P52. Traitement massif de données satellitaires multi-source de l'océan

Année 2015

Encadrants : R. FABLET (Département Signal et Communication), P-H. HORREIN et P. TANDEO (Département Électronique).

Partenaires : E. AUTRET et JF. PIOLLE, Centre de recherche IFREMER.

Mots clés : Traitement de données, télédétection satellitaire, optimisation séquentielle, optimisation parallèle, traitement d'images, océanographie, interpolation de données manquantes, « big data ».

Résumé :

Le projet vise à développer une solution optimisée de reconstruction d'images hauterésolution journalières de la température à la surface de la mer à l'échelle globale. L'objectif a été de procéder à l'optimisation séquentielle et parallèle d'un d'algorithme existant, permettant de gérer efficacement les volumes de données à traiter et d'exploiter au mieux l'architecture multi-cœur en place chez le client.

1. Présentation et contexte du projet.

Pour son activité de recherche l'Ifremer exploite un grand nombre de données océanographiques, comme la température de surface de la mer, recueillies grâce à différents capteurs embarqués sur des satellites. Du fait de contaminations lors des mesures (couverture nuageuse), les données (de l'ordre du To par jour) sont incomplètes. L'Ifremer a donc développé un algorithme permettant d'effectuer une interpolation spatiotemporelle des données manquantes. Mais dans son état actuel l'exécution de cet algorithme est trop longue et le client se contente d'une version simplifiée, sans interpolation temporelle. Le but du projet est donc de proposer au client une version améliorée de l'algorithme d'interpolation spatiotemporelle en proposant différentes pistes d'optimisation (séquentielle et parallèle) permettant de réduire le temps de calcul et l'espace mémoire utilisé par l'exécution de cet algorithme.

2. Méthodologie développée pour aboutir.

Notre projet se décompose en deux principales activités que sont l'optimisation séquentielle et parallèle. Comme tout projet informatique, l'algorithme optimisé doit être régulièrement testé et validé. Ces 3 lots s'accompagnent d'une analyse préliminaire du besoin et d'un lot de gestion de projet inhérent à tout projet. Nous avons donc identifié 6 lots principaux :

Pour mener à bien nore projet nous avons utilisé les outils classiques de gestion de projet comme le Gantt et l'AMDEC. Nous avons fonctionné en début de projet en méthode « Scrum » avec des retours et des réunions très régulieres avec le client ce qui nous a permis de bien comprendre l'algorithme et d'aborder rapidement l'optimisation séquentielle.

3. Développement des différentes tâches et principaux résultats.

31. Prise en main de l'algorithme et optimisation séquentielle.

D'abord nous nous sommes appropriés les principes mathématiques et statistiques sur lesquels est basé le fonctionnement de l'algorithme[1]. Ensuite nous avons étudié en détail l'algorithme afin d'identifier les différentes parties optimisables. Nous avons mesuré à l'aide de la fonction Matlab « profile » les parties du code qui prenaient le plus de temps à l'exécution pour cibler notre optimisation. Nous avons donc ensuite éliminé certains calculs redondants ou simplifié certains calculs en utilisant certaines propriétés mathématiques comme la décomposition de Cholesky.

32. Transposition du code en C/C++.

Pour améliorer les performances de l'algorithme et ajouter plus de flexibilité pour la programmation concurrente, nous avons transposé le code en C/C++, un langage Open Source compatible avec la plupart des standards et environnements de parallélisation comme Hadoop.

33. Etude de la granularité et optimisation parallèle.

Pour implémenter la parallélisation des données, nous avons mis en place une stratégie de division des données afin de pouvoir assigner à chaque unité de calcul une plage de données à traiter. L'objectif a été ensuite de rassembler les résultats issus des différentes unités de calcul pour obtenir une carte complète des données. Les dimensions de ces plages de données sont identifiées par des granularités. Pour réaliser une parallélisation optimale nous avons identifié les valeurs des granularités pour lesquelles le temps de traitement est minimal.

4. Conclusions et perspectives.

Nous avons identifié les fonctions les plus chronophages et nous avons optimisé séquentiellement le code avec un gain de 59%. Nous avons transposé les principaux modules de l'algorithme en C++ pour gagner du temps par rapport à Matlab et nous avons entamé la réalisation de la parallélisation.
A partir de nos travaux réalisés sur un jeu de données de petite taille et notre étude de la parallélisation, le client possède les principaux éléments pour effectuer la parallélisation d'un jeu de données plus important sur ses serveurs.

[1] Voir Bibliographie

Bibliographie :

Autret E. "*Analyse de champs de température de surface de la mer à partir d'observations multi-source*". Thèse de doctorat, Octobre 2014.