Rapport du TP1 : Systèmes concurrents

Le dîner des philosophes

Myriam ROBBANA

Octobre 2024

Table des matières

Table des figures

1 Première approche : Les fourchettes sont des ressources critiques

1.1 Version de base

Dans cette version de base, chaque fourchette est associée à un sémaphore, et tous les philosophes commencent par saisir leur fourchette de droite avant de prendre celle de gauche.

Cependant, cette approche présente un risque d'interblocage. En effet, lorsqu'un philosophe prend sa fourchette de droite, tous les philosophes peuvent simultanément demander à manger, entraînant ainsi une situation où chacun attend d'accéder à sa fourchette de gauche. Ce phénomène a été mis en évidence grâce à l'introduction d'un délai de 400 ms entre la prise de la fourchette droite et celle de gauche. Cela permet à un philosophe de saisir sa fourchette de droite, puis d'attendre avant de prendre celle de gauche, tandis qu'au même moment, le philosophe suivant peut également saisir sa fourchette de droite, et ainsi de suite.

Voici un exemple illustrant cette situation d'interblocage avec quatre philosophes.

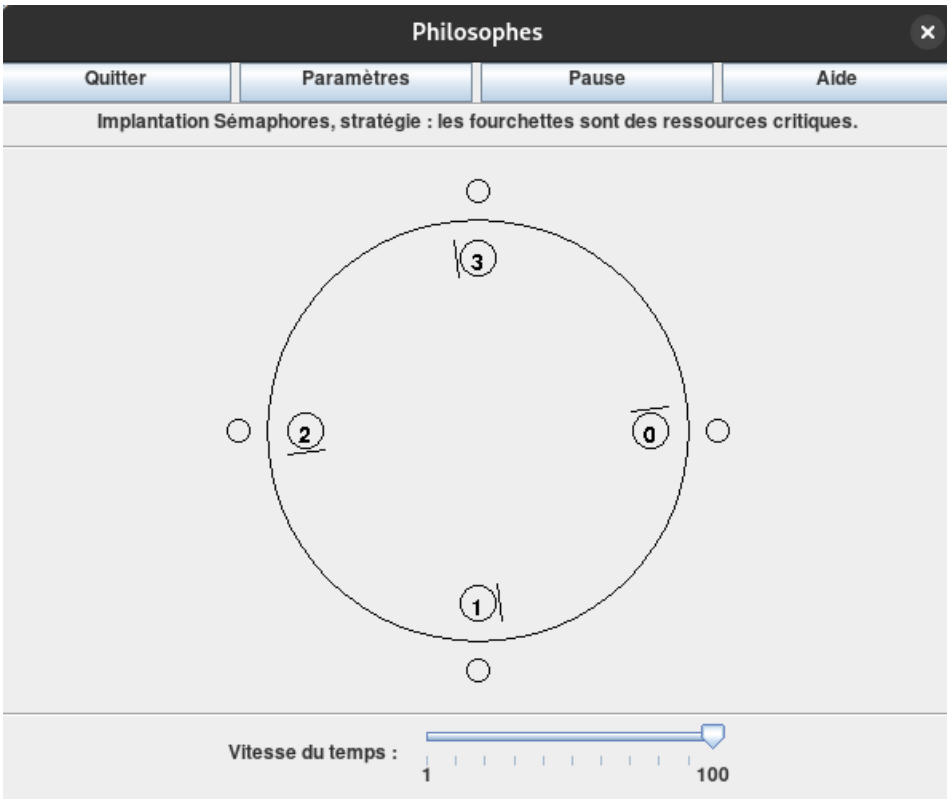

FIGURE 1 – Version de base : Situation d'interblocage

Le fichier fourni pour cette partie est : **StrategiePhiloPremiereApproche.java**.

1.2 Adaptations évitant les interblocages

Par la suite, j'ai implanter deux solution différentes de cette version de base afin d'éviter les interblocages.

1.2.1 Première solution

La première solution consiste à faire en sorte que le premier philosophe prenne d'abord sa fourchette de gauche, puis celle de droite, tandis que tous les autres philosophes commenceront par saisir leur fourchette de droite avant de prendre celle de gauche.

Cette approche permet d'éviter l'interblocage en assurant qu'au moins un philosophe peut toujours accéder aux deux fourchettes, empêchant ainsi que tous les philosophes se retrouvent dans une attente circulaire pour les fourchettes.

Le fichier fourni pour cette partie est : **StrategiePhiloPremiereApprocheSansInterblocage1.java**.

1.2.2 Deuxième solution

Dans cette deuxième solution, un sémaphore (mutex) est utilisé pour synchroniser l'accès aux fourchettes, ce qui garantit qu'un philosophe peut prendre ses deux fourchettes adjacentes sans être interrompu par un autre.

Le fichier fourni pour cette partie est : **StrategiePhiloPremiereApprocheSansInterblocage2.java**.

2 Seconde approche : Contrôler la progression d'un philosophe en fonction de l'état de ses voisins

La seconde approche consiste à introduire explicitement la notion d'état des philosophes, et d'associer un sémaphore "privé" à chaque philosophe. Un philosophe peut manger si aucun de ses voisins ne mange, il doit attendre sinon.

2.1 Analyse de l'approche

La deuxième approche se révèle plus efficace en matière de gestion des ressources par rapport à la première approche sans interblocage 1 et à la deuxième version sans interblocage 2. En vérifiant l'état des philosophes voisins avant de demander des fourchettes, cette méthode permet à un philosophe de commencer à manger immédiatement si ses voisins ne le sont pas, ce qui augmente considérablement le degré de parallélisme. En revanche, la première approche impose une séquence stricte d'acquisition des fourchettes, pouvant engendrer des périodes d'attente inutiles. La deuxième version sans interblocage 2, bien qu'elle utilise un mutex pour contrôler l'accès, n'introduit pas de vérification d'état, ce qui peut aussi entraîner des délais d'attente prolongés.

2.2 Stratégie équitable

On peut remarquer un problème dans la deuxième approche, bien que meilleure que la première. Imaginons un scénario avec 5 philosophes (P0, P1, P2, P3, P4) autour d'une table. Selon la deuxième approche, chaque philosophe vérifie si ses voisins mangent avant de demander les fourchettes. Supposons que tous les philosophes commencent à manger simultanément : P0 mange. P1 attend car P0 mange. P2 attend car P1 mange. P3 attend car P2 mange. P4 attend car P3 mange.

Dans cette configuration, P0 peut finir de manger et reposer ses fourchettes, mais si P1 reprend immédiatement son repas, cela peut entraîner une situation où P0, P1, P2, P3, et P4 alternent entre manger et attendre, laissant potentiellement un philosophe dans un état d'attente sans fin, et donc de famine.

Solution proposée :

Pour résoudre le problème de famine des philosophes, une solution basée sur un système de priorités peut être mise en place. Chaque philosophe se voit attribuer un niveau de priorité qui détermine son accès aux fourchettes, lui permettant de patienter si ses voisins mangent et d'acquérir les fourchettes s'il est le premier dans l'ordre de priorité.

Attente maximale pour un philosophe demandeur, et limite de la solution proposée :

Bien que cette approche améliore la gestion des ressources en éliminant le risque de famine, elle peut limiter le degré de parallélisme, notamment lorsque tous les philosophes ont la même priorité, ce qui pourrait réduire le nombre de philosophes mangeant simultanément à un. Le temps d'attente maximal pour un philosophe pourrait correspondre au nombre total de philosophes autour de la table, tandis que des défis subsistent, tels qu'une attente accrue pour ceux ayant une priorité plus faible et la difficulté d'équilibrer les priorités de manière équitable. Ainsi, bien que la solution de gestion des priorités soit prometteuse, elle introduit des complexités qui nécessitent une attention particulière.

Le fichier fourni pour cette partie est : **StrategiePhiloDeuxiemeApproche.java**.