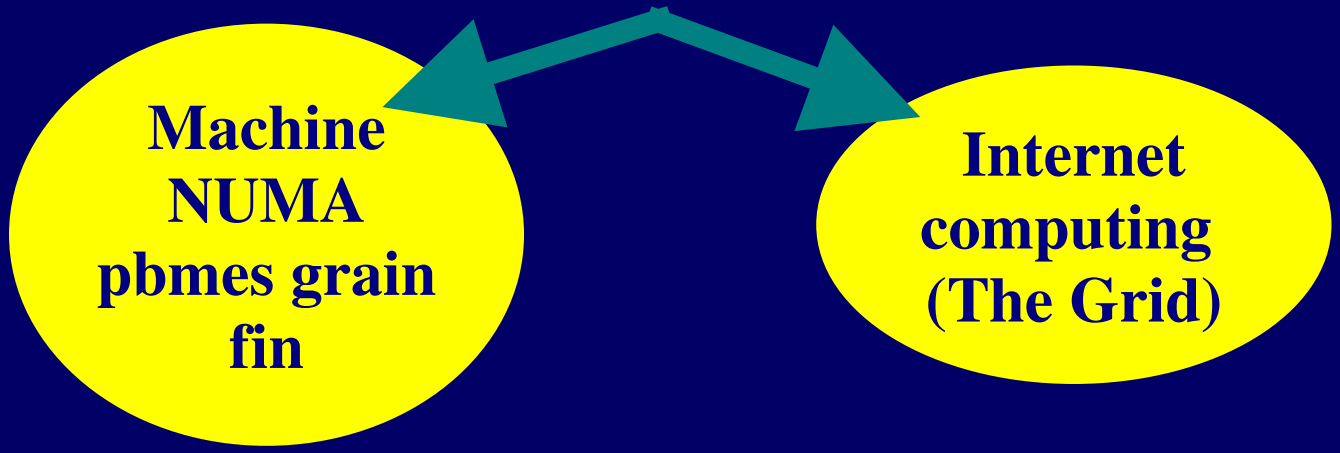
Cours 3. Algorithmes Parallèles avec prise en compte des Communications



www-id.imag.fr/Laboratoire/Membres/Roch_Jean-Louis/perso.html/enseignement.html/

Cours 3. Algorithmes Parallèles avec prise en compte des Communications

I. **Modèlisation des communications**

-LogP -BSP -Délai

II. Exemples d'ordonnancements avec com.

-Glouton: ETF - Placement de données

III. Contrôle algorithmique de granularité

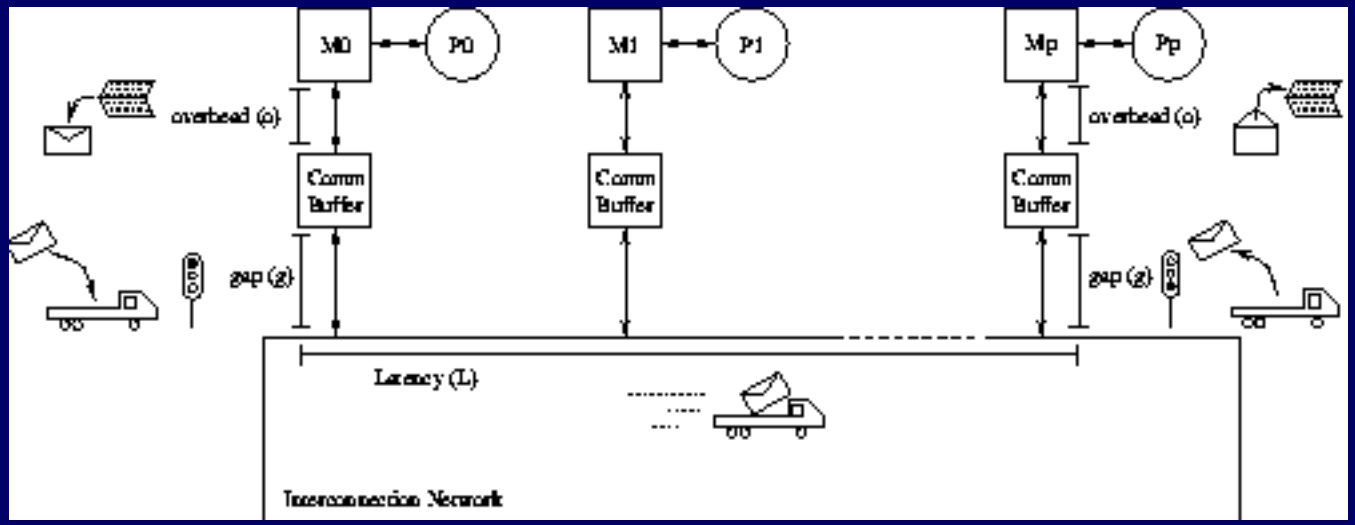
-Tri -Jeu de la vie -FFT

IV. Exemple de synthèse : produit de matrices

-Systolique -Distributions bloc -Minimisation com.

V. Application : HPL sur grappe

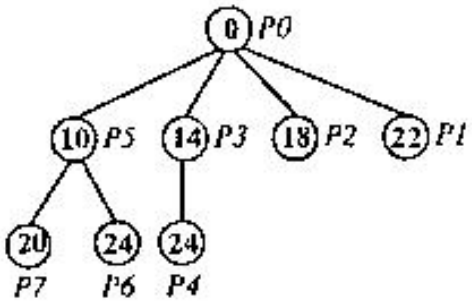
Modélisation des coms : LogP



L : Latence = délai de traversée du réseau
o: overhead = surcoût emballage/déballage
g : gap = attente dûe à la contention

Diffusion sur LogP

Diffusion optimale avec les paramètres L=6, o=2, g=4, P=3



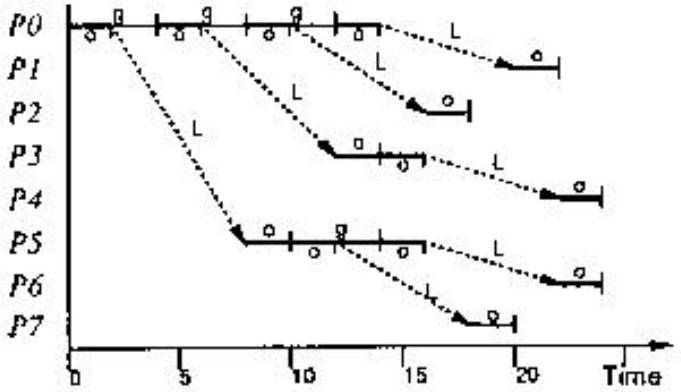


Figure 3: *Optimal broadcast tree for $P = 8, L = 6, g = 4, o = 2$ (left) and the activity of each processor over time (right). The number shown for each node is the time at which it has received the datum and can begin sending it on. The last value is received at time 24.*

« LogP : Towards a realistic model of Parallel Computation » D. Culler & al.

Recouvrement communications

Multithreading :

recouvrement latence réseau (débit supposé « infini »)
pipelininig des accès distants

Surcoût : g, o et temps de changement de contexte

Principe :

q « threads » s'exécutant alternativement sur chacun des p processeurs

Ordonnancement à la volée sur les (p.q) threads

Exemples: machines data-flow [..., Tera Computer]

Recouvrement: modèles théoriques

- Hachage universel [Karp&al 96]
  - Tous les accès en mémoire sont distribués sur les processeurs par hachage uniforme
  - Masquage de la latence par recouvrement
  - Surcoût d'accès:
    - évaluation de la fonction de hachage
    - Contention
- BSP [Valliant&al 90 ...]
  - Programme parallèle = séquence de superpas
  - Un superpas = n blocs de calcul indépendants
  - Communication : seulement entre 2 superpas:
    - h-relation

Communications et coût

Mesures des communications sur le graphe bipartite qui décrit le flot de données [cours1]:

C_1 : volume des accès distants

C^*_∞ : volume maximal de com. sur un chemin

« *chemin critique en communications* »

Ordt glouton [Graham] / work-stealing avec com : modèle délai : coût com. unitaire = h(p,q):

$T_p < (T_1/p) + T_\infty$ **$+ h(p,q)(C_1/p) + C_\infty$** $+ \sigma$

Bilan : algo. C_1 petit / meilleur ordt

Cours 3. Algorithmes Parallèles avec prise en compte des Communications

I. Modèlisation des communications

-LogP -BSP -Délai

II. Exemples d'ordonnancements avec com.

- Glouton: ETF

- Placement de données

ETF : earliest task first

Principe glouton : affectation de la tâche qui pourra démarrer son exécution au plus tôt sur un processeur inactif.

Théorème : modèle délai - latence = h

$$T_p < (T_1 / p) + T_\infty + h.C^*_\infty$$

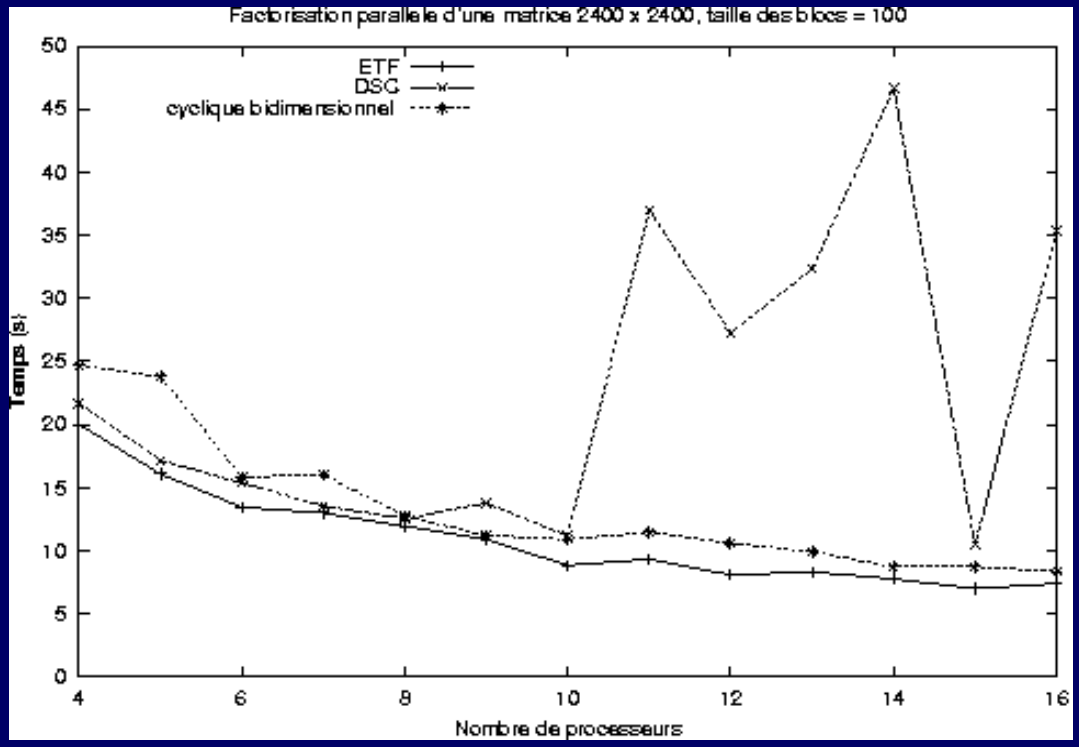
Surcoût de mise en oeuvre : O(p.n.*log* n)

Application : diffusion

ETF : earliest task first

Difficultés :

- Estimation du coût des tâches
- Non prise en compte des diffusions
- Estimation du coût des communications : instabilité



Alternative stable : placement de données

Placement de données

Tableau initial :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Distribution bloc :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Distribution cyclique

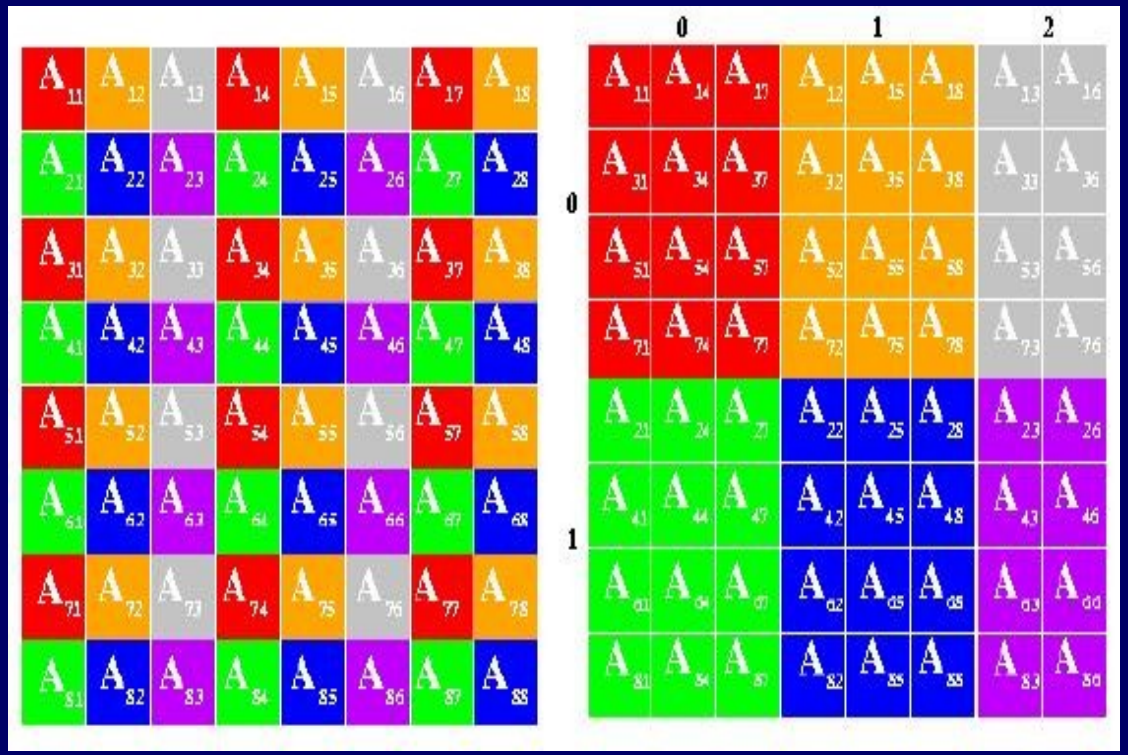
1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

Distribution bloc-cyclique / taille bloc = 2

1	2	9	10	3	4	11	12	5	6	13	14	7	8	15	16

Distribution bloc-cyclique bi-dimensionnelle

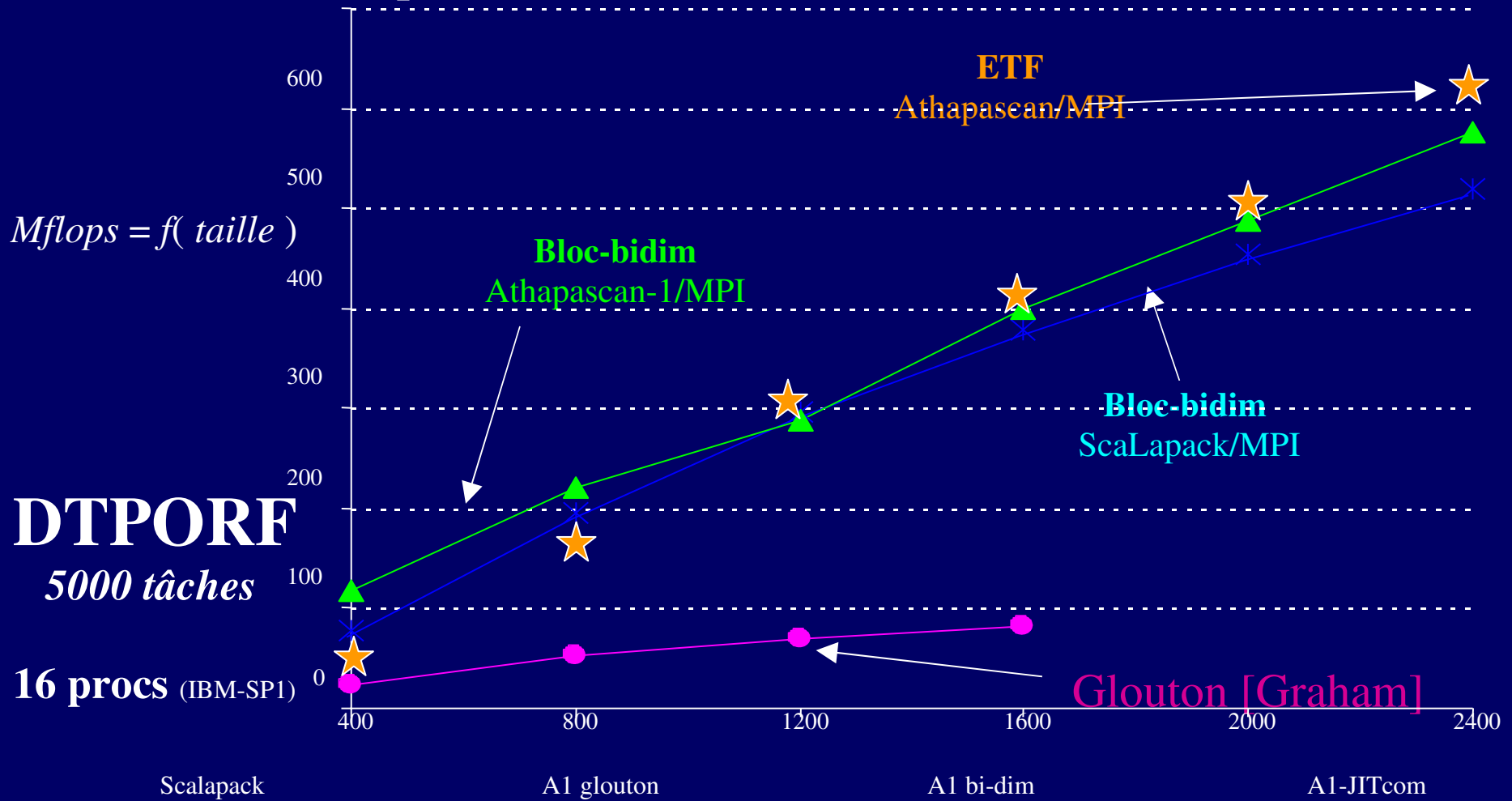
Sur 6 Processeurs, agencés en grille 2x3



Comparaison ETF / Bloc cyclique

Bloc-cyclique « stable » mais processeurs identiques, algo « simple »

ETF « automatique » mais : instabilité, modèle de coût de com.



Cours 3. Algorithmes Parallèles avec prise en compte des Communications

I. Modèlisation des communications

-LogP -BSP -Délai

II. Exemples d'ordonnancements avec com.

-Glouton: ETF - Placement de données

III. Contrôle algorithmique de granularité

-Tri

-Jeu de la vie

-FFT

Exemple 1 : Tri

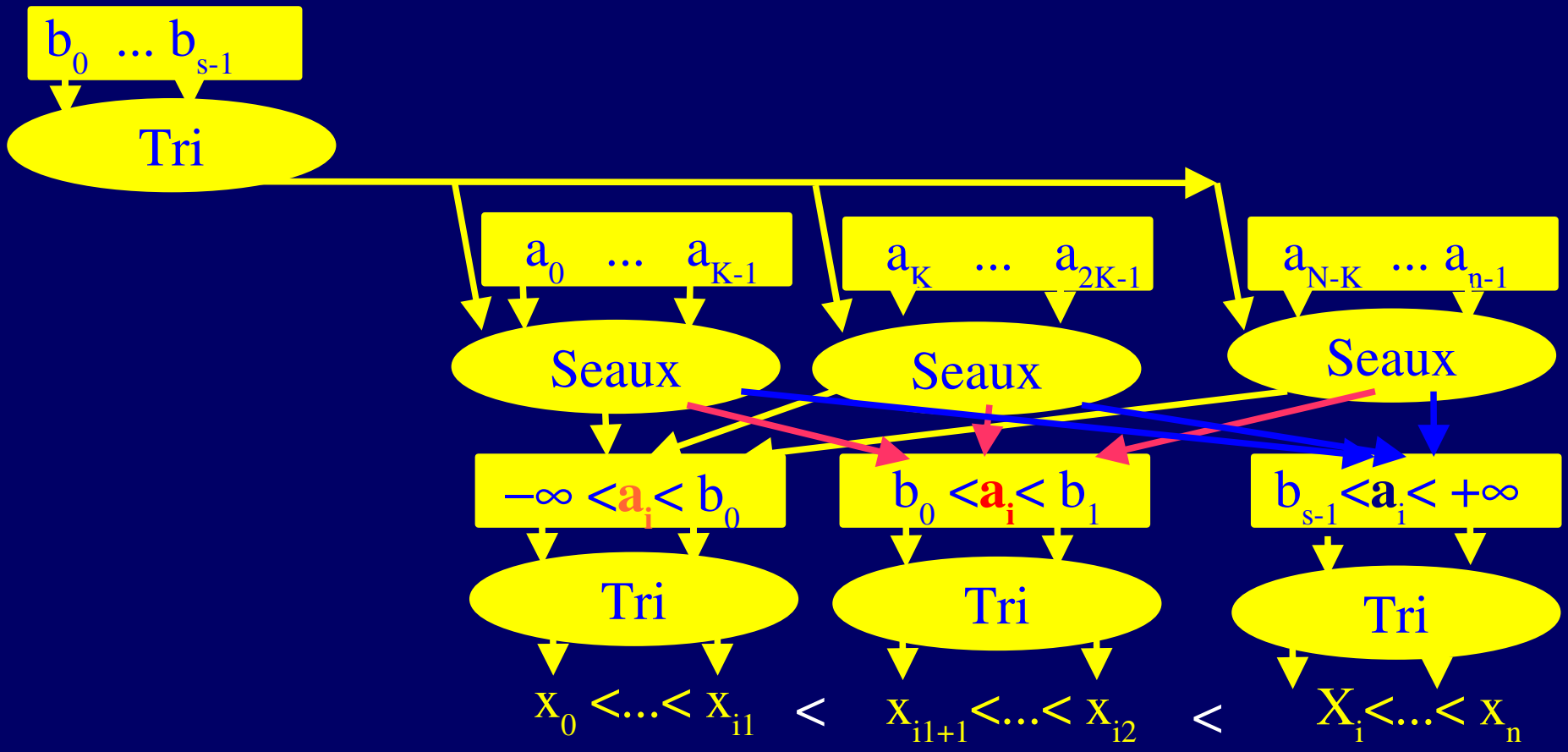
Algorithme séquentiel efficace : quick-sort

Parallélisation : OK mais communications...

...*Regrouper* pour réduire les communications

Tri par seaux

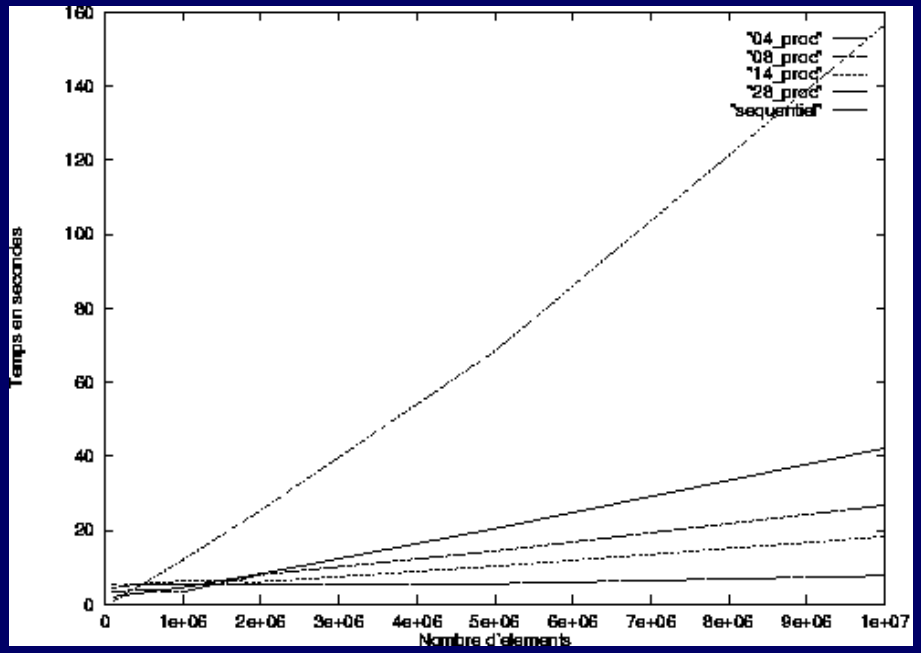
s « pivots » : $b_0 \ldots b_{s-1}$ tirés au hasard parmi les a_i



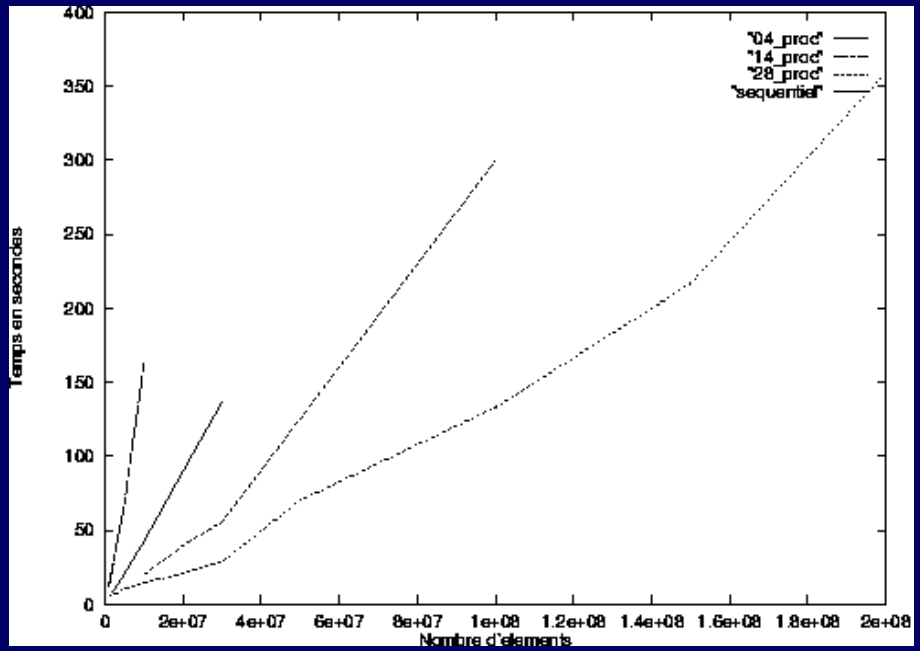
Exercice : Choix de s et K ? Ordonnancement ?

Expérimentations

Architecture : IBM SP1 - 1, 4, 8, 16, 28 Processeurs
Découpe : √n avec seuil, non récursive
Ordonnancement : pré-placement bloc et work-stealing sur inactivité



N petit



N grand

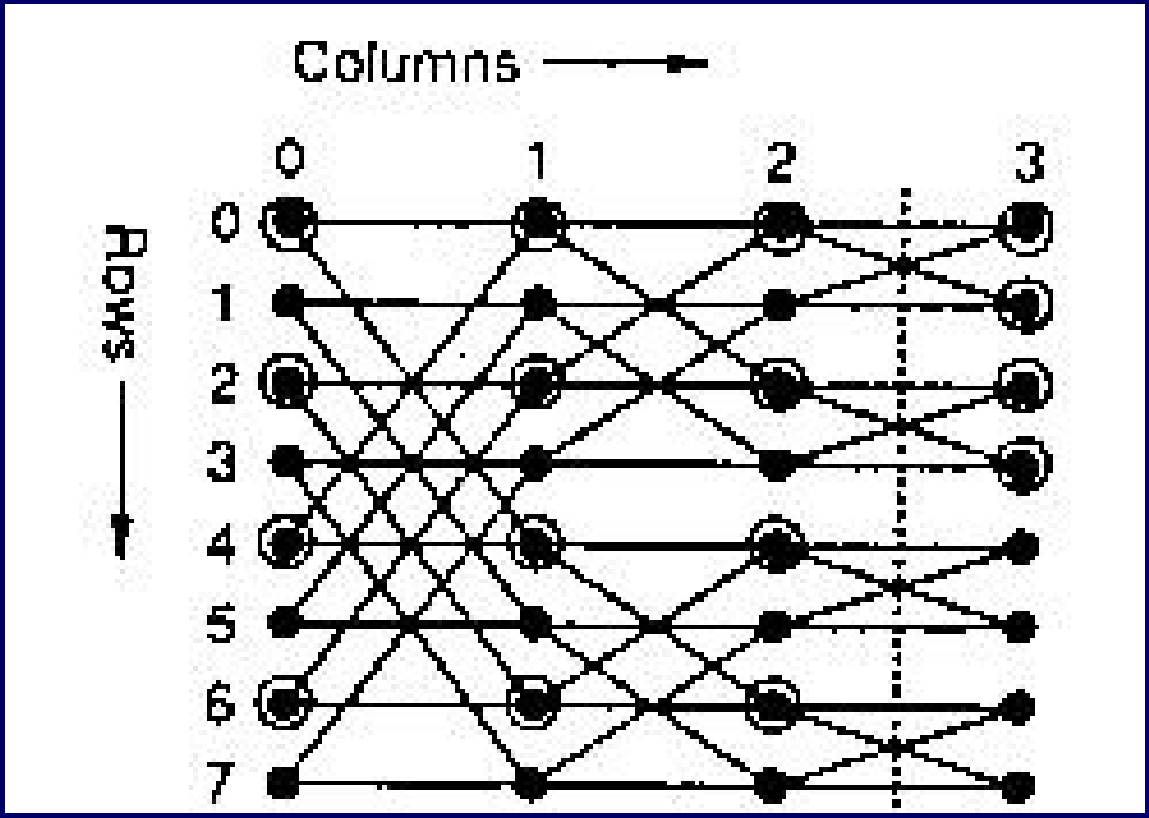
Exemple 2 : Jeu de la vie

En dimension 2

Application :
  Jacobi vs Gauss-Seidel/SOR vs Red-Black

Exercice : en dimension 3

Exemple 3 : FFT

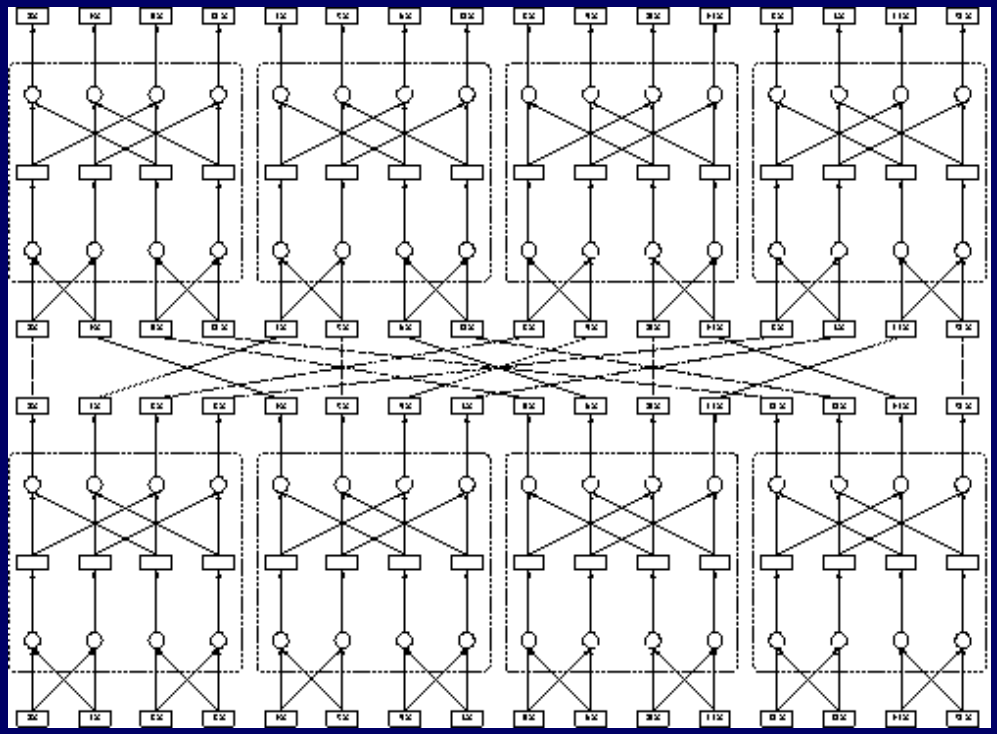


FFT : redistribution

$T_1 = \sqrt{n}$

$T_\infty = \sqrt{n}.\log n$

$C_1 = n$



FFT : performances

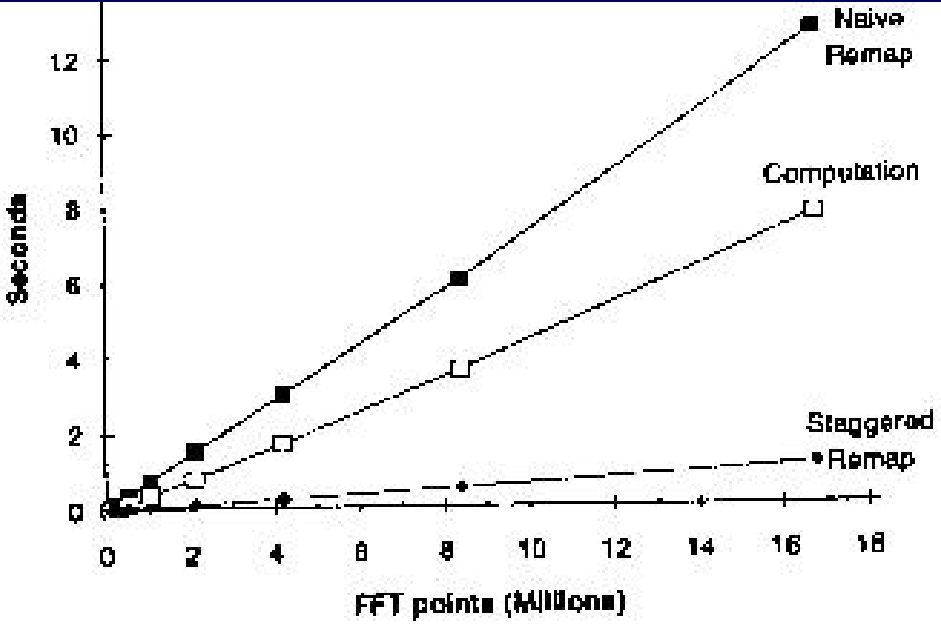


Figure 6: *Execution times for FFTs of various sizes on a 128 processor CM-5. The compute curve represents the time spent computing locally. The bad remap curve shows the time spent remapping the data from a cyclic layout to a blocked layout if a naive communication schedule is used. The good remap curve shows the time for the same remapping, but with a contention free communication schedule, which is an order of magnitude faster. The X axis scale refers to the entire FFT size.*

« LogP : Towards a realistic model of Parallel Computation » D. Culler & al.

Techniques introduites

Construire un algorithme en minimisant les communications :

exemple : tri

Regrouper les calculs parallèles pour diminuer les communications

exemples : jeu de la vie, FFT

Ordonnancements pour l'exécution :

au choix : ETF, placement, ...

Cours 3. Algorithmes Parallèles avec prise en compte des Communications

I. Modélisation des communications

-LogP -BSP -Délai

II. Exemples d'ordonnancements avec com.

-Glouton: ETF - Placement de données

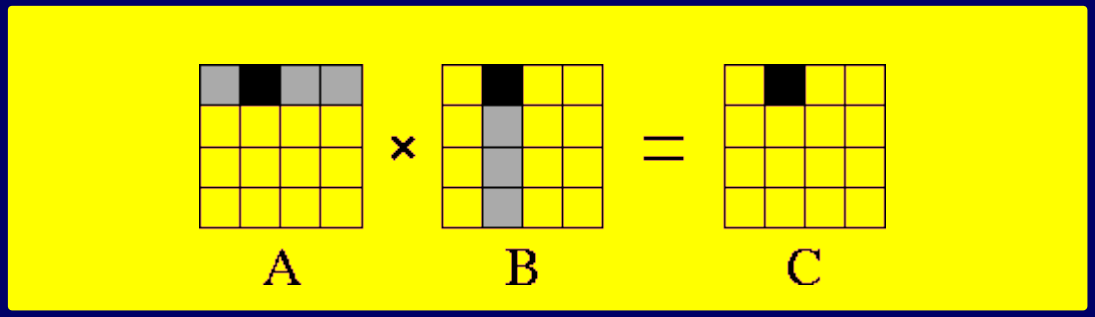
III. Contrôle algorithmique de granularité

-Tri -Jeu de la vie -FFT

IV. **Exemple de synthèse : produit de matrices**

-Minimisation com.
-Systolique
-Distributions bloc

Construction de l'algorithme



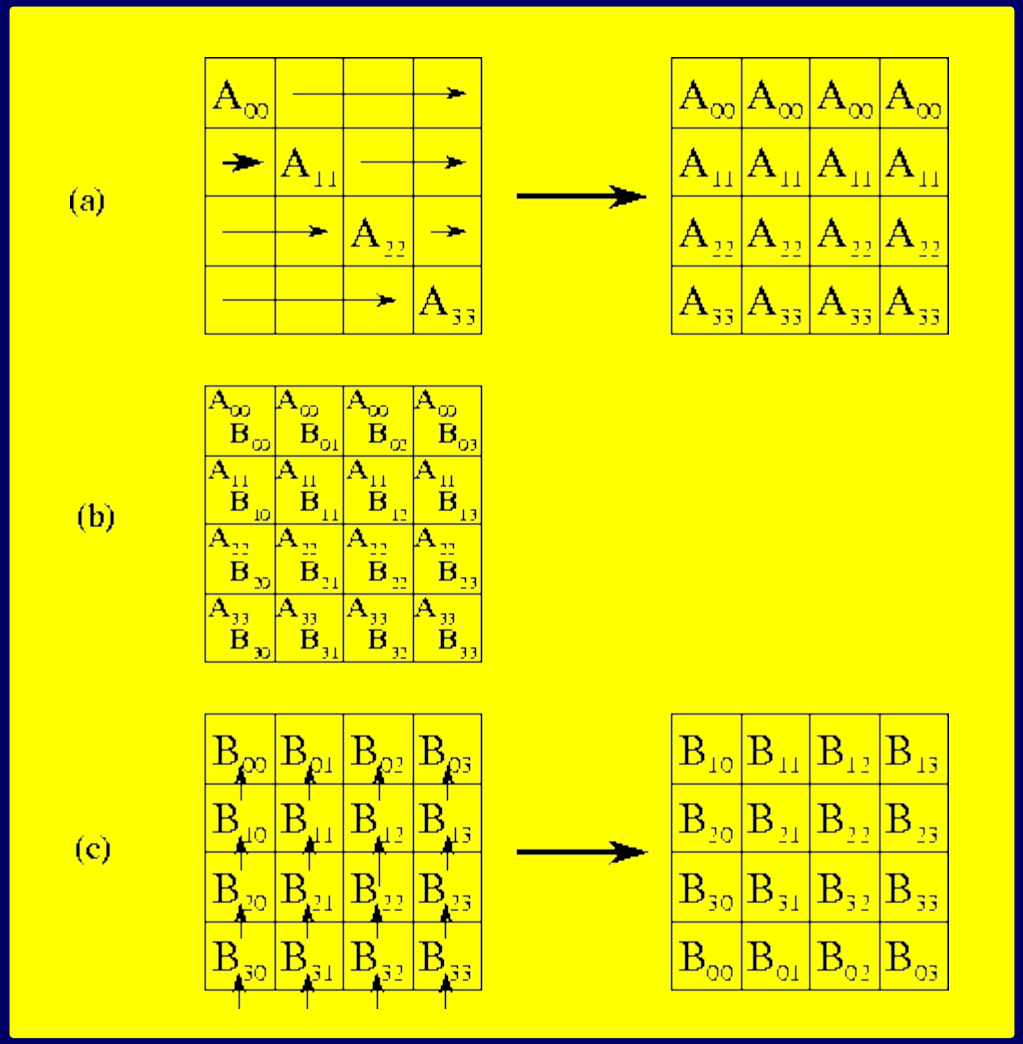
Algorithme parallèle grain fin

Adaptation de granularité :
limiter le surcôut d'ordonnancement

Regrouper calculs/données :
limiter les communications

Produit de matrices en bloc : √Px√P procs

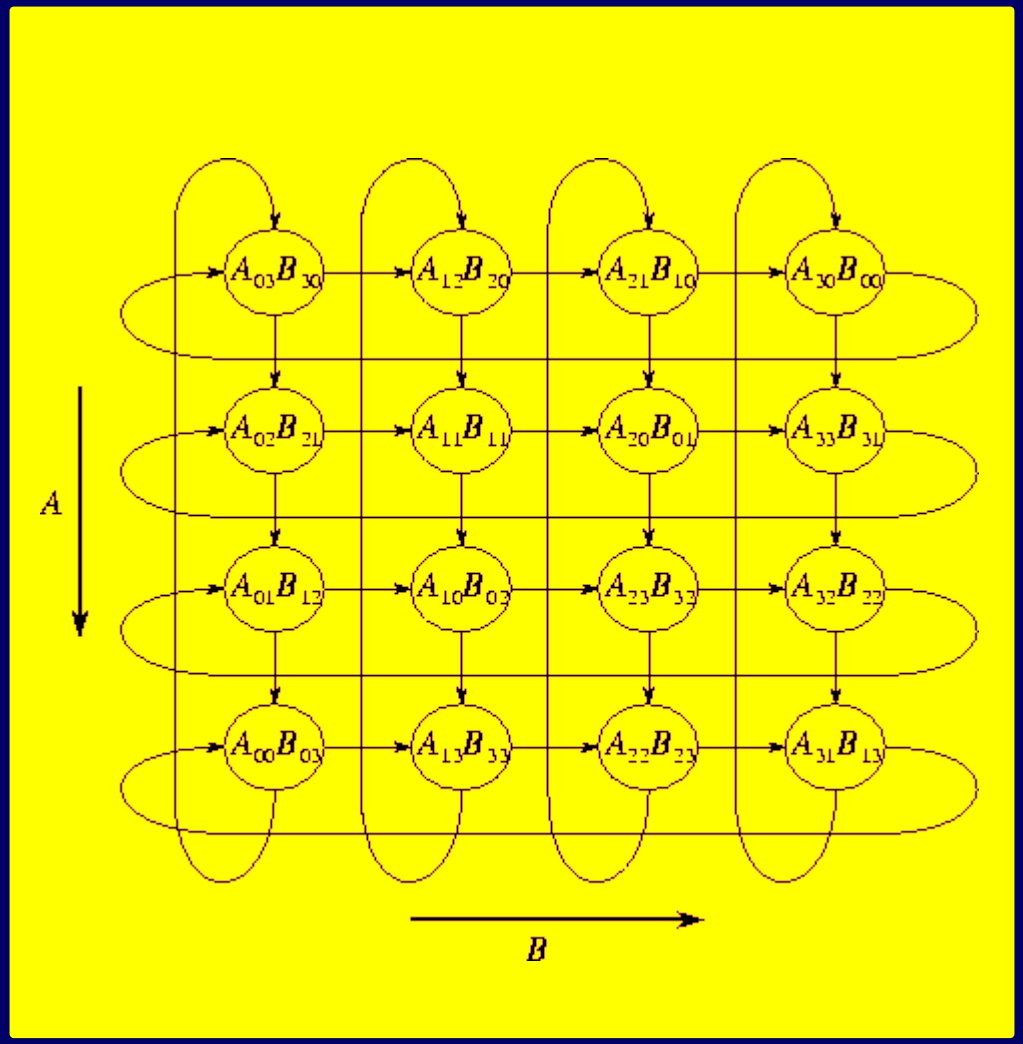
A : circulation en ligne



B : circulation en colonne

Produit de matrices en bloc : √Px√P procs

A : circulation
en ligne



B : circulation
en colonne

Cours 3. Algorithmes Parallèles avec prise en compte des Communications

I. Modèlisation des communications

-LogP -BSP -Délai

II. Exemples d'ordonnancements avec com.

-Glouton: ETF - Placement de données

III. Contrôle algorithmique de granularité

-Tri -Jeu de la vie -FFT

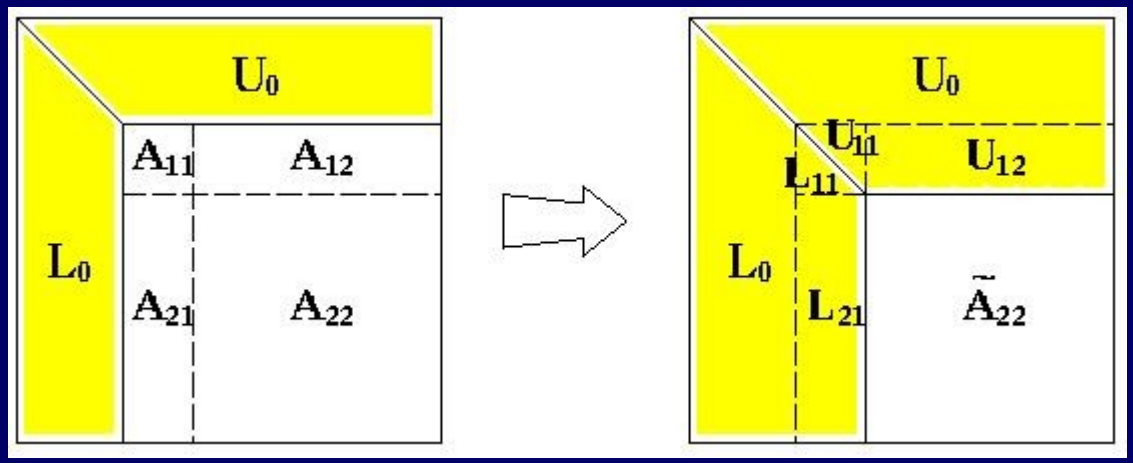
IV. Exemple de synthèse : produit de matrices

-Systolique -Distributions bloc -Minimisation com.

V. **Application : HPL sur grappe**

HPL : factorisation P.LU récursive

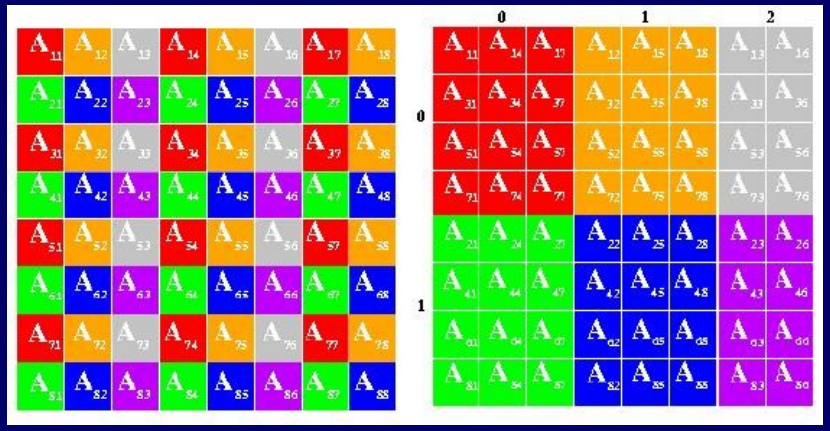
A =



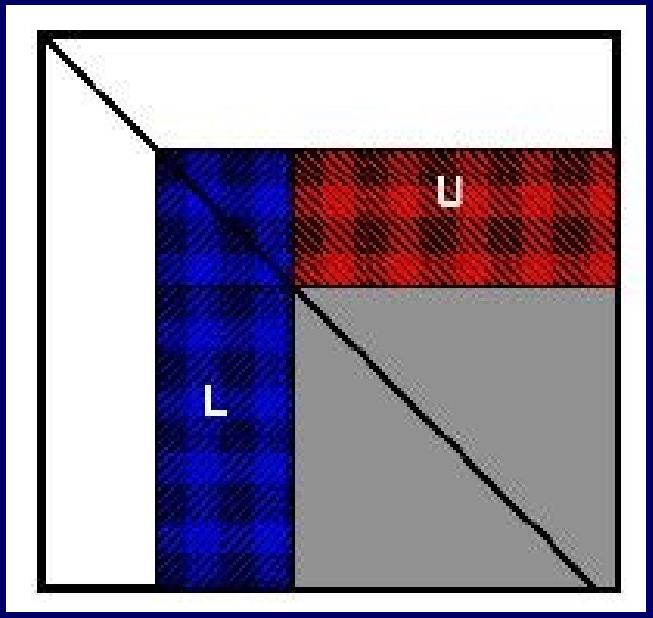
= P.L.U

$$A_{1,1} = L_{11}U_{11}$$
$$A_{21} = L_{21}U_{11}$$
$$A_{12} = L_{11}U_{12} \Leftrightarrow U_{12} = (L_{11})^{-1}A_{12}$$
$$A_{22} - L_{21}U_{12} = L_{22}U_{22}$$

Parallélisation: bloc-cyclique bi-dim



Mapping des blocs Nb x Nb
sur une grille virtuelle de
PxQ Pes
(blocs cycliques)

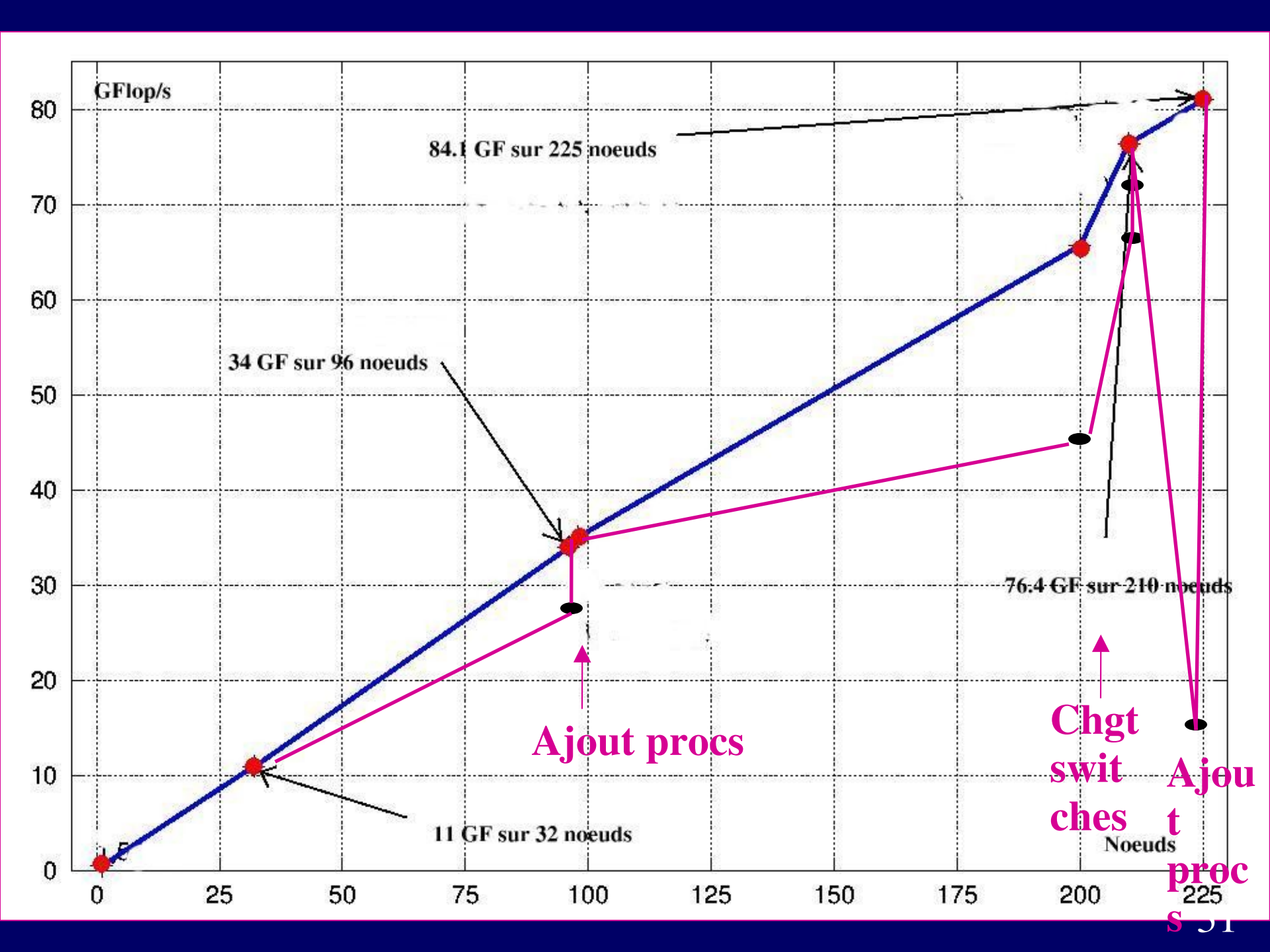


- Nb colonnes calculées sur une colonne de Pes
- Mise à jour sur une ligne

Programme MPI « portable » : HPL
5 Janvier 2001 :28.6 Gf sur 96 processeurs

Tuning : paramètres de HPL

- Algo de factorisation séquentielle (blas)
- N, Nb : taille de A et d'un bloc
- Profondeur de « pipe »
- P, Q : taille de la grille de PEs
- Algo de diffusion (BCAST)


GFlop/s
84.1 GF sur 225 noeuds
34 GF sur 96 noeuds
11 GF sur 32 noeuds
76.4 GF sur 210 noeuds
Ajout procs
Chgt swit ches
Ajou t proc s
Noeuds
0
10
20
30
40
50
60
70
80
0
25
50
75
100
125
150
175
200
225

Conclusion : portabilité sur grappes

Matériel standard : prix (réseau) + portabilité logicielle

mais :

- caractéristiques spécifiques
- Evolue : ajout de processeurs, chgt de switchs...
- Programme difficile à porter à une architecture hétérogène

Algorithme	<>	Programme
|		|
Gauss PLU par blocs		HPL : MPI + paramètres