

Modélisation et implémentation de parallélisme implicite pour les simulations scientifiques basées sur des maillages

Soutenance de thèse

Hélène Coullon

Thèse dirigée par **Sébastien Limet**, professeur à l'Université d'Orléans
Convention CIFRE entre **Géo-Hyd** Antea group, et le LIFO

helene.coullon@univ-orleans.fr

29 septembre 2014



Le parallélisme ?



Ma thèse ne porte pas sur un réglage automobile !

Années 60



Obladi Oblada 🎵

Le parallélisme

Années 60



Calcul



1 seul processeur



Résultat

1 processeur & plus de puissance → attente & puissance limitée

Le parallélisme

Années 60



Calcul



1 seul processeur

Résultat

Parallélisme !

Calcul

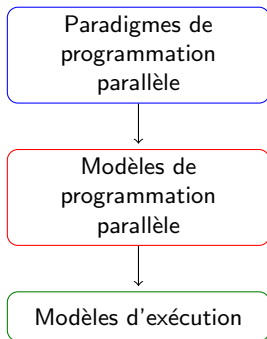


Gamma 60 - Multi-tâches
et multi-processeurs

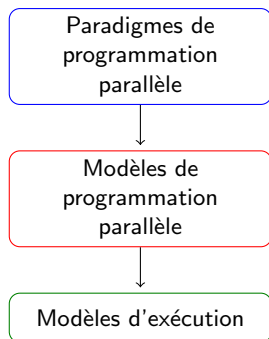
Résultat

n processeurs & plus de puissance → pas d'attente & puissance moins limitée

Le parallélisme et la programmation parallèle en 2014



Le parallélisme et la programmation parallèle en 2014



Parallélisme de données

Parallélisme de tâches

Passage de messages ...

Grappes

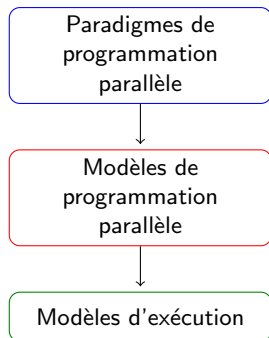
Multi-cœurs

GPGPU

Many-cœurs ...



Le parallélisme et la programmation parallèle en 2014



Parallélisme de données

Parallélisme de tâches

Passage de messages ...

BSP

MPI

OpenMP

Cuda

OpenCL ...

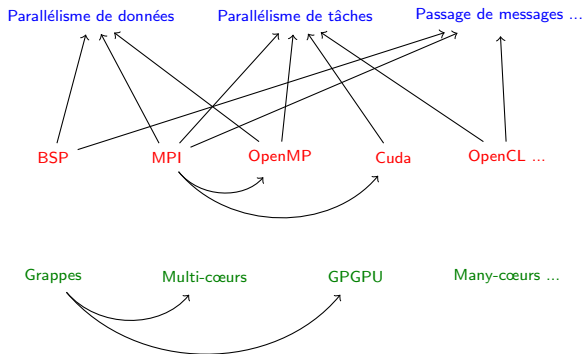
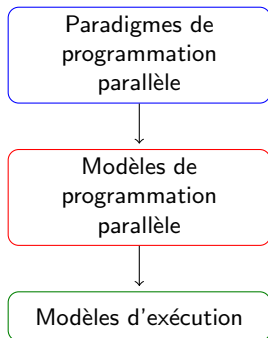
Grappes

Multi-cœurs

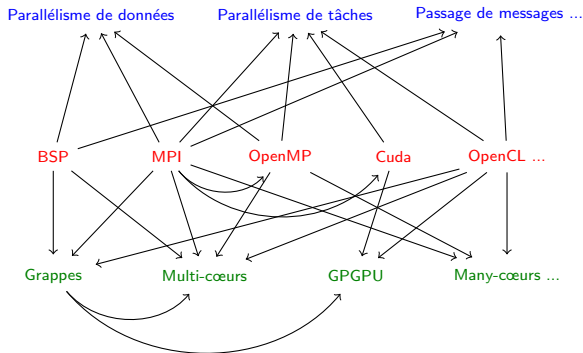
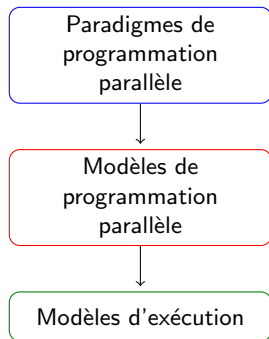
GPGPU

Many-cœurs ...

Le parallélisme et la programmation parallèle en 2014



Le parallélisme et la programmation parallèle en 2014

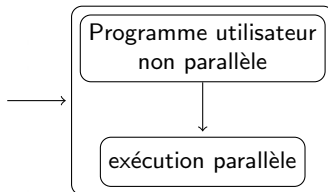


Domaine très complexe !

Le parallélisme implicite

Le parallélisme implicite

Proposer des solutions de programmation où le parallélisme est totalement implicite pour l'utilisateur

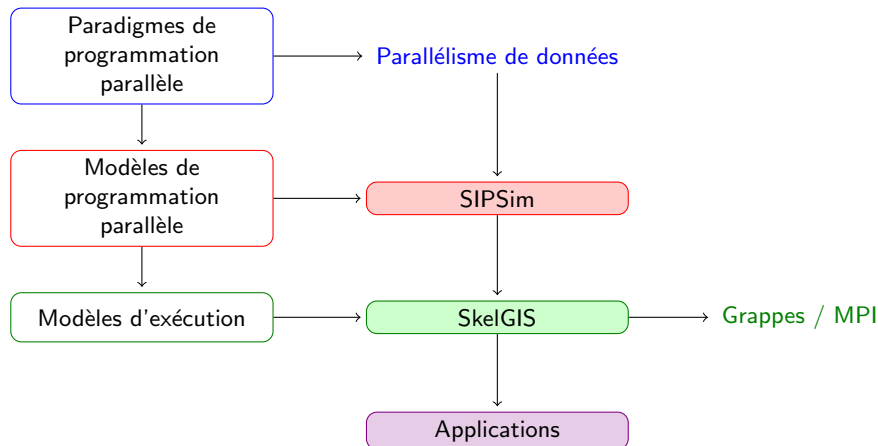


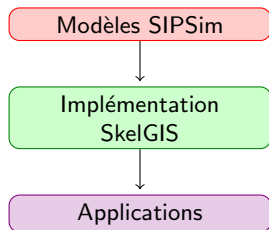
Modèle de programmation parallèle implicite

Modèles de programmation parallèle implicite :

- Conteneurs parallèles
- Patrons
- Solutions de domaine spécifiques

Parallélisme implicite pour les simulations numériques

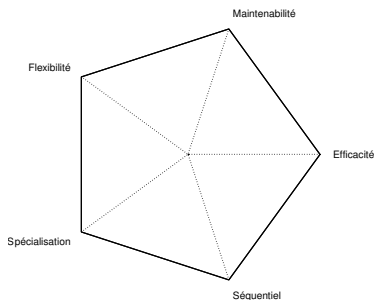




- 1 Positionnement du travail
- 2 Simulations numériques et modèle SIPSim
- 3 Implémentation : SkelGIS
- 4 Applications et résultats
- 5 Conclusion et perspectives

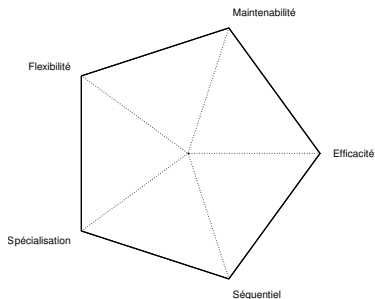
Positionnement

Critères de positionnement



Trois solutions majeures de parallélisme implicite : (1) Les solutions générales à conteneurs parallèles ; (2) les solutions à patrons ; (3) les solutions de domaine spécifique

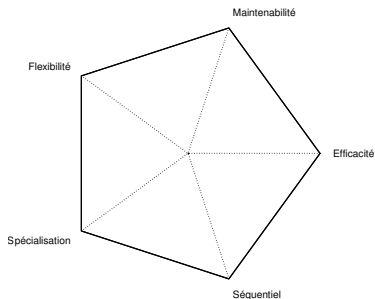
Critères de positionnement



- **Efficacité** du programme obtenu

Trois solutions majeures de parallélisme implicite : (1) Les solutions générales à conteneurs parallèles ; (2) les solutions à patrons ; (3) les solutions de domaine spécifique

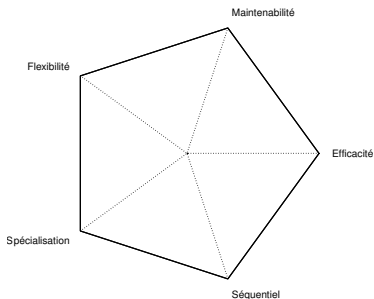
Critères de positionnement



- **Efficacité** du programme obtenu
- Reste-t-on proche d'un programme **séquentiel** ?

Trois solutions majeures de parallélisme implicite : (1) Les solutions générales à conteneurs parallèles ; (2) les solutions à patrons ; (3) les solutions de domaine spécifique

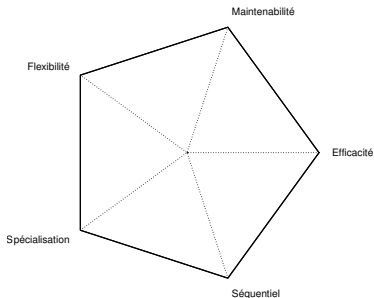
Critères de positionnement



- **Efficacité** du programme obtenu
- Reste-t-on proche d'un programme **séquentiel** ?
- La solution est-elle **spécifique** à un domaine ?

Trois solutions majeures de parallélisme implicite : (1) Les solutions générales à conteneurs parallèles ; (2) les solutions à patrons ; (3) les solutions de domaine spécifique

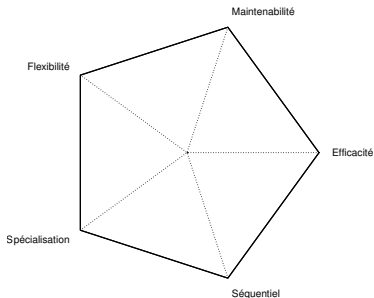
Critères de positionnement



- **Efficacité** du programme obtenu
- Reste-t-on proche d'un programme **séquentiel** ?
- La solution est-elle **spécifique** à un domaine ?
- Le code est-il **flexible** (cas non-standards...)

Trois solutions majeures de parallélisme implicite : (1) Les solutions générales à conteneurs parallèles ; (2) les solutions à patrons ; (3) les solutions de domaine spécifique

Critères de positionnement



- **Efficacité** du programme obtenu
- Reste-t-on proche d'un programme **séquentiel** ?
- La solution est-elle **spécifique** à un domaine ?
- Le code est-il **flexible** (cas non-standards...)
- La solution est-elle facile à **maintenir** ?

Trois solutions majeures de parallélisme implicite : (1) Les solutions générales à conteneurs parallèles ; (2) les solutions à patrons ; (3) les solutions de domaine spécifique

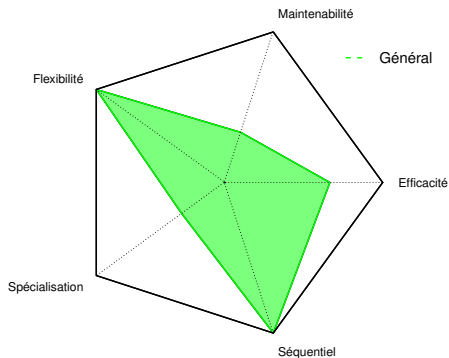
Solutions générales à conteneurs parallèles

Ces solutions proposent une bibliothèque de conteneurs distribués, sur lesquels appliquer des algorithmes, et un ensemble d'outils pour les manipuler.

Liens entre les paramètres

- flexibilité $\wedge$ séquentiel

Exemple : STL parallèle



Solutions à patrons

Ces solutions proposent une programmation fonctionnelle où les fonctions représentent des paradigmes de parallélisation automatiquement mis en place.

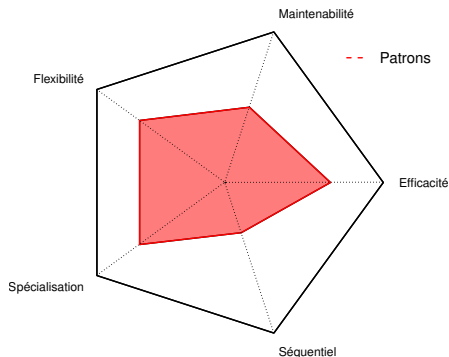
Liens entre les paramètres

- flexibilité $\wedge$ séquentiel
- spécialisation $\wedge$ maintenabilité

Blocages entre les paramètres

- séquentiel $\oplus$ spécialisation
- flexibilité $\oplus$ spécialisation

Exemple : SkeTo



Solutions de domaine spécifique

Solutions très spécialisées, conçues pour un domaine précis.

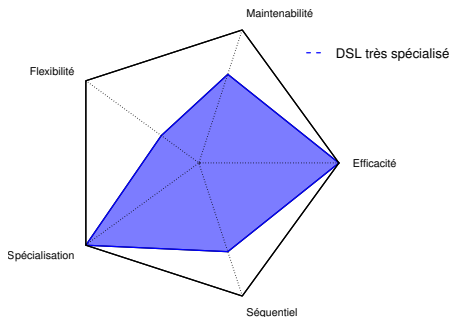
Liens entre les paramètres

- flexibilité $\wedge$ séquentiel
- spécialisation $\wedge$ maintenabilité
- spécialisation $\wedge$ efficacité

Blocages entre les paramètres

- séquentiel $\oplus$ spécialisation
- flexibilité $\oplus$ spécialisation

Exemples : ZPL, PETSc



Solutions de domaine spécifique

Solutions très spécialisées, conçues pour un domaine précis.

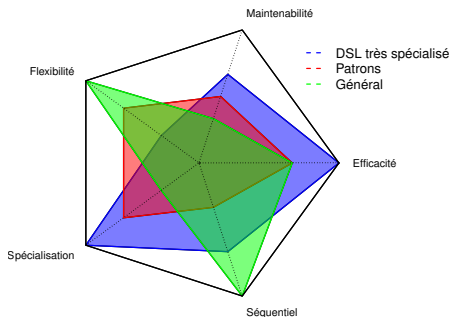
Liens entre les paramètres

- flexibilité $\wedge$ séquentiel
- spécialisation $\wedge$ maintenabilité
- spécialisation $\wedge$ efficacité

Blocages entre les paramètres

- séquentiel $\oplus$ spécialisation
- flexibilité $\oplus$ spécialisation

Exemples : ZPL, PETSc



Structured Implicit Parallelism for scientific SIMulations

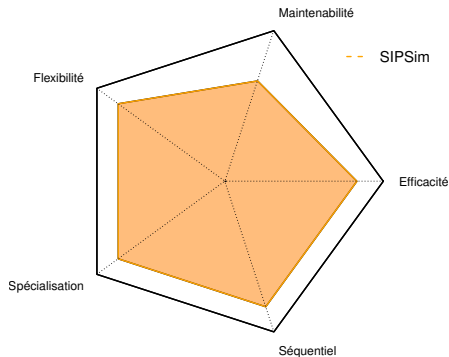
*SIPSIm est un modèle de parallélisme implicite pour le **domaine des simulations numériques basées sur les maillages**.*

Liens entre les paramètres

- flexibilité $\wedge$ séquentiel
- spécialisation $\wedge$ maintenabilité
- spécialisation $\wedge$ efficacité

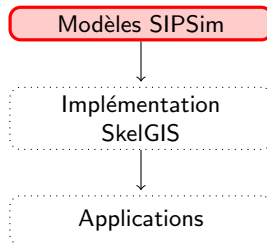
Blocages entre les paramètres

- séquentiel $\oplus$ spécialisation
- flexibilité $\oplus$ spécialisation

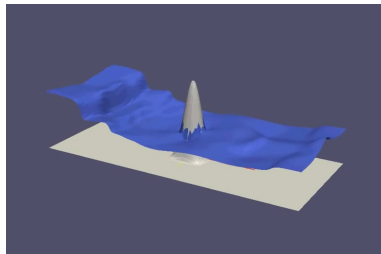
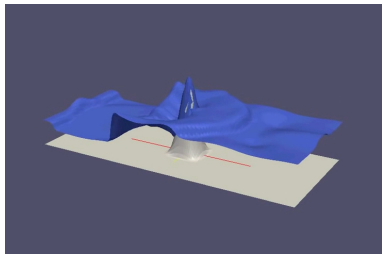
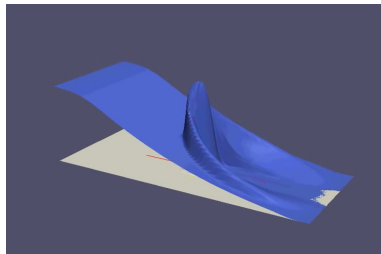
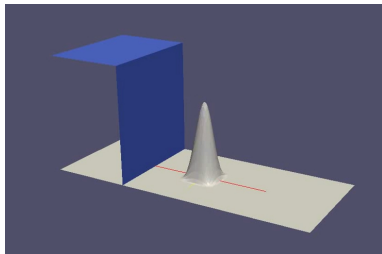


Son implémentation est libre, sous forme de bibliothèque, de framework ou encore de DSL etc., et peut être envisagée pour plusieurs types de modèles d'exécution.

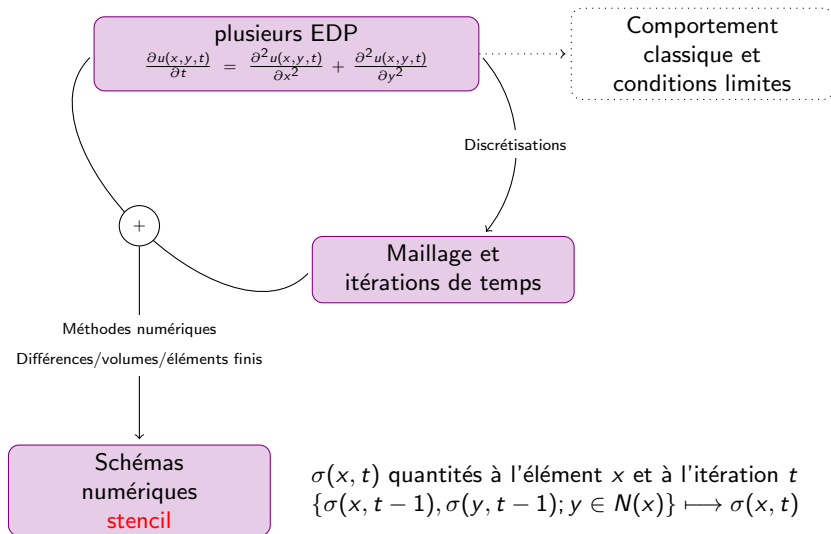
Simulations numériques et modèle SIPSim



Simulations numériques



Simulations numériques



Programme séquentiel

Création du maillage μ

Création des quantités à simuler appliquées sur μ

Initialisation des quantités et des paramètres

Définition du pas de temps : t

Définition du temps maximal : t_{max}

tant que $t < t_{max}$ **faire**

pour chaque élément de la bordure physique **faire**

 Calcul des stencils des conditions limites

fin

pour chaque $x \in \mu$ **faire**

 Calcul des stencils des éléments du maillage

fin

$t = t+1$

fin

Parallélisme de données

Création du maillage μ

Partitionnement du maillage $\mu = \{\mu_0, \mu_1, \dots, \mu_{p-1}\}$

Création des quantités à simuler appliquées sur μ

Initialisation des quantités et des paramètres

Définition du pas de temps, commun à tous les processeurs : t

Définition du temps maximal, commun à tous les processeurs : t_{max}

tant que $t < t_{max}$ **faire**

Communication/accès du voisinage stencil N entre les processeurs

pour chaque élément de la bordure physique **faire**

| Calcul des stencils des conditions limites avec N

fin

pour chaque $x \in \mu_i$ **faire**

| Calcul des stencils des éléments du maillage avec N

fin

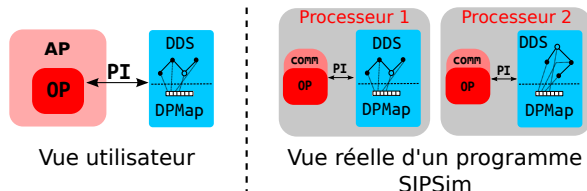
$t = t+1$

fin

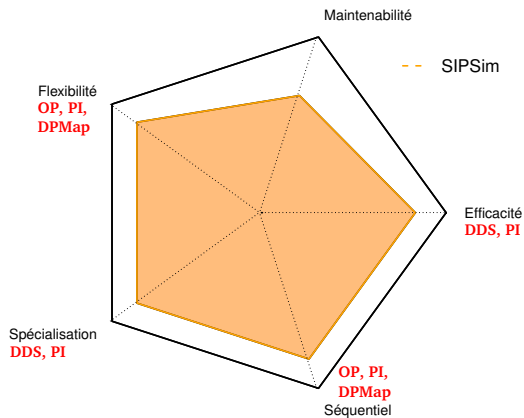
Le modèle SIPSIm

5 concepts SIPSIm pour cacher les lignes rouges

- Structure de données distribuée (DDS)
- Application de propriétés distribuées (DMap)
- Appicateurs et opérations (AP, OP)
- Interfaces pour écrire les opérations (PI)
 - ▶ itérateurs
 - ▶ accesseurs aux données d'un itérateur
 - ▶ accesseurs aux données voisines d'un itérateur



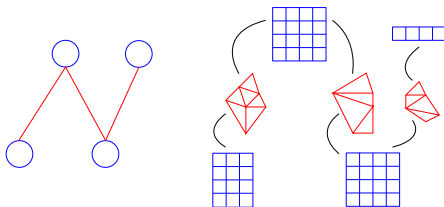
Le modèle SIPSim



Extension du modèle à d'autres simulations ?

Simulation sur un réseau

Simulation de deux phénomènes physiques reliés par un réseau



Exemples : simulation sur les réseaux pétrolières, le trafic routier, le réseau sanguin etc.

- Potentiellement deux discrétisations différentes
- Deux ensembles d'EDP différents
- Des schémas numériques différents

Parallélisme de données

Création du réseau *net*

Partitionnement du réseau $net = \{net_0, net_1, \dots, net_{p-1}\}$

Création des quantités à simuler appliquées sur *net*

Initialisation des quantités et des paramètres

Définition du pas de temps, commun à tous les processeurs : *t*

Définition du temps maximal, commun à tous les processeurs : *tmax*

tant que $t < tmax$ **faire**

Communication/accès du voisinage de *net* N_{net} **entre les processeurs**

pour chaque élément de la bordure physique de *net* **faire**

 Calcul des stencils des conditions limites du réseau *net* avec N_{net}

fin

pour chaque $node \in net_i$ **faire**

 Calcul des stencils des nœuds avec N_{net}

fin

pour chaque $edge \in net_i$ **faire**

 Calcul des stencils des arêtes avec N_{net}

fin

$t = t+1$

fin

Le modèle SIPSIm pour les réseaux

Réseau $\neq$ Maillage

MAIS on retrouve la même structure de programme parallèle

ET on retrouve les mêmes problématiques :

- partitionnement du réseau
- application de données au réseau
- manipulation des données et des voisinages
- cacher les communications

Application de SIPSIm aux réseaux

- Structure de données distribuée (DDS) = réseau distribué
- Application de propriétés distribuées (DPMa) = DPMa_Nodes, DPMa_Edges
- Appicateurs et opérations (AP, OP)
- Interfaces pour écrire les opérations (PI) = voisinage de réseau

Programmation SIPSIm

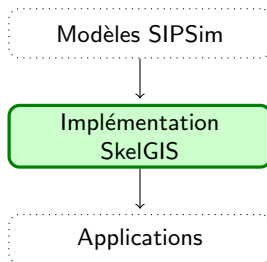
Fonction principale

Création du maillage distribué *DDS*
 Instanciation des *DPMMap* sur le *DDS*
 Initialisations des instances *DPmap* et des paramètres
 Définition du pas de temps, commun à tous les processeurs : Δt
 Définition du temps maximal, commun à tous les processeurs : t_{max}
tant que $t < t_{max}$ **faire**
 applicateur(*DPMMap**,*opération1*)
 applicateur(*DPMMap**,*opération2*)
 $t = t + \Delta t$
fin

Exemple d'opération

$itB :=$ *itérateur* de démarrage de la bordure physique
 $enditB :=$ *itérateur* de fin de la bordure physique
tant que $itB \leq enditB$ **faire**
 Accéder aux valeurs des *DPMMap* à itB
 Accéder les valeurs des *DPMMap* voisines à itB
 Calcul du stencil
 $itB++$
fin
 $it :=$ *itérateur* de démarrage des autres éléments
 $endit :=$ *itérateur* de fin des autres éléments
tant que $it \leq endit$ **faire**
 Accéder aux valeurs des *DPMMap* à it
 Accéder aux valeurs des *DPMMap* voisines à it
 Calcul du stencil
 $it++$
fin

Implémentation : SkelGIS



SkelGIS

Implémentation du modèle implicite SIPSIm

- librairie C++ “header-only” (orientée template)
- architectures à mémoire distribuée (grappes)
- utilisation de la bibliothèque MPI

SkelGIS implémente 2 solutions :

- maillages cartésiens 2D



Helene Coullon, Sébastien Limet - HPCS 2013

Algorithmic skeleton library for scientific simulations : SkelGIS

- réseaux (Réduction du problème aux graphes dirigés acycliques (DAG) et aux maillages a une dimension dans les nœuds et les arêtes)



Helene Coullon, Sébastien Limet - Euro-Par 2014

Implementation and Performance Analysis of SkelGIS for Network Mesh-based Simulations

Implémentation d'un DDS

Le DDS doit fournir

- structure de données **distribuée**
- **partitionnement** automatique de la structure
- accès efficace aux **voisinages**

Optimisations

- recouvrement des communications par les calculs
- éviter les conditions coûteuses dans le code
- optimiser l'utilisation de la mémoire cache

Le format CSR

Pour un graphe $G = (V, E)$ où $V = \{v_0, \dots, v_{n-1}\}$, et pour un sommet $v_i \in V$,

- $N(v_i)$ représente l'ensemble des **voisins** pour le sommet v_i ,
- le degré du sommet v_i est noté $\deg(v_i) = |N(v_i)|$,
- le **degré cumulatif** est représenté par $cdeg(v_i) = \sum_{j=0}^i \deg(v_j) = \sum_{j=0}^i |N(v_j)|$

Compressed Sparse Row format

Pour un graphe G , si $Sp(G)$ est la matrice d'adjacence (creuse) de G , alors ce format correspond au format CSR de $Sp(G)$.

Le format CSR

Pour un graphe $G = (V, E)$ où $V = \{v_0, \dots, v_{n-1}\}$, et pour un sommet $v_i \in V$,

- $N(v_i)$ représente l'ensemble des **voisins** pour le sommet v_i ,
- le degré du sommet v_i est noté $deg(v_i) = |N(v_i)|$,
- le **degré cumulatif** est représenté par $cdeg(v_i) = \sum_{j=0}^i deg(v_j) = \sum_{j=0}^i |N(v_j)|$

Compressed Sparse Row format

Pour un graphe G , si $Sp(G)$ est la matrice d'adjacence (creuse) de G , alors ce format correspond au format CSR de $Sp(G)$.



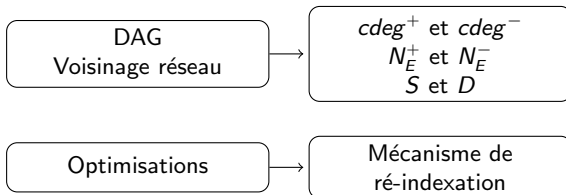
Le format CSR

Pour un graphe $G = (V, E)$ où $V = \{v_0, \dots, v_{n-1}\}$, et pour un sommet $v_i \in V$,

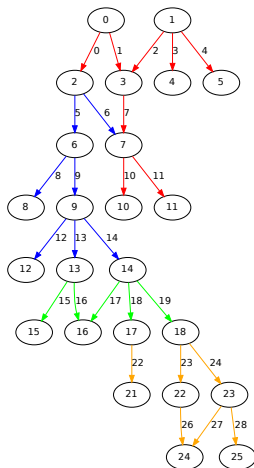
- $N(v_i)$ représente l'ensemble des **voisins** pour le sommet v_i ,
- le degré du sommet v_i est noté $\text{deg}(v_i) = |N(v_i)|$,
- le **degré cumulatif** est représenté par $\text{cdeg}(v_i) = \sum_{j=0}^i \text{deg}(v_j) = \sum_{j=0}^i |N(v_j)|$

Compressed Sparse Row format

Pour un graphe G , si $\text{Sp}(G)$ est la matrice d'adjacence (creuse) de G , alors ce format correspond au format CSR de $\text{Sp}(G)$.



Partitionnement



Partitionnement automatique du réseau par le DDS :

- partitions équilibrées
- limitation des communications

travaux en cours avec Rob Bisseling

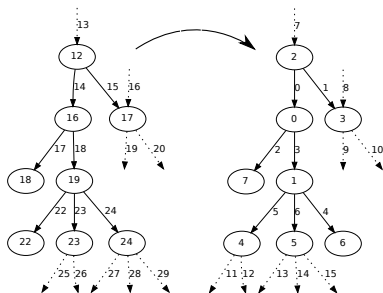
Mécanisme de ré-indexation

Arêtes ne nécessitant pas de comm	Arêtes nécessitant des comm	Arêtes issues d'autres processeurs

Arêtes locales

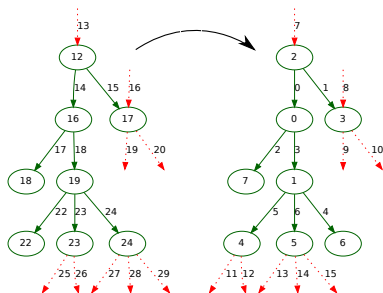
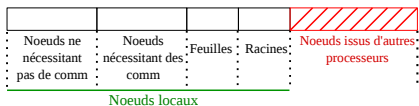
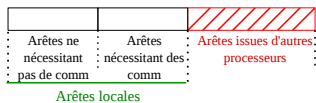
Noeuds ne nécessitant pas de comm	Noeuds nécessitant des comm	Feuilles	Racines	Noeuds issus d'autres processeurs

Noeuds locaux



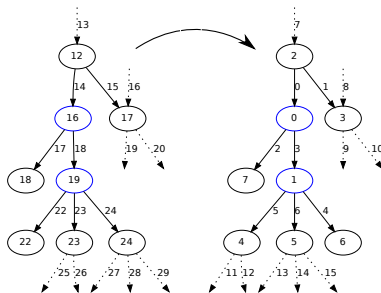
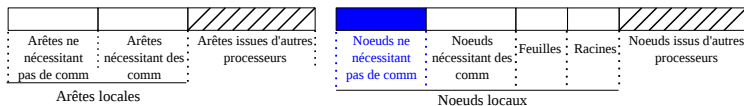
- moins de conditions coûteuses
- meilleure utilisation de la mémoire cache

Mécanisme de ré-indexation



- moins de conditions coûteuses
- meilleure utilisation de la mémoire cache

Mécanisme de ré-indexation



- moins de conditions coûteuses
- meilleure utilisation de la mémoire cache

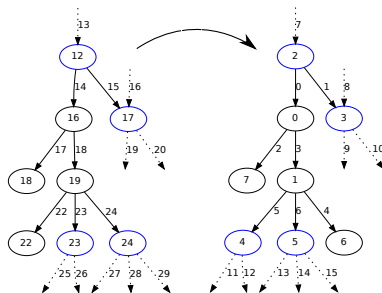
Mécanisme de ré-indexation

Arêtes ne nécessitant pas de comm	Arêtes nécessitant des comm	Arêtes issues d'autres processeurs
-----------------------------------	-----------------------------	------------------------------------

Arêtes locales

Noeuds ne nécessitant pas de comm	Noeuds nécessitant des comm	Feuilles	Racines	Noeuds issus d'autres processeurs
-----------------------------------	-----------------------------	----------	---------	-----------------------------------

Noeuds locaux



- moins de conditions coûteuses
- meilleure utilisation de la mémoire cache

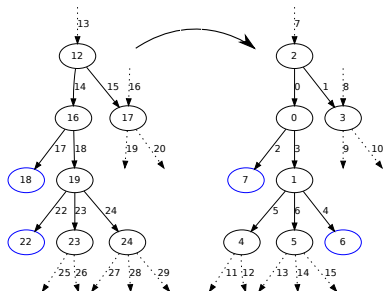
Mécanisme de ré-indexation

Arêtes ne nécessitant pas de comm	Arêtes nécessitant des comm	Arêtes issues d'autres processeurs
-----------------------------------	-----------------------------	------------------------------------

Arêtes locales

Noeuds ne nécessitant pas de comm	Noeuds nécessitant des comm	Feuilles	Racines	Noeuds issus d'autres processeurs
-----------------------------------	-----------------------------	----------	---------	-----------------------------------

Noeuds locaux



- moins de conditions coûteuses
- meilleure utilisation de la mémoire cache

Optimisations des conditions

Dans une simulation contenant t **itérations de temps** et n **éléments de maillage** à traiter :

- Certaines conditions du code utilisateur sont exécutées $t \times n$ fois
 - ▶ Il est difficile d'agir sur ces conditions
 - ▶ Nous évitons toutefois les conditions dues aux **bordures physiques** grâce aux **DDS et itérateurs**
- Certaines conditions dans SkelGIS sont exécutées $t \times n$ fois
 - ▶ **Il faut agir sur ces conditions**
 - ▶ Nous évitons les conditions dues au **recouvrement** grâce aux **DDS et itérateurs**
 - ▶ D'autres conditions sont associées à des **paramètres** de l'objet (DDS, DPMMap etc.)

↪ Il faut spécifier le code des objets suivant certains paramètres !

Spécialisation partielle de template

SkelGIS

```
template<class T,int Or,bool  
box=false> struct DMatrix  
{...};
```

```
template<class T> struct  
DMatrix<T,0>  
{...};
```

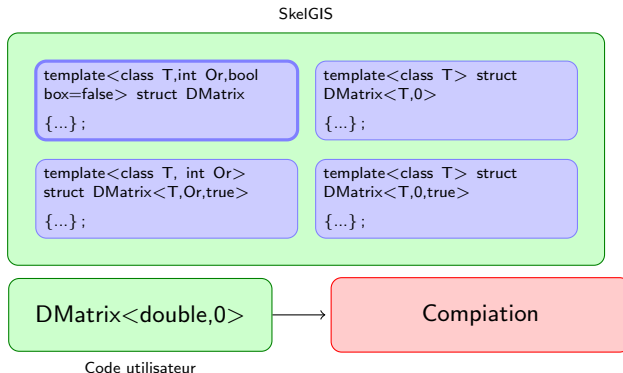
```
template<class T, int Or>  
struct DMatrix<T,Or,true>  
{...};
```

```
template<class T> struct  
DMatrix<T,0,true>  
{...};
```

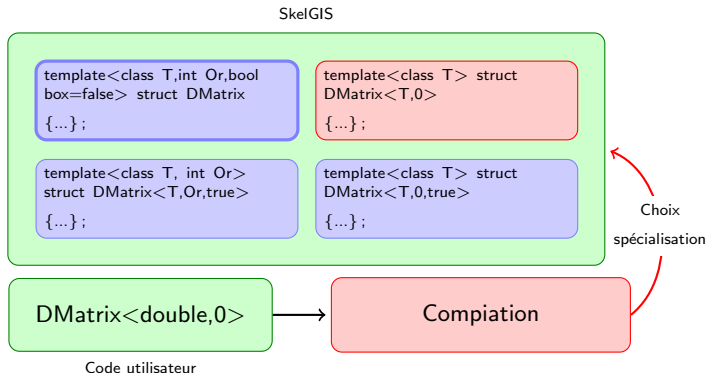
```
DMatrix<double,0>
```

Code utilisateur

Spécialisation partielle de template



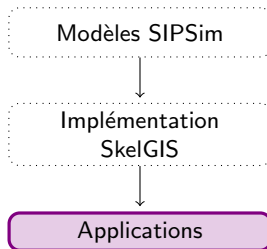
Spécialisation partielle de template



Applications et résultats



Multi-scale heterogeneous bulk-synchronous
Gaetan Hains



FullSWOF2D ré-écrit en SkelGIS

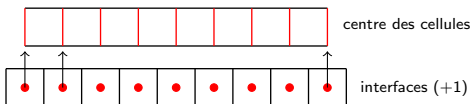
FullSWOF2D

Logiciel de simulation de l'écoulement de l'eau. Développé au *MAPMO* (Université d'Orléans)

Résolution des équations de Saint Venant (Navier-stokes 2D) par la méthode numérique des volumes finis sur un **maillage cartésien à deux dimensions**.

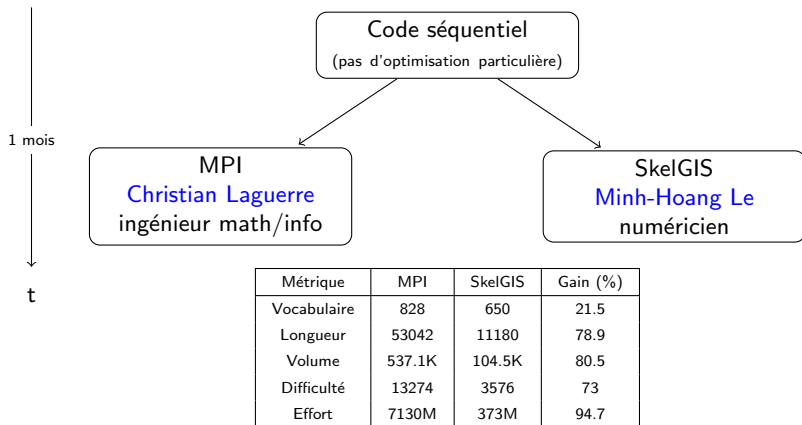
Calculs stencil complexes : **flexibilité du modèle mise à l'épreuve**

- intervention de plusieurs données : centre et interfaces
- première et dernière lignes traitées différemment
- Premier et dernier éléments d'une ligne traités différemment
- Traitement spécifique pour les bordures physiques



Résultats

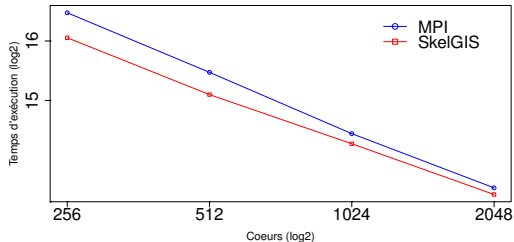
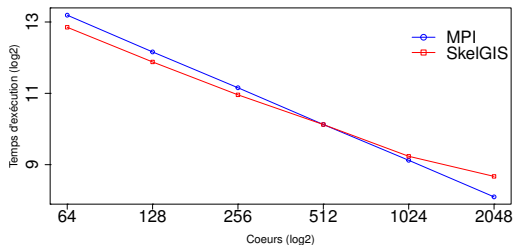
Projet CEMRACS 2012 FullSWOF-Paral



Maurice H. Halstead - Elsevier Science 1977

Elements of Software Science (Operating and programming systems series)

Résultats

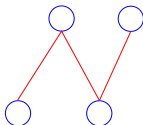


Helene Coullon, Minh-Hoang Le, Sébastien Limet - ICCS 2013

Parallelization of Shallow-Water Equations with the Algorithmic Skeleton Library SkelGIS

Simulation d'un réseau artériel

Simulation d'écoulement du sang dans un réseau artériel. Développée à l'UPMC Paris 6.

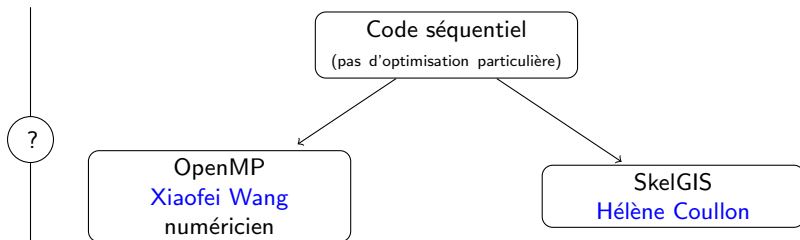


- équations Navier-Stokes 1D, volumes finis
- loi de conservation et de re-distribution entre les quantités entrantes et sortantes

Calculs relativement simples mais sur une structure complexe

Spécialisation du modèle mise à l'épreuve

Résultats



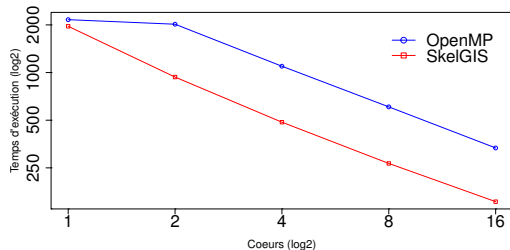
Métriques	OpenMP	SkelGIS	Gain %
Vocabulaire	866	513	40.7
Longueur	13344	10116	24.2
Volume	130213.73	91072.48	30
Difficulté	3911.18	3040.68	22.2
Effort	509.3M	276.9M	45.6



Maurice H. Halstead - Elsevier Science 1977

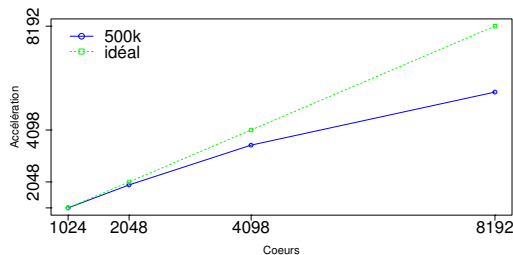
Elements of Software Science (Operating and programming systems series)

Résultats



1 nœud "thin" du
TGCC-Curie

2× Sandy Bridge (2.7GHz)
64 Go par nœuds
16 cœurs par nœud



réseau de 500k
nœuds et 500k arêtes

nœuds Blue Gene/Q Juqueen
IBM Power PC (1.6GHz)
16Go par nœud
16 cœurs par nœud



Helene Coullon, Jose-Maria Fullana, Pierre-Yves Lagrée, Sébastien Limet, Xiaofei Wang - ICCS 2014

Blood Flow Arterial Network Simulation with the Implicit Parallelism Library *SkelGIS*

Conclusion et perspectives

Conclusion

Parallélisme implicite pour
les simulations numériques

Parallélisme de données

Modèles SIPSim

Implémentation
SkelGIS

Applications

Conclusion

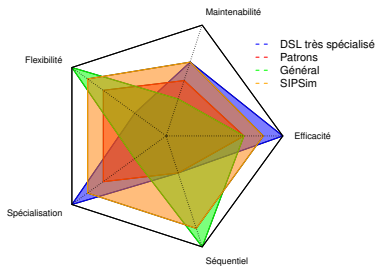
Parallélisme implicite pour
les simulations numériques

Parallélisme de données

Modèles SIPSim

Implémentation
SkelGIS

Applications



Conclusion

Parallélisme implicite pour les simulations numériques

Parallélisme de données

Modèles SIPSim

Implémentation
SkelGIS

Applications

Cinq concepts SIPSim :

- Structure de données distribuée
- Application de propriétés distribuées
- Appicateurs et opérations
- Interfaces pour écrire les opérations
 - ▶ itérateurs
 - ▶ accesseurs aux données
 - ▶ accesseurs aux données voisines

Conclusion

Parallélisme implicite pour les simulations numériques

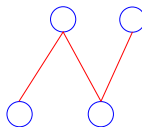
Parallélisme de données

Modèles SIPSim

Implémentation
SkelGIS

Applications

Modèle applicable aux réseaux



Conclusion

Parallélisme implicite pour les simulations numériques

Parallélisme de données

Modèles SIPSim

Implémentation
SkelGIS

Applications

Bibliothèque C++ orientée template

- maillages cartésiens 2D
- réseaux de type DAG et contenant des maillages 1D

Implémentation pour les réseaux :

- format CSR modifié
- mécanisme de ré-indexation
- optimisations de conditions

Conclusion

Parallélisme implicite pour
les simulations numériques

Parallélisme de données

Modèles SIPSim

Implémentation
SkelGIS

Applications



Helene Coullon, Sébastien Limet - Euro-Par 2014
*Implementation and Performance Analysis of SkelGIS
for Network Mesh-based Simulations*



Helene Coullon, Sébastien Limet - HPCS 2013
*Algorithmic skeleton library for scientific
simulations : SkelGIS*

Conclusion

Parallélisme implicite pour les simulations numériques

Parallélisme de données

Modèles SIPSim

Implémentation
SkelGIS

Applications

Cas réels et complexes

- FullSWOF SkelGIS

- ▶ comparaison MPI et SkelGIS en effort de programmation
- ▶ comparaison MPI et SkelGIS en performances

- simulation artérielle

- ▶ comparaison OpenMP et SkelGIS en effort de programmation
- ▶ performances de SkelGIS

Expériences sur : flexibilité, spécialisation, séquentiel (effort de programmation) et efficacité du programme obtenu.

Conclusion

Parallélisme implicite pour les simulations numériques

Parallélisme de données

Modèles SIPSim

Implémentation
SkelGIS

Applications



Helene Coullon, Jose-Maria Fullana, Pierre-Yves Lagrée, Sébastien Limet, Xiaofei Wang - ICCS 2014
Blood Flow Arterial Network Simulation with the Implicit Parallelism Library SkelGIS



Helene Coullon, Minh-Hoang Le, Sébastien Limet - ICCS 2013
Parallelization of Shallow-Water Equations with the Algorithmic Skeleton Library SkelGIS



Stéphane Cordier, Helene Coullon, Olivier Delestre, Christian Laguerre, Minh-Hoang Le, Daniel Pierre, Georges Sadaka - ESAIM : Proceedings CEMRACS 2012
FullSWOF Paral : Comparison of two parallelization strategies (MPI and SKELGIS) on a software designed for hydrology applications



Morgan Abily, Helene Coullon, Florian Cordier, Olivier Delestre - Springer book "Advances in Hydroinformatics - SimHydro 2014"
Comparison and validation of two parallelization approaches of FullSWOF_2D software on real cases

Perspectives

Évolution du modèle SIPSim

Généralisation aux simulations réseaux et multi-physiques

- composition de DDS
- composition d'opérations
- définition avancée d'un voisinage

Autres maillages

- Maillages non-structurés (OP2, Liszt)
- Maillages adaptatifs

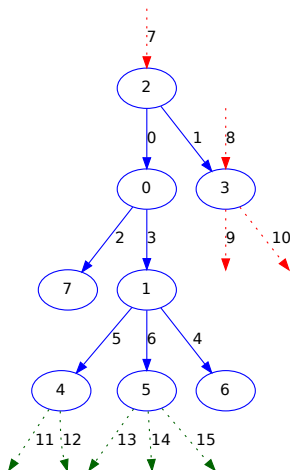
Implémentation du modèle

- aller vers l'implémentation d'un langage SIPSim (DSL)
- autres modèles d'exécution (mémoire partagée, GPGPU)

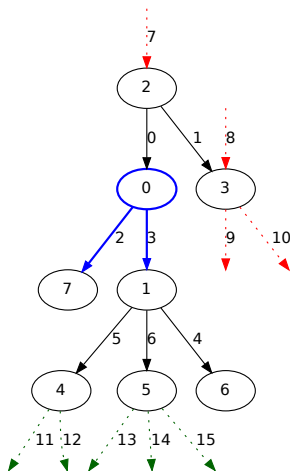
Merci de votre attention



Modification du format CSR



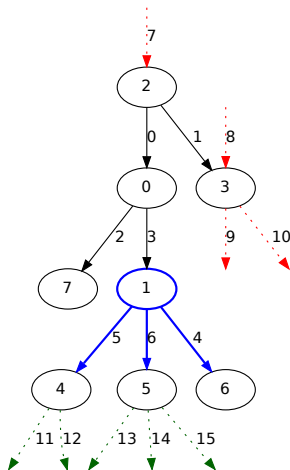
Modification du format CSR



$$cdeg^+ = [0, 2, \dots],$$

$$N_E^+ = [2, 3, \dots],$$

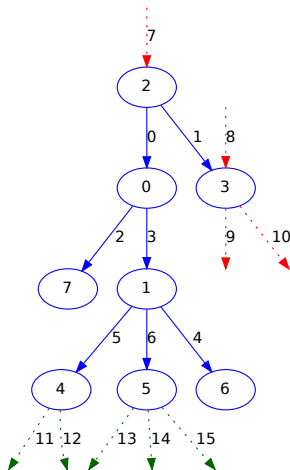
Modification du format CSR



$$cdeg^+ = [0, 2, \mathbf{5}, \dots],$$

$$N_E^+ = [2, 3, \mathbf{5}, \mathbf{6}, 4, \dots],$$

Modification du format CSR



$$cdeg^+ = [0, 2, 5, 7, 9, 11, 14, 14, 14],$$

$$cdeg^- = [0, 1, 2, 3, 5, 6, 7, 8, 9],$$

$$N_E^+ =$$

$$[2, 3, 5, 6, 4, 0, 1, 9, 10, 11, 12, 13, 14, 15],$$

$$N_E^- = [0, 3, 7, 1, 8, 5, 6, 4, 2],$$

$$S = [2, 2, 0, 0, 1, 1, 1],$$

$$D = [0, 3, 7, 1, 6, 4, 5]$$

Coût du DDAG

Tableau	Taille
$cdeg^+$	$ V_i $
$cdeg^-$	$ V_i $
S	$ E_i $
D	$ E_i $
$N_E^+ + N_E^-$	$\sum_{j=0}^{ V_i } deg(v_j)$
$cdeg^{tor}$	p
$cdeg^{tos}$	p
N_E^{tor}	$cdeg^{tor}[p]$
N_V^{tor}	$cdeg^{tor}[p]$
N_E^{tos}	$cdeg^{tos}[p]$
N_V^{tos}	$cdeg^{tos}[p]$

La taille totale représentée par cette structure de donnée, pour chaque processeur, est donc

$$Taille = L + P + Conn + Comm, \quad (1)$$

avec

$$L = 2 \times |V_i| + 2 \times |E_i|,$$

$$P = 2 \times p,$$

$$Conn = \sum_{j=0}^{|V_i|} deg(v_j),$$

$$Comm = 2 \times cdeg^{tor}[p] + 2 \times cdeg^{tos}[p].$$

Partitionnement

Implémentation actuelle

Méthode par regroupement d'arêtes soeurs

- résultats satisfaisants sur le cas de test étudié
- ne résoud pas réellement le problème de partitionnement des réseaux
- se limite aux graphes acycliques

Méthodes en cours d'étude

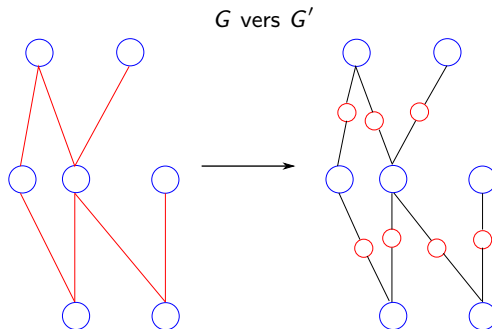
Utilisation du partitionneur **Mondriaan**

- **Méthode à partitionnement unique** (en cours)
- **Méthode à partitionnement double** (en cours)

Formulation du problème de partitionnement

- Calculs différents effectués sur les nœuds et les arêtes
↔ deux contraintes d'équilibrage de charge
- Calcul contenant plusieurs phases
Si $T1$ et $T2$ correspondent aux arêtes et aux nœuds ou vice versa
 - ① Communication des $T1$ calculés à t
 - ② Calcul des $T2$ à $t + 1$
 - ③ Communication des $T2$ calculés à $t + 1$
 - ④ Calcul des $T1$ à $t + 1$

Transformation du réseau



Si V^r est l'ensemble des sommets **rouges**, V^b l'ensemble des sommets **bleus** de G' :

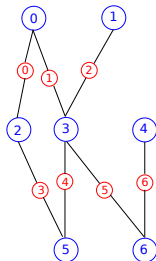
$$\max_i \omega(V_i^r) \leq (1 + \epsilon) \frac{\omega(V^r)}{p}, \quad \epsilon > 0$$

$$\max_i \omega(V_i^b) \leq (1 + \epsilon) \frac{\omega(V^b)}{p}, \quad \epsilon > 0$$

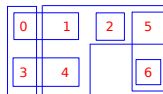
Méthode à partitionnement simple

- ① L'étape de communication des sommets bleus vers les sommets rouges est exprimée comme un problème de partitionnement d'hypergraphe en 1D, ce qui permet de distribuer les sommets rouges.
- ② Une heuristique est appliquée pour distribuer les sommets bleus, en prenant en compte la distribution des sommets rouges.

1. Distribution des sommets rouges



	0	1	2	3	4	5	6
0	1	1	0	0	0	0	0
1	0	0	1	0	0	0	0
2	1	0	0	1	0	0	0
3	0	1	1	0	1	1	0
4	0	0	0	0	0	0	1
5	0	0	0	1	1	0	0
6	0	0	0	0	0	1	1

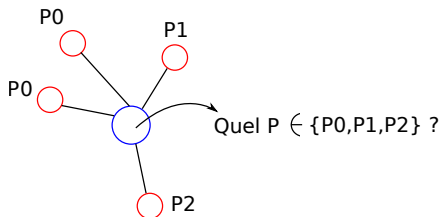


Utilisation du partitionneur Mondriann pour le partitionnement “1D row-net”

$$\max_i \omega(V_i^r) \leq (1 + \epsilon) \frac{\omega(V^r)}{p}, \quad \epsilon > 0$$

↪ Cette étape va essayer de minimiser les communications des sommets **bleus** vers les sommets **rouges**.

2. Distribution des sommets bleus



L'heuristique choisit $P(v) = P_i$ tel que $\deg(v, P_i)$ est maximal et tel que

$$\max_i \omega(V_i^b) \leq (1 + \epsilon) \frac{\omega(V^b)}{p}, \quad \epsilon > 0$$

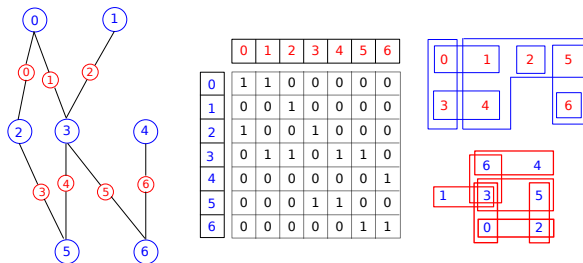
Cas d'égalité : si $\exists i, j$ tels que $\deg(v, P_i) = \deg(v, P_j)$, $P(v) = P_i$ si $\omega(V_i^b) < \omega(V_j^b)$

↪ Cette étape va essayer de minimiser les communications des sommets **rouges** vers les sommets **bleus**.

Méthode à partitionnement double

- ① L'étape de communication des sommets bleus vers les sommets rouges est exprimée comme un problème de partitionnement d'hypergraphe en 1D, ce qui permet de distribuer les sommets rouges.
- ② L'étape de communication des sommets rouges vers les sommets bleus est exprimée comme un problème de partitionnement d'hypergraphe en 1D, ce qui permet de distribuer les sommets bleus.
- ③ Un problème de permutation est résolu pour ré-organiser les deux distributions.

1. and 2. Distribution des sommets rouges et bleus



Utilisation du partitionneur Mondriann pour le partitionnement “1D row-net” et “1D column-net”

$$\max_i \omega(V_i^r) \leq (1 + \epsilon) \frac{\omega(V^r)}{p}, \quad \epsilon > 0$$

$$\max_i \omega(V_i^b) \leq (1 + \epsilon) \frac{\omega(V^b)}{p}, \quad \epsilon > 0$$

3. Petit problème de matching

Permutation des processeurs attribués dans chaque partitionnement de façon optimale

W est une matrice de taille $p \times p$ calculée à partir de la matrice A :

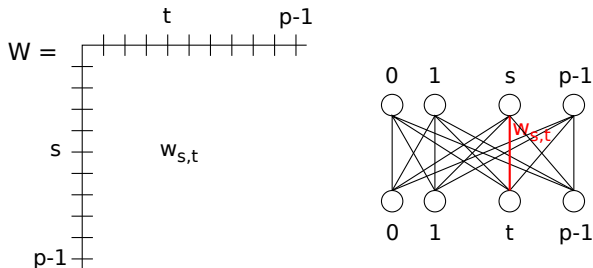
- Si un sommet bleu i (ligne de A) est lié à un sommet rouge j (colonne de A), on a $a_{i,j} = 1$
- $\Phi(i)$ représente le processeur attribué au sommet bleu i
- $\Psi(j)$ représente le processeur attribué au sommet rouge j

$$w_{s,t} = \sum_{i : \Phi(i)=s} \delta_i(t) \quad (2)$$

$$\delta_i(t) = \begin{cases} 1 & \text{if } \exists j : a_{i,j} = 1 \wedge \Psi(j) = t \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

↪ le nombre de communication du processeur s au processeur t pour les communications des sommets bleus vers rouges

3. Petit problème de matching



- Calcul de la matrice W' de la même façon, en utilisant A^T
- Résolution du problème de matching maximal sur $\overline{W} = W + W'^T$
- Algorithme de Hongrois $O(p^4)$

Métriques de Halstead

Symbole	Mesure
N_1	Nombre total d'opérateurs
N_2	Nombre total d'opérandes
η_1	Nombre d'opérateurs distincts
η_2	Nombre d'opérandes distincts

Symbole	Valeur	Métrique
η	$\eta_1 + \eta_2$	Vocabulaire
N	$N_1 + N_2$	Longueur
V	$N \times \log_2 \eta$	Volume
D	$\frac{\eta_1}{2} \times \frac{N_2}{\eta_2}$	Difficulté
E	$D \times V$	Effort

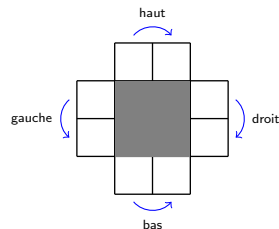
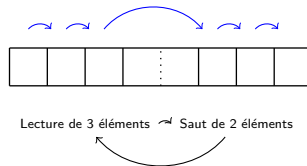
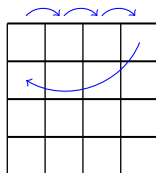
SkelGIS pour les maillages cartésiens à deux dimensions

Fusion DDS et DPmap

- Structure de données légère
- Simplicité d'utilisation avec un concept en moins

Itérateurs

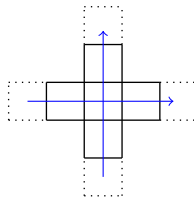
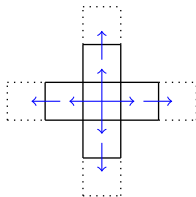
Standard / ligne / saut régulier / bordure physique 4 côtés



SkelGIS pour les maillages cartésiens à deux dimensions

Accès aux voisinages

- accès directionnel
- accès horizontal et vertical
- accès à toutes les valeurs voisines



Exemple de code : l'équation de la chaleur

```

#include "skelgis/skelgis.hpp"
using namespace skelgis;

int main(int argc, char** argv)
{
    INITSKELGIS;
    HEADER head;
    head.x=0;head.y=0;
    head.width=100;head.height=100;
    head.spacing=1;head.nodata=-9999;

    DMatrix<double,1> m(head,0);
    m.setGlobalMiddleValue(1);
    DMatrix<double,1> m2(head,0);
    for(int i=0;i<100;i++)
    {
        ApplyUnary<double,1,double,1>::apply(laplacien,m,m2);
        DMatrix<double,1> m3(m);
        m = m2;
        m2 = m3;
    }
    ENDSKELGIS;
}

```

Exemple de code : l'équation de la chaleur

```
BEGINApplyUnary(laplacien ,m, double ,1 ,m2, double ,1)
{
  double a = 0.05;
  iterator<double,1> it = m.begin();
  iterator<double,1> itEnd = m.end();
  for(it; it<itEnd; ++it)
  {
    double val = m[it];
    double res = (1-4*a)*val+a*(m.getRight(it) + m.getLeft(it)+
m.getUp(it) + m.getDown(it));
    m2[it] = res;
  }
END(laplacien)
```