Parallélisme à base de thread

(PBT 2022-2023)

TD2

Modèle de programmation Pthread

hugo.taboada@cea.fr
marc.perache@cea.fr

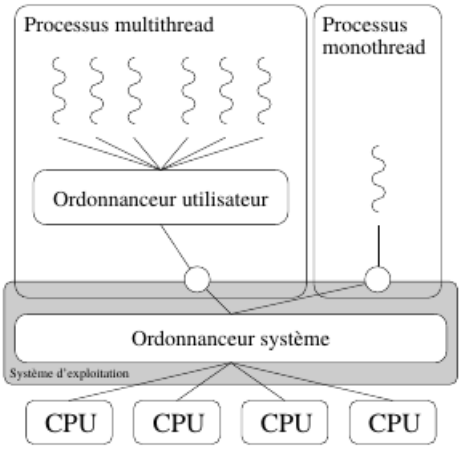
Click here or scan to download TD'files from pcloud link.

Le but de ce TD est d'évaluer différentes bibliothèques de threads (fin du TD1) et ensuite d'implémenter partiellement une bibliothèque mixte. Comme nous l'avons vu avec le TD1, les threads peuvent être gérés dans une bibliothèque entièrement en espace utilisateur, ou bien au sein même du noyau avec l'aide éventuelle d'une bibliothèque en espace utilisateur. Il existe donc trois types de bibliothèques de threads :

— Les bibliothèques de niveau utilisateur.
— Les bibliothèques de niveau système.
— Les bibliothèques mixtes.

I Fonctionnement d'une bibliothèque de niveau utilisateur : GnuPth

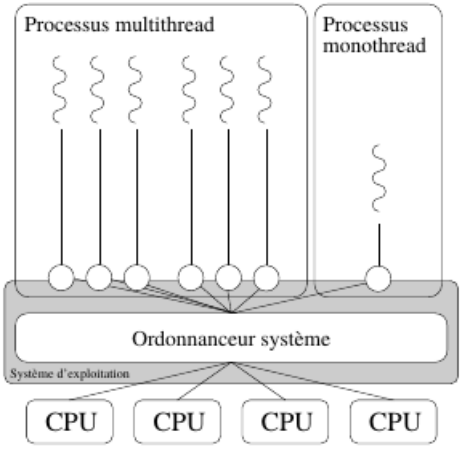
Les bibliothèques de threads de niveau utilisateur telle que GnuPth sont les premières apparues sur les systèmes d'exploitation, sans doute parce qu'il est plus facile de travailler en espace utilisateur que de modifier les systèmes d'exploitation. Il est facile de les adapter à un besoin particulier et, n'ayant pas à dialoguer avec le système d'exploitation, elles sont très efficaces. Le schéma suivant illustre ce type de bibliothèques :



Q.1: D'après vous, quels sont les caractéristiques de ce type de bibliothèque ? (performances, flexibilité, SMP, Appels systèmes bloquants)

II Fonctionnement d'une bibliothèque de niveau système : Pthread

À l'opposé des bibliothèques de niveau utilisateur, les bibliothèques de niveau système sont généralement dédiées à un système particulier. Ces bibliothèques utilisent l'ordonnanceur du système pour gérer leurs threads ; une étroite intégration entre le système et la bibliothèque s'avère donc nécessaire. Les bibliothèques fournies avec les systèmes d'exploitation récents sont généralement des bibliothèques de niveau système (plus rarement à deux niveaux). Le schéma suivant illustre ce type de bibliothèques :



Q.2: D'après vous, quels sont les caractéristiques de ce type de bibliothèque ? (performances, flexibilité, SMP, Appels systèmes bloquants)

III Évaluation des bibliothèques Pthread et GnuPth sur SMP (processeur multi-coeur)

Nous vous proposons d'évaluer deux bibliothèques afin de pouvoir valider les réponses précédentes.

1 Évaluation des capacitées sur architectures SMP

Q.3: Décrire un moyen qui permet de caractériser si une bibliothèque utilise correctement tous les processeurs/coeurs à sa disposition.

Q.4: Évaluer les caractéristiques SMP de la bibliothèque GnuPth.

Q.5: Évaluer les caractéristiques SMP de la bibliothèque Pthread.

2 Évaluation des capacitées envers les appels bloquants

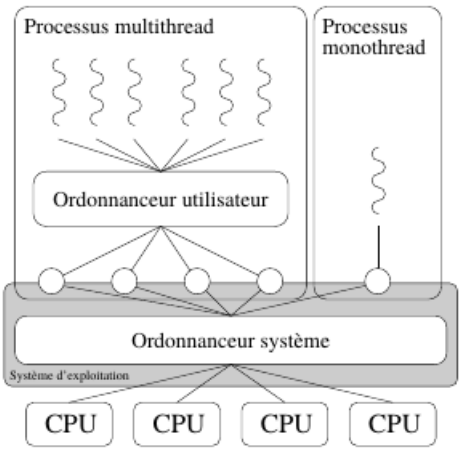
Q.6: Trouver un appel système bloquant permettant d'évaluer la capacité de gestion des appels bloquants des bibliothèques de threads.

Q.7: Évaluer les caractéristiques envers les appels bloquants de la bibliothèque GnuPth.

Q.8: Évaluer les caractéristiques envers les appels bloquants de la bibliothèque Pthread.

IV Fonctionnement d'une bibliothèque mixte : mthread

Les bibliothèque mixte ont l'avantage de garder un ordonnanceur en espace utilisateur et donc d'être efficaces et flexibles. Toutefois, la nécessité de faire co-opérer deux ordonnanceurs rend l'écriture de telles bibliothèques plus difficile. Le schéma suivant illustre ce type de bibliothèques :



Q.9: D'après vous, quels sont les caractéristiques de ce type de bibliothèque ? (performances, flexibilité, SMP, Appels systèmes bloquants)

1 Découverte du code de la bibliothèque mthread

Q.10: Localiser l'ordonnanceur dans le code de mthread.

Q.11: Décrire le fonctionnement de l'ordonnanceur.

Q.12: Localiser les fonctions de manipulation des listes dans le code de mthread.

Q.13: Décrire comment insérer un thread dans la liste des threads prêts de l'ordonnanceur.

Q.14: Décrire comment bloquer un thread.

2 Mutex

Q.15: Décrire à l'aide d'un schéma l'utilité des mutex.

3 Mise en place des mutex dans mthread

Q.16: Mettre en place et expliquer l'implémentation de la fonction mthread_mutex_init.

Q.17: Mettre en place et expliquer l'implémentation de la fonction mthread_mutex_lock.

Q.18: Mettre en place et expliquer l'implémentation de la fonction mthread_mutex_unlock.

Q.19: Mettre en place et expliquer l'implémentation de la fonction mthread_mutex_destroy.

Q.20: Mettre en place et expliquer l'implémentation de la fonction mthread_mutex_trylock.

Q.21: Mettre en place et expliquer l'implémentation de la macro MTHREAD_MUTEX_INITIALIZER

Q.22: Pour chacune des fonctions/macros précédentes, construire un programme d'exemple qui teste leur bon fonctionnement.

Ce TD est noté. Le code source et le rapport sont à fournir. Attention à bien faire des Makefile. Merci de bien préciser dans le rapport à quelle ligne, et dans quel fichier, se trouvent vos réponses dans le code source. Détaillez l'implémentation dans le rapport. Tout cela est à envoyer avant le 19/02/2023 à 23h59 sur le lien pcloud suivant :

https://e.pcloud.com/#page=puplink&code=JDxZNPUMc4ndgyBK8Ha9veVXa8snCScX

Pour toutes questions n'hésitez pas à m'envoyer un mail à l'adresse suivante : *hugo.taboada@cea.fr*