TP Ptolemy II

22/09/09

LECRUBIER Vincent

1 Modelisation de l'evolution de la taille d'une fille d'attente

Pour cet exercice, il faut utiliser l'acteur « Counter » qui dispose d'une entrée d'incrémentation et d'une entrée de décrémentation. Le compteur représente la longueur de la file d'attente, il sera incrémenté lors de l'arrivée d'un événement, et décrémenté lors de la sortie d'un évenement.

Le problème rencontré est le fait que la file d'attente ne doit pas avoir une longueur négative.

La première solution envisagé a été le positionnement d'un composant appellé « Limiter » en sortie du compteur, ce qui a pour effet d'écreter toutes les valeurs négatives en sortie, en les remplaçant par la valeur zéro. Cependant la valeur du compteur pouvait parfois être négative pendant de longues périodes de temps, ce qui avait pour effet une valeur nulle de la sortie du composant, malgré la présence d'évenements en entrée. Cette solution n'est donc pas valable.

La solution valable consiste à filtrer l'entrée représentant les sorties, en ne laissant pas passer les messages de décrémentation du compteur si la valeur de celui ci est négative ou nulle. Un comparateur est utilisé à cet effet, combiné à un « Boolean switch » qui ne bloque les messages indésirables. La valeur de la constante « zéro » alimentant le comparateur a besoin d'être activée à chaque fois que la sortie du compteur est modifié, c'est pourquoi le composant « Const » a une entrée branchée sur la sortie du compteur.

Lors de l'exécution, une boucle infinie est créée, matérialisée par la chaine « Counter – Comparator – Boolean Switch - Counter ». Afin de casser la boucle, il a fallu inserer un composant « Timed Delay » qui découple les éléments et permet au solveur de simuler le modèle.

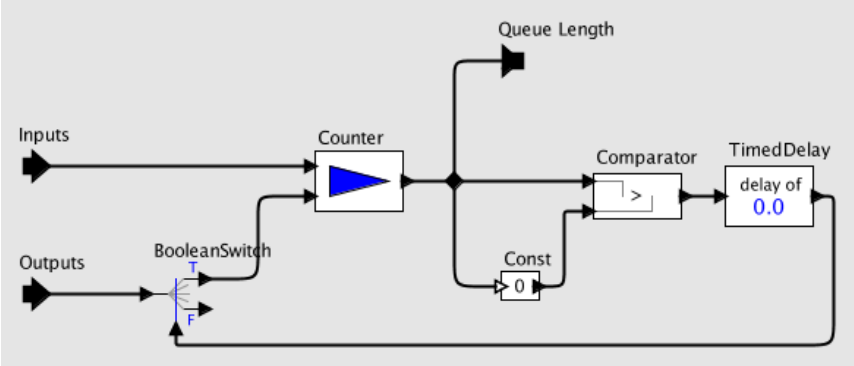

Illustration 1: Schéma du composant modelisé

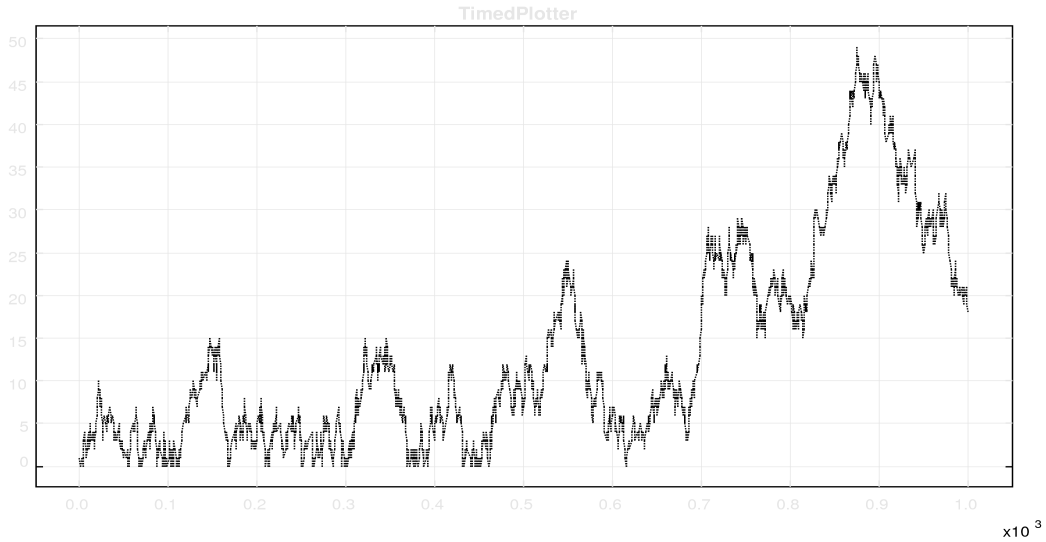

Illustration 2: Evolution de la longueur de la file en fonction du temps

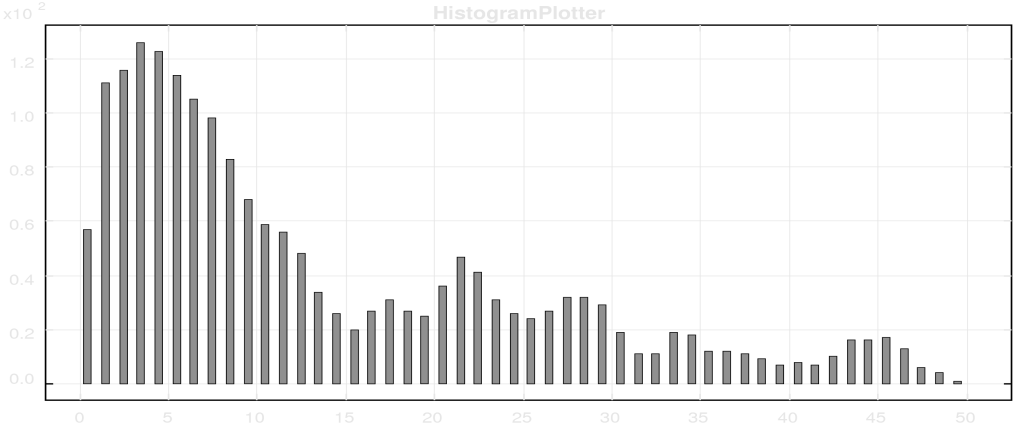

Illustration 3: Histogramme de la distribution de la longueur de la file.

2 Mesure d'une latence de bout en bout

L'objectif de ce problème est de modeliser deux serveurs en série, et de mesurer le temps de latence entre les entrées et les sorties.

Afin de modeliser des serveurs à temps de traitement variable, nous utilisons un composant « Gaussian » qui permet de modifier le temps de traitement de chaque serveur. Ces composants « Gaussian » sont reliés à l'entrée de chaque serveur, afin d'actualiser leur valeur à chaque nouvel événement.

Pour mesurer les temps de traitement, nous utilisons un sous système, grace au composant « Composite actor ». Ce sous système prend deux entrées : les évenements en entrée de la série de serveurs, et les évenements en sortie. Un composant « Current Time » est placé à la suite de chaque entrée, afin de récuperer la date associée à chaque évenement. Puis la date correspondant à l'évenement d'entrée est stockée dans une file d'attente, et est dépilée lorsqu'un événement advient en sortie de la série de serveurs. Les serveurs traitant les événements dans l'ordre chronologique, l'évenement en sortie du dernier serveur correspond forcément à l'évenement en sortie de la file d'attente. Il n'y a donc pas d'interversion possible entre les évenements.

En faisant simplement la différence entre la date d'entrée et la date de sortie, nous trouvons le temps de latence associé à l'évenement, qui est envoyé en sortie afin d'être analysé.

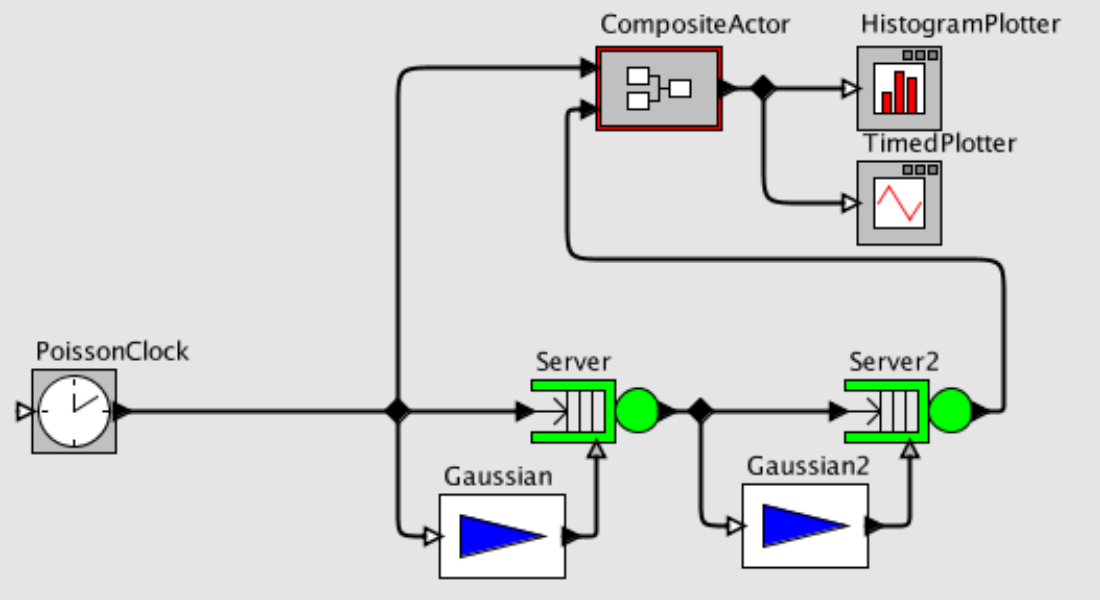

Illustration 4: Schéma global du problème. Le système de mesure du temps de latence est regroupé en un sous système "Composite Actor"

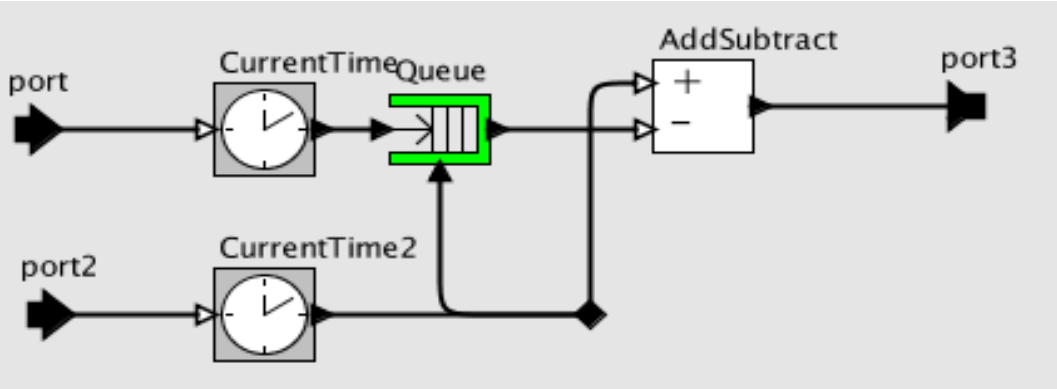

Illustration 5: Schéma du sous système de mesure du temps de latence entre l'entrée 1 et l'entrée 2.

En fonction des parametres, deux situations ont été identifiées.

- Dans la première, l'intervalle caracteristique d'arrivée des éléments dans le processus de poisson en entrée est du même ordre de grandeur ou plus petit que la somme des temps de traitment des serveurs. Dans ce cas, le temps de latence restera toujours borné, et son ordre de grandeur sera la somme des temps de latence des serveurs.
- Dans la seconde, les évenements arrivent trop rapidement pour les serveurs, qui saturent, et le temps de latence ne fait qu'augmenter dans le temps, les files d'attentes ont une longueur croissante qui tend vers l'infini. Cette situation a été rencontré au début de mon test, ce qui me laissait penser que le modèle était faux. En fait seul le paramètre de temps de l'horloge de poisson était trop faible par rapport au temps de traitement.

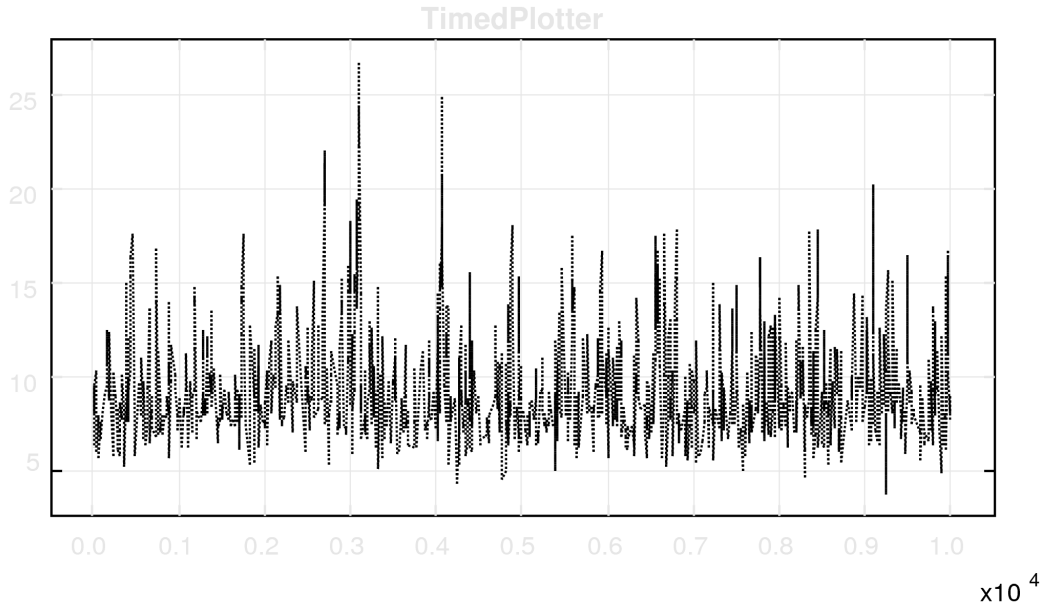

Illustration 6: Evolution du temps de latence de bout à bout, en fonction du temps, dans la situation favorable.

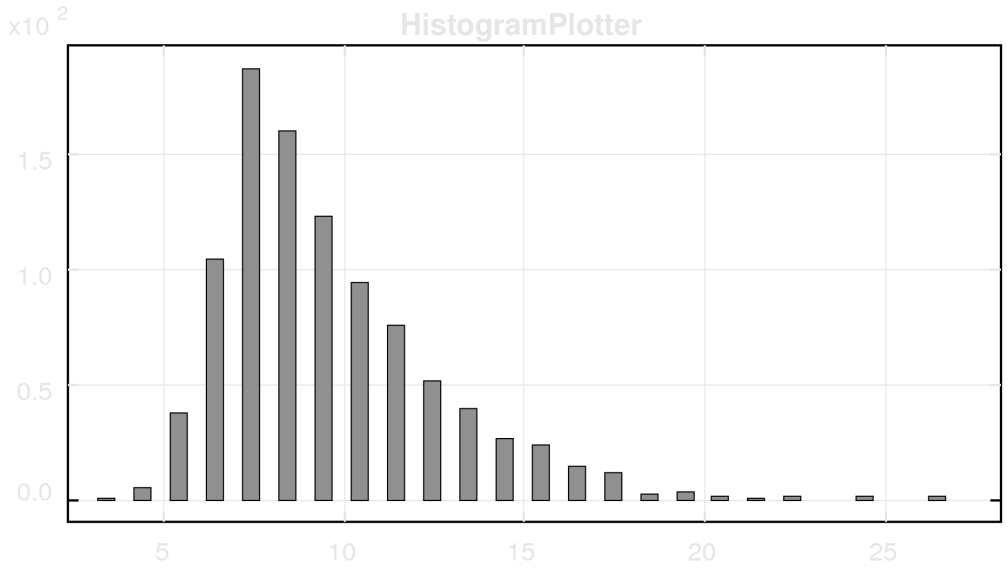

Illustration 7: Répartition des temps de latence, dans le cas favorable. Le pic correspond exactement à la somme des temps de traitement des serveurs.

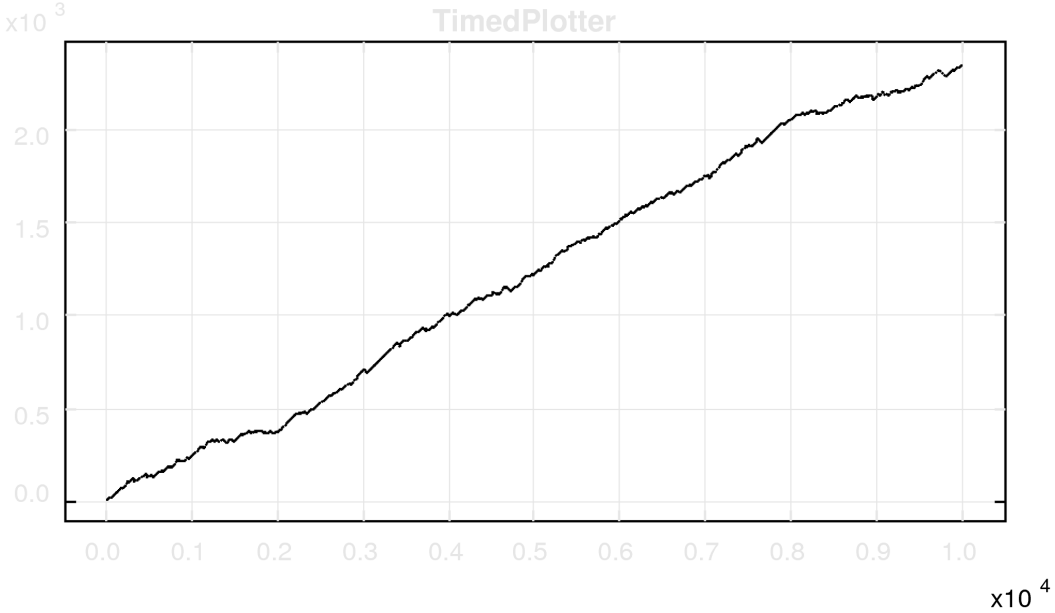

Illustration 8: Evolution du temps de latence de bout à bout, en fonction du temps, dans la situation défavorable. Le temps de latence tend vers l'infini.