DM2 : optimisation de routage

Hoël Boëdec, Mickaël Fournier

11 Janvier 2016

Contents

Routage Bernoulli

1.

Définition : Avec le processus de comptage N([a,b]) = nb de points dans [a,b] de T_i pour i entier naturel, on dit que T_i pour i entier natuel est un processus de Poisson de paramètre λ) si N a les 2 propriétés : - N[a,b] et N[c,d] sont indépendants si $[a, b] \cap [c, d] = \emptyset$ - N[a,b] suit une loi de poisson de taux $\lambda \cdot (b - a)$

On note X la variable aléatoire qui définie le nombre de paquets qui arrivent au routeur dans un intervalle de temps [a, b] avec $a \neq b$ et $a > 0$, $b > 0$. Cette variable aléatoire suit une loi de Poisson de paramètre λ. On suppose qu'un paquet a une probabilité p d'être routé dans la file 1 et une probabilité p-1 d'être routé dans la file 2.
Notons maintenant Y la variable aléatoire qui définie le nombre de paquets qui arrivent dans la file 1 dans l'intervalle de temps [a, b]. Si on suppose que n paquets arrivent au routeur dans [a, b] alors Y est la somme de n variable aléatoires indépendantes suivant une loi de Bernouilli de paramètre p. Ainsi, Y suit une loi Binomiale de paramètres n et p. Sa loi de probabilité est donc :

$$\begin{aligned}
P_Y(i) &= \sum_{n=0}^{\infty} P(Y = i|X = n) \cdot P_X(n) \\
&= \sum_{n=i}^{\infty} \binom{n}{i} \cdot p^i \cdot q^{n-i} \cdot e^{-\lambda} \cdot \frac{\lambda^n}{n!} \\
&= \sum_{n=i}^{\infty} \frac{p^i \cdot q^{n-i} \cdot e^{-\lambda} \cdot \lambda^n}{i! \cdot (n-i)!} \\
&= \frac{e^{-\lambda} \cdot (\lambda \cdot p)^i}{i!} \sum_{n=i}^{\infty} \frac{(\lambda \cdot q)^{n-i}}{(n-i)!} \\
&= \frac{e^{-\lambda} \cdot (\lambda \cdot p)^i}{i!} e^{\lambda \cdot q} \\
&= \frac{e^{-\lambda \cdot p} \cdot (\lambda \cdot p)^i}{i!}
\end{aligned}$$

Ainsi, on a prouvé que Y suit une loi de poisson de paramètre $\lambda \cdot p$. De plus, comme les paquets qui arrivent dans la file 1 ne dépendent que des paquets arrivant au routeur alors pour a, b, c et d tels que $[a, b] \cap [c, d] = \emptyset$ on a que le nombre de paquets arrivant dans [a, b] est indépendant du nombre de paquets arrivant dans [c ,d].

D'après la définition notée au début de la question, nous pouvons alors affirmé que les paquets qui arrivent dans la file 1 forment un Processus de Poisson de taux $\lambda \cdot p$.

2.

Il faut que $\lambda \cdot p < \mu_1$ et $\lambda \cdot (1 - p) < \mu_2$. On évitera ainsi que les messages s'accumulent dans le système.

3.

On sait que la quantité moyenne de paquets en régime stationnaire dans le système vaut $\bar{N} = \frac{\lambda \cdot p}{\mu_1 \times (1 - \frac{\lambda \cdot p}{\mu_1})} + \frac{\lambda \cdot (1-p)}{\mu_2 \times (1 - \frac{\lambda \cdot (1-p)}{\mu_2})}$.

Si on considère ce résultat comme une fonction en fonction de p. C'est un polynôme du second degré dont le coefficient du monôme de plus haut degré est positif. Ainsi, il existe un p qu'on note p^* tel que $\bar{N}$ est minimum. Après calcul on trouve $p^= \frac{1}{\lambda} \times (\mu_1 - \frac{\mu_2 + \mu_1 - \lambda}{1 + \sqrt{\mu_2 \cdot \mu_1}})$.

4.

Pour $\mu_1 = \mu_2 = \lambda = 1$ on trouve $p^* = 0.5$. Ce résultat est intuitif car le temps de service dans les deux serveurs sont égaux. Ainsi, autant partager les tâches équitablement entre les deux serveurs.

Pour $\mu_2 = \lambda = 1$ et $\mu_1 = 2$ on trouve $p^* = 0.82$ environ. Le fait que le résultat soit supérieur à 0.5 paraît intuitif car le serveur 1 a un temps de service doublement supérieur à celui du serveur 2. Ainsi, autant donné plus de paquets par seconde au serveur 1 afin de minimiser le nombre moyen total de paquets dans le système.

Routage périodique

1.

Le routage est ici périodique :

- À partir d'une date t_a le système envoie n paquets sur un serveur.
- Puis, à partir d'une date t_b le système envoie un paquet sur l'autre serveur.
- À une date t_c le système recommence à envoyer n paquets sur le premier serveur.

Pendant l'intervalle t_c - t_a le système doit donc envoyer exactement n paquets sur la file 1.
Si le processus d'arrivée sur la file 1 était de Poisson, on aurait :
Pour tout t_0=0 $< t_1 < \ldots < t_k$, (N_{t_k} - $N_{t_{k-1}}$, ..., (N_{t_1} - N_{t_0}) indépendantes. (En notant (N_t) le processus de poisson).

On aurait donc notamment (N_{t_c} - N_{t_b}) et (N_{t_b} - N_{t_a}) qui seraient indépendants. Or si (N_{t_c} - N_{t_b}) = n, alors (N_{t_b} - N_{t_a}) = 0 donc l'indépendance n'est pas vérifiée et le processus de la file 1 n'est pas de Poisson.

2.

Il faut que $\lambda * n < \mu_1$ et $\lambda < \mu_2$.

3.

Notre simulateur a deux éxécutables : le premier (Main.java) permet de faire afficher le nombre de paquets moyens en attente pour λ, $\mu 1$ et $\mu 2$ donnés à l'aide d'une boucle sur le nombre de paquets envoyé au serveur 1. Dans la classe principale qu'est Simulator on alimente une queue de priorité triée suivant un temps. On considère deux types de requêtes : une requête d'arrivée au routeur et une requête de départ du Service.

Le deuxième (GraphSimu.java) permet de faire afficher le nombre de paquets moyen en attente en fonction du temps pour λ, $\mu 1$, $\mu 2$ et n donnés.

Dans notre simulateur on retrouve les classes principales suivantes :

- Simulator : définie par deux files d'attente, deux services, une queue de priorité pour stocker les événements, ainsi que d'autres attributs servant à son bon fonctionnement.
- Requete : définie par un type (départ ou arrivée) et un numéro de file, elle permet de stocker les événements dans la queue de priorité du Simulator.
- Service : définie par une instance de Exponentielle (permettant d'implémenter la distribution exponentielle de paramètre μ) et un temps représentant son occupation.
- Stat : définie par un temps et un nombre de paquets dans la file 1 et dans la file 2, elle permet d'exploiter les résultats à la fin de la simulation.
- FileAtt : définie par le nombre de personnes la composant.

Les intervalles de confiance de la moyenne sont calculés grâce aux méthodes intervalle_de_confiance, calculateAverage, variance et la classe statique Intervalle. La classe Intervalle a deux attributs qui représentent l'intervalle de confiance de la moyenne. Nous avons utilisé la formule suivante : $[moyenne - 2\sqrt{\frac{variance}{N}}; moyenne + 2\sqrt{\frac{variance}{N}}]$.

4.

1. Cas où $\lambda = \mu 1 = \mu 2 = 1$
   Par simulation on trouve $n^* = 1$ avec un nombre de paquets moyen en attente de 0.97 avec un intervalle de confiance [0.96, 0.98] (10 simulations).
2. Cas où $\lambda = \mu 2 = 1$ et $\mu 1 = 2$
   Par simulation on trouve $n^* = 2$ avec un nombre de paquets moyen en attente de 0.47 avec un intervalle de confiance [0.46, 0.47] (10 simulations).

5.

Pour $\mu_1 = 1$ alors p = 0.5 : ($\rho_1 = p \cdot \lambda \cdot \mu_1 = 0.5$ et $\rho_2 = (1-p) \cdot \lambda \cdot \mu_2 = 0.5$) donc $\bar{N} = 2$.
Pour $\mu_1 = 2$ alors p = 0.82 : ($\rho_1 = 0.41$ et $\rho_2 = 0.18$) donc $\bar{N} = 0.91$.

On voit avec ces résultats que le routage périodique est plus efficace que le routage Bernouilli. En effet, pour les deux cas le nombre de paquets moyen en attente est inférieur pour le premier que pour le second.
Cependant pour d'autres valeurs de μ_1, μ_2 et λ ceci n'est pas le cas. !!compléter!!

Routage optimal

1.

Les paquets qui sont envoyés en haut et en bas sont indépendants de ce qu'il se passe dans les files. Ainsi, nous pensons que si au moment de router vers la file 1 ou vers la file 2 le routeur voit l'état du système alors on peut prendre une meilleure décision que les précédentes.