Construction parallèle de l'enveloppe convexe haute d'un ensemble de points dans le plan

Auteurs :

- Jules PROUST
- Angel GEZAT

Descriptif du Problème : Construction de l'Enveloppe Convexe Haute

Introduction

La construction de l'enveloppe convexe haute conciste à trouver le polygone convexe le plus petit qui englobe un ensemble de points donné dans un plan. L'approche "diviser et conquérir" va être utiliser pour résoudre efficacement ce problème en le subdivisant en sous-problèmes plus simples.

Enveloppe Convexe

L'enveloppe convexe d'un ensemble de points dans le plan est définie comme le plus petit polygone convexe qui contient tous les points de l'ensemble. Un polygone est dit convexe si, pour tout couple de points appartenant à l'ensemble, le segment de droite défini par le couple est entièrement contenu dans le polygone. En d'autres termes, l'enveloppe convexe est la frontière extérieure la plus simple qui englobe tous les points.

Diviser et Conquérir

La méthode "diviser et conquérir" est une stratégie algorithmique où un problème est divisé en sous-problèmes plus petits et plus simples qui sont résolus de manière récursive. Ensuite, les solutions partielles sont combinées pour obtenir la solution globale du problème. Dans le contexte de la construction de l'enveloppe convexe haute, cette approche est utilisée pour subdiviser l'ensemble de points en deux sous-

ensembles plus petits, calculer l'enveloppe convexe de chaque sous-ensemble, puis fusionner les résultats pour obtenir l'enveloppe convexe haute globale.

Objectif de l'Algorithme

L'objectif principal de l'algorithme est de parvenir à une solution efficace pour construire l'enveloppe convexe haute d'un ensemble de points. En utilisant la méthode "diviser et conquérir", la complexité du problème est réduite en résolvant des sous-problèmes plus simples, ce qui permet une solution plus efficace.

Ce chapitre a jeté les bases du problème de construction de l'enveloppe convexe haute, en introduisant les termes clés tels que l'enveloppe convexe et la stratégie "diviser et conquérir". La suite du rapport explorera en détail l'algorithme parallèle utilisé, les structures de données choisies, et fournira une trace d'exécution avec une courbe d'enveloppe convexe si le projet est abouti.

Présentation des structures de données utilisées

Structure de Point (point)

- **Description :** La structure `point` représente un point dans le plan avec des coordonnées entières (x, y). Elle est utilisée pour stocker les points de l'enveloppe convexe.
  - **Champs :**
    - `int x, y` : Coordonnées du point.
    - `point *next` : Pointeur vers le prochain point dans la liste chaînée des points de l'enveloppe.

Tableau pour le calcul de l'enveloppe convexe haute (to_UH)

- **Description :** Une tableau de points utilisée pour stocker les points qui nécessitent le calcul de l'enveloppe convexe haute. La pile est utilisée de manière dynamique pendant l'exécution de l'algorithme.
  - **Champs :**
    - `point **to_UH` : Tableau de pointeurs vers les points à calculer.
    - `int to_UH_nb` : Index de tête de pile.

Tableau pour la fusion de points (to_merge)

- **Description :** Un tableau de points utilisée pour stocker les points qui nécessitent une fusion dans le processus de construction de l'enveloppe convexe haute.
  - **Champs :**
    - `point **to_merge` : Tableau de pointeurs vers les points à fusionner.
    - `int to_merge_nb` : Index de tête de pile.

Tableau pour le suivi de l'état de fusion (to_merge_status)

- **Description :** Un tableau de statuts utilisé pour suivre l'état des points à fusionner (en calcul, fusionné, prêt à être fusionné).
  - **Champs :**
    - `int *to_merge_status` : Tableau d'entiers représentant le statut des points à fusionner.

        - `int to_merge_merged_nb` : Index de tête de pile pour les points fusionnés.
    - **Constantes :**
        - `MERGING` : Statut indiquant que le point est en cours de fusion.
        - `MERGED` : Statut indiquant que le point a été fusionné.
        - `READY_TO_MERGE` : Statut indiquant que le point est prêt à être fusionné.

Justification des Choix de Structures pour Optimiser la Performance Parallèle

- **Liste Chaînée pour les Points :** La structure de point utilisant une liste chaînée est adaptée pour ajouter et retirer efficacement des points lors du calcul de l'enveloppe convexe.
- **Piles Dynamiques :** Les piles (`to_UH` et `to_merge`) sont des structures de données efficaces pour gérer les points à traiter de manière ordonnée, en suivant le principe de la méthode "diviser et conquérir".
- **Statuts pour le Suivi :** L'utilisation d'un tableau de statuts (`to_merge_status`) offre un moyen efficace de suivre l'état de fusion des points, facilitant la gestion parallèle des opérations.

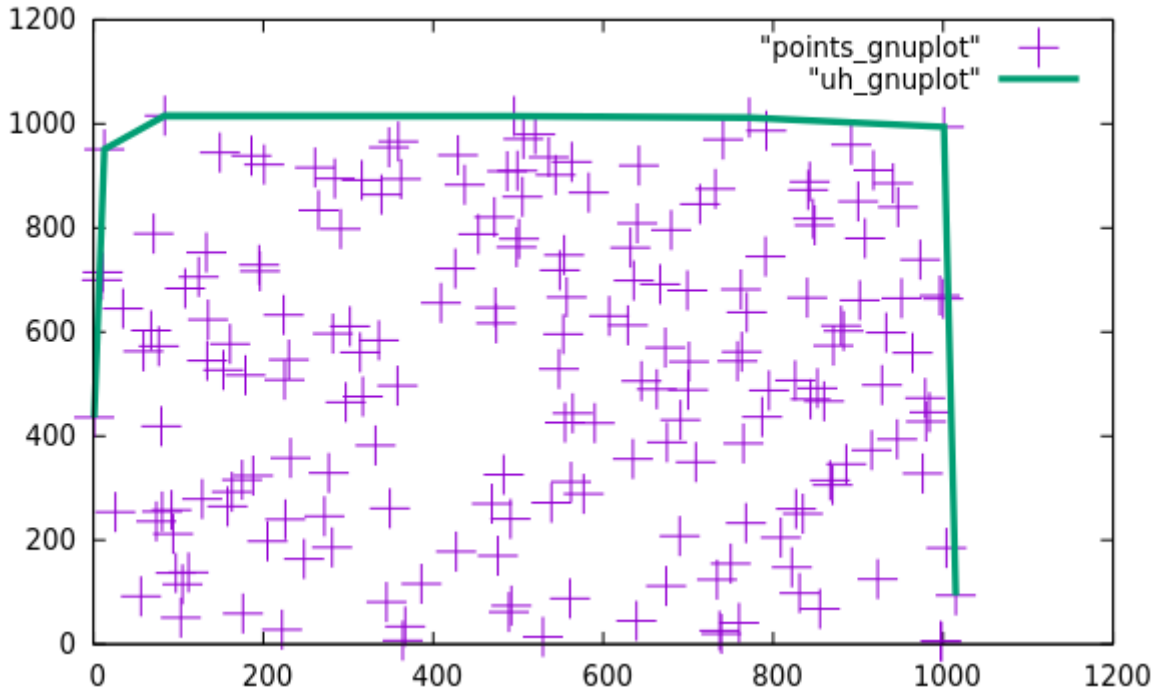
Ces choix de structures de données visent à optimiser l'efficacité et la parallélisation de l'algorithme tout en maintenant une gestion claire et ordonnée des opérations.

Parallélisation

Le code implémente un algorithme parallèle pour le calcul de l'enveloppe convexe haute d'un ensemble de points en utilisant PVM. Les esclaves sont instanciés pour effectuer le calcul parallèle, et le processus maître envoie chaque sous-ensembles de points et leur opération à chaque esclave à l'aide de `master_send_point`. Les esclaves effectuent les opérations nécessaires, puis le maître les reçoit via `master_receive_point`. Le processus maître gère l'envoi de nouvelles tâches aux esclaves, la fusion des résultats et la gestion des piles de points à calculer ou à fusionner. Enfin, le programme se termine en envoyant un message de fin aux esclaves via `pvm_mcast` et se quitte le système PVM avec `pvm_exit`. Il existe les fonctions `slave_receive_point` et `slave_send_point` qui reçoit les points et les opérations et qui les renvoit.

Conclusion

Le projet est une réussite ! Voici le graphique final :



Une enveloppe convexe haute avec 200 points.