Approche placement pour la redistribution de données irrégulières

Aurélien Esnard

RenPar'17 / SympA'2006 / CFSE'5 / JC'2006
Canet en Roussillon, 4 au 6 octobre 2006

Approche placement pour la redistribution de données irrégulières

Aurélien Esnard

Projet ScAlApplix, INRIA Futurs et LaBRI UMR CNRS 5800,
351, cours de la Libération, 33405 Talence - France
esnard@labri.fr

Résumé
Dans le cadre du couplage de codes, la redistribution efficace des données est un enjeu majeur pour obtenir de bonnes performances. Or la plupart des travaux dans ce domaine se limitent le plus souvent à l'étude d'objets structurés avec des distributions régulières (e.g. tableaux denses avec des distributions bloc-cycliques). Les applications multi-physiques modernes ou les environnements de pilotage (couplage simulation-visualisation) manipule fréquemment des données irrégulières, comme des ensembles de particules ou des maillages non structurés. Il s'avère donc important de proposer des méthodes de redistribution pouvant s'appliquer à ces objets. Dans ce papier, nous proposons une nouvelle approche de la redistribution, l'approche placement, bien adaptée au contexte du pilotage.

Mots-clés : redistribution de données, couplage de codes, pilotage, données non structurées.

---

1. Introduction

Le problème de la redistribution des données est devenue aujourd'hui primordiale dans le contexte du couplage de codes. Une application est alors vue comme un assemblage de plusieurs codes, le plus souvent parallèles, collaborant à la réalisation d'un travail commun (e.g. simulations multi-physiques). Ces codes sont typiquement distribués sur une grille de calcul, interconnectés par un réseau rapide. La communication entre les codes parallèles couplés nécessite de passer d'une distribution à une autre et donc de « redistribuer les données ». On subdivise généralement le problème de la redistribution en quatre sous-problèmes : la description des données distribuées (distribution et stockage), la génération des messages (à envoyer et recevoir), l'ordonnancement des communications et la réalisation effective des transferts. Nous nous intéressons principalement dans cet article au problème de la génération des messages, qui pour être standard doit reposer sur un modèle de description des données distribuées également standard. Nous n'aborderons pas ici le problème de l'ordonnancement des communications, pour lequel nous souhaitons reposer sur des résultats existants, en particulier [1]. Les travaux présentés dans ce papier sont largement motivés par le projet EPSN (Environnement pour le Pilotage de Simulations Numériques Parallèles et Distribuées) [2] qui cherche à définir un environnement pour le pilotage des simulations numériques, permettant de coupler une simulation parallèle avec un code de visualisation lui-même parallèle, ce que nous appelons le problème du pilotage M $\times$ N (Fig. 1). Un tel couplage peut conduire à des redistributions de données plus complexes que celles étudiées par le passé.

La plupart des travaux existants dans ce domaine comme HPF [3], ScaLAPACK [4], PAWS [5] ou CCA M$\times$N [6] traitent uniquement de la redistribution de données régulières : des tableaux denses, multidimensionnels et parallélépipédiques avec des distributions par blocs plus ou moins complexes (i.e. essentiellement du bloc-cyclique). Nous avons proposé dans [7] un algorithme de redistribution visant à étendre ces travaux pour prendre en compte des distributions de tableaux plus complexes. Concernant la redistribution de données irrégulières, on distingue deux types de travaux : d'une part, des travaux spécifiques comme MpCCI (Mesh based Parallel Code Coupling Interface) [8] pour la redistribution de maillages non structurés ; et d'autre part des travaux plus généralistes basés sur le principe de linéarisation comme MetaChaos [9],ou MPI-I/O M$\times$N [10]. Le principe de linéarisation consiste à ordonner

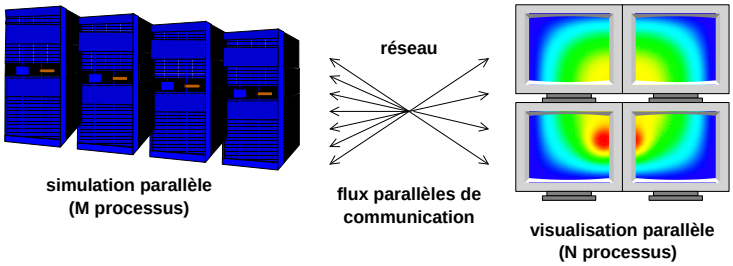

FIG. 1 – Problématique du pilotage $M \times N$ dans EPSN.

totalement les éléments d'un objet « source » et d'un objet « destination », ce qui permet de mettre implicitement et indirectement en correspondance tous les éléments. Cette approche est très puissante car elle permet de redistribuer des structures de données *a priori* quelconques (e.g. arbres, graphes, maillages, *etc.*), mais s'avère relativement coûteuse au regard des structures de données intermédiaires qui sont utilisées pour générer les messages.

Dans ce papier, nous proposons une nouvelle approche de la redistribution, baptisée *l'approche placement*, pouvant s'appliquer de manière générale à des données irrégulières comme des ensembles de particules ou des maillages non structurés. Cette approche se trouve particulièrement bien adaptée au contexte du pilotage $M \times N$, puisqu'elle exploite le fait que le code de visualisation parallèle peut choisir dynamiquement sa distribution à partir de celle de la simulation. Ainsi posé, le problème de la redistribution s'apparente simplement à un problème de placement des éléments d'un code vers l'autre. Dans la section suivante, nous introduisons les notions préliminaires utiles à la compréhension de nos algorithmes, concernant notamment les objets complexes et les messages symboliques. Puis, nous présentons aux sections 3 et 4 *l'approche placement* de la redistribution sans découpage des régions, puis avec découpage ce qui permet d'améliorer le placement. Pour terminer, nous validons nos algorithmes en présentant quelques résultats expérimentaux obtenus avec les bibliothèques RedSYM et RedCORBA (Sec. 5).

2. Redistribution symbolique d'objets complexes

2.1. Objets complexes

Le modèle de description des données que nous proposons est basé sur la notion *d'objet complexe*. Ce modèle a pour objectif de prendre en compte une grande variété de structures de données présentes dans les codes de calcul scientifique, tout en permettant une génération efficace des messages pour la redistribution. Considérons un ensemble d'éléments E totalement ordonné (ordre global). Un objet complexe se présente comme une partie de E découpée en sous-ensembles appelées *régions* et composée de plusieurs *séries de données*. Chaque élément de l'objet possède une valeur dans chaque série de données, dont une fonction d'adressage sert à préciser le stockage en mémoire pour chaque région. Lorsqu'un objet complexe est distribué sur un ensemble de processeurs P de taille M, cela revient a affecté à chaque processeur P_i un ensemble de régions, notées $R_i = (R_{i,0}, R_{i,1}, \ldots)$. Dans notre modèle, la distribution des éléments de E sur P est définie par une fonction $\sigma_P : k \longrightarrow (i, r, x)$ qui à l'indice global k d'un élément de E associe un processeur de rang i dans P, une région d'indice r sur P_i et un indice local x dans la région $R_{i,r}$. On dit alors que E est distribué sur P selon σ_P et on note $R_{i,r}[x]$ l'élément d'indice local x dans la région $R_{i,r}$ du processeur P_i avec $R_{i,r}[x] = E[k]$. Ainsi un objet complexe distribué sur P selon σ_P se définit comme une collection d'objets complexes associés à chaque P_i et notés $\mathcal{O}_i$, tels que les régions $R_{i,r}$ sont les ensembles d'éléments induits par σ_P.

Sur la base de ce modèle, nous avons proposé la définition de plusieurs classes d'objets complexes : les grilles structurées, les boîtes d'atomes, les ensembles de particules et les maillages non structurés. Nous nous intéresserons plus particulièrement dans ce papier à ces deux dernières classes. Les *ensembles de particules* sont des objets complexes qui servent à représenter des particules dispersées dans l'espace physique à n dimensions. Ces objets sont très fréquents en dynamique moléculaire ou en astrophysique.

Une série de données particulière sert à décrire la position des particules dans l'espace. L'ensemble des particules formant une région est *a priori* quelconque et permet de regrouper des « paquets de particules » ayant le plus souvent une caractéristique commune dans la simulation, comme par exemple des particules chargées positivement ou négativement ou encore différentes espèces chimiques, *etc*. Les *maillages non structurés* sont des objets[1] complexes représentant un ensemble de cellules géométriques dans l'espace physique à 2 ou 3 dimensions. Actuellement, nous ne prenons en compte que le cas de maillages formés d'un seul type ce cellule : triangle, quadrilatère, hexaèdre ou tetraèdre. Les éléments de l'objet complexe (que nous considérons du point de vue de la redistribution) sont à proprement parler les cellules du maillage. Chaque région décrit une partie du maillage (typiquement une composante connexe) regroupant un ensemble cellules reliant plusieurs nœuds. Chaque cellule est défini à l'aide d'une série de données particulière représentant la liste des connectivités. Cette liste contient les indices locaux des nœuds dans la région servant à former chaque élément. Les coordonnées des nœuds associées à la région sont représentées à l'aide d'une série de données particulière.

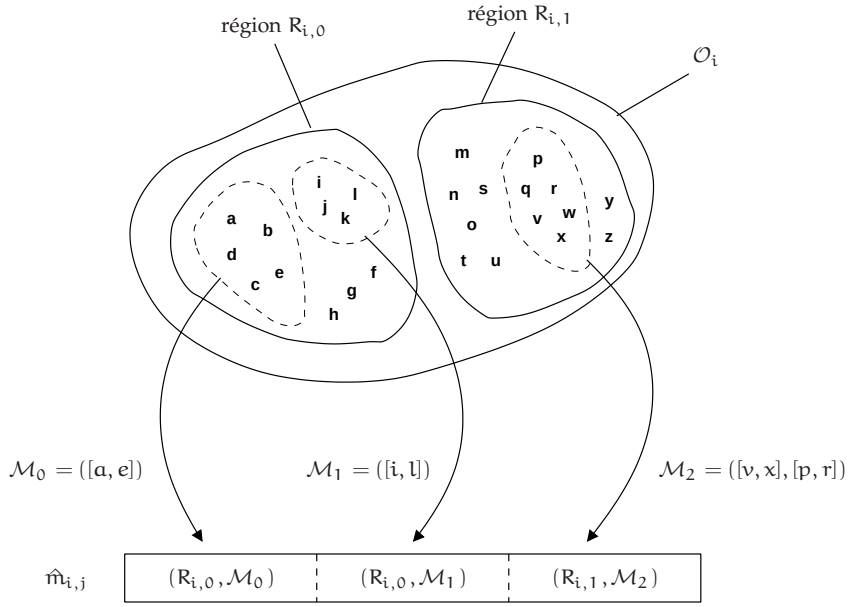

FIG. 2 – Construction d'un message symbolique $\hat{m}_{i,j}$ à l'aide de masque d'extraction.

2.2. Messages symboliques et physiques

Pour résoudre le problème de la redistribution, il faut calculer l'ensemble des messages échangés entre P et Q. Par définition, le message $m_{i,j}$ est l'ensemble des éléments qu'il faut échanger entre P_i et Q_j. Nous allons maintenant poser quelques définitions et introduire la notion de *message symbolique*, une représentation intermédiaire des messages que nous utilisons pour exprimer les messages indépendamment de l'espace de stockage et de la couche de communication. Un message symbolique $\hat{m}_{i,j}$ de P_i vers Q_j est une séquence de sous-messages, notée $(\hat{m}_{i,j,0}, \hat{m}_{i,j,1}, \hat{m}_{i,j,2}, \ldots)$, tel que le k-ème sous-message symbolique, $\hat{m}_{i,j,k}$, soit un triplet de la forme $(r_k, r'_k, \mathcal{M}_k)$ avec : r_k le numéro de la région source R_{i,r_k} de P_i, r'_k le numéro de la région cible R'_{j,r'_k} de Q_j, et $\mathcal{M}_k$ un masque d'extraction sur la région source. Le masque $\mathcal{M}_k$ est simplement défini par une séquence d'intervalles d'indices qui désigne la liste ordonnée des éléments de la région source relatifs au k-ème sous-message (Fig. 2). Un message physique $\bar{m}_{i,j,s}$ de P_i vers Q_j désigne la séquence des adresses et des tailles de données représentant l'emplacement

[1] Les maillages non structurés sont en fait définis comme une paire d'objet complexe, $\mathcal{O}^{cell}$ et $\mathcal{O}^{node}$, pour les cellules et pour les nœuds. Nous les considérerons abusivement dans notre discours comme un objet complexe classique $\mathcal{O} = \mathcal{O}^{cell}$, car la distribution des nœuds est induite par celle des cellules et réciproquement.

en mémoire des données de la série S_s à envoyer ou à recevoir, d'une manière compréhensible par une couche de communication. Notons que le message physique peut être généré très simplement à partir de la connaissance du message symbolique et de l'espace de stockage relatif à la série considérée.

2.3. Approche placement de la redistribution

Considérons deux codes couplés A et B, respectivement distribués sur deux ensembles de processeurs P et Q, de tailles M et N. Nous présentons dans cette section un algorithme parallèle de redistribution, dont le but est de générer l'ensemble des messages de P vers Q (et réciproquement) à partir de la description des objets complexes. Chaque P_i est uniquement responsable de calculer les messages qu'il devra transmettre à tous les Q_j. Cet algorithme se décompose en deux étapes principales : (1) le calcul des messages symboliques selon une opération abstraite *redistribute* et (2) la génération des messages physiques à partir des messages symboliques pour chaque série de données. La définition de l'opération *redistribute* va dépendre d'une part de la classe de l'objet considérée et d'autre part de l'approche choisie pour redistribuer les objets.

Dans cette article, nous présentons une nouvelle approche de la redistribution, baptisée *l'approche placement*, s'appliquant plus spécifiquement dans le contexte du pilotage $M \times N$. On supposera alors que le code A joue le rôle d'une simulation parallèle sur M processeurs et B celui d'un programme de visualisation parallèle sur N processeurs ($M \gg N$). Dans le cadre du pilotage $M \times N$, seul le code de simulation (code A) possède initialement un objet complexe $\mathcal{O}$ distribué sur P. Côté visualisation, le code B n'a *a priori* aucune information sur l'objet distant qu'il souhaite visualiser. L'objet $\mathcal{O}'$ est ainsi vierge de toute distribution. Il faut alors choisir cette distribution à l'initialisation du couplage, à partir des informations relatives à la distribution de l'objet distant $\mathcal{O}$. En pratique, l'acquisition de ces informations implique une étape de communication entre chaque paire de processeurs (P_i, Q_j). On peut alors formuler le problème de la redistribution comme un problème plus simple s'apparentant à un *problème de placement* des éléments d'un code vers l'autre. Dans ce cas, la distribution des éléments pour l'objet vierge peut être choisie « au mieux » afin de faciliter le couplage et minimiser le coût des communications inter-codes. Nous avons choisi d'appliquer cette approche à des objets complexes irréguliers, comme les ensembles de particules ou les maillages non structurés. Nous allons présenter aux sections suivantes deux variantes de cette approche : l'approche placement sans découpage des régions (Sec. 3) et avec découpage (Sec. 4). Les messages symboliques $\hat{m}_{i,j}$ étant construits à partir de masques d'extraction sur les régions, le problème de la redistribution va principalement consister dans les sections suivantes à calculer des masques d'extraction.

3. Placement des régions sans découpage

Le placement des régions sans découpage est une stratégie de redistribution très simple, qui consiste à mettre explicitement en relation les régions de $\mathcal{O}_i$ avec les régions de $\mathcal{O}'_j$, à l'aide d'un *tag* numérique unique servant à identifier les régions sur chaque code. On note $tag(R_{i,r})$ le *tag* de la r-ème région du processeur P_i et $tag(R'_{j,l})$ le *tag* de la l-ème région du processeur Q_j. La génération du message symbolique $\hat{m}_{i,j}$ nécessite de comparer simplement les *tags* associés aux régions des processeurs P_i et Q_j. Ainsi, si la région locale $R_{i,r}$ possède un *tag* identique à la région distante $R'_{j,l}$, cela signifie que ces régions vont partager le même ensemble d'éléments. Dans ce cas, tous les éléments de la région $R_{i,r}$ vont appartenir au sous-message $\hat{m}_{i,j,k} = (r, l, \mathcal{M})$ destiné à la région $R'_{j,l}$ avec le masque $\mathcal{M} = ([0, |R_{i,r}|-1])$. On peut ainsi résumer cette stratégie en disant que l'on échange des régions entières (une ou plusieurs) entre chaque paire de processeur (P_i, Q_j). La construction d'un message symbolique canonique nécessite de trier les sous-messages $\hat{m}_{i,j,k}$ dans un ordre identique pour P_i et Q_j. Pour ce faire, nous utilisons simplement l'ordre croissant des *tags* numériques sur les régions.

Le placement des régions du code A sur le code B peut être fixé explicitement par l'utilisateur à partir de la description de $\mathcal{O}$ ou être calculé automatiquement par *une fonction de placement*. Afin de simplifier notre propos, nous supposons que le nombre d'éléments dans les régions de $\mathcal{O}$ est globalement équilibré, ce qui est généralement le cas pour la plupart des codes de simulations numériques. Sous cet hypothèses, il est possible de définir une fonction de placement très simple ne dépendant que de M, de

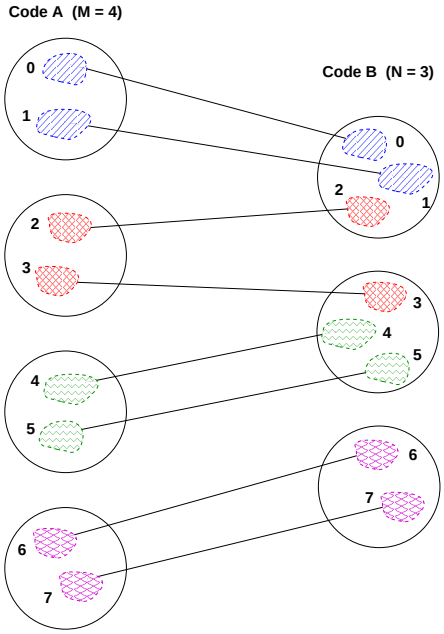

FIG. 3 – Placement sans découpage.

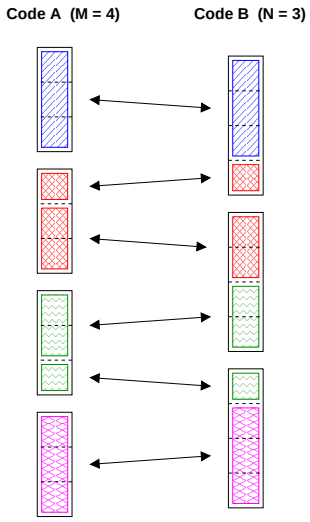

FIG. 4 – Placement avec découpage.

N et du nombre de régions $|R_i|$ associé à chaque P_i. Soient $|R|$ le nombre total de régions et z le reste de la division entière de $|R|$ par M. Le nombre de régions $|R'_j|$ affectés à Q_j est $\lceil |R|/M \rceil$ si $j < z$ et $\lfloor |R|/M \rfloor$ sinon. Si $|R|$ est divisible par M, tous les Q_j possèdent exactement le même nombre de régions, sinon il diffère au pire de 1. Dans l'exemple de la figure 3, le code A est distribué sur $M = 4$ processeurs avec 2 régions par processeur, et B est distribué sur $N = 3$ processeurs. Dans ce cas, nous choisissons de placer les $|R| = 8$ régions de la façon suivante : 3 régions sur Q_0 , 3 régions sur Q_1 et 2 régions sur Q_2.

En conclusion, il faut souligner que cette stratégie de redistribution est très simple à mettre en œuvre et peut s'appliquer de manière très générale à tout type d'objet complexe, et pas uniquement aux ensembles de particules ou aux maillages non structurés. En revanche, cette approche peut conduire à un mauvais équilibrage de la charge du code B si les régions du code A ne sont pas à grain suffisamment fin par rapport au nombre total d'éléments.

4. Placement avec découpage

Une idée simple pour surmonter les difficultés de la méthode précédente consiste à autoriser le découpage des régions pour ajuster la taille des messages échangés entre A et B. Afin de simplifier notre propos, nous allons nous placer dans le cas où le nombre de régions est fixé à 1 pour chaque processeur P_i (région $R_{i,0}$) et chaque processeur Q_j (région $R'_{j,0}$). De plus, nous supposerons que le nombre d'éléments dans chaque région de P_i est identique et égal à m. Notons qu'il est possible de généraliser cet algorithme sans trop de difficultés au cas où les tailles des régions sont différentes.

Cet algorithme repose sur la définition de l'opération *split*, une opération abstraite de découpage d'une région en n parties de poids $(p_0, p_1, \ldots, p_{n-1})$, tels que la somme des poids $p_k > 0$ soit égal à 1. La k-ème partie est simplement définie par un masque d'extraction $\mathcal{M}_k$ sur la région, désignant un ensemble de $p_k.m$ éléments. L'opération *split* permet de calculer les masques d'extraction servant à générer les messages symboliques. Dans le cas des ensembles de particules, il est possible de considérer un découpage en n intervalles d'éléments successifs (selon l'ordre local au région). En revanche, le découpage d'un maillage non structuré en n parties de poids différents nécessite d'utiliser un algorithme de par-

titionnement, comme ceux implantés dans Scotch [11] ou Metis [12]. Notons que le partitionnement des cellules d'un maillage induit implicitement un partitionnement des nœuds portées par les cellules. Par conséquent, nous manipulerons en plus du masque classique sur les éléments (i.e. les cellules) noté $\mathcal{M}^{cell}$, un masque sur les nœuds noté $\mathcal{M}^{node}$, simplement calculé à partir de $\mathcal{M}^{cell}$.

Le calcul d'un placement des éléments de A vers B se base sur un découpage en $M \times N$ unités logiques de l'objet $\mathcal{O}$ et de l'objet distant $\mathcal{O}'$. Les unités logiques sont simplement une abstraction qui va nous permettre de calculer plus facilement les poids servant au découpage des régions. Chaque région locale $R_{i,0}$ est décomposée en N unités logiques et chaque région distante $R'_{j,0}$ est décomposée en M unités logiques. On note $u_{i,k}$ la k-ème unité logique de la région locale $R_{i,0}$ ($0 \leq k < N$) et $u'_{j,k'}$ la k'-ème unité logique de la région distante $R'_{j,0}$ ($0 \leq k' < M$). La figure 4 illustre un découpage logique en 12 unités logiques ($M = 4$ et $N = 3$) et leur placement sur les processeurs distants selon notre algorithme. Les unités logiques sont numérotées par ordre de processeurs croissants de 0 à $M.N - 1$ et chaque unité locale $u_{i,k}$ d'indice global $i \times N + k$ est associée à l'unité distante $u'_{j,k'}$ de même indice, c'est-à-dire telle que $i \times N + k = j \times M + k'$. Plus précisément, si l'on note $U_i = [i.N, (i+1).N - 1]$ l'intervalle des indices globaux relatifs aux unités logiques de P_i et de même pour l'intervalle $U'_j = [j.M, (j+1).M - 1]$ sur Q_j, alors le message symbolique $\hat{m}_{i,j}$ est constitué des unités logiques dont l'indice global est à l'intersection de ces deux intervalles : $U_i \cap U'_j$. Lorsque l'intersection est nulle, il n'y a aucun message échangé entre P_i et Q_j. Il est facile de voir que chaque P_i envoie au moins 1 message aux processeurs de Q et au plus 2 messages. On peut démontrer que le nombre total de messages échangés est alors de $M + N - PGCD(M, N)$[2]. Notons que dans le cas particulier où M et N sont des multiples (en supposant $M \geq N$) alors chaque P_i envoie exactement 1 message de N unités logiques consécutives (i.e. pas de découpage) et chaque Q_j en reçoit exactement M/N. Dans l'exemple de la figure 4, cette stratégie de placement génère un total de 6 messages pour $M = 4$ et $N = 3$, tels que $U_0 \cap U'_0 = [0, 2], U_1 \cap U'_0 = [3, 3], U_1 \cap U'_1 = [4, 5], U_2 \cap U'_1 = [6, 7], U_2 \cap U'_2 = [8, 8]$ et $U_3 \cap U'_2 = [9, 11]$. Les poids relatifs au découpage des régions des processeurs de P sont alors respectivement $(3/3)$ pour P_0, $(1/3, 2/3)$ pour P_1, $(2/3, 1/3)$ pour P_2 et $(3/3)$ pour P_3.

D'une manière générale, la réception des messages pour Q_j se base sur l'opération abstraite *merge*, dont le rôle est de fusionner plusieurs messages à destination d'une même région $R'_{j,0}$. Dans le cas des ensembles de particules, cette opération correspond simplement à une accumulation des données reçues, message par message, dans un ordre bien déterminé pour conserver les éléments de la région $R'_{j,0}$ dans un ordre identique d'une réception à l'autre. Dans le cas des maillages non structurés, nous avons dû adapté ce mécanisme pour prendre également en compte la série des connectivités. En effet, les indices des nœuds dans la série des connectivités font référence aux indices locaux dans la région $R_{i,0}$. Par conséquent, ces indices doivent être convertis en une numérotation locale à la région $R'_{j,0}$. Cette conversion s'effectue en deux étapes, une première qui consiste à convertir les indices des nœuds reçus en une numérotation interne au message de rang k s'étendant de 0 à $n_k - 1$, avec n_k le nombre total de nœuds utilisés dans ce message. Notons que cette nouvelle numérotation doit respecter l'ordre croissant des indices de nœuds définis dans la région source. La deuxième étape de la conversion consiste simplement à translater les indices des nœuds de $d = \sum_{k'=0}^{k-1} n_{k'}$ pour prendre en compte l'accumulation des nœuds dans la région $R'_{j,0}$ relative aux $k - 1$ messages précédents.

5. Résultats

Afin de valider notre approche, nous avons développé un environnement logiciel pour le couplage de codes parallèles et plus précisément pour la redistribution des objets complexes. Cet environnement se compose de deux bibliothèques, appelées RedSYM et RedCORBA[3]. La bibliothèque RedSYM implante

[2] Pour établir ce résultat, il faut se ramener au cas où M et N sont premiers entre eux. Si l'on examine « le motif d'intersection » dans ce cas, il est facile de voir que le nombre de messages est exactement de $M + N - 1$. Lorsque M et N ont un multiple commun, il est alors possible de se ramener au cas précédent en posant $M' = M/PGCD(M, N)$ et $N' = N/PGCD(M, N)$. Le motif d'intersection obtenu pour M' et N' se répète autant de fois que le PGCD de M et N. Le nombre de messages total est alors $PGCD(M, N) \times (M' + N' - 1) = M + N - PGCD(M, N)$.

[3] Ces travaux ont été réalisés dans le cadre de l'ACI GRID EPSN et de l'ARC RedGRID.

le modèle de description des données et les algorithmes présentés dans ce papier. Plus précisément, RedSYM calcule (en parallèle) les messages symboliques à partir de la description des objets complexes. Cette bibliothèque est dite *symbolique* dans le sens où son interface est clairement indépendante d'une couche de communication. La bibliothèque RedCORBA implante une couche de communication au dessus de RedSYM, basée sur la technologie CORBA. Elle permet de coupler dynamiquement plusieurs codes parallèles (e.g. MPI, PVM, *etc*.) sur un réseau hétérogène. Elle prend en charge l'échange des descriptions d'objet entre les codes couplés, et le transfert parallèle des données d'un code à l'autre. Pour plus de détails concernant RedSYM et RedCORBA, nous renvoyons le lecteur à l'article [7].

Les expériences pour l'évaluation de la redistribution ont entièrement été réalisées sur le cluster *Grid'5000* à Bordeaux. Les PCs de ce cluster sont interconnectés par un réseau Giga-Ethernet via un switch. Dans ces expériences, nous examinons le problème de la redistribution pour les ensembles de particules et des maillages non structurés en considérant différentes configurations $M \times N$ en comparant les stratégies de placement avec ou sans découpage. Nous mesurons le débit cumulé (en Mo/s), défini comme la taille totale des données distribuées divisée par le temps total du transfert, pour différentes tailles de données. Nous effectuons principalement des mesures de débits cumulés, car elles permettent de mettre en évidence l'agrégation possible de la bande passante lors des flux parallèles de communication.

5.1. Cas des ensembles de particules

Nous allons maintenant présenter quelques résultats expérimentaux dans le cas des ensembles de particules pour les stratégies de placement avec ou sans découpage, dans les configurations 8×8 et 8×7 (Fig. 5). Les particules sont initialement distribuées de manière équilibrée entre tous les processeurs émetteurs. La stratégie de placement avec découpage donne des résultats identiques à la stratégie sans découpage pour des cas triviaux (8×8), mais offre de bien meilleures performances lorsque M et N sont premiers entre eux. Dans le cas 8×7, le débit cumulé est même divisé par 2, ce qui s'explique par le fait que le premier processeur côté réception se voit attribuer lors du placement sans découpage deux régions contre une seule pour les autres récepteurs. Le temps nécessaire pour générer les messages CORBA avec RedSYM et RedCORBA (incluant l'échange initial des descriptions d'objet entre les codes couplés) ne dépend pas du nombre de particules ; il est de l'ordre de 10 ms dans les expériences précédentes.

5.2. Cas des maillages non structurés

Nous reproduisons des expériences similaires dans le cas des maillages non structurés. Il s'agit de comparer le comportement des redistributions de maillages en 8×8 et en 8×7 pour les stratégies de placement avec ou sans découpage (Fig. 6). Le maillage considéré est une grille de quadrilatères définie comme un maillage non structuré (i.e. liste des connectivités explicite), initialement découpée en colonne sur chaque processeur côté émetteur. Dans le cas trivial 8×8, le placement calculé est identique pour les deux stratégies et l'on pourrait s'attendre à obtenir d'aussi bonnes performances avec découpage que sans découpage. La différence de performance tient au fait que nous utilisons un cache à l'envoi pour la stratégie avec découpage, car cette stratégie produit des messages naturellement très décousus, sauf dans certains cas triviaux comme celui-ci. Le cas 8×7 est là pour mettre en évidence l'intérêt de la stratégie avec découpage pour équilibrer les flux de communication et obtenir de meilleures performances. En effet, on constate que le débit est multiplié par 1.5 par rapport à la stratégie sans découpage. Notons que le débit cumulé n'est pas tout à fait multiplié par 2, à cause de l'utilisation d'un cache à l'envoi pour construire un message contigu. Le temps nécessaire pour générer ces messages est très supérieure pour la stratégie avec découpage où nous avons recours à un algorithme plus complexe de partitionnement des maillages actuellement basé sur Metis, de l'ordre de 250 ms pour un maillage de 160 000 quadrilatères, contre 10 ms dans l'autre cas. D'une manière générale, le temps de génération des messages est constant (dépendant de M et N uniquement) dans la stratégie sans découpage alors qu'il est sur-linéaire par rapport au nombre de cellules du maillage dans la stratégie avec découpage.

6. Conclusion

Dans ce papier, nous abordons la problématique de la redistribution des données pour des données irrégulières, tels que les ensembles de particules ou les maillages non structurés, des objets très fréquents

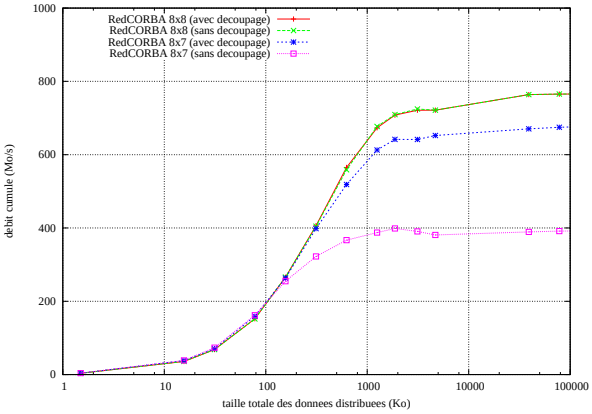

FIG. 5 – Cas des ensembles de particules.

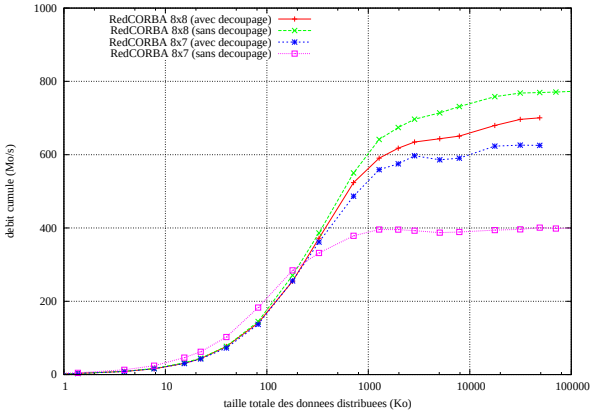

FIG. 6 – Cas des maillages non structurés.

dans les codes de simulations numériques. Nous avons présenté une nouvelle approche de la redistribution, l'approche placement avec ou sans découpage, qui se trouve particulièrement bien adaptée dans le contexte du pilotage M $\times$ N, mais offre également certaine perspective pour le contexte du couplage de codes. Ces algorithmes sont déjà utilisés dans l'environnement EPSN, grâce aux bibliothèques RedSYM et RedCORBA, pour piloter Les expériences ont montré l'intérêt de l'approche placement avec découpage pour des cas irrégulier : bien que plus complexe du point de vue algorithmique, cette approche génère des messages plus équilibrés et offre donc de meilleures performances pour les transferts.

Bibliographie

1. E. Jeannot and F. Wagner. Two Fast and Efficient Message Scheduling Algorithms for Data Redistribution through a Backbone. *IPDPS*, 2004.
2. A. Esnard, M. Dussere, and O. Coulaud. A Time-coherent Model for the Steering of Parallel Simulations. In *Euro-Par 2004 Parallel Processing*, pages 90–97. Springer LNCS, 2004.
3. C. Koebel, D. Loveman, R. Schreiber, G. Steele Jr., and M. Zosel. *The High Performance Fortran Handbook*. MIT Press, Cambridge, 1994.
4. L.S. Blackford, J. Choi, A. Cleary, E. D'Azevado, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R.C. Whaley. ScaLAPACK Users'Guide. Technical report, SIAM, 1997.
5. P. H. Beckman, P. K. Fasel, W. F. Humphrey, and S. M. Mniszewski. Efficient Coupling of Parallel Applications Using PAWS. *The Tenth IEEE International Symposium on High Performance Distributed Computing (HPDC)*, 1998.
6. F. Bertrand, R. Bramley, K. B. Damevski, J. A. Kohl, D. E. Bernholdt, J. W. Larson, and A. Sussman. Data Redistribution and Remote Method Invocation in Parallel Component Architectures. In *Proceedings of the 19th International Parallel and Distributed Processing Symposium : IPDPS 2005*, 2005.
7. A. Esnard. Modèle pour la redistribution de données complexes. 16ème Rencontres francophones du parallélisme (RenPar'16), 2005.
8. MpCCI : Mesh-based parallel code coupling Interface. http ://www.mpcci.org.
9. M. Ranganathan, A. Acharya, G. Edjlali, A. Sussman, and J. Saltz. Runtime Coupling of Data-Parallel Programs. In *ICS '96 : Proceedings of the 10th international conference on Supercomputing*, pages 229–236, New York, NY, USA, 1996. ACM Press.
10. F. Bertrand, Y. Yuan, K. Chiu, and R. Bramley. An Approach to Parallel MxN Communication. In *12th High Performance Distributed Computing (HPDC)*, Seattle, WA, June 2003.
11. F. Pellegrini. Graph Partitioning based Methods and Tools for Scientific Computing. *Parallel Computing ETPSC'3*, 23 :153–164, 1997.
12. G. Karypis and V. Kumar. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. Technical Report 95-035, University of Minnesota, june 1995.