Synthèse

1 Sujet

Le but de ce projet est de mettre en place une architecture Clients/Serveur ou le serveur se chargera de récupérer et traiter des requêtes des clients. Une requête consiste en la demande de calcul des sommes préfixes d'un tableau d'entiers transmis au serveur via un canal de communication (ici segment de mémoire partagée).

Si $\oplus$ est une opération associative, par exemple ($+$, $\times$, min, max, pgcd, ...) et A un tableau de n éléments $[a_0, a_1, \ldots, a_{n-1}]$, les sommes préfixes de A forment le tableau

$$[a_0,\ a_0 \oplus a_1,\ a_0 \oplus a_1 \oplus a_2,\ \ldots, a_0 \oplus a_1 \oplus \cdots a_{n-1}]$$

Par exemple si :

$$A = [\ 4,\ -1,\ 3,\ 0,\ 1,\ 2,\ -2,\ 5\ ]$$

et $\oplus$ est l'addition ($+$), alors les sommes préfixes de A sont :

$$[\ 4,\ 3,\ 6,\ 6,\ 7,\ 9,\ 7,\ 12\ ]$$

2 Architecture globale

La communication entre le serveur et les clients devra se faire selon le schéma suivant (voir aussi Figure 1) :

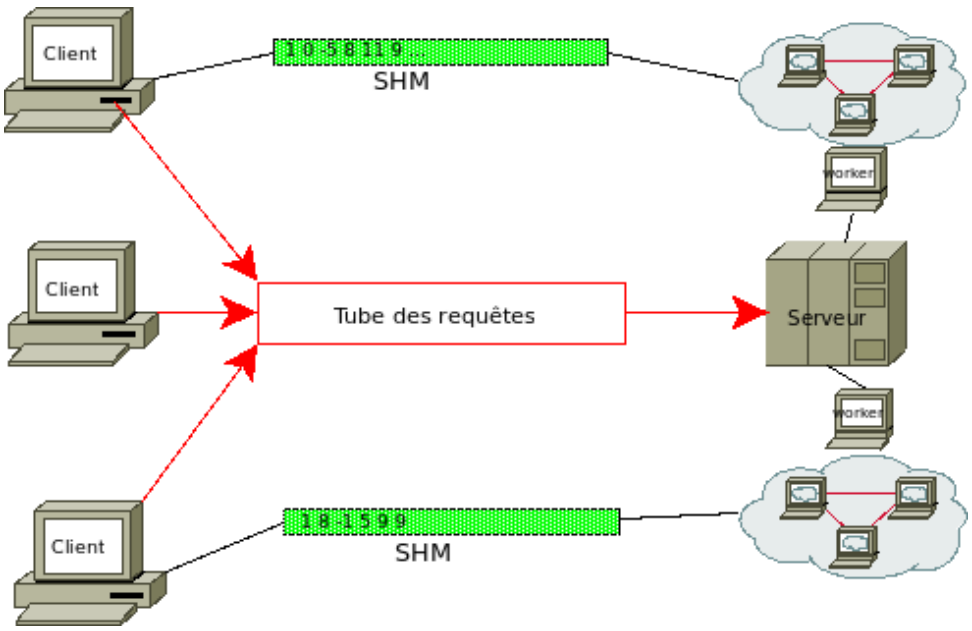

Figure 1 – Schéma de l'architecture clients/serveur

— Les clients demandent l'attribution d'une ressource de calcul au serveur via un tube nommé de taille fixée au démarage du serveur ;

— La demande d'un client doit comporter son pid, la taille du tableau A ainsi que l'opération associative $\oplus$;
— A la réception d'une requête, le serveur crée un processus de travail (worker) et lui communique le pid du client ainsi que l'opération $\oplus$. Le processus worker procédera au calcul parallèle des sommes préfixes sur le tableau A représenté par un segment de mémoire partagé créé par le client sous le nom "/pid" ou pid est le pid du client ;
— Le client attend la fin du calcul du processus de travail (worker) pour afficher le résultat.

3 Contraintes

Votre application sera composée d'un démon et d'un client répondant aux impératifs suivants :
— Le démon sera capable de gérer plusieurs clients en parallèle ;
— Le calcul des sommes préfixes doit être effectué par p, ($p \geq 2$) threads ou processus concurrent selon l'algorithme parallèle "Hillis-Steele Scan" ou "Blelloch Scan" ;
— Le démon devra gérer correctement les zombies et les demandes de terminaisons via des signaux ;
— Le tube nommé permettant au client de déposer sa requête sera à une taille fixe (paramétrable), il faudra donc gérer le cas où il n'est pas possible pour un client de déposer une requête ;
— Les éléments définis en paramètres seront enregistrés dans un fichier de configuration, et les valeurs des paramètres prises en compte au démarrage de l'application.

4 Travail à réaliser

En plus des codes sources correctement documentés et d'un makefile respectant les flags spécifiés en début de semestre, vous rendrez :
— Un petit manuel utilisateur explicitant comment utiliser votre application ;
— Un manuel technique décrivant les solutions que vous avez été amené.e.s à développer pour réaliser les différents modes de communication entre le démon et ses clients ;

Le projet peut être réalisé seul.e ou en binôme. La date limite de rendu est fixée au Samedi 31 decembre 2022. Une soutenance sera organisée en début d'année 2023.