Simulation de serveurs en parallèle et en série avec QNAP2

ISIMA |1 rue de la Chebarde, 63170 Aubière

TP1 - Simulation

Jael Vavasseur – ClÉment Mesnil

2019

Compte rendu

TP 1

Simulation à flux discrets

Jaël Vavasseur – Clément Mesnil Philippe Lacomme

Table des matières

[Introduction 3](#_Toc28729165)

[Partie 1 – Structures utilisées 5](#_Toc28729166)

[Partie 2 – Serveurs en série 6](#_Toc28729167)

[Partie 3 – Serveurs en parallèle avec probabilité 8](#_Toc28729168)

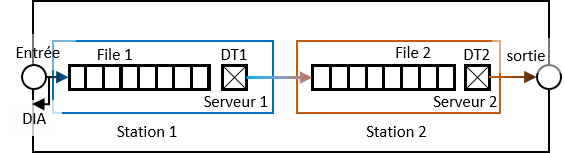
[Partie 4 - Serveurs en parallèle et réunion de pièces 9](#_Toc28729169)

# Introduction

Le but de ce TP est de modéliser et simuler en C++ des serveurs avec des files d’attentes en parallèle ou en série. Ce TP comporte plusieurs parties.

Tout d’abord, nous allons modéliser deux serveurs avec des files de capacités limitées, en série. Les durées de traitement des machines diffèrent et leurs capacités seront identiques. Évidemment, si la durée de traitement de la deuxième machine est trop importante par rapport à celle de la machine n°1, les pièces vont s’entasser dans la première file voire même bloquer l’arrivée des pièces.

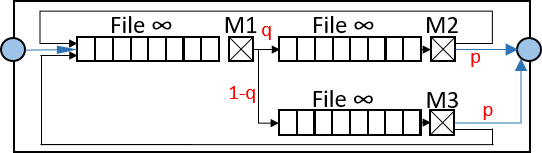
Nous allons donc étudier ce phénomène. De plus, nous modéliserons une péremption des pièces, c’est-à-dire une durée maximale d’attente dans la première file qui, si dépassée, rendra la pièce invalide.



*Figure 1. Schéma représentant la première modélisation :  
Serveurs en série*

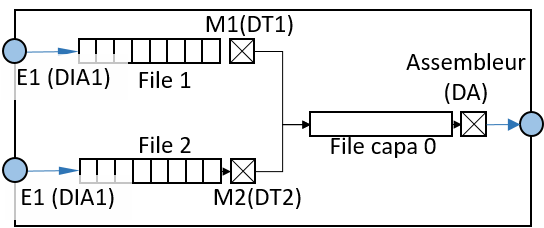
Dans un second temps, la modélisation effectuée sera composée de deux étapes : une première composée d’une seule machine sur laquelle toutes les pièces doivent passer avant d’aller vers, et une deuxième composée de deux machines en parallèle.

Après être passées sur la première machine, les pièces sont redirigées vers la machine M2 **ou bien** la machine M3, selon une probabilité **q**. Les files considérées dans cette partie sont supposées de capacités infinies.



*Figure 2. Schéma représentant la deuxième modélisation :  
Serveurs en série probabilisés*

Pour finir, nous avons modélisé un assemblage de pièces, c’est-à-dire que la modélisation est composée de deux groupes entrée / file / machine en parallèle et qu’une fois que deux pièces sont prêtes (1 pour chaque machine), elles sont mises sur une autre machine, appelée assembleur. Dans le cas où une pièce est prête mais pas la deuxième, la pièce prête est mise en attente sur sa machine et bloque donc celles qui la suivent.

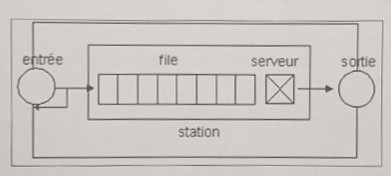


*Figure 3. Schéma représentant la troisième modélisation :  
Assembleur*

Pour réaliser ces différentes modélisations, nous avons dû mettre en place des structures en C++ nous permettant de gérer des pièces, des files, des machines et des entrées/sorties. C’est ce que nous allons détailler maintenant.

# Partie 1 – Préalable : File MM1

Avant de nous lancer dans des modélisations complexes, nous avons modélisé une file MM1, c’est-à-dire avec une entrée, une file et un serveur.



Nous avons choisi comme paramètres pour la simulation :

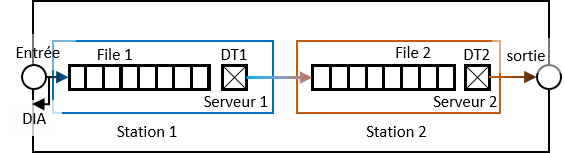
* T = 2 : durée inter-arrivée
* S = 4 : temps de travail de la pièce sur le serveur

De plus, si le nombre de clients sur le serveur est plus grand que 5 lorsqu’une pièce est censée arriver sur le serveur, on choisit de considérer la pièce comme perdue.

En lançant la simulation avec ces paramètres, pour une durée maximale de simulation de 300 unités, on obtient un nombre de pièces perdues de 20. Cette modélisation est satisfaisante et représente bien le modèle souhaité.

# Partie 2 – Serveurs en série

On souhaite maintenant modéliser deux files MM1 en série et vérifier son bon fonctionnement.



*Figure 1. Schéma représentant la première modélisation :  
Serveurs en série*

Nous avons choisi comme paramètres :

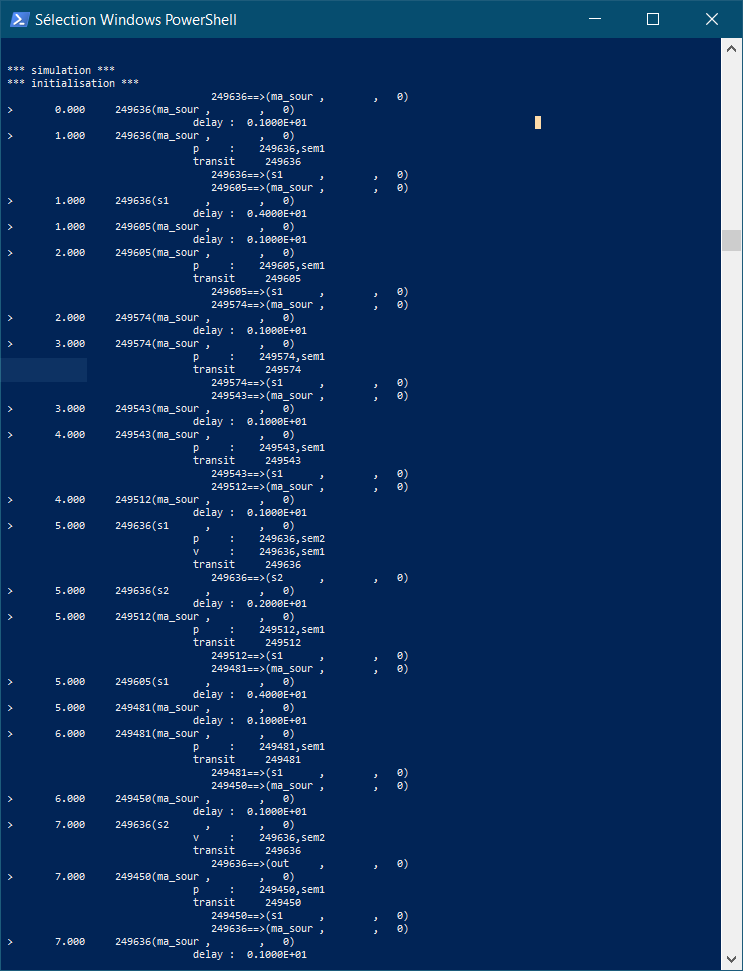
* Sa = 4 : durée de traitement sur le serveur 1
* Sb = 2 : durée de traitement sur le serveur 2
* DIA = 1 : durée inter-arrivée

En regardant la trace sur l’invité de commande, on s’aperçoit rapidement que la simulation est fonctionnelle et cohérente. En effet, on voit qu’à chaque nouvelle unité de temps, la source génère un nouveau client et l’envoie vers le serveur 1.

Après 4 unités de temps sur le serveur 1, on observe un autre transit du s1 vers le s2. Pour la première pièce générée à T = 1, ce transit s’effectue donc bien à T = 5.

Enfin, à T = 7, une nouvelle transition entre le serveur 2 et la sortie est effectuée. Cela est bien cohérent avec la théorie. A cela s’ajoute le fait que les autres clients générés continuent d’évoluer selon les paramètres fixés.

On peut donc affirmer que le déroulement du modèle est valide.



Envoie du client généré vers le serveur 1

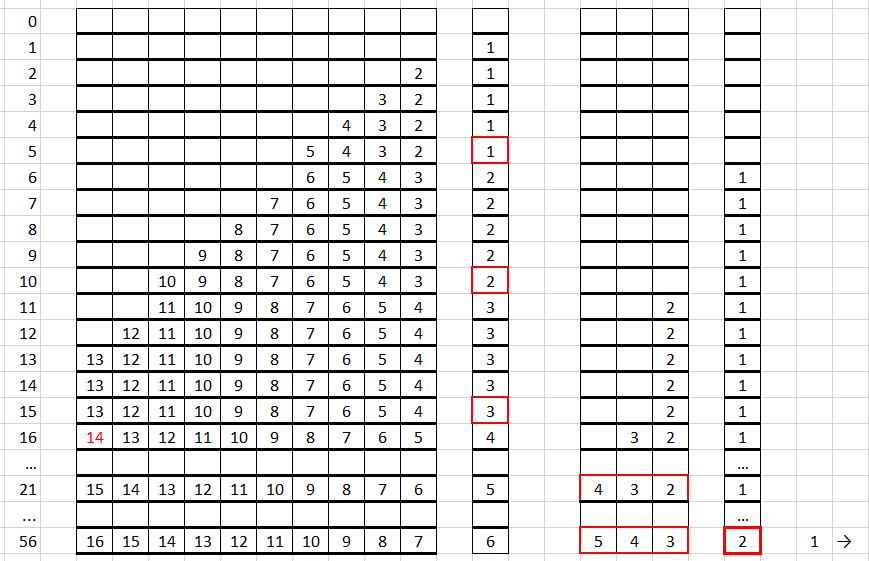
Transition entre le serveur 1 et le serveur 2

Le client sort de la simulation

Nous avons réalisé le même modèle avec différents paramètres :

* Sa = 5 : temps de traitement sur le serveur 1
* Sb = 50 : temps de traitement sur le serveur 1
* DIA = 1 : durée inter-arrivée
* L1 = 10 : taille de la file 1
* L2 = 3 : taille de la file 2

De cette manière, on va rapidement bloquer l’arrivée des nouveaux clients.

Trace théorique :

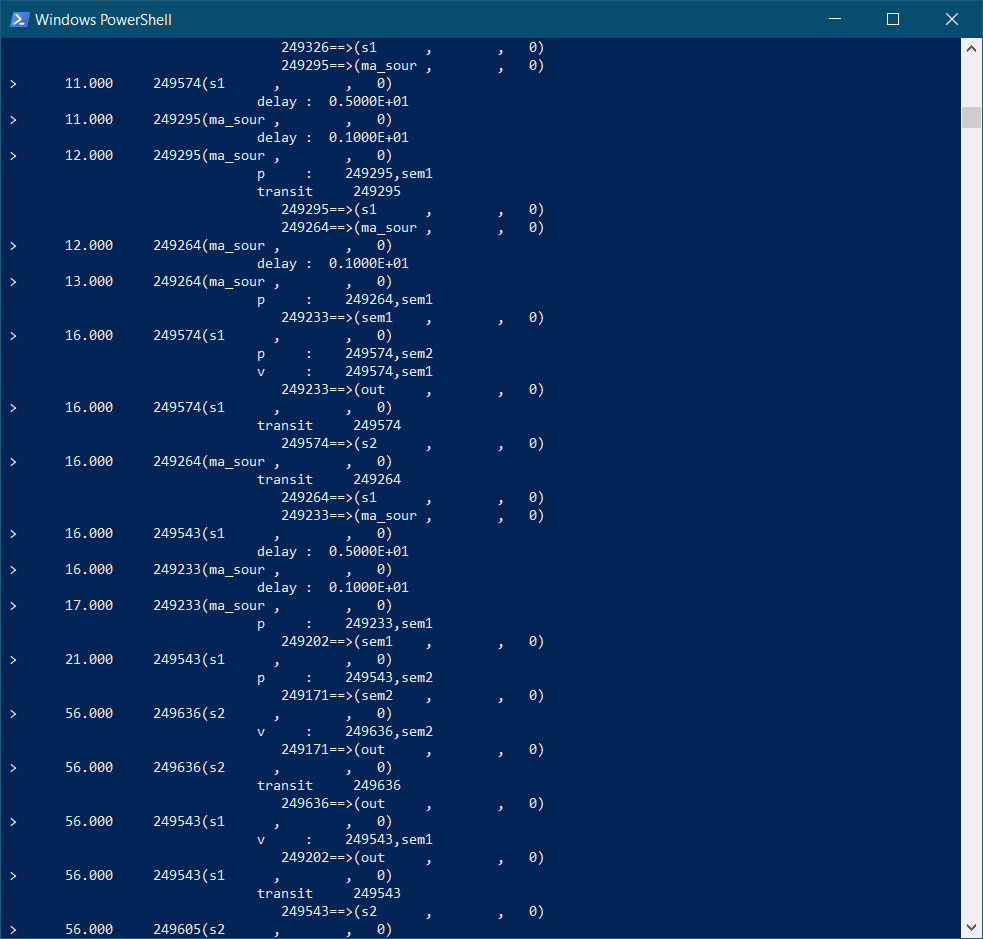
Il semblerait qu’à T = 13, la file 1 soit pleine et que l’entrée soit alors bloquée jusqu’à la date T = 16. Ce problème réapparaitra alors dès  
T = 17 et ce jusqu’à T = 21.

De plus, à la date T = 21, la 2e file serait bloquée jusqu’à ce que le client 1 quitte le 2e serveur donc jusqu’à T = 6 + 50 = 56.

Observons ce qui se passe lors de la simulation.

La simulation reprend vie lorsque le 1er client quitte le 2e serveur : à   
T = 56

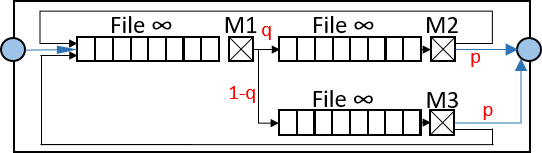
Même chose

Ainsi, notre simulation est correcte.

La file 1 est bien plein à T = 13, ce qui bloque l’entrée jusqu’à la transition de la pièce qui était sur le serveur S1 vers le serveur S2 :   
à T = 16

# Partie 3 – Serveurs en parallèle avec probabilité

Ce système est constitué de 3 files mais à chaque passage dans le système, une pièce va choisir aléatoirement une file parmi les 2 proposées et passera seulement dans celle-là.(cf. figure 2).

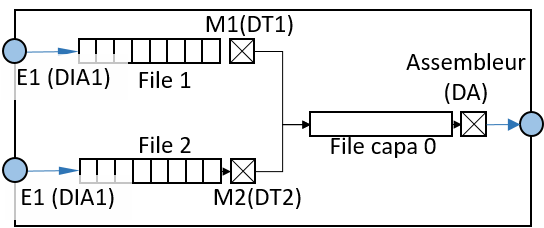


*Figure 2. Schéma représentant la deuxième modélisation :  
Serveurs en série probabilisés*

On ne s’intéresse pas ici au blocage des files ou au nombre de pièces perdues puisqu’on suppose les files comme étant infinies.

On retient toutefois la liste des machines par lesquelles les pièces sorties sont passées. De plus, avec cette modélisation, les temps à l’intérieur du système sont très longs voire, théoriquement infinis. On le voit avec la capture d’écran précédente : la pièce n°16 est entrée dans le système à la date **t = 30** et est sortie à la date **t = 906** pour une durée moyenne de séjour de **420 unités de temps**.

# Partie 4 - Serveurs en parallèle et réunion de pièces



*Figure 3. Schéma représentant la troisième modélisation :  
Assembleur*

Ce système (cf. figure 3) consiste en la réunion de 2 éléments pour former la pièce lors d’une troisième manipulation. La fabrication des 2 éléments doit être terminée avant défaire sortir n’importe quelle pièce des machines précédents. C’est-à-dire que si la première machine a terminé avant la deuxième, la première doit attendre la seconde et passe donc dans un état d’attente jusqu’à ce que la deuxième machine ait fini. Quand les 2 machines ont fini, on peut vider les 2 machines et on lance l’assemblage. La machine d’assemblage ne possède pas de file.

Nous avons modélisé ce problème avec les paramètres suivants :

* DIA1 = 1
* DIA2 = 10
* DT1 = 5
* DT2 = 2
* DAssemblage = X∼exp(10)