**Programmation Concurrente**

**TD - Ne plus s'endormir pour attendre**

**Wait Free Synchronisation**

**La pile partagé (en Java)**

Dans un environnement multi-threads, les algorithmes sans verrouillage fournissent un moyen pour les threads d'accéder aux ressources partagées sans bloquer les threads. Ces algorithmes deviennent le choix des programmeurs car ils offrent un débit plus élevé et évitent les blocages.

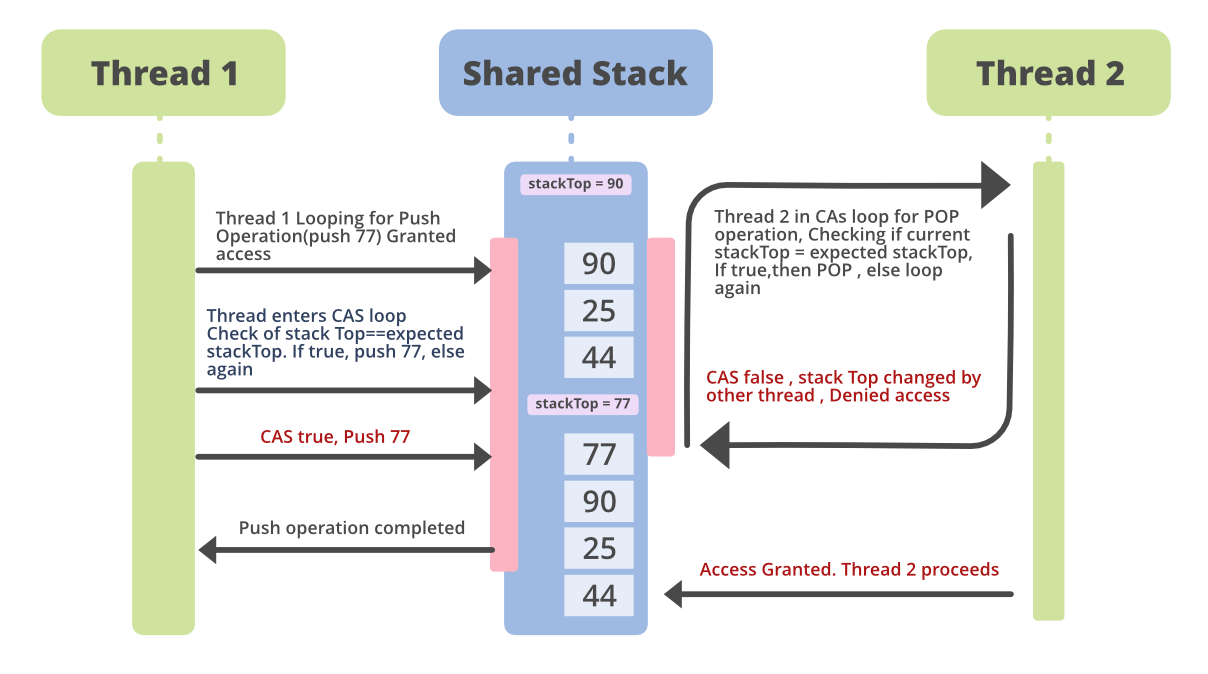
Nous allons étudier la manière dont on peut appliquer un algorithme sans verrouillage à l'une des structures de données largement utilisées en programmation : la pile. En effet la Pile est utilisé dans de nombreuses applications de la vie réelle comme les fonctions Annuler/Refaire dans un traitement de texte, l'évaluation des expressions et l'analyse syntaxique, le support de la récursivité, etc. Voyons donc comment écrire une pile sans verrou.

L'opération la plus importante qui constitue la base des algorithmes sans verrouillage est la comparaison et l'échange. Elle se compile en une seule opération matérielle, ce qui la rend plus rapide car la synchronisation apparaît à un niveau granulaire. Cette opération est également disponible dans toutes les classes atomiques. Le CAS vise à mettre à jour la valeur d'une variable/référence en la comparant avec sa valeur actuelle.

**Utilisation du CAS pour construire une pile non bloquante :**

Une pile non bloquante signifie essentiellement que les opérations de la pile sont disponibles pour tous les threads et qu'aucun thread n'est bloqué. Pour utiliser le CAS dans les opérations de la pile, on écrit une boucle dans laquelle la valeur du nœud supérieur (appelé sommet de la pile) de la pile est vérifiée à l'aide du CAS. Si la valeur du sommet de la pile est conforme aux attentes, elle est remplacée par la nouvelle valeur du sommet, sinon rien n'est modifié et le fil est à nouveau inséré dans la boucle.

Supposons que nous ayons une pile entière. Supposons que le thread1 veuille pousser une valeur de 77 sur la pile alors que la valeur du haut de la pile est de 90. Et le fil 2 veut faire sauter le sommet de la pile qui est actuellement à 90. Si le fil 1 essaie d'accéder à la pile et se voit accorder l'accès parce qu'aucun autre fil n'y accède à ce moment-là, alors le fil obtient d'abord la dernière valeur du sommet de la pile. Ensuite, il entre dans la boucle du CAS et vérifie le sommet de la pile avec la valeur attendue (90). Si les deux valeurs sont identiques, c'est-à-dire que le CAS a renvoyé true, ce qui signifie qu'aucun autre fil ne l'a modifié, la nouvelle valeur (77 dans notre cas) est poussée sur la pile. Et 77 devient le nouveau sommet de la pile. Pendant ce temps, le fil 2 continue à faire une boucle avec le CAS, jusqu'à ce que le CAS retourne vrai, pour avoir fait sauter un objet du haut de la pile. Ceci est illustré dans le diagramme ci-dessous.



Maintenant à vous de jouer à l’aide du code ci-joint. La mise en œuvre la pile traditionnelle avec les verrous vous est offerte.