

# DS 2013 Systèmes avancés

Damien Crémilleux

13 mai 2014

## 1 Questions à réponse courte

### 1.1 Parallélisme

**Q1 : Quel est le principal avantage des machines à mémoire partagée sur les machines à mémoire distribuée ?** Le principal avantage des machines à mémoire partagée est l'espace d'adressage unifié. Ainsi il est possible d'utiliser des variables partagées, de mettre en place de la synchronisation. C'est bien un système adapté aux tâches légères.

**Q2 : Quel est le principal avantage des machines à mémoire distribuée sur les machines à mémoire partagée ?** Le principal avantage des machines à mémoire distribuée est l'utilisation de communication à distance. Il s'agit donc d'un système bien adapté aux processus plus qu'aux tâches légères, et qui nécessite un réseau pour communiquer.

**Q3 : Pourquoi un programme ne peut-il (en général) s'exécuter P fois plus vite sur P processeurs que sur 1 processeur ?** Un programme ne peut pas en général s'exécuter P fois plus vite sur P processeurs que sur 1 processeur car toutes les tâches ne sont pas parallélisables. Et même lorsque les tâches sont parallélisables, le coût de la parallélisation (synchronisation, etc) diminue la vitesse d'exécution du programme.

### 1.2 MPI

**Q4 : Pourquoi ne peut-on pas partager de variables en MPI ?** On ne peut pas partager de variables en MPI car il s'agit seulement d'une bibliothèque de passage de message, pour des machines distribuées. Il n'y a donc pas de mémoire partagée et de variable partagée.

**Q5 : Pourquoi n'utilise t'on pas le rendez-vous pour la communication en MPI ?** En MPI, les communications sont non-bloquantes afin d'être plus efficaces. En outre, cela évite le risque de *deadlock* si un message se perd sur le réseau.

### 1.3 Système de fichiers distribués

**Q6 : Quel est l'intérêt d'avoir un serveur de fichier distribué stateless ?** L'intérêt d'avoir un serveur de fichier stateless (comme NFS v3) est la résistance

aux problèmes de réseau. Ainsi le serveur peut être situé dans un environnement instable, subir un crash et redémarrer sans problème.

When NFS was developed there were many problems with the stability of networks and servers. NFS was designed to be an efficient protocol that could successfully handle an unstable environment without causing harm to the client machine. To operate in such a potentially harsh environment, NFS was built to be stateless. Being stateless means that there is no "state" that any operation could hold NFS in. For instance, in NFS version 2, when you perform a write, it will not return a result until the data has been written onto permanent storage. If the write never returns, it is up to the client to decide to continue waiting for the write to finish, or to not make the write.

So being stateless means that the server can crash and the client will wait for the server to come back up. However, for some technical reasons as we'll explain below, there can be minor problems. The most notable one is the "Stale File Handle" error that can be seen after a server is rebooted. The cause of this one is, related to the file handles that the server uses with its filesystem. If those should change, then NFS mounts that the client will not have the correct filehandle and it will need to be unmounted and then remounted.

**Q7 : Quelles sont les limitations de NFS pour une utilisation dans un cadre massivement parallèle ?** NFS a été désigné pour des petites infrastructures et présente donc des limites à cause de l'approche client/serveur. Ainsi, face à un grand nombre de client, le serveur NFS se trouvera surchargé.

## 1.4 Mémoire virtuellement partagée

**Q8 : Quelles sont les deux fonctions essentielles d'une mémoire virtuellement partagée ?** Les deux fonctions essentielles d'une mémoire virtuellement partagée sont :

- Les données sont distribuées entre différentes mémoires.
- Un accès transparent est fourni aux données.

**Q9 : Dans le cas d'une gestion de mémoire avec invalidation ; que se passe-t-il lorsque deux processus situés sur des processeurs différents accèdent régulièrement à une même variable ? Que peut-on faire pour améliorer l'efficacité dans ce cas ?**

**Q10 : Deux techniques de gestion de la cohérence existent : le *write back* et l'invalidation. Donner des arguments en faveur ou en défaveur de chacune de ces techniques (coût des défaut de page, nombre de messages, ...)**

	<i>write back</i>	invalidation
avantages	Le défaut de page a uniquement lieu au premier accès	L'écriture est immédiate
inconvénients	Il faut attendre pour écrire	Beaucoup de mise à jour sont envoyées à chaque écriture