend du volume de l'autre flux. En dessous des deux droites limites, le système est stable. Le point d'intersection joue un rôle central dans l'analyse. Ses coordonnées sont $(\lambda_1^*, \lambda_2^*)$:

$$\lambda_i^* = \frac{l_i}{\Theta + l_1 S_1 + l_2 S_2}$$

Plaçons nous dans la région $(\lambda_1 < \lambda_1^*, \lambda_2 > \lambda_2^{\max})$. La file 2 est instable mais la file 1 reste stable. En effet, la file 2 instable, consomme le service maximum qu'elle peut recevoir. Son trafic *écoulé* n'est plus donné par $\lambda_2 S_2$ (la file 2, de capacité inévitablement finie, subit un taux de perte important). Dans le pire cas pour la file 1, chaque bloc de l_1 clients qui est servi est séparé du suivant par une absence pour servir un bloc à l'autre file. l_1 clients sont servis en un temps $\theta_1 + l_1 S_1 + \theta_2 + l_2 S_2$, soit un taux maximum de $l_i/(\Theta + l_1 S_1 + l_2 S_2)$, qui est précisément λ_1^*: la file 1 est donc bien stable ici.

Une analyse semblable vaut pour la région $(\lambda_1 > \lambda_1^{\max}, \lambda_2 < \lambda_2^*)$. Finalement, dans la région $\lambda_1 > \lambda_1^*, \lambda_2 > \lambda_2^*)$, les deux files sont instables.

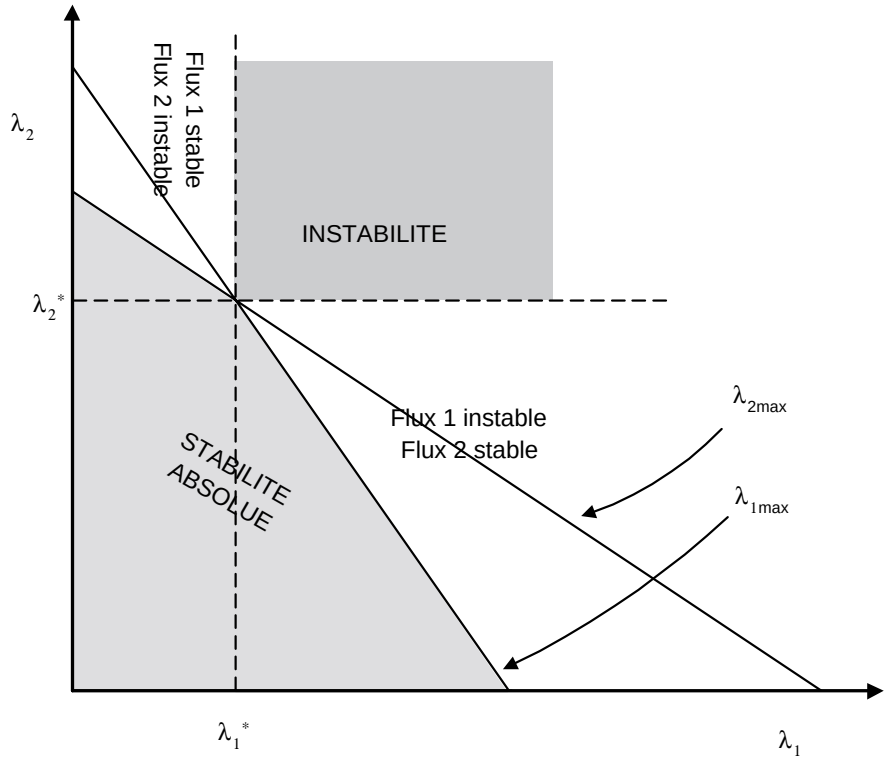


FIG. 5.3 – *Zones de stabilité du système à accès limité, 2 flux*

Pour comprendre l'intérêt de ce mécanisme, il faut le comparer avec la situation résultant d'un mécanisme de type exhaustif quand la charge augmente. L'augmentation d'un flux provoque automatiquement la saturation de toutes les files, dès que la condition globale $\sum \rho_i < 1$ est dépassée.

Analyse des délais

L'analyse proposée dans la référence qu'on utilise ici est assez lourde à mettre en œuvre, mais est très générale. Elle permet de traiter le cas de systèmes non symétriques, où les λ_i et ρ_i diffèrent. On décompose les cycles selon qu'un client y est servi ou non, on note c'_j (resp. c''_j) le temps moyen d'un cycle sans (resp. avec) service dans la file j. On calcule la moyenne et la variance de ces temps de cycle, en se plaçant dans la file j. Pour les moyennes:

$$c'_j = \frac{\Theta}{1-\rho+\rho_j} \tag{5.8}$$

$$c''_j = \frac{\Theta + Es_j}{1-\rho+\rho_j} \tag{5.9}$$

Pour les variances:

$$\mathrm{Var}(c'_j) = \sum_{i\neq j}(\lambda_i c'_j Es_i^2 - \rho_i^2 c_j'^2) \tag{5.10}$$

$$\mathrm{Var}(c''_j) = \sum_{i\neq j}(\lambda_i c''_j Es_i^2 - \rho_i^2 c_j''^2) + Es_j^2 - (Es_j)^2 \tag{5.11}$$

Et finalement,

$$W = \frac{\mathrm{Var}(c'_j) + c_j'^2}{2c'_j} + \frac{\lambda_j\left(\mathrm{Var}(c''_j) + c_j''^2\right)}{2(1-\lambda_j c''_j)} \tag{5.12}$$

L'application numérique du paragraphe précédent est laissée en exercice au lecteur.

Compléments et Exercices

Repères bibliographiques

Les systèmes avec priorités sont étudiés dans les ouvrages classiques [6, 11]. Pour le cas des approximations du "polling", voir [16].

L'étude des modèles de calculateurs a donné lieu à une littérature fournie. Voir [9, 12], et [8] pour le cas particulier des systèmes téléphoniques temps-réel.

Exercices

5.1 - Reprendre l'exemple en fin de la Section 3, en inversant le rôle des priorités. Que remarque-t-on? Pour une généralisation, voir [11], tome 2.

5.2 - Dans un gros calculateur, un processeur spécialisé gère les relations entre l'UC et les périphériques (le *processeur d'échanges*). Il explore, cycliquement, la file des tâches de chaque périphérique, et le cas échéant traite les tâches en attente. Le temps de service est exponentiel, de moyenne 15 ms. La commutation d'une file de tâches vers la suivante consomme un temps (constant) $t = 10$ ms. Les tâches arrivent à chaque périphérique, avec un taux λ_i (Poisson). Application numérique, N=5 périphériques, $\lambda_1 = \lambda_2 = ... = 10/s$.

a) Quel est le temps moyen d'un cycle d'exploration?

b) Quelle est l'attente moyenne d'une tâche?

c) On garde les λ_i constants, sauf pour i=1. On augmente λ_1. Quelle valeur ne doit-on pas dépasser?

5.3 - Le processeur d'une machine temps-réel reçoit 3 types de tâches. Elles se rangent dans 3 files spécialisées. Les caractéristiques en sont les suivantes:

- Tâches "urgentes"; taux d'arrivée = 15/s, durée de service *constante* 10 ms.
- Tâches normales; taux d'arrivée = 20/s, durée de service *exponentielle*, de moyenne 20 ms.
- Tâches "de fond"; taux d'arrivée = 5/s, durée *exponentielle*, de moyenne 40 ms.

a) Supposons les tâches traitées en un seul flot (sans priorité). Quel est le temps moyen d'attente?

b.1) On ordonnance (tâches urgentes prioritaires, puis normales, et tâches de fond en dernière priorité), selon un schéma non préemptif ("head of the line"). Que deviennent les attentes?

b.2) Calculer, dans ce cas, l'attente moyenne (i.e. toutes tâches confondues).

b.3) Combien y a t'il de tâches en attente dans chaque file (en moyenne)?

5.4 - Un réseau de transmission de données utilise des lignes de transmission à 64 kilobits/seconde. Il reçoit et transmet 2 types de blocs d'information (des "paquets", dans la terminologie en usage):

- des "paquets courts", de longueur constante - 80 octets, qui arrivent selon un flux poissonnien λ_1=30 paquets/s.
- des "paquets longs", longueur distribuée exponentiellement de moyenne 400 octets, selon un flux poissonnien λ_2=10 pq/s.

a) Quelle est l'attente subie par chaque flux (la gestion est sans priorité).

b) Les paquets courts sont des données précieuses et urgentes (signalisation dans le réseau), on veut en limiter le temps d'attente. Quelle limitation apporter au flux λ_2, pour que l'attente moyenne du flux 1 soit inférieure à 40 ms.

c) Même question, si le flux 1 devient prioritaire.

5.5 - **Time is money**

Un atelier comporte 2 séries de machines, qui produisent en série des pièces quelconques. Les machines de type A produisent des pièces "de luxe", qui seront vendues 4 F chaque, tandis que les machines de type B produisent des pièces ordinaires, qui rapportent 3 F chaque. Le rythme de production est le même pour les machines A et B.

Ces machines, fragiles, réclament le concours permanent d'un spécialiste, pour l'alimentation, le réglage, etc. Chaque classe de machine C ($C = A$ ou B) provoque ainsi un flux de demandes qu'on suppose Poissonnien, de taux λ_c (on suppose le nombre total de machines très grand, pour justifier l'hypothèse poissonnienne). La durée d'intervention du spécialiste

est exponentielle, de durée moyenne X_C. Application numérique: λ_A=0.2/mn, X_A=2mn; λ_B=0.5/mn, X_B=1mn.

Comment ordonnancer le traitement des "pannes" (les interventions du spécialiste)? C'est la question que se pose le responsable de l'atelier.

a) Calculer le trafic offert par chaque flux. Vérifier qu'un seul opérateur suffit.

b) Calculer le paramètre W_0 caractérisant le système. Ecrire la relation de conservation.

c) D'abord, on traite les demandes dans l'ordre de leur arrivée. Quels sont les temps d'attente, et quel est le coût de cette solution?

d) Donnons maintenant la priorité au flux A: priorité "HOL". Quels sont les temps d'attente, et quel est le coût de cette solution?

e) Même question en donnant la priorité au flux B.

f) Essayer de généraliser, en montrant la règle suivante: L'ordonnancement optimal consiste à traiter les flux dans l'ordre des rapports C_j/X_j décroissants (c.a.d priorité au flux de rapport C/X maximum, etc). Cf. [11], tome 2.

5.6 - Un système informatique multiprocesseurs est organisé autour d'un bus, fonctionnant sur le principe de l'anneau à jeton. Vitesse du bus: 1 Mbit/s; le jeton est un paquet de longueur 100 bits. Le bus interconnecte N=50 stations.

Les processeurs utilisent le bus pour leurs besoins de communication. Ils échangent pour cela des paquets de longueur exponentielle, moyenne 1000 bits. Chaque station émet un flot de 14 paquets/seconde.

a) Calculer le temps moyen de circulation du jeton.

b) Quel est le temps moyen d'attente avant émission d'un paquet. En déduire le nombre moyen de paquets en attente dans la file d'émission du processeur.

c) Lorsqu'un processeur réclame une information à un autre processeur, il utilise cette procédure. Admettons que c'est là la seule fonction du bus. Supposons un trafic équilibré (demandes dirigées uniformément vers tous les processeurs). Quel est le trafic de demandes reçu par un processeur?

d) Le processeur est un serveur qui traite un trafic local (nécessitant de temps à autre l'émission d'une demande), et qui en même temps répond aux demandes de ses compagnons. Supposons que le traitement local demande s_L=4 ms, avec un trafic correspondant ρ_L=0.6; la réponse aux requêtes du bus demande s_D=2 ms. Calculer le trafic total du processeur.

e) On donne priorité aux demandes distantes. Quel est le temps de réponse. Quel est le temps qui s'écoule entre l'émission d'une requête vers un processeur distant et la réception de la réponse (ce temps est l'attente devant le bus, puis dans la file du processeur, puis à nouveau devant le bus).

Chapitre 6

Réseaux de Files d'Attente: 1- Réseaux de Jackson

6.1 Introduction

Le concept de "réseau de files d'attentes" apparait naturellement, aussitôt qu'on tente de construire un modèle d'un système un peu complexe. Un réseau de télécommunications, un système informatique moderne mettent en œuvre plusieurs stations, entre lesquelles les clients vont circuler, selon un *routage* plus ou moins complexe.

La théorie des réseaux de files d'attentes est née dans la fin des années 50, avec (entre autres) les travaux de J.R. Jackson: 1ère publication en 1957, généralisation en 1963. La version "fermée", dûe à Gordon et Newell, date de 1967. Il faut ensuite attendre 1972 pour une généralisation majeure, grâce aux travaux de Baskett, Chandy, Munz et Palacios (travaux précédés évidemment d'autres résultats préparatoires).

Cette première partie va nous permettre d'introduire le sujet, avec la présentation des résultats de Jackson et Gordon-Newell.

6.2 Le Réseau de Jackson ouvert

6.2.1 Hypothèses, notations

On considère un réseau de N stations monoserveurs, dans lequel des clients évoluent de façon aléatoire. Plux précisément, au départ d'une station i, le client tire au sort la prochaine station j. On note $r(i,j)$ la probabilité de passer à la station j, à la fin du service en station i.

Le client peut aussi choisir de quitter le réseau, ce qu'on imagera par une $N+1$ ième station "out" Il faut que $\sum_{j=1}^{N+1} r(i,j) = 1$. Le terme $r(i,i)$, non nul dans le cas général, représente le rebouclage d'un client en fin de service.

Chaque serveur est exponentiel, on notera μ_i le taux de service de la station i. Le serveur

est précédé d'une file, de capacité infinie, dont les clients sont extraits dans l'ordre de leur arrivée. Les temps de service reçus dans les serveurs successifs sont indépendants.

Les clients arrivent de l'extérieur, selon un flux de Poisson. On peut donner du processus des arrivées deux descriptions équivalentes:

- Chaque station i voit arriver un flux externe de taux λ_i;
- Une station fictive de numéro 0 voit arriver un flux Λ, les clients sont ensuite aiguillés avec des probabilités $q_i = r(0,i)$, et $\lambda_i = \Lambda.q_i$.

Finalement, on notera n_i le nombre de clients présents dans la station i (en attente ou en service).

On représentera schématiquement le réseau par le graphe suivant:

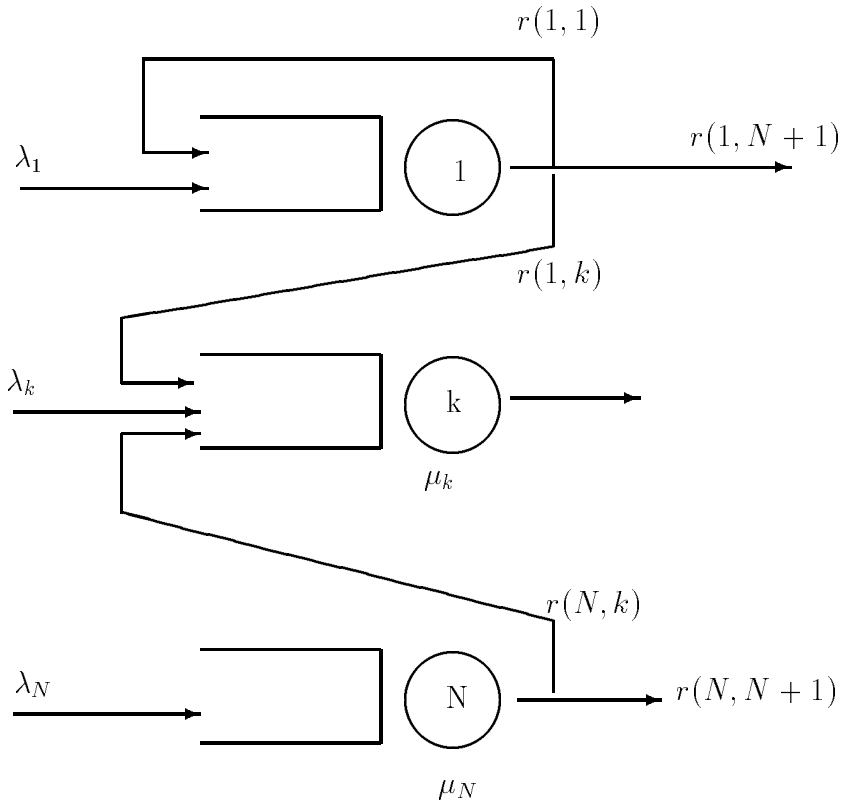


FIG. 6.1 – *Le "Réseau de Jackson" le plus général*

6.2.2 L'équation de flux

Notons e_i le taux total des arrivées (ou départs) dans la station i. Attention: ce taux ne préjuge pas d'une hypothèse Poissonnienne; e_i ne représente rien de plus que le nombre

moyen d'arrivants pas unité de temps, qu'il s'agisse d'arrivées externes ou de mouvements internes au réseau.

Alors, on a la relation suivante:

$$e_i = \lambda_i + \sum r(j,i).e_j \tag{6.1}$$

La preuve de cette relation est simple: elle affirme simplement la conservation du nombre total de clients dans le réseau (englobant la station fictive $N+1$).

Exemple. Il est laissé en vérification au lecteur que, pour le réseau qui suit, on a $e_1 = 2\lambda_1 + \lambda_2$ et $e_2 = 1.2\lambda_1 + 1.6\lambda_2$

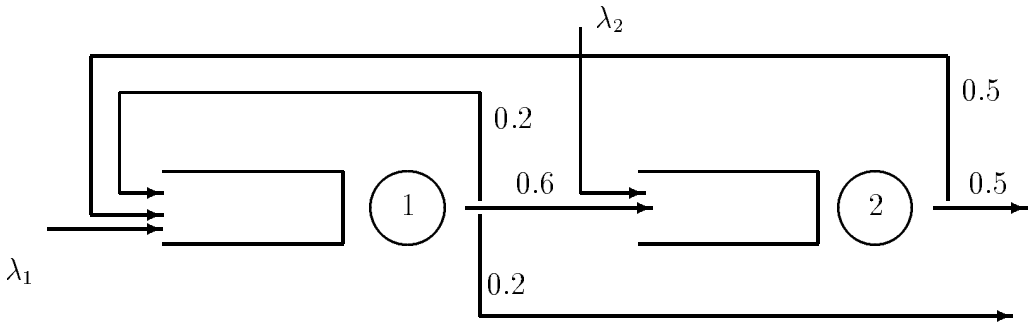


FIG. 6.2 – *Calcul des Flux dans le Réseau*

Puisque les e_i représentent les taux d'arrivée des clients (c'est à dire du "travail"), il faut évidemment, pour qu'un régime stable puisse exister que pour chaque serveur, la condition suivante soit vérifiée:

$$e_i < \mu_i$$

(NB: c'est une condition nécessaire, jusqu'à ce point).

6.2.3 Description d'état et équations d'évolution

Première constatation. On a fait des hypothèses exponentielles assez larges: il y a de fortes chances qu'une description d'état markovienne soit possible.

Effectivement, on se rend vite compte que la donnée du N-uplet $(n_1, n_2, \cdots, n_N)$ constitue une telle description (voir exercice). Par description d'état, rappelons bien que nous entendons la donnée du nombre minimum de variables (entières ou réelles) nécessaires à la prévision (stochastique) du futur du système:

Une arrivée externe en file i dans l'intervalle $[t, t+dt[$ a lieu avec probabilité $\lambda_i dt$, indépendamment des autres éléments du réseau. Un départ de la file k vers la file m a lieu avec la probabilité $\mu_k r(k,m)\, dt$. Les hypothèses qu'on a adoptées permettent ces affirmations.

On augmente l'intérêt de la description d'état en introduisant la notation $\underline{1}_j$. Si $\underline{n}$ désigne le vecteur d'état $(n_1, \cdots, n_N)$, alors:

$$\underline{n} - \underline{1}_j = (n_1, \cdots, n_j - 1, \cdots, n_N)$$

En d'autres termes, $\underline{n} - \underline{1}_j$ désigne l'état où la file j a perdu un client. Le mouvement d'un client de k vers m se marquera par $-\underline{1}_k + \underline{1}_m$, etc.

A partir de la description d'état et des hypothèses d'évolution, il est facile d'écrire l'ensemble des équations d'évolution. Ces équations, dites de "balance globale", traduisent l'égalité des flux de probabilité entrant et sortant pour chaque état.

$$\left[\Lambda + \sum_{i=1}^{N} \mu_i\right] p(\underline{n}) = \sum \lambda_i \, p(\underline{n} - \underline{1}_i) + \sum \mu_i \, p(\underline{n} + \underline{1}_i) \, r(i, N+1) + \sum_i \sum_j \mu_i \, p(\underline{n} + \underline{1}_i - \underline{1}_j) \, r(i,j) \tag{6.2}$$

En bref, le taux de départs de l'état $\underline{n}$ est égal au taux des arrivées: par arrivée d'un client, par départ d'un client, ou encore par un transfert de i vers j.

(Remarque: on a écrit l'équation "générale". Pour les cas limites, des termes peuvent manquer: pas de départ d'une station vide, notamment).

Cas particulier

Pour illustrer l'écriture et la signification des équations, supposons un réseau à 2 stations. L'espace d'états est un réseau dans le plan, ce qui permet de le dessiner. On posera pour simplifier l'écriture: $n_1 = i$ et $n_2 = j$. On représente seulement les transitions sortant de l'état (i, j).

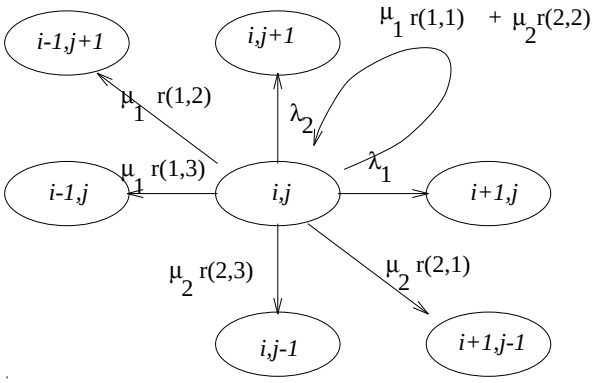


FIG. 6.3 – *Les transitions, dans un réseau à 2 stations (NB: station 3 = sortie du réseau).*

6.2.4 Solution des équations

Soit (e) le vecteur solution de l'équation de conservation (1). Le **théorème de Jackson** (dont la formulation générale date de 1963) donne la solution du système d'équations ci-dessus.

$$p(n) = \prod_{i=1}^{N} p_i(n_i) \tag{6.3}$$

$$p_i(n_i) \quad = \quad (1-\rho_i).(\rho_i)^{n_i} \qquad \text{avec} \qquad \rho_i = e_i/\mu_i \tag{6.4}$$

En clair: tout se passe comme si le réseau était décomposé en stations indépendantes (d'où le produit des p_i), chaque station se voyant alimentée par un flux Poisson (d'où une solution identique à celle d'une M/M/1).

En réalité, les stations ne se comportent pas de façon indépendante. De même, on peut montrer que les flux ne sont généralement pas Poissonniens, à l'intérieur du réseau. Bien noter le "tout se passe comme si".

La démonstration de ce théorème est relativement simple (mais laborieuse): elle consiste à vérifier que l'expression donnée vérifie effectivement le système. Puisque une solution existe et est unique, on en exhibe une, c'est donc la solution.

6.3 Réseau de Jackson fermé

Dans un réseau fermé, aucun client ne rentre ni ne sort ($\lambda_i = 0$, $r(i, N+1) = 0$ pour tout i). Le nombre de clients dans le réseau est constant, on le notera M. Un exemple élémentaire de tel réseau, le réseau cyclique:

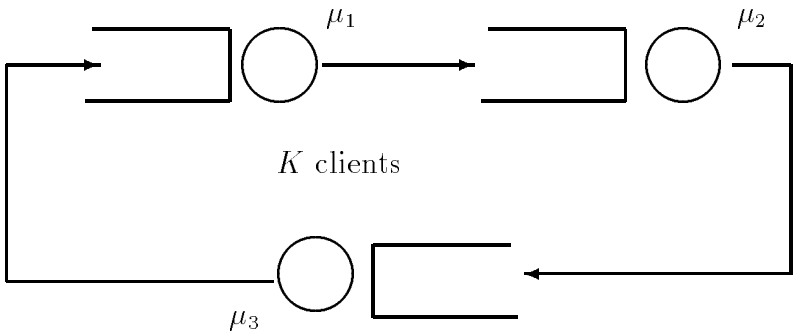


FIG. 6.4 – *Un réseau fermé, cyclique à trois stations.*

6.3.1 Equation de flux, Equations d'état

L'équation de flux prend la forme simplifiée:

$$e_i = \sum r(j,i).e_j \tag{6.5}$$

La preuve en est identique (conservation des flux). Malgré tout, une difficulté surgit: les e_i ne peuvent pas désigner les taux d'arrivée. Egalement, on vérifie facilement que le système n'a plus de solution unique. Ainsi, dans l'exemple du réseau cyclique, on voit très vite que le système donne comme solution:

$$e_1 = e_2 = e_3$$

L'interprétation de la solution est la suivante: les e_i donnent les fréquences de passage relatives dans chaque station (d'où l'égalité, dans le réseau cyclique). On peut choisir arbitrairement l'un des e, prendre par exemple $e_1 = 1$. Les autres e_i donnent alors le nombre de

passages à travers i pour chaque traversée de la station 1. On vérifiera sur la solution, donnée plus loin, qu'elle ne dépend pas de ce choix arbitraire. Quant aux équations qui gouvernent le réseau, elles s'écrivent directement - ou bien se traduisent immédiatement du système (2), en y interdisant la possibilité de sortie:

$$\sum_i \mu_i p(\underline{n}) = \sum_i \sum_j \mu_j \, r(j,i) \, p(\underline{n} + \underline{1}_j - \underline{1}_i) \tag{6.6}$$

Comme dans le cas ouvert, on posera $\rho_i = e_i/\mu_i$ pour simplifier l'écriture - même si ici le "ρ" n'a pas directement le sens usuel.

L'interprétation du système découle immédiatement des réflexions qui ont conduit à (2).

6.3.2 Solution des équations

On va, là encore, exhiber une *forme produit*, comme d'un chapeau. Montrons comment on peut en justifier l'introduction (a posteriori...).

A partir de l'équation des flux (5), on écrit

$$1 = \sum_j r(j,i) \, \frac{e_j}{e_i} \qquad i = 1, \cdots, N$$

Dans l'équation (6), on multiplie le terme de gauche par cette dernière relation (multiplier par 1 ne change rien!).

$$\sum_i \sum_j \mu_i r(j,i) \frac{e_j}{e_i} p(\underline{n}) = \sum_i \sum_j \mu_j r(j,i) p(\underline{n} + \underline{1}_j - \underline{1}_i)$$

C'est à dire

$$\sum_i \sum_j r(j,i) \left[\mu_i \frac{e_j}{e_i} p(\underline{n}) - \mu_j p(\underline{n} + \underline{1}_j - \underline{1}_i) \right] = 0$$

Une façon simple de satisfaire cette équation est de rendre nul le terme entre crochets.

$$p(\underline{n}) = \rho_i (\rho_j)^{-1} p(\underline{n} + \underline{1}_j - \underline{1}_i)$$

Cette égalité, à vérifier pour tous les couples i et j, sera satisfaite si, par exemple,

$$p(\underline{n}) = \rho_i p(\underline{n} - \underline{1}_i)$$

En clair, l'équation qui précède s'écrit:

$$\begin{aligned} lp(n_1, \cdots, n_N) &= \rho_i \, p(n_1, \cdots, n_i - 1, \cdots, n_N) \\ &= \rho_i^2 \, p(n_1, \cdots, n_i - 2, \cdots, n_N) \\ &= \rho_i^{n_i} \, p(n_1, \cdots, 0, \cdots, n_N) \end{aligned}$$

En déroulant la même récurrence pour les autres indices, on arrive à la forme finale:

$$p(n_1, \cdots) = \text{Cste} \prod_{i=1}^{N} (\rho_i)^{n_i}$$

La constante pourrait s'écrire $p(\underline{0})$. On la note traditionnellement $G_N(K)^{-1}$. C'est, comme à l'habitude, la *constante de normalisation*:

$$G_N(K) = \sum_{\mathcal{C}} \prod_i (\rho_i)^{n_i} \tag{6.7}$$

$\mathcal{C}$ désigne le domaine de sommation: $\mathcal{C} = \{\underline{n} \mid n_i \geq 0, \sum n_i = K\}$. Le calcul effectif de $G_N(K)$ fera l'objet d'un paragraphe spécial: il faut en effet perdre l'espoir d'une évaluation directe de cette quantité.

6.3.3 Calcul des figures de performances

Toutes les quantités utiles à la caractérisation des performances du réseau fermé peuvent se déduire des $p(\underline{n})$. En fait, on peut aussi les évaluer à partir de la constante $G_N(K)$.

Taux d'utilisation

On note u_i le taux d'utilisation de la station i (fraction du temps où le serveur est actif):

$$u_i = P\{n_i \neq 0\}$$

C'est la probabilité de trouver le réseau dans un état $\{\underline{n}\}$, tel que $n_i > 0$. Soit:

$$u_i = \frac{1}{G_N(K)} \sum_{\mathcal{C}'} \prod_k (\rho_k)^{n_k}$$

$\mathcal{C}'$ désigne le domaine: $n_j \geq 0$, $j \neq i$, $n_i > 0$, $\sum n_k = K$. On renomme la variable n_i: $m_i = n_i - 1$, pour réécrire:

$$u_i = \frac{1}{G_N(K)} \rho_i \sum_{\mathcal{C}''} \prod_k (\rho_k)^{n_k}$$

$\mathcal{C}'$ désigne cette fois le domaine: $n_j \geq 0$, $j \neq i$, $m_i \geq 0$, $\sum n_k = K - 1$. En clair, on retrouve la même somme que pour G_N, avec cette fois 1 client en moins:

$$u_i = \rho_i \frac{G_N(K-1)}{G_N(K)} \tag{6.8}$$

Remarque: On a $\dfrac{u_i}{u_j} = \dfrac{\rho_i}{\rho_j}$ (théorème de "Chang-Lavenberg").

Occupations moyennes

Nombre moyen de clients dans la file i:

$$E(n_i) = \sum_k k\, P\{n_i = k\} = \sum_k P\{n_i \geq k\}$$

On laisse au lecteur le soin de montrer, par un calcul analogue au précédent, que:

$$P\{n_i \geq k\} = (\rho_i)^k \frac{G_N(K-k)}{G_N(K)}$$

Il s'ensuit que:

$$E(n_i) = \frac{1}{G_N(K)} \sum_{k=1}^{K} (\rho_i)^k G_N(K-k) \tag{6.9}$$

Remarque: Le choix de la solution particulière du système (5), par exemple en y faisant $e_1 = 1$, est sans importance: toutes les quantités qu'on a calculées dépendent non pas des (e_i), mais bien des quotients entre ces quantités, comme le montrent les formules ci-dessus.

Compléments

Exercice 1: Deux files exponentielles en tandem. Considérons le système suivant:

- Une première station est traitée par un serveur exponentiel (paramètre μ_1). Elle reçoit en entrée un flux Poissonnien, de paramètre λ.
- A leur sortie, les clients sont dirigés vers une seconde file, à serveur exponentiel (paramètre μ_2).
- Après le second service, les clients quittent le système.

La première station est une file M/M/1. L'ensemble constitue un exemple élémentaire de réseau de files d'attentes.

1) Notons n_1 le nombre de clients dans la 1ère station, n_2 le nombre dans la seconde. Montrer brièvement que (n_1, n_2) est une description d'état, qui rend le système markovien.

2) Ecrire les transitions possibles, construire le diagramme (dans le plan!). Ecrire les équations d'état.

3) Il est possible, mais délicat, de résoudre brutalement le système d'équations. Vérifier qu'il existe une solution de la forme:

$$P\{n_1, n_2\} = A \left(\frac{\lambda}{\mu_1}\right)^{n_1} \left(\frac{\lambda}{\mu_2}\right)^{n_2}$$

Chapitre 7

Réseaux de Files d'Attente: 2- BCMP, et Algorithmes

7.1 Introduction

Les théorèmes de Jackson et de Gordon-Newell permettent de calculer les probabilités d'état d'un réseau grâce à la propriété miraculeuse que constitue l'existence d'une *forme produit*. Dès la publication du théorème, on a cherché à en étendre l'appplication: pourrait-on traiter aussi efficacement d'autres types de réseaux, d'autres catégories de stations, etc.

Premier exemple d'extension, presqu'immédiat: on voit assez facilement que les théorèmes sont encore valides si les stations ont un taux de service fonction du nombre de clients qu'elles contiennent: $\mu_i = f(n_i)$. C'est à dire qu'on a encore, dans ces conditions, une forme produit. Cela permet de traiter le cas de stations multiserveurs (avec c_i serveurs), où $\mu_i(n_i) = \min(n_i, c_i)\,\mu$. Le théorème BCMP (pour Baskett, Chandy, Muntz et Pallacios, ses inventeurs) réalise l'avancée décisive, en décrivant les conditions sous lesquelles une forme produit est possible.

7.2 Le Théorème BCMP

7.2.1 Comment représenter une "loi générale?

On va utiliser une astuce, pour représenter une loi de service quelconque à partir de serveurs exponentiels (seuls serveurs faciles à manipuler, dans une représentation markovienne). On fait pour cela appel aux lois de Cox.

Règle du jeu: un seul client à la fois dans le réseau ci dessus. La transformés de Laplace du réseau de Cox ci-dessus s'écrit:

$$F(s) = \sum_m (1 - b_{m+1}) \prod_{k=1}^{m} \frac{b_k . \mu_k}{\mu_k + s}$$

On définit A_l, probabilité que le client aille jusqu'au serveur de rang l, avant de quitter le

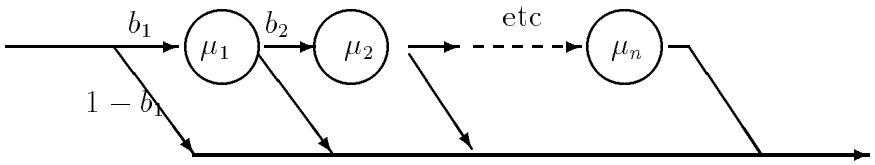


FIG. 7.1 – *Un "Réseau de Cox". Serveurs exponentiels modélisant une loi générale*

réseau. Le temps moyen de service s'en déduit:

$$\begin{aligned} A_l &= b_1.b_2.\ldots.b_l.(1-b_{l+1}) \\ \frac{1}{\mu} &= \sum_k \frac{A_k}{\mu_k} \end{aligned} \tag{7.1}$$

On en tire le résultat suivant: *Toute loi de service à transformée de Laplace rationnelle peut se représenter par une telle construction.* On sait d'autre part que toute loi de service peut être approximée (sa transformée de Laplace) par une telle fraction. On pourra donc représenter une loi de service quelconque par un réseau de Cox, avec une précision aussi grande que nécessaire.

Quel intérêt présente cette approximation? L'état du réseau de Cox est décrit (avec le sens habituel de ce mot) par le numéro de l'étage atteint par le client: représentation markovienne, donc.

7.2.2 Hypothèses et Notations

Les clients du réseau peuvent appartenir à plusieurs *classes*. Les classes se différencient, selon le cas, par le temps de service ou le routage dans le réseau. Les clients peuvent changer de classe en changeant de station. On notera R le nombre de classes.

Plus spécifiquement on introduit le jeu de probabilités de passage $p(i,r;j,s)$: probabilité qu'un client de classe r qui quitte la station i se dirige vers la station j avec la classe s. Comme dans le réseau de Jackson, une $N+1$-ième station représente la sortie du réseau.

De même, on introduit les probabilités d'arrivée: $q_{i,t}$, probabilité qu'une arrivée produise un client de classe t se dirigeant vers la station i. On note λ le taux des arrivées total. Le taux d'arrivées global peut dépendre du nombre total de clients dans le système: on notera alors $\lambda(K)$ le taux d'arrivées avec K clients dans le réseau.

On notera $k_{i,r}$ le nombre de clients de classe r dans la station i, k_i le nombre total de clients dans cette station, et K le nombre total de clients dans le réseau. On a évidemment:

$$k_i = \sum_r k_{i,r} \qquad K = \sum_i k_i$$

On autorise, pour les stations (ou les serveurs qui les alimentent), des mécanismes plus généraux que pour le théorème de Jackson. Les types possibles sont énumérés ci-après:

7.2.3 Les types de stations

Station de type 1.
Le serveur est de type exponentiel, le taux (μ_i) peut être fonction du nombre de clients dans la station. Tous les clients (quelle que soit leur classe) ont la même loi de service. La discipline de service est FIFO. Pour simplifier, on note $\mu(k_i) = C_i(k_i).\mu$. Pour une station à 1 serveur, $C_i = 1$. Pour une station à m_i serveurs, $C_i(k_i) = \min(k_i, m_i)$.

Station de type 2.
La loi de service est *quelconque* (plus exactement, à transformée de Laplace rationnelle), elle peut différer selon les classes de clients. La discipline est *Processor Sharing*.

Station de type 3.
La loi de service est générale (à transformée de Laplace rationelle). Le nombre de serveurs est infini, c'est à dire qu'il n'y a pas d'attente (on parle de *délai pur*).

Station de type 4.
La loi de service est générale (à transformée de Laplace rationnelle). Elle peut différer selon les classes. La discipline de service est *LIFO avec préemption* (le dernier client qui arrive prend la place de celui qui était en cours de service).

7.2.4 Description de l'état des stations

La description d'état reste *discrète*, mais va se compliquer, par rapport au cas de Jackson.

Considérons la station i. Si elle est de type 1, son état est décrit par la donnée du vecteur $(r_1, r_2, \cdots, r_{k_i}$, où k_i est le nombre de clients de la station, et où r_m représente la classe du client de rang m en attente dans la file.

Pour les stations de type 2 et 3, l'état est un vecteur $(v_1, \ldots, v_R)$. v_p est attaché à la classe p: $v_p = (s_1, s_2, ..., s_{k_p})$ – on omet l'indice i de la station, par simplicité); s_j désigne l'étape du serveur de Cox atteinte par le $j-$ième client de la classe considérée (la station compte k_p clients de classe p).

Pour les stations de type 4, l'état est le vecteur $(r_1, m_1, r_2, m_2, ..., r_p, m_p)$, avec r_j: la classe du j-ième client, et m_j l'étape du serveur de Cox qu'il a atteint.

Pour toutes les stations à serveurs de Cox, on note $k_{i,r,l}$ le nombre de clients de la station i, de classe r, ayant atteint l'étage l de leur serveur de Cox (stations de type 2, 3, 4, où plusieurs serveurs sont actifs simultanément).

Comme on le voit, la description d'état est lourde, en général. Cette description assure cependant une *évolution markovienne* du système: on pourra écrire une équation de balance – et la résoudre.

On n'écrira pas cette équation ici, tant elle est lourde et de peu d'intérêt; c'est l'analogue de l'équation (2) du Chapitre précédent, elle égale les taux de transition entrant et sortant de chaque état du réseau.

7.2.5 Le théorème BCMP

Notons $e_{i,r}$ la solution du système d'équations:

$$e_{i,r} = q_{i,r} + \sum_{j,s} e_{j,s} p(j,s;i,r) \tag{7.2}$$

C'est encore l'équation des flux, où cette fois, les e sont proportionnels (et non égaux) aux taux d'arrivée dans chaque file. Si le système est fermé, on a simplement les q tous nuls.

La solution générale du réseau de files d'attentes, c'est à dire la probabilité d'observer l'état global $\mathbf{S} = (\mathbf{S_1}, \mathbf{S_2}, \ldots, \mathbf{S_N})$, du réseau à N stations et R classes de clients, est donnée par:

$$P(\mathbf{S}) = \frac{1}{G} d(\mathbf{S}) \prod_{i=1}^{i=N} f_i(\mathbf{S_i}) \tag{7.3}$$

Les fonctions f_i sont données par les expressions suivantes:

$$\begin{aligned}
\text{Type 1:} \quad f_i(\mathbf{S_i}) &= \prod_{j=1}^{k_i} \frac{e_{i,r_j}}{\mu_i C_i(k_i)} \\
\text{Type 2:} \quad f_i(\mathbf{S_i}) &= k_i! \left[\prod_{r,l} \left(\frac{e_{i,r} A_{i,r,l}}{\mu_{i,r,l}}\right)^{k_{i,r,l}} / k_{i,r,l}! \right] / \prod_{j=1}^{k_i} C_i(j) \\
\text{Type 3:} \quad f_i(\mathbf{S_i}) &= \prod_{r,l} \left(\frac{e_{i,r} A_{i,r,l}}{\mu_{i,r,l}}\right)^{k_{i,r,l}} / k_{i,r,l}! \\
\text{Type 4:} \quad f_i(\mathbf{S_i}) &= \prod_{j=1}^{k_i} \frac{e_{i,r_j} A_{i,r_j,l_j}}{\mu_{i,r_j,l_j}}
\end{aligned} \tag{7.4}$$

K désigne le nombre total de clients (toutes files, toutes classes confondues). G désigne la constante de normalisation, d est donnée par:

$$\begin{aligned}
d(\mathbf{S}) &= 1 && \text{si le réseau est fermé} \\
d(\mathbf{S}) &= \textstyle\prod_{k=0}^{K-1} \lambda(k) && \text{si le réseau est ouvert}
\end{aligned}$$

La description d'état donnée par le vecteur $\mathbf{S}$ est habituellement trop riche. On est rarement intéressé, par exemple, par le détail des étages de service. On simplifiera alors; en donnant les probabilités marginales: sommes sur les états de mêmes $k_{i,r}$, sans tenir compte de l'étage de service atteint. Il vient alors:

$$P(\underline{n}) = \frac{d(\underline{n})}{G} . g_1(k_1) . g_2(k_2) . \cdots g_N(k_N) \tag{7.5}$$

où les g sont donnés par:

$$\begin{aligned}
\text{type 1:} \quad g_i(k_i) &= \left[k_i! \prod_r \frac{(e_{i,r})^{k_{i,r}}}{k_{i,r}!} \right] / \prod_{j=1}^{k_i} \mu_i C_i(j) \\
\text{type 2 et 4:} \quad g_i(k_i) &= \left[k_i! \prod_r \left(\frac{e_{i,r}}{\mu_{i,r}}\right)^{k_{i,r}} / k_{i,r}! \right] / \prod_{j=1}^{k_i} C_i(j)
\end{aligned} \tag{7.6}$$

$$\text{type 3:} \qquad g_i(k_i) \;=\; \prod_r \left(\frac{e_{i,r}}{\mu_{i,r}}\right)^{k_{i,r}} / k_{i,r}!$$

Comment arriver à cette forme, en partant de (3-4)? On fait la somme des probabilités pour les états de même valeur des $k_{i,r}$ (somme sur les indices l dans (4). Cela revient, grâce à la forme produit, à sommer séparément les termes de chaque file f_i. Pour une station de type 2, par exemple, on a le terme:

$$\sum_{k_{i,r,l}} k_i! \prod_r \prod_l {\omega_{i,r,l}}^{k_{i,r,l}} / k_{i,r,l}$$

la somme est sur les k tels que $\sum_l k_{i,r,l} = k_{i,r}$. Le terme ω désigne le facteur eA/μ. On remarque (!) que la somme sur les $k_{i,r,l}$ des produits sur r n'est rien d'autre que le développement de $k_{i,r}!(\sum_l \omega_{i,r,l})^{k_{i,r}}$. On arrive ainsi à la forme donnée en (4), en se souvenant que, selon l'équation (1), $\sum_l A_{i,r,l}/\mu_{i,r,l} = 1/\mu_{i,r}$.

D'autres simplifications sont possibles (d'autres probabilités marginales), si on ne s'intéresse pas aux différentes classes de clients. Enfin, si le réseau sert une seule classe de clients, dans des stations toutes de type 1, il est immédiat de vérifier que l'on retrouve les résultats de Jackson et Gordon-Newell.

7.3 Algorithmes de Calcul

Dans tout ce qui suit, on se place dans le cadre simplifié du théorème de Gordon-Newell. Tous les résultats de cette section se généralisent au théorème BCMP (c'est à dire avec certaines des stations de type 2, 3 ou 4). Dans le but de garder un exposé simple et des algorithmes compréhensibles, on se restreint ici aux stations de type 1 avec des taux μ constants (ne dépendant pas des n_i). Pour le cas général (qui ne présente pas de difficulté théorique supplémentaire) on se reportera à la bibliographie.

Le calcul complet de la solution du réseau fermé comportant N stations et K clients passe, on l'a vu, par l'évaluation de la constante de normalisation $G_N(K)$. En général, le *calcul direct* de cette constante consomme des ressources (de calcul) prohibitives. En effet, ce calcul porte sur un nombre d'états égal à $\mathcal{N}(N,K) = \begin{pmatrix} K+N-1 \\ K \end{pmatrix}$.

Pour le montrer, on remarque simplement qu'on peut dénombrer ces états par l'artifice suivant: alignons K symboles "1", représentant les clients. Disposons "entre" ces symboles $N-1$ séparateurs (des "0", si l'on veut). Chaque configuration définit un état particulier.

Ainsi, pour un réseau à 4 stations, où évoluent 6 clients, la configuration 111010011 est à interpréter comme $n_1 = 3$, $n_2 = 1$, $n_3 = 0$ et $n_4 = 2$. Il y a $K+N-1$ symboles au total, le nombre de configurations est le nombre de façons de choisir $N-1$ "0" parmi eux. CQFD.

Exemple: $K = 25$, $N = 10$: le nombre de configurations est de l'ordre de 52.10^6.

En termes de *complexité algorithmique*, le nombre d'opérations pour une estimation directe de la constante est dit "en $O(\mathcal{N}(N,K))$".

Tous les éléments caractéristiques des performances du réseau seront déduits des G. En effet, on a vu que les taux d'utilisation, les valeurs moyennes se déduisaient du jeu des $G_N(j)$ $j = 1, \cdots, K$ (formules (8,9) du Chapitre précédent).

7.3.1 L'Algorithme de Convolution

Définissons les fonctions génératrices partielles:

$$g_i(z) = \sum_{k \geq 0} (\rho_i z)^k = \frac{1}{1 - \rho_i z}$$

$$g(z) = \prod_i g_i(z)$$

$G_N(K)$ est le coefficient de z^K dans $g(z)$. Pour calculer $g(z)$, on effectue les produits partiels:

$$h_1(z) = g_1(z)$$
$$h_m(z) = h_{m-1}.g_m(z)$$

Le produit partiel h_m correspond à la fonction G_m: $h_m(z) = \sum G_m(j) z^j$.

La relation de récurrence des h_m se met sous la forme:

$$h_m(z) = h_{m-1}(z) + h_m(z)\rho_m.z$$

soit:

$$G_m(j) = G_{m-1}(j) + \rho_m G_m(j)$$

C'est la récurrence cherchée, qui débute avec $G_1(k) = (\rho_1)^k$, $G_k(0) = 1$.

On opère par "lignes", calculant les $G_m(k)$, $k = 1, ..., K$, puis passant à la ligne suivante $(m := m+1)$, jusque à $m = K$. On voit facilement que la complexité du calcul est en $O(NK)$. Dans le cas général, où les taux de service sont fonctions des occupations, l'algorithme est en $O(NK^2)$. On comparera pour l'exemple évoqué $K = 25$, $N = 10$: on passe de $\mathcal{N}(N, K) = 50.10^6$ à $NK = 250$ ou $NK^2 = 2500$.

7.3.2 Mean Value Analysis

Il existe même une façon encore plus radicale de simplifier l'estimation des performances, en ne calculant pas la constante $G_N(K)$. Cette méthode, connue sous le vocable de *"Mean Value Analysis"*, est dûe à Reiser et Lavenberg, et date de 1979.

Montrons en le principe, sur le cas particulier d'un réseau cyclique. Cherchons à écrire l'expression du temps de traversée de la i-ième station, $E(T_i)$:

$$E(T_i) = \frac{1}{\mu_i} + \frac{1}{\mu_i}.\{\text{Nbre moyen de clients à l'arrivée}\}$$

Le point clé de la méthode est le suivant: on relie ce temps avec les caractéristiques d'un réseau comprenant 1 client de moins. Ceci grâce à un résultat, *"Arrival theorem"* (dont la paternité est attribuée, tantôt à Lavenberg, tantôt à Sevsik et Mitrani), qui dit: le nombre moyen de clients dans une file à l'arrivée d'un client a la même distribution que la distribution d'équilibre dans la file pour un réseau comptant un client en moins. D'où:

$$E(T_i)[K] = \frac{1}{\mu_i} + \frac{1}{\mu_i} E(n_i[K-1])$$

De façon générale, on indicera par $[K]$ dans la suite les quantités estimées dans le réseau fermé contenant K clients. L'expression ci dessus est à rapprocher du calcul de la perte dans le système à population finie.

Maintenant, on relie le débit du réseau avec les nombre de clients. Soit $\Lambda[K]$ le débit du réseau (de chaque station, puisqu'elles sont en série) qui contient K clients. La formule de Little relie temps de séjour et nombre de clients:

$$\Lambda[K].E(T_i)[K] = E(n_i)[K]$$

D'autre part, le nombre total des clients est somme des nombre de chaque station. Soit

$$\Lambda[K].\sum_i E(T_i)[K] = K$$

Les deux relations relient donc $E(T_i)[K]$ et $E(n_i)[K]$. D'où finalement l'algorithme:

- $En_i[0] = 0$ pour i=1,...,N
- Pour $k = 1, ..., K$, faire:
  - pour $i = 1, ..., N$, faire
    $E(T_i)[k] = \frac{1}{\mu_i} + \frac{1}{\mu_i} E(n_i)[k-1]$
  - fpour
  - $\Lambda[k] = k / \sum_i E(T_i)[k]$
  - $E(n_i)[k] = \Lambda[k].E(T_i)[k]$
- fpour

Dans le cas général, il faut prendre en compte la structure exacte de réseau: les débits de sstations ne sont pas identiques; d'autre part, la relation entre $E(T)$ et $E(n)$ changera avec le type de station. On consultera les références pour les formes complètes de l'algorithme.

Pour en savoir plus

Tous les cours modernes disent quelques mots des réseaux, de façon plus ou moins complète. Pour un exposé élémentaire (version simple des résultats de Jackson et Gordon-Newell), voir [13, 15]. Pour un exposé plus complet, voir [6, 5, 10]. Le plus complet est certainement [12], qui donne, entre autres, les versions détaillées et complètes des algorithmes de convolution et de MVA. Mais attention aux notations! La référence [13] contient un chapitre complet et très lisible sur les algorithmes, ainsi qu'un exposé sur PANACEA (algorithme traitant des réseaux de très grande taille).

La découverte de la "forme produit" a suscité de nombreuses recherches, dans la fin des années 70. Pour des compléments sur les extensions et les implications, probabilistes et "philosophiques" de cette propriété, on consultera [10, 13].

Chapitre 8

La Surcharge : 1- Description des phénomènes

Ce chapitre a comme premier objectif d'explorer les phénomènes qu'on observera dans les systèmes quand la charge offerte approche ou dépasse la valeur limite 100%. Ceci doit permettre de comprendre comment en pallier les faiblesses par les mécanisme de *contrôle de surcharge*.

La surcharge provoque des comportements "non-linéaires", dont la cause est à chercher:

- dans le comportement des usagers (humains ou machines),
- dans les effets de variation des temps de service,
- dans le partage de capacités limitées.

Les conséquences sont assez faciles à imaginer: dégradation de la Qualité de Service, mais aussi aggravation et persistance de la surcharge (effets d'hystérésis), et exportation de ses conséquences loin du lieu d'origine. Dans certains cas, on observera des effets de "blocage" (le deadlock).

8.1 Capacités Limitées

La capacité limitée permet en principe de faire fonctionner des systèmes au delà du point limite où le trafic offert dépasse la capacité (en bref, elle autorise $\rho > 1$). Remarque: **tous** les systèmes sont à capacité limitée. On parle de système à capacité limitée lorsque celle-ci fait sentir son effet pendant le service nominal. La capacité limitée devient un problème lorsqu'elle est partagée entre flux distincts. La Figure 8.1 en donne une illustration:

A l'entrée, il y a rejet dès que les files 1 ou 2 sont pleines. Il y a blocage des serveurs S1 et S2 si la file 3 est pleine (obligatoire). On voit alors que si λ_1 augmente, le flux 2 va aussi être perturbé.

La solution la plus simple consistera à scinder N3 en 2 parties réservées. Mais on a vu l'intérêt de partager. Une autre solution sera de partager une partie du tampon, et d'en privatiser

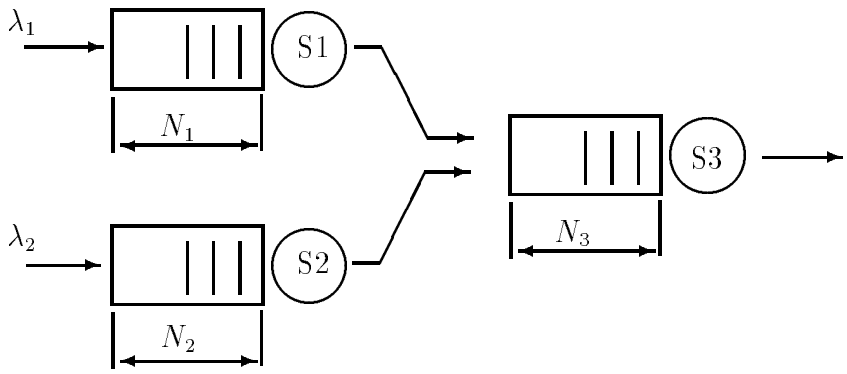


FIG. 8.1 – *Propagation au flux 2 d'une surcharge du flux 1*

une autre. Une règle de partage très performante a pu être proposée:[1] on n'autorise pas plus de $N/\sqrt{L}$ clients de chaque flux (ici, N joue le rôle de N3 et L est le nombre de flux partageant la ressource). Le même mécanisme se rencontrera chaque fois que deux flux aux caractéristiques différentes se mélangeront. Le problème de multiplexage statistique dans les réseaux haut-débit est une forme de ce phénomène.

8.1.1 L'Importance du Phénomène

C'est dans cet effet que réside la source des engorgements du trafic routier - qu'on peut comparer au *deadlock* des systèmes distribués informatiques: ces configurations s'observent aussi dans les systèmes informatiques. A noter qu'on observe de tels blocages même en l'absence de surcharge avérée, par le simple jeu des fluctuations statistiques. Dans le cas du trafic routier, le comportement des usagers perturbe naturellement le phénomène observé.

Dans les réseaux de type téléinformatique le phénomène est le même, quoique plus simple à observer. Imaginons deux flux, le premier allant du noeud A au noeud B, le second de P au noeud Q; ces deux flux transitent par le même noeud C. Si le 1er flux sature, par congestion en B par exemple, alors C va engorger et le flux de P à Q va être perturbé, voire bloqué.

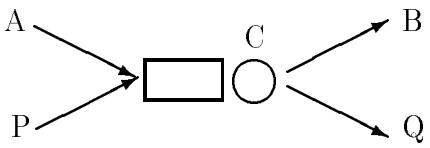


FIG. 8.2 – *Interblocage par mélange des flux*

1. M. Irland, *Buffer Management in a Packet Switch*, IEEE Transactions on Communications, Mars 1978.

8.2 Comportement des sources

Le comportement des sources se caractérise par 2 phénomènes, également importants:

- abandon prématuré (impatience ou temporisation) lorsque le temps de réponse s'allonge; conséquence: le serveur peut être en train de traiter le client qui abandonne, d'où un travail perdu;
- répétition de la demande; conséquence: le trafic offert mesuré se trouve faussé, la même demande étant enregistrée plusieurs fois.

On caractérise le phénomène par les paramètres suivants:

Tentative: tout client qui se présente devant le service constitue une "tentative", qui est un 1er essai ou un renouvellement. On notera λ_0 le flux de 1ères demandes (tentatives fraîches), et λ le flux total observé.

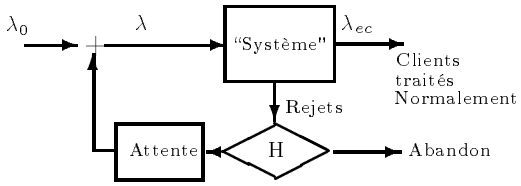
Coefficient de répétition: c'est le rapport $\beta = \lambda/\lambda_0$, qui mesure l'amplification provoquée par l'encombrement; il permet de relier la demande réelle à l'observation.

Taux de persévérance: dans un modèle très simple, on tient compte du comportement de persévérance de la demande de façon probabiliste: soit H la probabilité qu'une tentative qui échoue se représente une nouvelle fois (Raffinement: prendre H fonction du rang de la tentative, de la nature des échecs, etc).

Taux de perte: on note p la probabilité de rejet (ou abandon); en réalité, le taux d'échec est différent selon le rang de la tentative et l'intervalle de temps entre tentatives. $r = 1 - p$ est le "taux d'efficacité".

Il faut bien réaliser qu'on ne peut pas, la plupart du temps, percevoir les quantités "originelles", telles que λ_0. Il est même le plus souvent difficile et très coûteux de séparer dans une campagne de mesure λ_0 de λ (Essayer d'imaginer des protocoles de mesure pour les exemples de ce chapitre).

8.2.1 Un modèle de répétition



On notera λ_{ec} le débit écoulé. I vient:

$$\begin{aligned} \lambda_{ec} &= (1-p)\lambda \\ \lambda &= \lambda_0 + \lambda H p \end{aligned}$$

soit encore:

$$\lambda = \frac{\lambda_0}{1 - Hp} \qquad \lambda_{ec} = \frac{\lambda_0(1-p)}{1 - Hp}$$

8.2.2 Un exemple de conséquence

Supposons que le "système" soit une file M/M/1/N. On injecte une charge λ_0, inconnue. La procédure de mesure fournit λ. Voici un échantillon de résultats (on suppose que le flux total résultant reste Poisson); on a fait $N = 4$, et supposé un taux de persévérance $H = 0.9$:

- pour $\rho_0 = 0.7$, on mesure $\rho = 0.78$ - $p = 0.115$ (la M/M/1/4 aurait $p = 0.086$);
- pour $\rho_0 = 0.9$, on mesure $\rho = 1.20$ - $p = 0.28$ (la M/M/1/4 aurait $p = 0.16$);
- pour ρ_0=1.2, on mesure $\rho = 3.07$ - $p = 0.67$ (la M/M/1/4 aurait $p = 0.28$).

On imagine aisément les fausses interprétations qui en découlent: ou bien une mesure de trafic sous-estime le rejet, ou bien une mesure du rejet réel fait surestimer ρ_0.

Remarque
Peut-on mettre le système en équations? Voici une méthode approchée, qui consiste à supposer que le flux résultant des rebouclages conserve son caractère poissonnien – ce qui est faux, en toute rigueur. On écrira une équation reliant (ρ et p (c'est à dire λ_0, λ et p, et on estimera p par le rejet d'une M/M/1/N, reliant p à ρ:

$$\rho = \frac{\rho_0}{1 - Hp} \qquad \text{et} \qquad p = \frac{\rho^N(1-\rho)}{1-\rho^{N+1}}$$

La résolution est itérative: pour ρ_0 donné, on prend une valeur initiale $\rho = \rho_0$, on en déduit la perte par $p = g(\rho)$, puis une nouvelle valeur de charge par $\rho = f(p)$, et ainsi de suite jusqu'à convergence (précision de 1/1000 en quelques itérations). L'ensemble des résultats est présenté sur le graphe suivant:

(a) Trafic écoule et perte (b) Trafic reellement observe

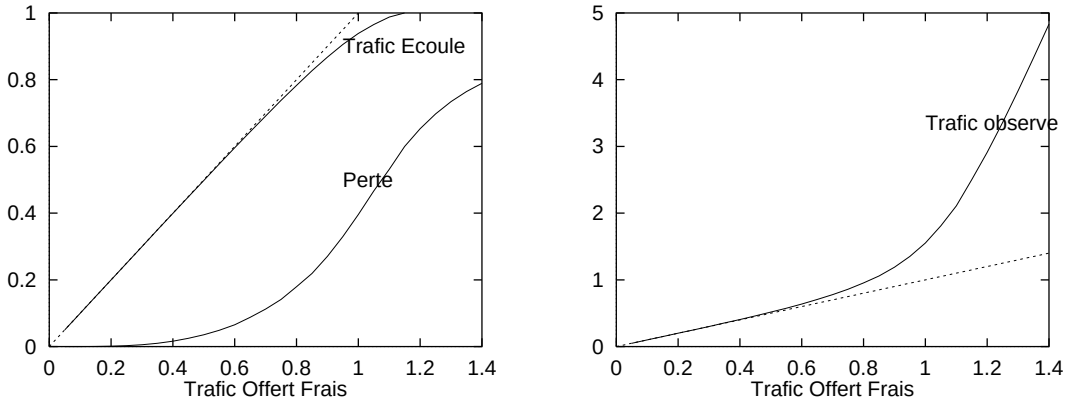
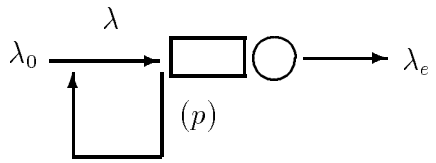


FIG. 8.3 – *Observation des temps d'attente*

8.3 Variation des temps de service

Ce qui est génant c'est que ce temps va croître. Les phénomènes physiques qui provoquent cette variation sont à chaque fois spécifiques du système étudié. On en donne qq exemples.



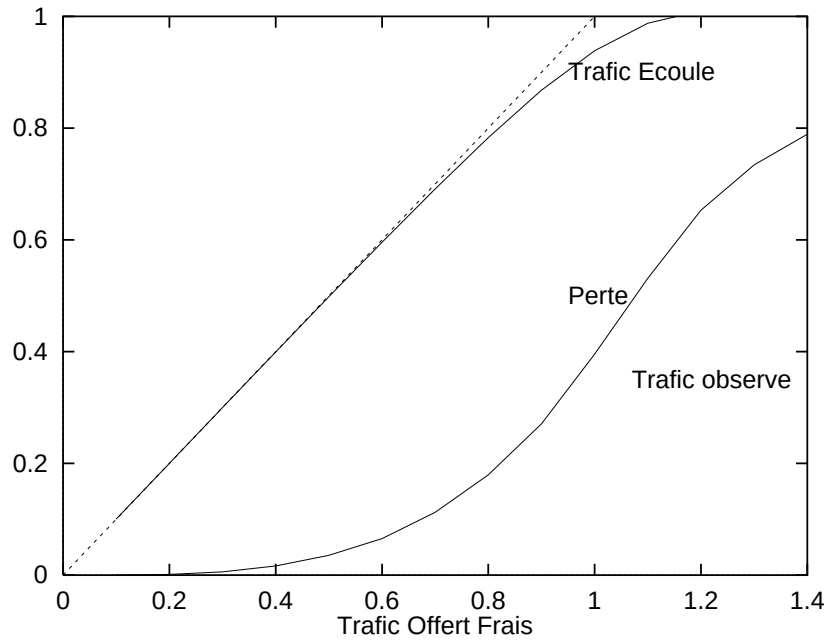
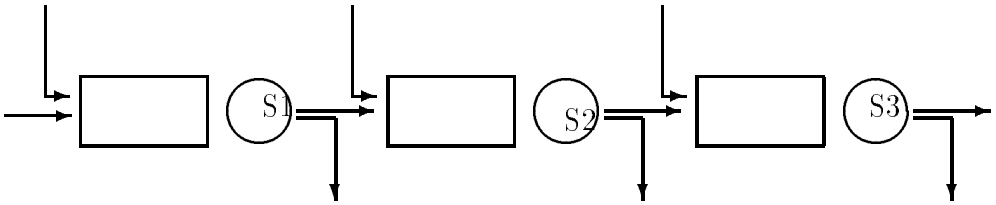


FIG. 8.4 – *Comportement en surcharge d'une file M/M/1/N. $N = 4, H = 0.9$. A gauche, échelles de la perte et du trafic écoulé; à droite, trafic offert observé.*

8.3.1 Nécessité d'un Contrôle de Congestion dans un réseau

On observe un réseau de transmission de données, ou plutôt une liaison de bout en bout de ce réseau (Figure 8.3.1).



Quand un noeud est saturé (le troisième par exemple), le serveur qui le précède se trouve, par là même, bloqué. En fait, le mécanisme exact est plus complexe: S2 va traiter et émettre le paquet (le niveau 2 va en faire une trame...); arrivé devant le tampon plein, le paquet est rejeté (que faire d'autre), même si le niveau 2 est OK. S2, qui ne reçoit pas l'acquittement attendu, doit réémettre une, deux, trois... fois: tout se passe comme si le paquet attendait en S2.

Si on note B la probabilité de rejet (on suppose la liaison homogène, B à chaque étage), et qu'on suppose que chaque tentative a le même B (hypothèse scabreuse), alors il y a:

- une émission, soit 1 temps de service, avec probabilité $1 - B$
- deux émissions, soit 2 temps de service, avec probabilité $B(1 - B)$
- trois émissions, soit 3 temps de service, avec probabilité $B^2(1 - B)$, etc

soit en fin de compte en moyenne $1/(1 - B)$ émissions, ce qui revient à prendre un temps moyen de service équivalent $E(s)/(1 - B)$: **le temps de service croît avec le rejet.**

Côté entrée, on note λ le taux d'arrivée (identique à chaque étage par symétrie). La charge offerte est λ (avec le modèle de serveur équivalent, le paquet attend dans un noeud jusqu'à en sortir victorieusement, il n'y a pas de perte). On fait un modèle M/M/1/N du noeud (c'est vraiment très approché). D'où:

$$B = \frac{a^N(1-a)}{1-a^{N+1}} \qquad \text{avec} \qquad a = \frac{\lambda E(s)}{1-B}$$

Les équations sont tout-à-fait analogues à celles du modèle précédent dans lequel on ferait H=1, quoique l'interprétation en soit différente. Pour un modèle plus réaliste, il faut rajouter l'abandon, et sûrement raffiner le modèle de perte. La résolution numérique, et le résultat, seraient comparables à la courbe du 6.2.

8.3.2 Un commutateur

Les demandes des usagers sont détectées par un explorateur qui place une requête dans la file du processeur principal. Celui-ci traite en permanence un certain nombre de tâches en "concurrence" (admission de nouvelles tâches, E/S, avancement de tâches en cours). L'ordonnancement des tâches est le plus souvent complexe (mélange d'interruptions, de scheduling cyclique, de tranches de temps périodiques, etc). Le mécanisme d'admission doit examiner les requêtes, pour distinguer les appels prioritaires, puis rejeter les appels qu'il ne peut traiter faute de ressource (de temps). Un appel rejeté n'est jamais "éjecté": il vaudrait mieux le dire "ignoré". On note:

- L_{off} le nombre d'appels nouveaux arrivant par seconde,
- L_{ec} le nombre d'appels traités avec succès par seconde,
- L_{rej} le nombre d'appels traités *puis rejetés*,
- L_{tot} le nombre total d'appels présentés (/seconde),
- s le temps pour servir un appel en entier,
- τ le temps pour examiner puis rejeter un appel malheureux (on supposera $\tau < s$),
- ρ_0 la charge à vide (travail de fond du processeur)

A noter, les appels "rejetés" ne disparaissent pas, on les détecte à nouveau au cycle suivant; il y a répétition, et de temps en temps abandon, selon le schéma déjà exposé. *Ainsi, la commande va devoir traiter non pas L_{off}, mais $\beta L_{off} = L_{tot}$ demandes.*

Tant que l'écoulement du trafic est fluide, on a:

$$\begin{aligned} \rho_0 + sL_{ec} &\leq 1 \\ L_{ec} &= L_{off} \end{aligned}$$

Mais en surcharge,

$$\begin{aligned} L_{tot} &= L_{rej} + L_{ec} \\ L_{tot} &= L_{off} + H.L_{rej} \\ 1 &= \rho_0 + sL_{ec} + \tau L_{rej} \end{aligned}$$

La première relation exprime la conservation des appels (ils sont soit traités, soit rejetés), la seconde est, encore une fois, la caractéristique du rebouclage de répétition; la dernière relation exprime que, en surcharge, le processeur travaille 100% de son temps. De ces relation, on élimine L_{rej} et L_{tot}, et:

$$L_{ec} = \frac{(1-H)(1-\rho_0) - \tau L_{off}}{s(1-H) - \tau}$$

ce qui montre que dans cette zone, le trafic écoulé décroit quand le trafic offert augmente... Pour la valeur-charnière $L_{off} = (1-\rho_0)/s$, on vérifie que le segment "montant" du régime fluide se raccorde au segment descendant. Remarquer aussi que la pente du segment descendant dépend en premier lieu de la valeur de H, et que lorsque $H \to 1$, le segment devient vertical. L'interprétation de ces variations est laissée au lecteur.

Le phénomène, inévitable, résulte de ce qu'on doit dépenser de l'énergie sur les clients qu'on ne saura traiter. Selon le système étudié, la cause intime du phénomène pourra varier, mais elle se traduira par une mise en équation analogue. Noter que les équations aboutissent à un modèle "linéaire", évidemment simplifié.

8.4 Moralité

On a montré tous les méfaits possibles des phénomènes d'engorgement et de surcharge. Des remèdes sont possibles, qui ont noms "procédures de contrôle de congestion", "régulation de charge", etc. La conception et l'étude de telles procédures demande tout d'abord que le phénomène cause soit compris. On a donné les principes: pour chaque application il faudra chercher les mécanismes particuliers dans lesquels ces principes s'incarnent. Retenons de ce chapitre la conclusion suivante:

> Lorsque l'intensité du trafic augmente - dès que les sources sont en état de percevoir l'existence d'une attente - des phénomènes parasites apparaîssent, liés aux comportements individuels (temporisation, impatience, répétitions), qui peuvent entrainer le système dans des zones de fonctionnement instables. Il faut évidemment incorporer ces comportements au modèle pour rendre compte de façon réaliste des phénomènes observés.

La modélisation du régime de surcharge est assez tolérante: il faut trouver des modèles qualitatifs de fonctionnement, observer l'allure du phénomène, plus que des quantifications précises.

Reste à étudier les mécanismes de contrôle. Mais ça, c'est une autre histoire...

Annexes

Pour en savoir plus

L'étude des phénomènes de surcharge dans les commutateurs téléphoniques est reprise de la référence [6]. D'autres formes de tels phénomènes s'observent dans tous les systèmes. Par exemple, le mauvais contrôle du degré de multiprogrammation provoque un écroulement des performances des systèmes informatiques: voir par exemple [3, 4, 8].

Complément

1. Reprendre le modèle de la Figure 3.1, en distinguant le cas des rejets des nouveaux clients (on notera p_1 la probabilité de cet événement) des rejets des répétitions (probabilité p_2). Montrer que la relation (1) devient:

$$\lambda = \frac{\lambda_0[1-(p_2-p_1)H]}{1-p_2H} \tag{8.1}$$

2. Dans le même cas de la figure 3.1, expliquer pourquoi on observe habituellement $p_2 > p_1$. On pourra s'aider d'un modèle pour le "système": supposons qu'il s'agit d'une file M/M/1/N. Comment calculer les probabilités correspondantes (on adopte un modèle simplifié, où les trafics offerts sont Poissonniens).

3. Considérons un système représenté par une file M/M/1. de paramètres λ et μ. Les clients manifestent un comportement d'impatience; dès que le temps de séjour (attente et service) dépasse une valeur T (propre à chaque client, variable aléatoire exponentielle de taux γ), ila abandonnent le système, même s'ils sont en cours de service.

a) Montrer que le système peut s'analyser par une chaîne de Markov, dont on précisera les paramètres. Donner la solution de ce système.

b) Ecrire la probabilité de départ par impatience, en fonction de la solution des équations. Ecrire le trafic écoulé efficacement.

c) Comment varie ce dernier, lorsque λ grandit? Il faut pour cela résoudre numériquement le système. Application numérique avec $\mu = 1$, $\gamma = 2$ ou 5 par exemple.

4. Reprendre l'exemple du réseau à capacité limitée de la Section 4. Faire un modèle de simulation pour 5 files en cascade. Observer les phénomènes.

5. Ecrire le modèle des répétitions (section 3.1) lorsque H ou p dépendent du rang de la tentative; lorsque des "obstacles" (caractérisés par des p_i) sont traversés en cascade (définition d'un p équivalent).

Pourquoi p est-il fonction du temps entre intervalles, pour les cas usuels?

Chapitre 9

La Surcharge : 2- Mécanismes de Contrôle

Puisque la surcharge provient d'une augmentation inconsidérée du trafic offert au réseau, les mécanismes de contrôle vont avoir comme tâche de mesurer et de limiter ce trafic. Une autre contrainte qu'on leur impose est d'accomplir cette limitation de la manière la plus rentable, c'est à dire de permettre au réseau de fonctionner au plus près de sa capacité maximale.

Enfin, la contrainte *d'équité* est aussi invoquée. Si le réseau se trouve en situation de pénurie, celle-ci doit être partagée équitablement entre les sources. Si une source voit son trafic augmenter, il ne faut pas qu'elle puisse écraser les autres sources, même si, vu du réseau, l'écroulement typique de la congestion ne se manifeste pas.

La difficulté du contrôle provient en majeure partie du *caractère distribué* des sources de trafic: trafic émanant de sources non coordonnées, absence d'organe de contrôle central qui régulerait en fonction de l'état "global" du réseau.

9.1 Contrôle dans un système centralisé

Nous allons décrire très sommairement l'approche mise en œuvre pour la régulation de charge dans une machine de type centralisée – c'est à dire dans le cas le plus favorable où l'organe de commande peut avoir une vue *exhaustive* et *instantanée* de l'état des ressources.

Le système temps réel va s'analyser, selon la démarche esquissée au Chapitre 5. On peut se le représenter comme un réseau de files d'attentes, avec probablement des mécanismes d'ordonnancement complexes. Quoiqu'il en soit, il existera, compte tenu du profil de trafic traité, une ressource plus chargée. C'est cette ressource qui va saturer en premier, quand la charge va augmenter, et c'est donc elle qu'il faut surveiller. En réalité, les profils du trafic ne sont pas immuables, et par conséquent "la" ressource la plus chargée pourra varier, et la surveillance sera le plus souvent multiple.

Imaginons pour simplifier que la ressource fragile est unique, et qu'il s'agit d'un processeur central (c'est un cas assez fréquent). Le principe du contrôle est fort simple: mesure de la charge de la ressource, c'est à dire de son taux d'occupation. Par exemple, on mesurera le

temps passé par le processeur au traitement des tâches de priorité les plus faibles. La mesure se fait cycliquement; on définit un intervalle de mesure T, par exemple 10 secondes; selon la valeur du temps total d'inactivité I dans le cycle, le processeur sera décrété plus ou moins chargé. La charge sera estimée par $1 - I/T$. En fonction du niveau détecté, les actions de correction ou de prévention seront lancées.

Remarque: C'est la charge du processeur, qu'on va mesurer, et non le taux des arrivées. Pourquoi? Simplement, parce que dans un système à file d'attente, la charge ($\rho = \lambda/\mu$) est le principal paramètre. Et qu'en situation de surcharge, *le temps de service n'est pas constant.* La prudence commande donc d'observer ρ et non λ.

Le processus de mesure

Sous cette description simpliste se cachent de redoutables pièges, qui vont conduire à complexifier le schéma. En premier lieu, la mesure est entachée d'incertitude – parce que les processus d'arrivée des requêtes et des services sont aléatoires. Si j'observe 2, 3 ou 4 secondes à plusieurs reprises, à l'état stationnaire, je ne retirerai pas la même information. Il faut tenir compte de l'imprécision inhérente au procédé de mesure.

Pour illustrer le dilemme du processus de mesure, supposons une charge de 80 %, un processeur fonctionnant comme un système M/M/1, avec une durée moyenne de service de 30 ms. Le cycle de mesure opère en comptant le temps d'inactivité sur 1 seconde, puis sur 3 ou 10 secondes. La moyenne de ce temps est de 0.2 seconde. Voici la statistique des observations réalisées (Figure 9.1).

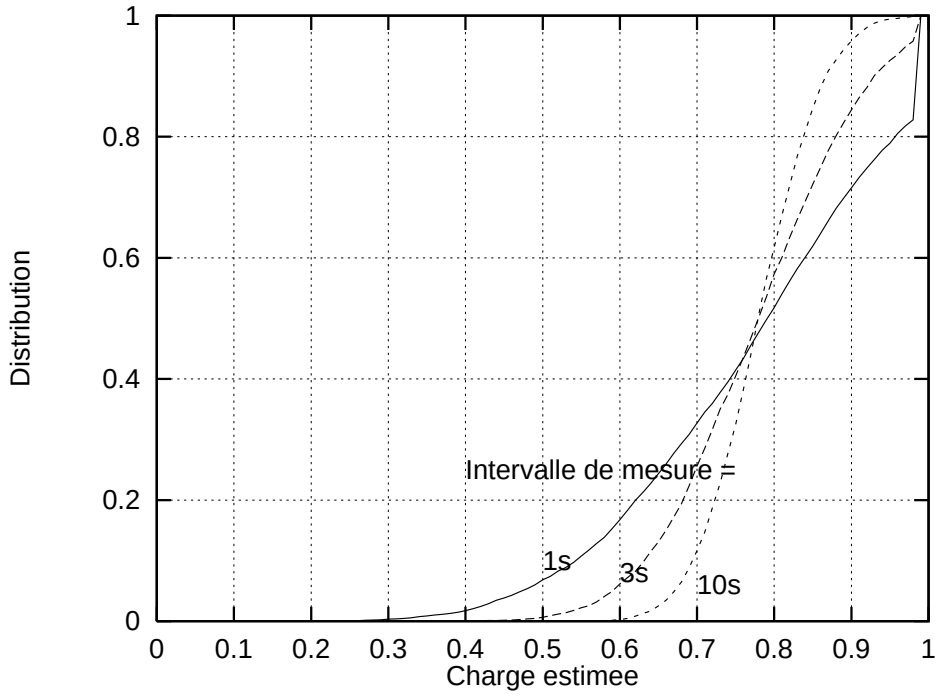


FIG. 9.1 – *Distribution des temps d'inactivité observés*

Comme on le remarque, une valeur de cycle de 1 s conduit à surestimer la charge, dans 35

cas sur 100 elle sera estimée supérieure à 90%; mais dans 20 cas sur 100 elle sera perçue comme inférieure à 65%.

On corrigera ce défaut en allongeant le cycle de mesure. Un cycle de 10 secondes donne une valeur supérieure à 0.9 pour 5% des mesures seulement.

Mais, en même temps, l'allongement de la durée du cycle ne règle pas la question. Ou plutôt, il introduit un autre défaut, celui de l'allongement du temps de réaction. Dans notre exemple, il faut 10 secondes avant de détecter de façon sûre une surcharge, temps pendant lequel elle pourra se propager et s'amplifier. C'est dire que le choix des paramètres d'un tel dispositif requiert un compromis obtenu après des réglages assez minutieux.

Les actions de défense

Quelle action entreprendre, une fois la surcharge détectée? Le plus souvent, la surveillance réagira progressivement, selon le franchissement de paliers dans la charge. Par exemple, un premier palier sera défini pour une charge de 80 %, un deuxième pour 85 %, etc. A chaque palier sera attachée une action d'efficacité (et de gravité) grandissante.

Les systèmes temps-réel sont assez souvent dotés de mécanismes d'autoserveillance (exploration et test cyclique des éléments actifs). Ce test consomme une partie de la ressource. La première action consistera à désactiver ces tests (dont le système peut évidemment se passer, pourvu qu'ils soient rétablis en régime moins chargé).

Les actions suivantes amèneront inévitablement à refuser l'accès du système à de nouvelles requêtes. Le plus souvent, on pourra établir une hiérarchie dans les rejets. Ainsi, dans un commutateur du réseau téléphonique on restreint d'abord l'accès des appels au départ (afin de favoriser les appels arrivants, qui ont eux déjà consommé une partie des ressources du réseau).

Une faiblesse de ce type d'action se trouvera dans les contraintes posées par l'organisation du système, par exemple la nécessité de détecter et d'analyser une demande avant de pouvoir statuer sur son sort: le traitement des requêtes conduit à un gaspillage inévitable de la ressource (cf. à ce sujet le modèle esquissé au chapitre précédent).

9.2 Les Réseaux “Store-and-Forward"

Les réseaux de paquets actuels (Transpac, par exemple) utilisent un mode de travail dit “Store-and-Forward". Cela signifie que l'information (structurée en “paquets") est transmise de proche en proche, de nœud en nœud, depuis l'origine jusqu'à la destination. Les nœuds intermédiaires accomplissent les fonctions des couches de protocoles de niveau 1 à 3: contrôle et correction des erreurs, acheminement, contrôle de flux, et de congestion.

9.2.1 Contrôle de flux

Le contrôle de flux vise à asservir le débit de la source à celui du récepteur. C'est une fonction “de bout-en-bout". C'est, aussi, une fonction de contrôle de congestion. C'est à dire qu'une mauvaise régulation du flux émis va provoquer une congestion sur le chemin de la connexion: si la source émet plus vite que le récepteur ne reçoit, les paquets vont s'accumuler dans le

réseau, d'abord dans le nœud de sortie, puis dans le précédent, etc.

Le mécanisme de Fenêtre

Un mécanisme simple et universel est celui de la *fenêtre*. La source se voit allouer un certain nombre de crédits W. Elle peut émettre un paquet à condition d'avoir un crédit (et le paquet consomme le crédit correspondant). Le récepteur acquitte les paquets reçus (et traités). L'accusé de réception régénère les crédits de l'émetteur, l'autorisant ainsi à poursuivre.

L'émetteur numérote les paquets qu'il envoie. A l'instant t, les paquets $n-2, n-1, n$ sont acquittés. Il peut envoyer le numéro $n+1$, mais aussi anticiper en envoyant $n+2, ..., n+W$: on parle d'un mécanisme de fenêtre d'anticipation (qu'on dit "fenêtre glissante", l'acquittement du $n+1$ faisant déplacer la fenêtre d'un cran).

Comment calculer la taille de la fenêtre W ("W" comme "window")? Un modèle classique par réseau de files peut aider. C'est un réseau fermé.

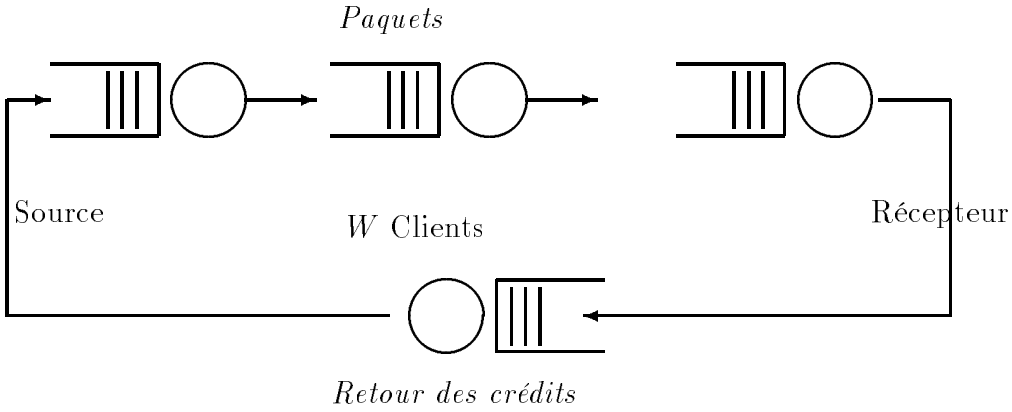


FIG. 9.2 – *Modèle à files d'attente du contrôle par fenêtre*

On peut le rendre plus réaliste, au prix d'une complication, en introduisant les flux incidents: dans chaque nœud traversé, le flux pisté doit affronter d'autres courants qui occupent les mêmes ressources. De même, chaque nœud peut être représenté par un modèle plus complexe.

Remarque: La terminologie n'est pas du tout fixée. La distinction contrôle de flux / contrôle de congestion est souvent omise. Voir par exemple la plupart des articles d'un numéro spécial IEEE Transactions on Communications, avril 1981.

9.2.2 Contrôle de Congestion

Le contrôle de flux s'exerce de bout en bout. Le contrôle de congestion vise à protéger le réseau, et pourra s'exercer de proche en proche. Il utilisera le cas échéant les mêmes outils.

Par exemple, on protège un nœud en instaurant une régulation par fenêtre sur chaque flux incident. Si la somme de chaque allocation accordée aux flux concernés est égale à la capacité mémoire disponible, il n'y a pas de phénomène de congestion lié aux capacités limitées.

Cette méthode est fiable: le noeud récepteur asservit parfaitement le débit de l'émetteur voisin à ses possibilités instantanées. C'est une heureuse conséquence du mécanisme"store-and-forward".

Malheureusement, cette méthode présente l'inconvénient majeur d'être très coûteuse en bande passante. Supposons que le nœud examiné se partage entre 100 circuits virtuels de débits identiques. Il divise sa capacité en 100 parties, et la partage entre les circuits. La variabilité des trafics issus de chaque source va rendre cette solution très défavorisante, la mémoire sera toujours vide, etc (des crédits sont alloués, et "gelés", sur des circuits inactifs momentanément).

On a donc dû imaginer d'autres méthodes, plus avantageuses. La première idée consiste à gérer les crédits globalement pour tout le réseau (le réseau peut accomoder au total N paquets, on crée N crédits, dispersés sur tous les nœuds). Le *contrôle isarithmique* utilise ce principe.

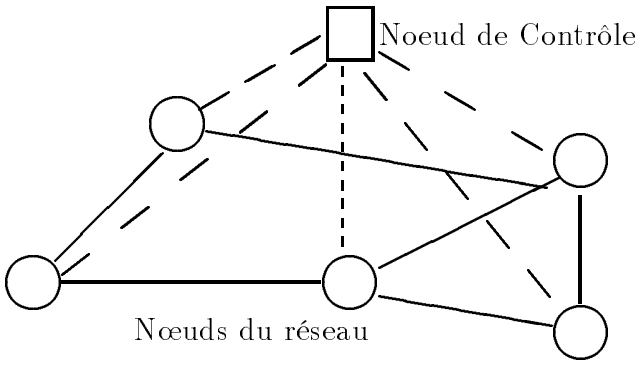


FIG. 9.3 – *Contrôle isarithmique d'un réseau maillé*

On construit un réseau de files d'attentes représentant le mouvement des jetons. C'est un réseau fermé – si le nombre des jetons est fixe. Dans le nœud de contrôle, on peut mettre en place des mécanismes complexes: génération ou destruction de jetons; accélération ou retard de l'envoi des jetons en fonction de l'état du réseau. Par exemple, on modélisera cette fonction par un taux de service dans le nœud de contrôle de la forme $\mu(n)$.

Une autre approche intéressante est appelée *input-buffer limiting*. Présentons l'argumentaire, dans un cadre simplifié. On supposera un réseau homogène: tous les nœuds sont"identiques", statistiquement. Le canal de sortie du nœud type se présente comme un serveur précédé d'une file de capacitéN, alimenté par un flux de transit λ_T et un flux de provenance locale λ_L. Chacun de ces flux subit un rejet dû à la capacité limitée, noté (respectivement) B_T et B_L. Le flux admis sera donc $\gamma_T = \lambda_T(1 - B_T)$ et $\gamma_L = \lambda_L(1 - B_L)$. Une fraction p du trafic est destinée au nœud, $1 - p$ est en transit. En moyenne, on renvoie autant de paquets qu'on n'en reçoit:

$$\gamma_T = (1 - p)\gamma_T + \gamma_L$$

Noter que $1/p$ est le nombre moyen de nœuds traversés par un paquet avant d'atteindre sa destination. Les paquets émis par le nœud vont peut-être se faire bloquer au nœud suivant. B_T est la probabilité de ce blocage (par homogénéité). Cela revient à dire que le débit

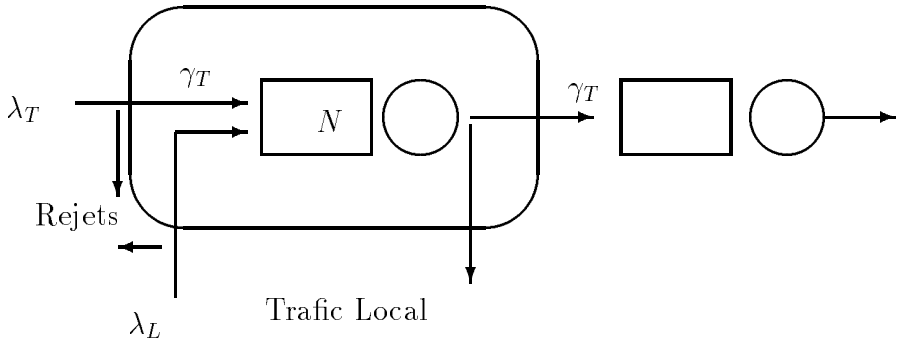


FIG. 9.4 – *Modèle d'un nœud en isolation*

effectif du serveur est non pas μ, mais $\mu(1 - B_T)$. On supposera enfin que chaque serveur est assimilable à un système M/M/1/N.

Lorsqu'aucun contrôle n'est institué, le fonctionnement est semblableà la cascade étudiée plus haut. Puisque les flux sont traités indistinctement, $B_T = B_L = B$.On a :

$$B = \frac{a^N(1-a)}{1-a^{N+1}} \qquad a = \frac{\rho}{1-B}$$

Comme plus haut, cette équation donne lieu au phénomène de congestion. Le contrôle consiste à discriminer les paquets selon leur provenance. On instaure un seuil N_L, si les paquets locaux atteignent ce seuil on les rejette. Le nœud obéit aux conditions:

$$\begin{aligned} 0 &\leq n_L \leq N_L \\ 0 &\leq n_L + n_T \leq N \qquad (N_L < N) \end{aligned}$$

Au terme d'une analyse approchée, on peut montrer qu'un choix judicieux du partage supprime le phénomène indésirable d'engorgement.

(la courbe provient de [15]. Le cas d'un réseau de configuration arbitraire donne lieu à une analyse analogue, quoique plus complexe. Voir la référence Lam et Reiser.

9.3 Le cas des Réseaux Haut Débit

L'étude du contrôle de la surcharge dans les réseaux haut débit prend une importance fondamentale. D'une part les conséquences des engorgements sont à la mesure des débits mis en jeu; en même temps, les mécanismes possibles atteignent les limtes de leur efficacité. D'autre part, la "surcharge" devient un élément normal du fonctionnement de ces réseaux afin d'en assurer un régime optimal. C'est notamment le cas de l'Internet comme de l'ATM (mode ABR), ou du Frame Relay.

C'est dire l'importance et la difficulté de cette opération. C'est dire, aussi, que son étude sort du cadre de cet exposé: on se reportera aux cours de présentation de ces techniques.

Repères bibliographiques

La plupart des références traitant de réseaux de files abordent le contrôle, à titre d'application. Sur le sujet d'actualité de ce chapitre, il faut consulter les revues pour avoir une présentation à jour des solutions et des tendances. Voici un échantillon de telles références.

IEEE Transactions on Communications, April 1980. Numéro spécial.

IEEE Transactions on Communications, April 1981. Numéro spécial sur le contrôle de congestion.

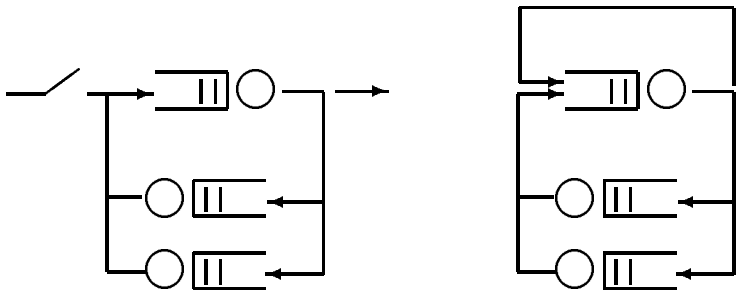
IEEE Journal of Selected Areas in Communications (JSAC). Avril 1991. Numéro spécial: évaluation des performances des réseaux large bande.

S. Lam, M. Reiser. *Congestion Control of Store-and-Forward networks by Input Buffer Limit - an analysis.* IEEE Transactions on Communications, janvier 1979.

Compléments

1. **Contrôle de la charge d'un système informatique.**On a vu au Chapitre précédent les méfaits du mauvais contrôle dansun système informatique temps-réel (le commutateur).

On corrige enlimitant le nombre de clients en service simultanément.



(a) (b)

Le réseau de files est un modèle (rudimentaire) du système. Le serveurS1 repésente le processeur central, S2 et S3 les périphériques. Les probabilités de routage en sortie de S1 sont p_2 et p_3 (versS2 et S3) et p (fin de service), avec $p_2 + p_3 + p = 1$ .Le principe du contrôle est de limiter à N le nombre de clients.On cherche à déterminer N optimal. Plaçons nous en surcharge.Montrer que le réseau figure (b) est un modèle du fonctionnement. Montrer qu'on peut obtenir le débit en fonction de N, ainsi que le temps moyen de réponse. Justifier l'utilisation de ces paramètrescomme critères de fonctionnement.

Application numérique:$p_2 = p_3 = 0.475, p = 0.05, \mu_2 = \mu_3 = 1000/s, \mu_1 = 3000/s$. Tracer les courbes donnant ces figures de performance en fonction de N.

Bibliographie

[1] W. Bux, H.L. Truong: *Mean delay approximation for cyclic service queueing systems.* Performance Evaluation, vol 3, pp 187-196, 1983.

[2] Ehran Cinlar: *Introduction to Stochastic Processes.* Prentice Hall 1975.

[3] G. Doyon: *Systèmes et Réseaux de Télécommunications en régime stochastique.* Coll Technique et Scientifique des Télécommunications, Masson 1989.

[4] William Feller: *Introduction to Probability Theory and its Applications*

[5] E. Gelenbe, I. Mitrani. *Analysis and Synthesis of Computer Systems.* Academic Press, 1978.

[6] E. Gelenbe, G. Pujolle. *Introduction aux réseaux de files d'attentes.* Eyrolles, 1982.

[7] Gross et Harris: *Fundamentals of Queueing Theory.* J. Wiley, 2nde édition, 1985.

[8] G. Hébuterne. *Ecoulement du trafic dans les autocommutateurs.* Masson, 1985.

[9] R. Jain: *The art of computer systems performance analysis.* J. Wiley, 1991.

[10] F.P. Kelly: *Reversibility and Stochastic Networks* J. Wiley, 1979 (2 volumes).

[11] L. Kleinrock: *Queueing Systems.* J. Wiley, 1975 (2 volumes).

[12] S.S. Lavenberg: *Computer Performance Modelling Handbook.* Academic Press, 1983.

[13] T.G. Robertazzi: *Computer Networks and Systems: Queueing Theory and Performance Evaluation.* Springer Verlag 1990.

[14] David Ruelle: *Hasard et Chaos.* Editions Odile Jacob, 1991.

[15] M. Schwartz: *Telecommunication Networks: Protocols, Modeling and Analysis.* Addison Wesley 1987.

[16] H. Takagi: *Queueing Analysis. Vol. 1, vacations and priority systems.* North Holland, 1991.