Cahier de laboratoire

Résumé

Ce document détaille les expérimentations réalisées dans le cadre des tests de performance et de validation des outils développés. Il contient les scripts utilisés, les fichiers de sortie générés, les commits associés pour traçabilité et les graphiques produits lors des analyses.

---

1. Mise en place d'un Make séquentiel

Objectif

Créer et vérifier le fonctionnement d'un Make séquentiel à l'aide de l'instance *premier*.

Détails techniques

- **Fichier principal :** `main.go`
- **Makefile utilisé :** `makefiles/premier/premier`
- **Commit SHA :** `cf8383f20cbd4ebc2df5b6d78b9991445ae4a8b1`

---

2. Implémentation de PingPong et PingPong IO

Objectif

Mettre en place deux programmes, PingPong et PingPong IO, pour tester leur fonctionnement via des échanges de messages, sans récupération des résultats.

Détails techniques

- **Fichiers utilisés :**
  - `pingpongIO/client.go`
  - `pingpongIO/server.go`
  - `pingpong/client.go`
  - `pingpong/server.go`
- **Fichier de log :** `pingpong/perf/logs/throughput.log`

- **Commit SHA :** `c33a2d6fe727e92dedb993efc6f7e26461fe3365`

---

3. Mesure de la latence et du Débit avec Ping Pong

Objectif

Mesurer la latence des échanges entre deux programmes Ping Pong et collecter les résultats pour tracer un graphique de la latence en fonction de la taille des messages échangés.

Remarque importante

Une erreur de calcul a été identifiée concernant l'unité de temps. Elle sera corrigée ultérieurement.

Détails techniques

- **Script utilisé :** `perf_measure/pingpong/measure_perf.sh`
- **Fichier de log :** `perf_measure/pingpong/perf/logs/latency.log`
- **Graphiques générés :**
  - Latence :

    `perf_measure/pingpong/perf/logs/perf_benchmarks/graphs/2024-11-20 143630.091073_latency%msg_size.png`
  - Débit :

    `perf_measure/pingpong/perf/logs/perf_benchmarks/graphs/2024-11-20 143630.215952_throughput%msg_size.png`
- **Commit SHA :** `7f5db612180a4be5c7665d83a4097157de8fa9f1`

---

4. Mesure de la latence et du Débit avec PingPong IO

Objectif

Mesurer la latence des échanges entre deux programmes PingPong IO et collecter les résultats pour tracer un graphique de la latence en fonction de la taille des messages échangés.

Plusieurs fichiers de type `.bin` ont été générés, avec des tailles variant de **100 MB à 1 GB**.

Processus :

1. **Lecture des fichiers :** Les fichiers sont lus par le client.
2. **Envoi au serveur :** Les fichiers sont transmis du client vers le serveur.
3. **Écriture par le serveur :** Le serveur écrit les fichiers reçus dans un emplacement spécifié.

Remarque importante

Une erreur de calcul a été identifiée concernant l'unité de temps. Elle sera corrigée ultérieurement.

Détails techniques

- **Script utilisé :** `perf_measure/pingpong_IO/measure_perf.sh`
- **Fichier de log :** `perf_measure/pingpong_IO/perf/logs/latency.log`
- **Graphiques générés :**
  - Latence :

    `perf_measure/pingpong_IO/perf/perf_benchmarks/graphs/2024-11-20 190759.804046_latency%msg_size.png`
  - Débit :

    `perf_measure/pingpong_IO/perf/perf_benchmarks/graphs/2024-11-20 190759.914805_throughput%msg_size.png`
- **Commit SHA :** `7f5db612180a4be5c7665d83a4097157de8fa9f1`

---

5. Mesure de la latence pour le transfert de fichiers avec SCP et RSYNC

Objectif

Comparer les performances des outils SCP et RSYNC pour le transfert de fichiers en mesurant la latence en fonction de la taille des fichiers transférés.

Détails techniques

- **Script utilisé :** `perf_measure/file_transferring_perf/start_measure.sh`
- **Graphiques générés :**
  - SCP : `perf_measure/file_transferring_perf/graphs/2024-11-27 14:30:36.320930_scp_delay%msg_size.png`
  - RSYNC : `perf_measure/file_transferring_perf/graphs/2024-11-27 14:30:38.637716_rsync_delay%msg_size.png`
- **Commit SHA :** `1aae9e72048f4f1762b682dcfb8918b17111d4ab`

---

6. Évaluation des performances de Make distribué vs Make local

Objectif

Comparer les performances entre Make distribué et Make local en mesurant le ratio des temps d'exécution. Les résultats sont présentés sous forme de graphique :

- **Axe des abscisses (Y) :** le ratio entre Make distribué et Make local.
- **Axe des ordonnées (X) :** le nombre de clients impliqués.

Remarque importante

Une erreur dans la légende du graphique a été identifiée et sera corrigée dans une itération ultérieure.

Détails techniques

- **Script utilisé :** `deploy_all.py`
- **Graphique généré :** `premier_efficiency_ratio.png`
- **Commit SHA :** `5ebf7e4fa1a293be64ef075cf417760a16dc5094`

---

7. Benchmarking des performances de Make, Maked with NFS, Maked without NFS

Objectif

Comparer les performances entre les différentes solutions sur des makefiles très différents. L'idée est de faire varier à la fois la durée d'execution des commandes, la taille des fichiers générés et le nombre de commandes.

Détails techniques

- **Script utilisé :** `gen-graph.py`
- **Graphique généré :** `*.png`
- **Commit SHA :** `0ed14bc36af23916394b0c05c8366836d8bf3fd5`
