Parallélisation d'Automates Cellulaires uni- et bi-dimensionnels et application à la modélisation du processus de subduction-érosion en tectonique des plaques

Thomas Leduc

Laboratoire LIP6
Thème Algorithmique Numérique et Parallélisme
Université Pierre et Marie Curie,
4, place Jussieu,
F-75252 PARIS Cedex 05
e-mail : Thomas.Leduc@lip6.fr

Résumé

Dans ce papier, nous présentons la parallélisation d'un automate cellulaire pour la modélisation numérique du processus de subduction-érosion en tectonique des plaques. Après un rappel de notre modélisation par automate cellulaire uni-dimensionnel, nous présentons notre modèle bi-dimensionnel puis nous abordons la question de la simulation parallèle et de son implémentation sur un CRAY T3E. Nous montrons en particulier l'extrême intérêt des *types de données dérivés* et des *topologies de processus* de la bibliothèque d'échange de messages MPI dans le cadre d'une méthode de décomposition de domaine sur une architecture parallèle à mémoire distribuée.

Mots-clés : Parallélisme, Décomposition de domaine, Automates cellulaires, systèmes dynamiques discrets, tectonique.

---

1. Introduction : le contexte géotectonique [1]

Le texte qui suit présente une partie de l'étude sur la modélisation informatique du processus de subduction-érosion en tectonique des plaques, que nous menons actuellement en collaboration avec le Laboratoire de Géodynamique Tectonique et Environnement de l'Université Pierre et Marie Curie. Après une présentation du contexte géotectonique et un rappel de notre simulation par automate cellulaire uni-dimensionnel, nous exposons notre modèle bi-dimensionnel et la simulation parallèle que nous avons développée avec la bibliothèque d'échange de messages MPI.

La tectonique des plaques est une hypothèse bien établie selon laquelle la lithosphère du globe terrestre est formée de plaques rigides d'une centaine de kilomètres d'épaisseur, flottant sur l'asthénosphère visqueuse et animées de mouvements dont la vitesse varie de 1 à 11 cm/an. Nous nous intéressons au cas des plaques (ou marges) convergentes, en subduction, avec extension. L'extension résulte d'un phénomène d'érosion de la base de la plaque chevauchante par hydrofracturation. C'est un processus qui consiste à entraîner dans la subduction, par le jeu des forces tectoniques, du matériel de la marge chevauchante dans l'asthénosphère visqueuse.

Il existe déjà actuellement des modélisations globales du phénomène par la méthode des éléments finis et des simulations en "bac à sable". Mais, comme nous l'expliquons dans [2], il nous a paru important de développer une simulation à base d'automates cellulaires. L'objectif sous-jacent à toutes ces études est clair : il s'agit de fournir une estimation suffisamment précise et détaillée du volume de matière absorbé en subduction, dans le but de procéder à un enfouissement sous-marin de déchets radioactifs.

2. Les résultats de la modélisation uni-dimensionnelle [2, 3]

Nous avons décidé de discrétiser la coupe verticale de subduction dans le sens longitudinal. L'élément de base, appelé cellule, est une bande verticale plane, sur toute la hauteur de la coupe, comportant plusieurs couches différentes (eau, sédiments, basaltes...). Ces cellules peuvent prendre un nombre fini d'états et sont influencées par un ensemble fini de cellules "proches". L'état d'une cellule est déterminé

par la donnée des épaisseurs des différentes couches qui la constituent, ces épaisseurs sont des entiers naturels majorés par la hauteur totale de la coupe étudiée, elles sont au nombre de sept.

Une fois la géométrie et l'état initial du réseau définis, il ne reste plus qu'à laisser le système évoluer de lui-même, par réévaluation parallèle synchrone de l'état de chacune des cellules en fonction de ceux de ses voisines (au sens où elles influencent la cellule courante...), à chaque pas d'horloge.

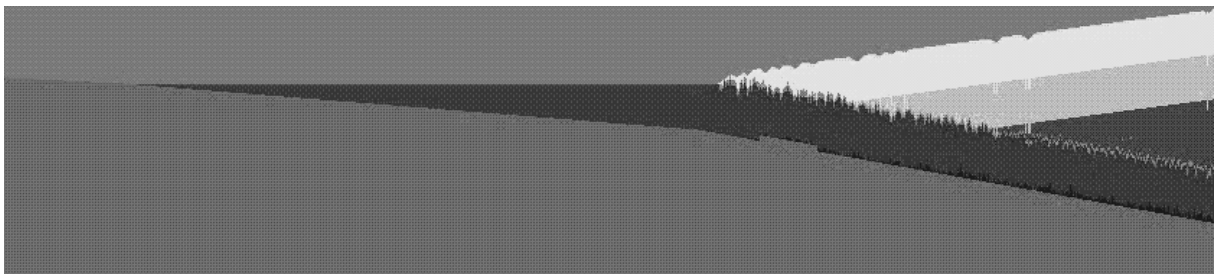

FIG. 1 – *Copie d'écran obtenue après 1100 itérations de calcul (simulation 1D, post-traitement graphique à l'aide de l'utilitaire Convert du logiciel ImageMagick sur une Silicon Graphics quadri-processeurs).*

L'avantage de ce modèle est clair : la quantité d'information à traiter est faible (de l'ordre du millier de cellules, chacune des cellules ne comportant pas plus de 7 champs entiers). Par contre, pour obtenir une représentation acceptable du phénomène physique (fig. 1), il nous a fallu introduire des notions relativement difficiles à mettre en place telles que : trois échelles de temps imbriquées, des phénomènes à "ampleur" locale et d'autres à implications plus globales.

3. La modélisation bi-dimensionnelle

Conscients du fait que le découpage en cellules du modèle uni-dimensionnel est trop "simplificateur" (7 entiers nous suffisent pour représenter une coupe verticale d'espace de 5 km de haut sur 25 m de large, correspondant au pourtour de la fosse océanique), nous avons cherché à modéliser ce phénomène géotectonique par un automate cellulaire bi-dimensionnel. Une cellule ne représente plus désormais qu'une portion d'espace homogène de 25 m par 25 m.

En pratique, notre automate cellulaire est représenté par un tableau bi-dimensionnel de 200000 cellules. Chacune de ces cellules n'a connaissance, d'un pas de temps à l'autre, que de son état propre et de celui de ses 24 voisines immédiates, puisque les cellules contigües et les cellules contigües des cellules contigües interagissent avec la cellule courante (les valeurs des états étant considérées au pas de temps précédent).

4. Utilisation de MPI [4] en décomposition de domaine pour notre automate cellulaire

Dans le but d'optimiser le temps d'exécution de notre simulation, nous avons envisagé d'utiliser une méthode de décomposition de domaine régulière pour le portage de notre code sur une plate-forme parallèle à mémoire distribuée. Ainsi, le domaine total – qui correspond dans notre cas à l'ensemble en deux dimensions des cellules du système discret – est partitionné en autant de sous-domaines que de processus et, c'est sur chacun de ces sous-domaines que l'on effectue les calculs. Un processus donné effectue donc un ensemble de traitements et, ceux-ci étant réalisés, il échange ses données aux frontières avec les huit sous-domaines voisins.

4.1. Intérêt de créer une topologie cartésienne de processus

Pour toutes les communications dites "locales", parce qu'elles ne concernent que des échanges de données aux interfaces entre les sous-domaines, il est intéressant de disposer les processus suivant une topologie régulière. MPI, contrairement à PVM, offre la possibilité de définir des topologies virtuelles de type cartésien. Cette possibilité s'avère très intéressante dans la mesure où elle constitue un moyen pratique de dénomination des processus qui correspond, en plus, au schéma de communication et permet donc à MPI d'optimiser les communications. Elle permet enfin de simplifier l'écriture même du code.

Pour notre simulation, nous avons partitionné le communicateur MPI_COMM_WORLD en deux sous communicateurs (fig. 2) avec la primitive MPI_Comm_split() : le premier constitué d'un seul PE appelé

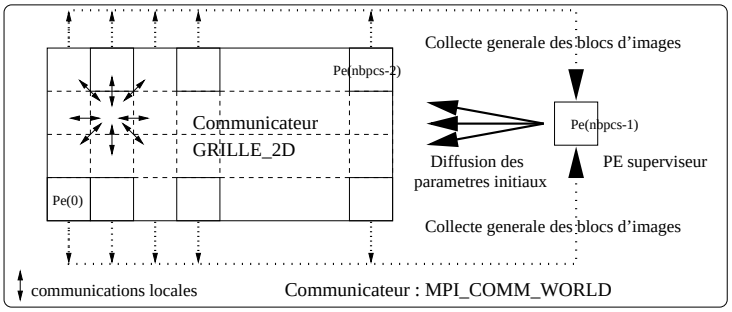

FIG. 2 – *Décomposition de domaine avec* MPI *: les PE (Processor Element) du communicateur* GRILLE_2D *partitionnent complètement une image de la zone de subduction à une date donnée.*

"superviseur" et le second constitué du reste des PE du communicateur global. Au cours de la simulation, alors que le superviseur est chargé de gérer les accès aux disques et les diverses entrées-sorties et de collecter les portions d'images, les autres PE sont, eux, chargés d'itérer l'automate cellulaire en pas de temps. Ces "itérateurs", parce qu'ils doivent échanger leurs données aux frontières avec leurs PE voisins, vont être disposés sur une topologie cartésienne virtuelle régulière à deux dimensions appelée GRILLE_2D. Nous créons cette topologie avec la primitive MPI_Cart_create() et offrons à l'utilisateur de la simulation, la possibilité d'imposer lui-même, dynamiquement, au lancement de l'exécutable, la taille de la grille des processus itérateurs dans chacune des dimensions.

Les primitives MPI_Cart_coords() et MPI_Cart_rank() de la bibliothèque permettent d'obtenir les coordonnées des PE (et de leurs voisins) à partir du rang du PE courant et inversement.

4.2. Intérêt des types dérivés

Dans les communications point à point de notre code, les données échangées sont bien souvent de types plus élaborés que les types "simples" classiques. Ainsi, nos processeurs de calcul échangent à chaque pas de temps deux lignes de cellules avec leurs voisins N et S, deux colonnes de cellules avec leurs voisins O et E, et des "angles" de quatre cellules avec leurs voisins NO, NE, SO, SE (fig. 3). De plus, régulièrement, ils doivent extraire de chacune des cellules de leur tableau local, un champ "couleur" et envoyer au superviseur leur "bloc d'image" (ou tableau de ces champs "couleur" mis bout à bout) propre. Ce même superviseur doit alors collecter l'ensemble des portions d'images distribuées sur les itérateurs et les réordonner, pour reconstituer chacune des images de la simulation.

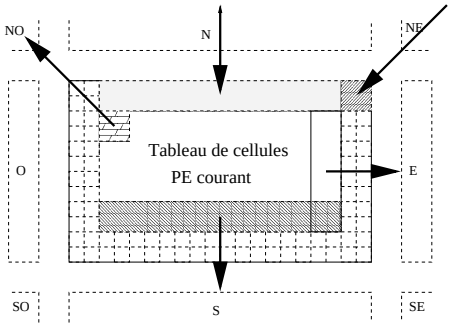

FIG. 3 – *Émission des blocs situés sur le "pourtour intérieur" du tableau de cellules et réception sur le "pourtour extérieur".*

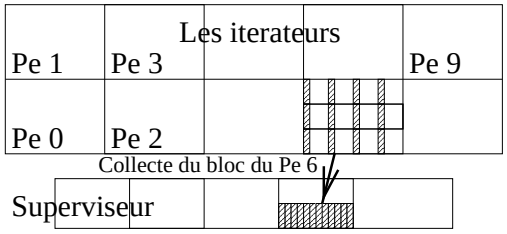

FIG. 4 – *Collecte des blocs d'image (par types dérivés) avec extraction du champ "couleur" au sein de chacune des cellules (Type_bloc_image) et ré-ordonnancement dans l'image globale (Type_image_globale) avant écriture sur le disque.*

Avec une bibliothèque d'échanges de message classique (comme PVM), l'envoi et la réception de tels blocs de données auraient nécessité l'écriture d'un certain de nombre de boucles et entraîné, du même coup, des sur-coûts de compactage/décompactage et de recopie des contenus de messages. Avec MPI, nous avons créé des types dérivés et obtenu, avec les primitives MPI_Type_struct() et MPI_Type_vector(), respectivement un type structuré appelé Type_cellule et trois types de données homogènes vectoriels à pas constant (Type_2lignes, Type_2angles, Type_2colonnes, voir fig. 3).

Par ailleurs, nous avons créé le type dérivé Type_bloc_image avec la primitive MPI_Type_indexed() en considérant le tableau des cellules comme un tableau d'octets et en en extrayant l'octet correspondant à la couleur de la cellule avec des décalages à pas non constants (puisqu'il s'agit de ne prendre en compte que les cellules non situées sur les bords). Pour obtenir des tailles de champs alignées en mémoire, nous avons utilisé la primitive MPI_Type_extent().

Enfin, après émission d'une donnée "Type_bloc_image" par un itérateur quelconque au cours de la simulation, il faut que le superviseur réceptionne le tableau d'octets correspondant et le stocke au bon endroit dans l'image globale. Pour ce faire, nous avons créé le type dérivé Type_image_globale avec la primitive MPI_Type_vector().

Une fois encore, ce dernier type nous permet de réaliser deux opérations en une seule puisque, avec une librairie "conventionnelle", il nous aurait d'abord fallu réceptionner le bloc d'image avant de le scinder en petits blocs à répartir (ou recopier) dans divers endroits de l'image globale.

5. Conclusion : simulation parallèle 1D et 2D

Ces deux simulations ont été écrites en C et fonctionnent sur réseau de PC sous Linux (sur lesquels nous avons installé l'environnement LAM 6.1, *Local Area Multicomputer*) et sur CRAY T3E en mode séquentiel mono-processeur et parallèle sur plusieurs processeurs.

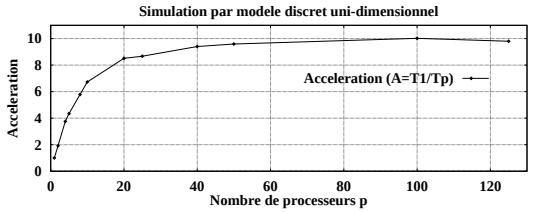

FIG. 5 – *Accélération des performances pour la simulation 1D sur* CRAY T3E *(*$1000 * 200$ *pixels, 250 clichés, 7500 pas de temps, tableau de 1000 cellules)*

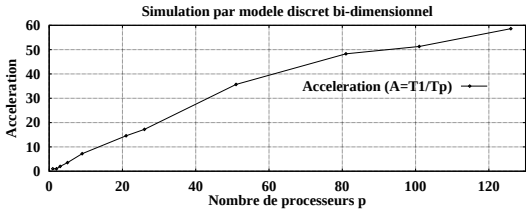

FIG. 6 – *Accélération des performances pour la simulation 2D sur* CRAY T3E *(*$1000 * 200$ *pixels, 90 clichés, 9000 pas de temps, tableau de 200000 cellules)*

Comme nous pouvions le prévoir étant donné les quantités respectives de cellules des cas 1D et 2D, l'accélération de la simulation 1D atteint un optimum de 10 (fig. 5) pour une centaine de PE, alors que celle de notre simulation 2D est quasi-linéaire et croissante sur l'intervalle d'étude (fig. 6).

Dans le cas bi-dimensionnel, l'intérêt d'une solution parallèle est flagrant puisque, alors que le temps de simulation séquentiel est de plus de 41 minutes (2462 secondes), il descend à 42 secondes dans le cas parallèle (sur 126 PE du CRAY T3E) !

Remerciements : Ces travaux ont été réalisés avec l'assistance et les moyens informatiques de l'Institut du Développement et des Ressources en Informatique Scientifique (IDRIS, France).

Bibliographie

1. BOURGOIS (Jacques). – La fosse d'Amérique Centrale : convergence, accrétion, érosion tectonique. *C.R. Acad. Sci., Sér. Gén., Vie Sci.*, vol. 10, n4, 1993, pp. 285–303.
2. LEDUC (Thomas). – *Modélisation par un système dynamique discret du processus de subduction-érosion en tectonique des plaques : première approche uni-dimensionnelle.* – Rapport interne, ftp://ftp.lip6.fr/lip6/reports/1997/lip6.1997.008.ps.gz, Laboratoire LIP6, Mai 1997.
3. LEDUC (Thomas). – A one-dimensional discrete computer model of the subduction erosion phenomenon. *In : CESA'98 Nabeul-Hammamet.* – (2e Congrés Mondial IMACS et IEEE/SMC).
4. SNIR (Marc), OTTO (Steve), HUSS-LEDERMAN (Steven), WALKER (David) et DONGARRA (Jack). – *MPI : The Complete Reference.* – MIT Press, 1995, *Scientific and Engineering Computation series.* http://www.netlib.org/utk/papers/mpi-book/mpi-book.html.