Problème des Lecteurs-Rédacteurs et Allocateur de Ressources

Assala Assellalou

March 21, 2025

1 Stratégies de Synchronisation

1.1 Priorité aux Rédacteurs

1.1.1 Présentation de la stratégie

La priorité est donnée aux rédacteurs et on utilise un moniteur pour synchroniser l'accès des lecteurs et rédacteurs, en donnant la priorité aux rédacteurs pour minimiser leur attente face aux lecteurs. Le code utilise plusieurs variables de contrôle :

- *EC* : un booléen qui indique si un rédacteur est en cours d'écriture.
- *nbLec* : un compteur qui enregistre le nombre de lecteurs actifs.
- *nbRatt*: un entier qui suit le nombre de rédacteurs en attente.
- *moniteur* : une instance de *ReentrantLock* permettant de gérer la synchronisation.
- *LireOK* et *EcrireOK* : deux conditions associées au moniteur, permettant de gérer les accès en fonction des états des lecteurs et rédacteurs.

1.1.2 Fonctionnement des méthodes

Le fonctionnement des méthodes est conçu pour donner la priorité aux rédacteurs :

- *demanderLecture* : Un lecteur peut lire si aucun rédacteur n'est actif et qu'il n'y a pas de rédacteurs en attente. Sinon, il attend. Une fois autorisé, il augmente le nombre de lecteurs actifs.
- *terminerLecture* : Lorsqu'un lecteur termine, il diminue le nombre de lecteurs actifs. Si aucun lecteur ne reste, un signal est envoyé pour permettre à un rédacteur de commencer.
- *demanderEcriture* **:** Un rédacteur peut écrire si aucun lecteur ni rédacteur n'est actif. Si ce n'est pas le cas, il attend et signale sa présence.
– *terminerEcriture* : Quand un rédacteur termine, il libère l'accès. Si aucun rédacteur n'attend, les lecteurs peuvent lire ; sinon, la priorité est donnée aux rédacteurs en attente.

1.2 Stratégie FIFO pour les Lecteurs/Rédacteurs

1.2.1 Présentation de la stratégie

l'ordonnancement des lecteurs et rédacteurs se fait selon la stratégie **First In, First Out (FIFO)**, où les processus sont servis dans l'ordre dans lequel ils arrivent. Pour cela, un moniteur est utilisé pour synchroniser les accès au fichier partagé et une file d'attente est mise en place pour gérer les demandes. Au variables de contrôle EC, nbLec et nbRatt s'ajoutent :
- File : Une file d'attente (de type LinkedList) qui stocke les conditions des lecteurs et rédacteurs en attente.
- SAS : Une condition associée à l'accès exclusif au fichier pour les rédacteurs.
- cond : Condition utilisée pour chaque processus en attente.
- nonvide : Un indicateur qui permet de savoir si la file d'attente des lecteurs contient des éléments.

1.2.2 Fonctionnement des méthodes

Les méthodes sont conçues pour garantir l'ordonnancement FIFO des processus en attente d'accès au fichier partagé :

- demanderLecture : Un lecteur attend que le fichier soit libre de toute écriture et que personne d'autre ne soit en attente pour commencer sa lecture.
- terminerLecture : Un lecteur termine sa lecture, décrémente le nombre de lecteurs actifs et libère un rédacteur ou un lecteur en attente si nécessaire.
- demanderEcriture : Un rédacteur attend que le fichier soit libre de toute lecture et écriture, puis obtient l'accès en respectant l'ordre FIFO.
- terminerEcriture : Un rédacteur termine son écriture, libère l'accès au fichier et libère le premier processus en attente (lecteur ou rédacteur).

2 Allocateur

2.1 Stratégie des petits demandeurs

La stratégie des petits demandeurs permet à une activité de démarrer si sa demande en ressources est inférieure ou égale aux ressources disponibles. En cas de blocage, elle priorise les demandes les plus petites, en augmentant progressivement jusqu'à ce que toutes les ressources soient allouées.

2.2 Implémentation de la stratégie

Un tableau de conditions est utilisé pour gérer les demandes et un réveil en chaîne permet de débloquer les processus dans l'ordre de taille des demandes. Cela garantit une allocation efficace et équitable des ressources.

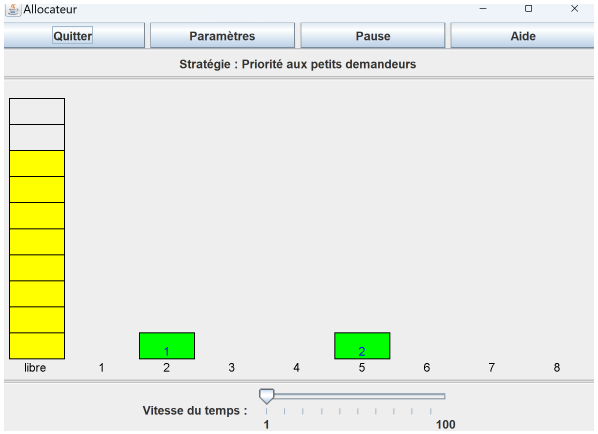

Figure 1: Figure 1 : Test 1 de la stratégie des petits demandeurs

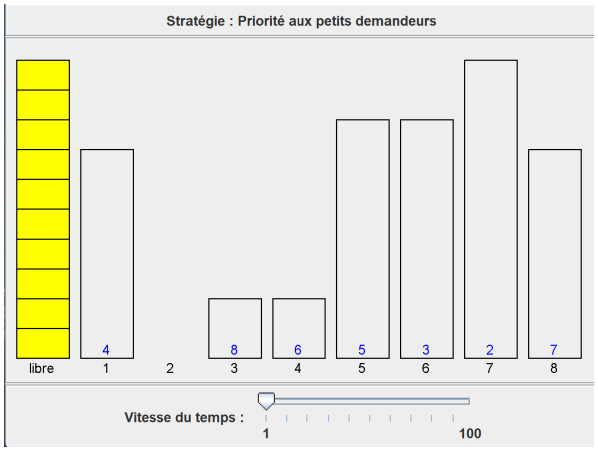

Figure 2: Figure 2 : Test 2 de la stratégie des petits demandeurs