Rapport (Queue Dependent Virtual Machine)

Projet : Analyse des performance et de lútilisation des ressources dans un data center

Abderrezak AGGOUN
Bin LIU

Master2 Informatique AMIS Paris Saclay
UVSQ 2019-2020

Table des matières

1 Introdution

Nous voulons analyser les performances et l'utilisation des ressources dans un data center. Les ressources sont des VMs (Virtuel Machines) qui sont activitée et désactivitées en fonction de la demande. On modélisera le data center avec une file d'attente multi-serveur (où chaque serveur représente une VM), et avec une politique à seuils.

Notre sujet est pour le but de modéliser et d'analyser le système que nous avons construit par matlab/simevnent et aussi de générer la chaîne de Markov afin d'obtenir les mesures de performances en fonction de la distribution stationnaire obtenue par les équations de balance de la chaîne de Markov.

2 Présentation des théorème

2.1 Modèle de la simulation sur VM

[2]

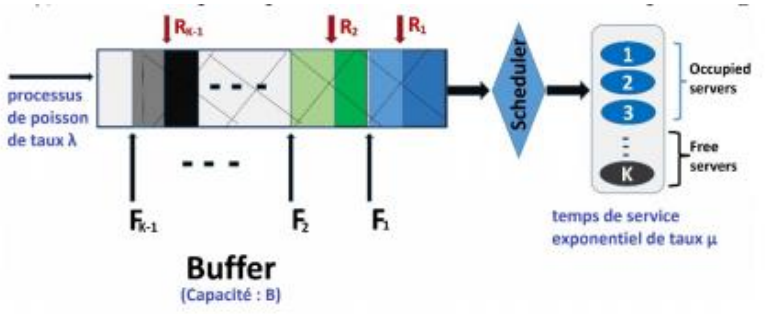
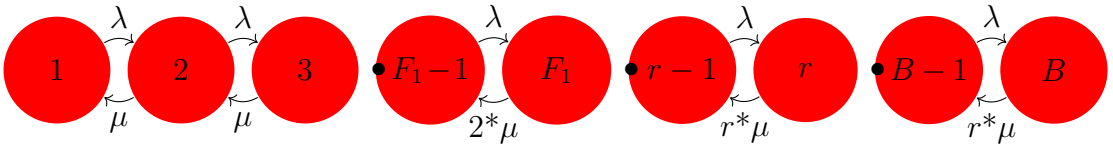


FIGURE 1 – le data center avec une file d'attente multi serveur et avec une politique à seuils.

Comme le modèle de machines virtuelles décrit par la figure ci dessus,il nous propose que les clients arrivent dans la file d'attente multi-serveur (nous considère que 3 serveurs, 6 serveurs et 12 serveurs à l'évoluation dans la partie de l'étude experimentale), et les serveurs sont activés selon l'occupation du système. Au départ, un seul serveur était activé, puis quand le nombre de clients dépassera F_1, on activera un deuxième serveur, et jusqu'au seuil $R_k - 1$ pour activer le k-ème serveur à la fois des désactivation des seuils R_1, ..., R_k-1.
Autrement dit, on contôle le nombre de clients. S'ils dépassent de R_1 (ou F_1) on désactive le deuxième serveur, et donc un seul serveur est toujours activé.
Dans notre modèle, on définie qu'au départ des processus avec chacune en temps d'inter-arrivée Exponentielles $\frac{1}{\lambda}$ et ceux en temps de service Exponentiel de moyeene $\frac{1}{\mu}$.

2.2 Chaîne de Markov



La chaîne de Markov est une suite de variables aléatoires qui permet de faire la modélisation d'évolution dynamique d'un système aléatoire. On note 0 , 1 , 2 ... , F1-1 , F1 , F1+1 ... r-1 , r , r+1 ... , B-1 , B la suite des instants de transition des états du système $\{X_n, n \geq 0\}$ dans notre suite. Et donc nous n'évoluons la distribution stationaire qu'au travers de la valeur actuelle.
Notre modèle va créer et supprimer dynamiquement une machine virtuelle afin de faire l'évoluation. Nous develpons une fonction récursive pour obtenir les probabilités d'état stable du système. Et chaque probabilité est obtenu par la multiplication

de P0 par la matrice A.
On calcule des versions successives de la distribution ainsi :

$$P_i = P_0 * A_i \tag{1}$$

Et à chaque itération, ils respectent toujours :

$$\sum_{i=0}^{B} P_i * A_i = 1 \tag{2}$$

Quand considère à cette chaîne, on résume que :

1. 1 er seuil : $0 \leq n \leq F_1 - 1$:

$$A_0 = 1; \tag{3}$$

$$A_1 = \left(\frac{\lambda}{\mu}\right)^1; \text{ .....} \tag{4}$$

$$A_{F_1-1} = \left(\frac{\lambda}{\mu}\right)^{F_1-1}; \tag{5}$$

2. 2 ème seuil : $F_1 \leq n \leq F_2 - 1$:

$$A_{F_1} = \left(\frac{\lambda}{2 * \mu}\right)^1 \times A_{F_1-1}; \tag{6}$$

$$A_{F_1+1} = \left(\frac{\lambda}{2 * \mu}\right)^2 \times A_{F_1-1}; \text{ .....} \tag{7}$$

$$A_{F_2-1} = \left(\frac{\lambda}{\mu}\right)^{F_2-1-(F_1-1)} \times A_{F_1-1}; \tag{8}$$

.....

3. n ème seuil (n est égal au nombre des serveurs) : $F_r \leq n \leq F_B$:

$$A_{F_r} = \left(\frac{\lambda}{r * \mu}\right)^1 \times A_{F_{r-1}-1}; \text{ .....} \tag{9}$$

$$A_{F_{B-1}} = \left(\frac{\lambda}{r * \mu}\right)^{F_{B-1}-1-(F_r-1)} \times A_{F_{r-1}-1}; \tag{10}$$

$$A_{F_B} = \left(\frac{\lambda}{r * \mu}\right)^{F_B-1-(F_r-1)} \times A_{F_{r-1}-1}; \tag{11}$$

2.3 Algorithmique de calculer $A_0 ... A_B$

Pour initialiser la matrice des seuils, on mise la distribution du nombre des états dans chacune. Et la capacité de file d'attente est initialisé à B. Dans ce cas là, si nous enregistrons tous les états de 0 à B , le nombre total des éléments posés dans la matrice est égal à B+1 (états).
L'algorithme qu'on aurait dût implémenter est le suivant :

Algorithm 1 Calculer des valeurs de $A_0 ... A_B$

$functionmat\,[] = Calculer\ A(index, \lambda, \mu\, tmp, pre)\{$
set $i = 1$
set $tmp = 0$
repeat
  if $index < Nb\, serveur$ **then**
    if $index == 0$ **then**
      for $i = 1$ to $seuils[index]$ **do**
        $tmp = \left(\frac{\lambda}{\mu}\right)^{i-1-0} \times pre;$
        $sum\ A = sum\ A + tmp;$
        $mat\ A[] = ajout\ matrice(mat, tmp, pos + i - 1);$
      end for
    else
      for $i = 1$ to $seuils[index]$ **do**
        $tmp = \left(\frac{\lambda}{\mu}\right)^{seuils[index-1]+i-1)-(seuils[index-1]-1)} \times pre;$
        $sum\ A = sum\ A + tmp;$
        $mat\ A[] = ajout\ matrice(mat, tmp, pos + i - 1);$
      end for
    end if
    $pos = pos + seuils[index];$
    $pre = tmp;$
    $index++;$
    $Calculer\ A(index, \lambda, \mu\, tmp + \mu, pre);$
  else
    $P_0 = \frac{1}{sum\ A};$
    $return\ mat\ A[];$
  end if
until $index < Nb\, serveur$
}

2.4 Les formules d'analyser le système

En fonction du nombre total de VMs, de la capacité du système et des seuils, nous analysons notre système avec des résultats calculés par l'algorithmique . Pour se faire :

1. Calculer P_0 :

$$P_0 + P_1 + ... + P_{F_1-1} + P_{F_1} + ...P_B = 1 \tag{12}$$

$$P_0 \times (A_0 + A_1 + ... + A_{F_1-1} + A_{F_1} + ...A_B) = 1 \tag{13}$$

$$P_0 = \frac{1}{\sum_{i=0}^{B} A_i} \tag{14}$$

2. Probabilité de blocage est égale à la probabilité qui se trouve dans l'état B :

$$Pr = 1 - \sum_{i=0}^{B-1} P_i = P_B = \left(\frac{\lambda}{\mu}\right)^{B-1-(r-1)} \times \cdots \left(\frac{\lambda}{\mu}\right)^{F_2-1-(F_1-1)} \times \left(\frac{\lambda}{\mu}\right)^{F_1-1} \times P_0 \tag{15}$$

3. Nombre moyen de requêts : [1]

$$Nb\,moyen\,de\,requetes = \sum_{i=0}^{B} (i \times P_i) \tag{16}$$

4. Nous considèrons que la distribution selon des probabilités à chaque VM comme par exemple (6 clients arrivent dans 3 VMs) :

VM1	VM2	VM3
2 états	3 états	2 états
$1 \times P_0 + 1 \times P_1$	$2 \times P_2 + 2 \times P_3 + 2 \times P_4$	$3 \times P_5 + 3 \times P_6$

Nombre de machine virtuelles en fonctionnement :

$$Nb\,de\,vm = \sum_{i=1}^{nb\,de\,serveurs} \left(i \times \sum_{j=F_\kappa}^{F_{\kappa+1}-1} P_j \right) \tag{17}$$

(Chaque VM fait lancer dans l'intervalle de état : $[F_\kappa, F_{\kappa+1} - 1]$.)

3 Etude Expérimentale avec des différentes paramètres

Pour nos expérimentations nous nous sommes concentrés sur les 3 modèles (avec des différents taux de service mais μ=5 requêtes/h) suivant :

- modèle A (12 VMs de 1 coeur avec chacune un taux de service μ)
- modèle B (6 VMs de 2 coeur avec chacune un taux de service 2*μ)
- modèle C (3 VMs de 4 coeur avec chacune un taux de service 4*μ)

On a également réalisé nos test avec différentes valeurs pour λ :

- 50
- 60
- 70
- 80
- 90
- 100

On définie que la capacité du système B est égal à 100.
Et enfin on l'a simulé sur la grande période à laquelle on l'effectue et attend son statut stable. Et après, on compare les résultats afin de déterminer la plus optimale.

3.1 Résultats

3.1.1 Tableaux de performances

modèle C : 3 MV		$\mu = 20$		
lambda	resultat	Pr blocage	nbr requetes	nbr MVs
$\lambda = 50$	simulation	0.0004011	61.7	2.236
	analytique	0.0001695	65.97	2.4995
$\lambda = 60$	simulation	0.01112	72.39	2.654
	analytique	0.0263157	81.42056	2.92105
$\lambda = 70$	simulation	0.1316	92.81	2.982
	analytique	0.1432906	94.113288	2.998482
$\lambda = 80$	simulation	0.2394	96.56	2.997
	analytique	0.25	97.000784	2.999979
$\lambda = 90$	simulation	0.3235	97.75	2.998
	analytique	0.33333	98.00001	2.9999999
$\lambda = 100$	simulation	0.3917	98.33	2.998
	analytique	0.4	98.5	2.99999999

modèle C : 6 MV		$\mu = 10$		
lambda	resultat	Pr blocage	nbr requetes	nbr MVs
$\lambda = 50$	simulation	0	59.94	4.243
	analytique	0.000841	71.6329	4.99579
$\lambda = 60$	simulation	0.002463	72.8	5.036
	analytique	0.03726	86.634695	5.776433
$\lambda = 70$	simulation	0.09933	88.66	5.726
	analytique	0.145723	94.491871	5.9799
$\lambda = 80$	simulation	0.2401	96.34	5.969
	analytique	0.250198	97.018786	5.99841
$\lambda = 90$	simulation	0.3307	97.82	5.99
	analytique	0.33335	98.001165	5.999849
$\lambda = 100$	simulation	0.3973	98.29	5.992
	analytique	0.4	98.500101	5.99998

modèle C : 12 MV		$\mu = 5$		
lambda	resultat	Pr blocage	nbr requetes	nbr MVs
$\lambda = 50$	simulation	0	45	6.37
	analytique	0.001881	75.48	9.98
$\lambda = 60$	simulation	0	53.4	7.457
	analytique	0.0459	88.83	11.44
$\lambda = 70$	simulation	0	66.85	9.068
	analytique	0.15	94.91	11.89
$\lambda = 80$	simulation	0.0369	76.24	10.03
	analytique	0.25	97.08	11.98
$\lambda = 90$	simulation	0.2254	90.81	11.23
	analytique	0.3335	98.011	11.995
$\lambda = 100$	simulation	0.3562	96.09	11.72
	analytique	0.40005	98.50	11.998

3.2 Analyses de graphes

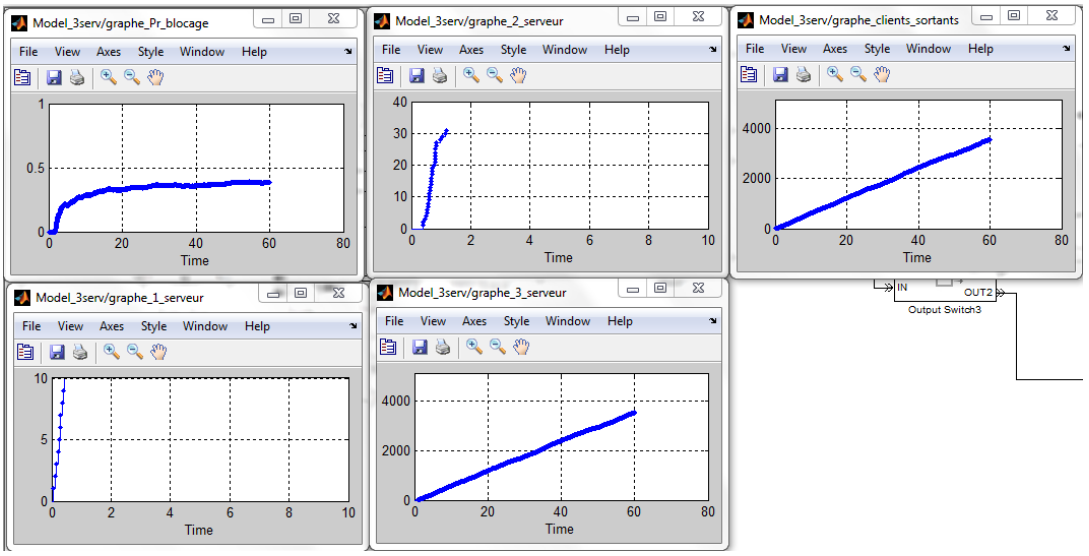


FIGURE 2 – Les graphes de simulation avec 3 MVs et le paramètre λ=100

Ces graphes représentent les résultats de la simulation du Modèle C avec les paramètres suivants : 3VM, Taux d'arrivées λ=100, Taux de service 4μ (μ=5 requêtes/h) et la Capacité du système B=100.

Nous obtenons 5 graphes (graphe de : probabilité de blocage, les clients(ou taches) passé par un serveur, les clients passé par deux serveurs, les clients passés par trois serveurs et les client passé par le système) en prenant un temps réduit de la simulation (60) pour avoir des graphes lisibles.

"graphe_Pr_blocage" c'est le graphe de probabilité de blocage qui montre que dans les premiers temps un blocage nul cela indique que tous les clients arrivés sont bien servis. Dès que la courbe dépasse la valeur 0, tous les serveurs sont activé car on a utilisé la distribution exponentielle pour générer des clients (le taux d'arrivées augmente selon la loi exponentielle), on remarque une augmentation progressive de la probabilité de blocage par rapport au temps car le taux d'arrivées est toujours en augmentation. au temps supérieur à 20 la courbe devient presque stable.

"graphe_1_serveur"," graphe_2_serveur ", "graphe_3_serveur" sont des graphes représentant le nombre de clients servis par 1, 2 ou 3 serveurs par rapport au temps. "graphe_1_serveur" montre que dans les premiers temps les clients sont servis par un seul serveur.

"graphe_2_serveur " commence après que le "graphe_1_serveur" atteint son seuil et qu'il a traité certains clients.

"graphe_3_serveur" commence quand le "graphe_2_serveur " atteint son seuil et il continu à augmenter jusqu'à la fin de la simulation.

"graphe_client_sortants" ce graphe montre que tous les clients sont servis par le système, on remarque qu'il a la même allure que le "graphe_3_serveur". cela indique que les clients servis par ce système sont presque tous traités par les 3 serveurs avec une probabilité de blocage due à un Taux d'arrivées plus élevé par rapport à la capacité de système.

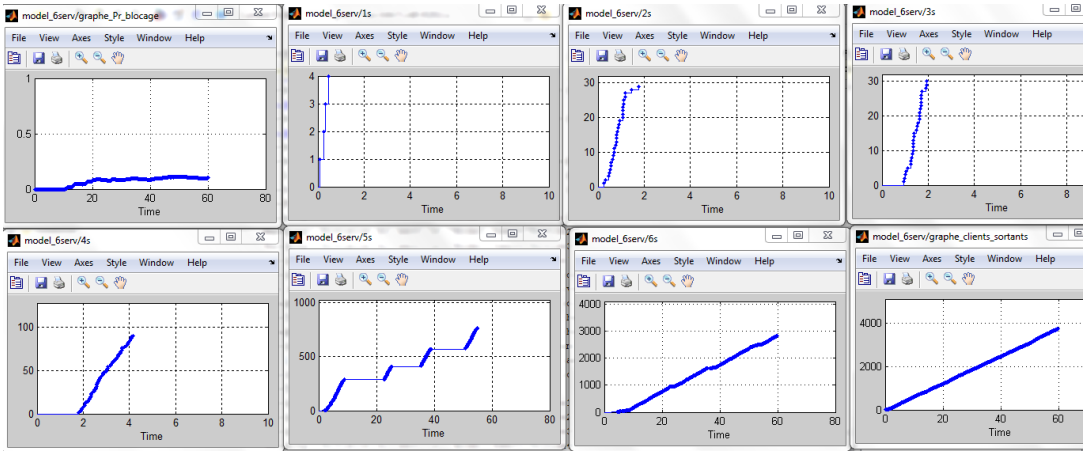


FIGURE 3 – Les graphes de simulation avec 6 MVs et le paramètre λ=70

Ces graphes représentent les résultats de la simulation du Modèle B avec les paramètres suivants : 6VM, Taux d'arrivées λ=70, Taux de service 2μ(μ=5 requêtes/h) et la Capacité du système B=100.
Dans ce modèle on constate que le fonctionnement des serveur est similaire au fonctionnement des serveurs du modèle précédant (à chaque déplacement de seuil d'un serveur, le prochain serveur se lance), le graphe "5s" montre que le les 5 serveurs fonctionnent de manière discontinue, cette discontinuité est due au déplacement de leur seuil ce qui entraine le fonctionnement avec 6 serveurs, lorsque le taux d'arrives est inferieur au seuil des 5 serveurs, le sixième serveur sera désactivé (fonctionnement en 5 serveurs).
On observe que la probabilité de blocage a diminué par rapport au modèle précédant car on a simuler avec un taux d'arrivés réduit

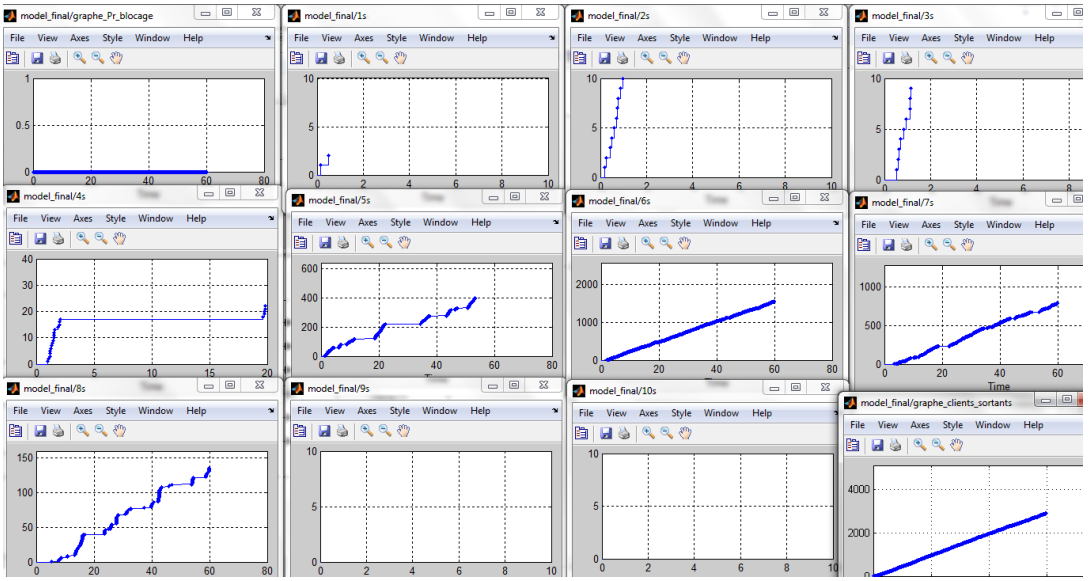


FIGURE 4 – Les graphes de simulation avec 12 MVs et le paramètre λ=50

Ces graphes représentent les résultats de la simulation du Modèle A, avec les paramètres suivants : 12VM, Taux d'arrivées λ=50, Taux de service μ=5 requêtes/h et la Capacité du système B=100.

Dans ce modèle on voie le même principe de fonctionnement des serveurs comme les modèles précédents. de plus, il n'existe aucune courbe sur les graphes "9s", "10s", "11s" et "12s" car le taux d'arrivées n'a pas dépasser le seuil de "8s" de ce fait, la probabilité de blocage est toujours nulle comme le montre le graphe"graphe_Pr_blocage".

4 Difficuliés rencontrées

4.1 Résultats de simulation et de programmation

les resultats obtenu par la simulation de notre modèle sont tres proche des resultats de la progmmation. Néanmoins nous avons rencontré plusieurs obstacles. Principalement, la probabilité de blocage a une marge d'erreur de l'ordre de 10^{-2}. Parfois les valeurs de nombre moyen de requêtes sont différentes lorsque λ est petit. les valeurs de sorties sont affichées, montrant qu'il y a des clients perdus dans notre système (ce sont pas des clients bloqués.).
Nous avons donc ignoré ces erreurs après une discussion avec notre professeur et nous considèrons que le programmation nous donne des résultats idéals sans perte pendant le traitement des serveurs.

5 Conclusion

Nous avons proposé une implémentation en C de l'algorithme de simulation de machine virtuelle selon la propriété de la distribution stationnaire.
Pour avoir les résultats les plus performants, on a proposé trois modèles de simulation avec des différentes configurations, les VMs sont saturées lorsque le Taux d'arrivées (λ) dépasse leurs seuils et la probabilité de blocage augmente aussi quand λ continu à augmenter après avoir déjà dépassé la capacité du système.
Après plusieurs simulations, nous avons trouvé que la performance du VM en modèle A (12 serveurs) est la plus efficace par rapport au modèle C (3 serveurs) et au modèle B (6 serveurs) car le cas de blocage en modèle A apparaît lorsque $\lambda = 70$, en modèle B quand $\lambda = 60$ et en modèle C quand $\lambda = 50$.
Nos résultats de simulation correspondent avec les résultats obtenus dans l'article présenté par V.Goswami, S.S.Patra G.B.Mund.

Références

[1] Pougne Pandore. *File d'Attente.* pandore, 2015.

[2] G. B. Mund Veena Goswami, Sudhansu SHEKHAR Patra. *Performance analysis of cloud with queue-dependent virtual machines.* KIIT University, 2012.