ECOLE POLYTECHNIQUE DE BRUXELLES

BA IRCI, M-IRIFS

INFO-F2O1

Systèmes d'exploitation

Olivier Vertu Ndinga Oba

000575921

Devon Disdero

000546864

Kevin Close

000539794

Professeurs :

Joël GOOSENS et Olivier Markowitch

2023

1. Introduction

Ce rapport présente une vue d'ensemble du projet Image Search, un programme conçu pour trouver l'image la plus similaire à une image donnée dans une base de données. Le projet comprend un programme C++, 'img-search', qui utilise les résultats de comparaison générés par 'img-dist', une application externe. Dans ce rapport, nous décrirons les objectifs du projet, explorerons les choix d'implémentation, examinerons les défis rencontrés au cours du développement et mettrons en évidence les solutions adoptées pour les surmonter.

2. Vue d'ensemble du Projet

Le projet de recherche d'images permet aux utilisateurs de trouver des images visuellement similaires dans une base de données. Le projet est divisé en plusieurs composantes principales :

— **Programme en C++ ('img-search')** : Au cœur du projet, ce programme gère la communication avec 'img-dist', une application externe responsable de la comparaison d'images, la récupération des résultats, et l'échange efficace de données entre les processus à l'aide de la mémoire partagée.

— **Scripts Shell** : Le projet inclut deux scripts shell - 'image-list.sh' pour lister les fichiers dans un répertoire et 'img-search.sh' pour exécuter l'application de recherche d'images. Ces scripts améliorent la convivialité et automatisent diverses tâches.

3. Choix d'Implémentation

Le programme prend en charge deux modes d'utilisation : interactif et automatique. En mode interactif, l'utilisateur saisit les chemins des images à comparer un par un jusqu'à ce qu'il tape "exit" pour terminer. En mode automatique, le programme lit la liste des images à partir d'un fichier à l'aide d'un script shell (`list-file`).

3..1 Traitement Parallèle des Comparaisons d'Images

Le programme utilise deux processus enfants créés avec `fork()` pour gérer simultanément les comparaisons d'images. Chaque processus enfant parcourt une moitié de la liste des images à comparer en utilisant la fonction `runImgDist()`.

3..2 Communication Interprocessus avec la Mémoire Partagée

La mémoire partagée (`shmget` et `shmat`) est utilisée pour partager les résultats des comparaisons entre les processus. Les processus enfants mettent à jour la distance minimale partagée et le chemin de l'image la plus similaire.

3..3 Gestion des Signaux

La gestion des signaux, en particulier `SIGINT`, permet une interruption propre du programme par l'utilisateur sans perturber le traitement en cours. L'utilisation d'une variable globale (`sigintReceived`) facilite la communication entre le gestionnaire de signal et les processus enfants.

3..4 Affichage des Résultats

Une fois que les processus enfants ont terminé leurs comparaisons, le processus parent affiche le chemin de l'image la plus similaire et la distance minimale.

4. Défis et Solutions

Tout au long du développement du projet, nous avons rencontré des défis et élaboré des solutions adéquates pour les résoudre :

4..1 Gestion des processus

Un des objectifs était de créer deux processus pour mener la recherche. On a fait le choix de diviser les images en deux nombres égaux; la première moitié comparée par le premier processus ("pid1") et la deuxième moitié par le deuxième processus ("pid2").

Dans le programme img-search, c'est à l'aide de la fonction fork() que les deux processus fils sont créées à partir du processus père. Ceux-ci exécutent leur propre boucle de traitement. Le processus parent utilise la fonction waitpid() pour attendre la fin de ses deux processus fils. Au terme de l'exécution des processus, le processus principal affiche le résultat final : l'image la plus ressemblante ou un message indiquant l'absence d'une image similaire.

Un défi était de faire partager les données des deux processus. En particulier, la distance minimale ("minDistance") et le chemin vers l'image la plus similaire ("imagePath") était partagée entre les deux processus par le biais d'une mémoire partagée. Les processus ont accès à la mémoire partagée en attachant la mémoire partagée à son espace d'adressage par la fonction shmat().

4..2 Gestion des signaux

Dans l'optique de réagir de manière appropriée à des événements tels que l'interruption par l'utilisateur, le programme img-search contient un gestionnaire de signal ("sigintHandler"). Celui-ci est déclaré dans une fonction qui sera appelée lorsque le programme recevra le signal SIGINT. Avant la création des processus fils, le programme main enregistre le gestionnaire de signal pour le signal SIGINT. Il est nécessaire d'effectuer cette étape avant la création le "fork" car le gestionnaire de signal sera hérité par les deux processus créés.

Au sein des deux processus fils, des vérifications sont incluses afin de savoir si le signal d'interruption SIGINT a été reçu. Si c'est le cas, le processus fils peut se terminer gracieusement en indiquant une terminaison normale ("exit(0)").

Solution : Pour atténuer ce risque, nous avons introduit un indicateur global pour suivre les signaux SIGINT. En cas de réception d'un signal SIGINT, les processus enfants se terminent de manière propre, préservant ainsi la mémoire partagée de toute corruption.

4..3 Mode Interactif vs. Automatique

L'implémentation de modes interactifs et automatiques pour l'entrée des chemins d'accès d'images nécessitait une conception et une mise en œuvre soignée.

Solution : Pour offrir de la flexibilité, nous avons établi deux modes - interactif et automatique. En mode interactif, les utilisateurs peuvent entrer les chemins d'accès aux images de manière interactive, offrant ainsi une expérience utilisateur intuitive. En mode automatique, un script extrait les chemins d'accès aux images à partir d'un fichier de liste.

5. Conclusion

Le projet de recherche d'images a débouché sur un outil robuste et flexible, "img-search" en C++, qui permet de rechercher des images similaires dans une base de données. L'interface utilisateur réactive, tant en mode interactif qu'en mode automatique, offre une expérience fluide. Les choix de mise en œuvre ont été délibérément faits pour assurer la robustesse, l'efficacité et la facilité d'utilisation, y compris la gestion des signaux pour une interruption en douceur des opérations. Ce projet démontre l'application réussie de concepts de programmation avancés et de la gestion de la mémoire partagée pour résoudre un problème complexe. L'interface conviviale, les diverses options de mode et la gestion des signaux en font un outil puissant pour la recherche d'images.