Polytech Sorbonne - MAIN 4

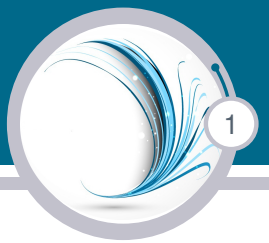
Projet HPC : le modèle shallow water

Parallélisation MPI

Fadwa Alozade Inès Benzenati

Lundi 9 avril 2018

Sommaire



Introduction
Le modèle shallow water

- Écoulement d'un fluide homogène sur la verticale
- Exemple : ondes à la surface de l'eau lorsque vous y jetez un caillou
- Modèle numérique

Introduction
Ce que fait le code séquentiel

- **parse_args :** Parser les arguments : x, y, t, export, ...
- **alloc :** Allouer de la mémoire : **hFil**, uFil, vFil, hPhy, uPhy, vPhy
- **gauss_init :** Initialiser l'image (**hFil**) au temps t=0
- **forward :** Remplir les images suivantes (et exporter le film)
- **dealloc :** Désallouer la mémoire

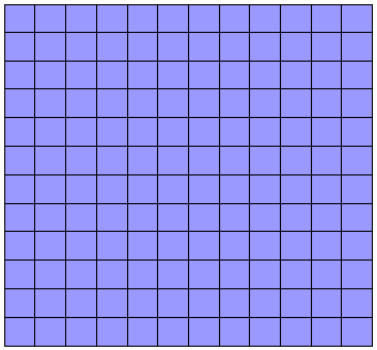
La parallélisation
Ce que nous parallélisons

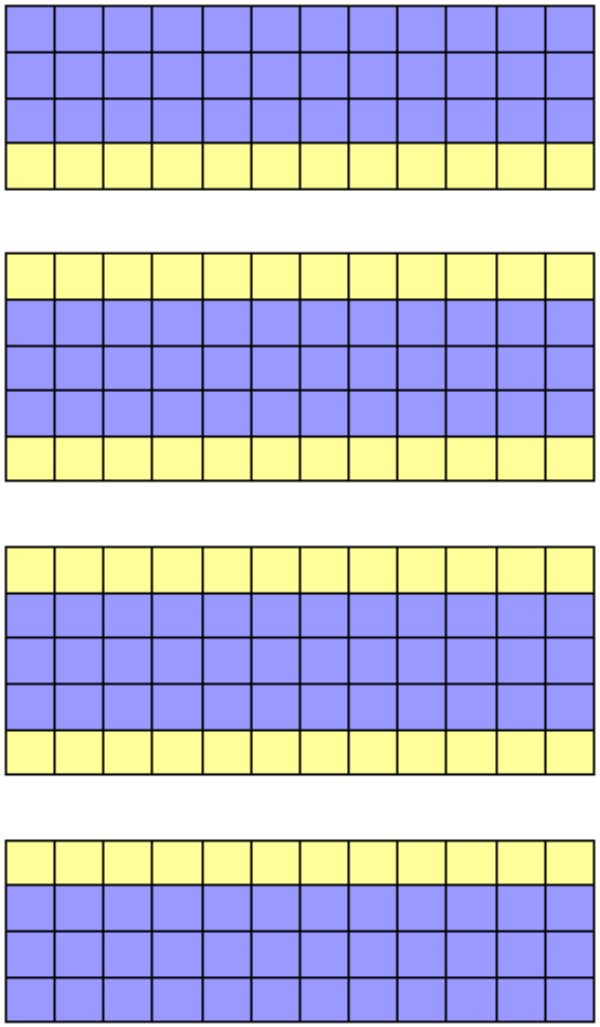
parse_args	Tous
alloc	Rang 0
loc_alloc	Tous
gauss_init	Rang 0
forward	Parallélisé par rapport à x et y
export	Rang 0
dealloc	Rang 0
loc_dealloc	Tous

Distribution des tâches

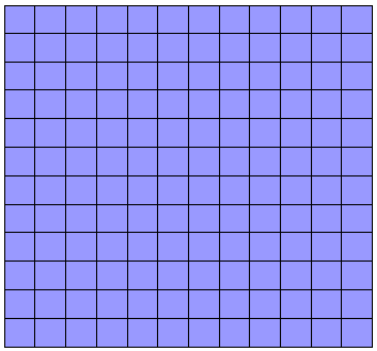
- **La parallélisation par bandes** : nombre de bandes = nombre de processeurs
- **La parallélisation par blocs** : nombre de blocs = nombre de processeurs

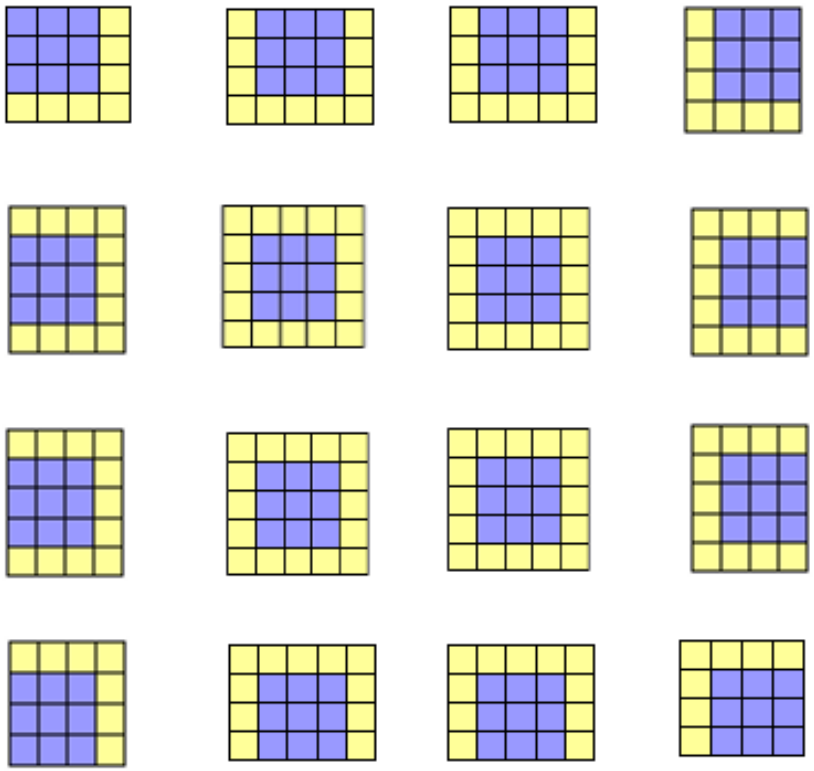
La parallélisation
Parallélisation par bandes

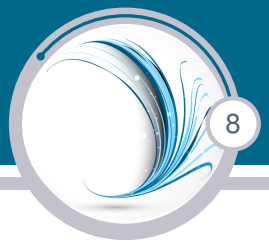




La parallélisation
Parallélisation par blocs





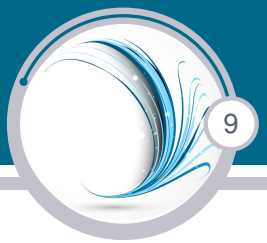


- Communication standard bloquante (MPI_Sendrecv).

Pour les valeurs $x = 8192$, $y = 8192$ et $t = 20$

Nb procs	Séquentiel	Parallèle	Speedup	Efficacité
1	433.487s	633.763s	0.68	0.68
2	433.487s	659.48s	0.66	0.33
4	433.487s	307.736s	1.41	0.35
8	433.487s	163.113s	2.66	0.33
16	433.487s	109.997s	3.94	0.25

- Le cas le plus efficace est celui avec 4 processeurs.



- Communication synchrone non bloquante (MPI_Issend et MPI_Irecv).

Pour les valeurs $x = 8192$, $y = 8192$ et $t = 20$

Nb procs	Séquentiel	Parallèle	Speedup	Efficacité
1	433.487s	397.179s	1.09	1.09
2	433.487s	234.326s	1.85	0.92
4	433.487s	153.293s	2.83	0.71
8	433.487s	124.12s	3.49	0.44
16	433.487s	106.09s	4.09	0.26

- Le cas le plus efficace est celui avec 2 processeurs.

Les performances

Performances de la parallélisation par bandes avec MPI-IO

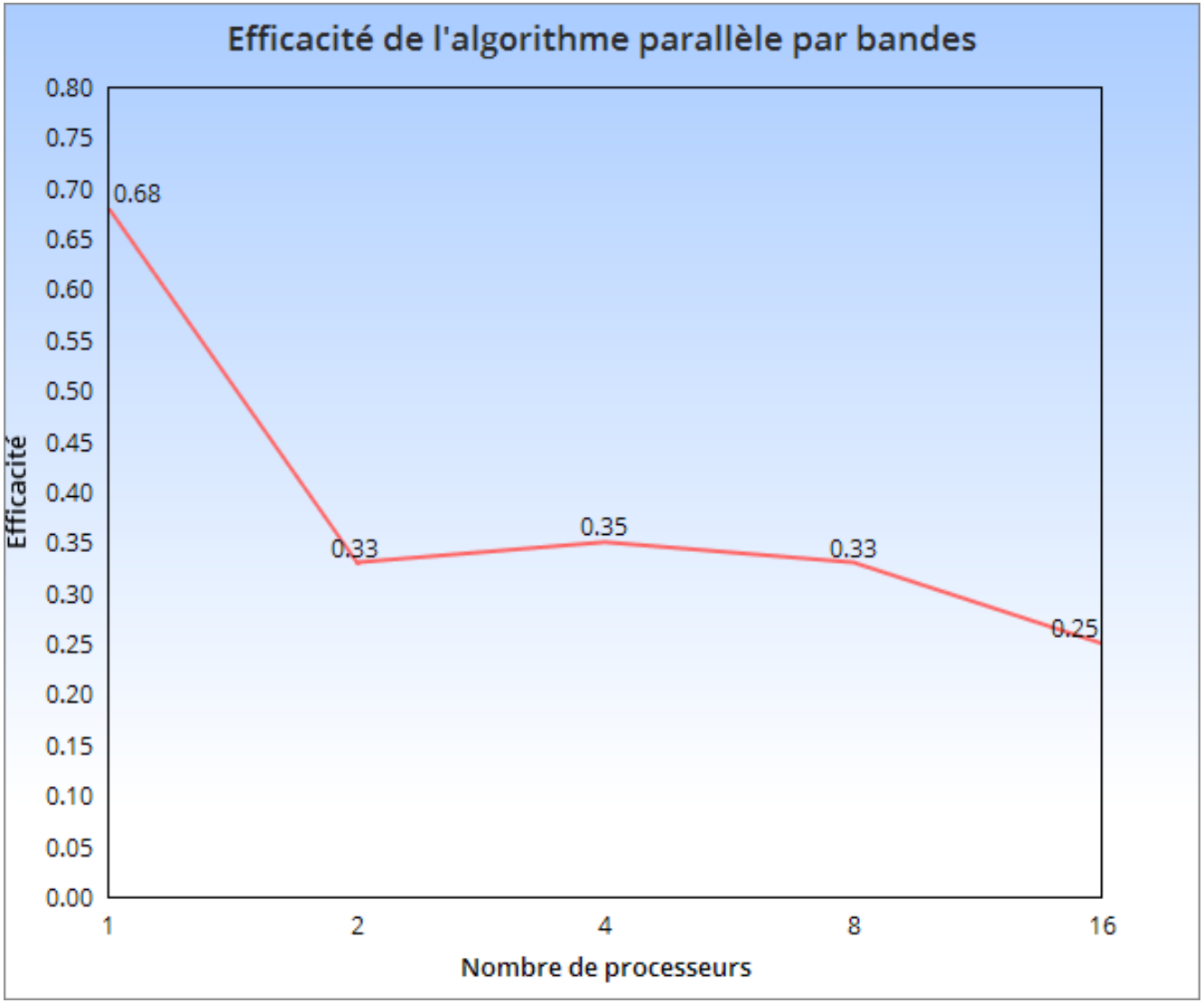
Pour les valeurs $x = 512$, $y = 512$, $t = 40$ et 4 processeurs

Séquentiel	Par bandes	MPI-IO
2.49s	3.23s	2.8s

- Pour des petites images, l'algorithme parallèle est moins efficace que l'algorithme séquentiel.
- MPI-IO permet d'améliorer les performances de l'algorithme parallèle.
- La vidéo visualisée est similaire dans les trois cas.

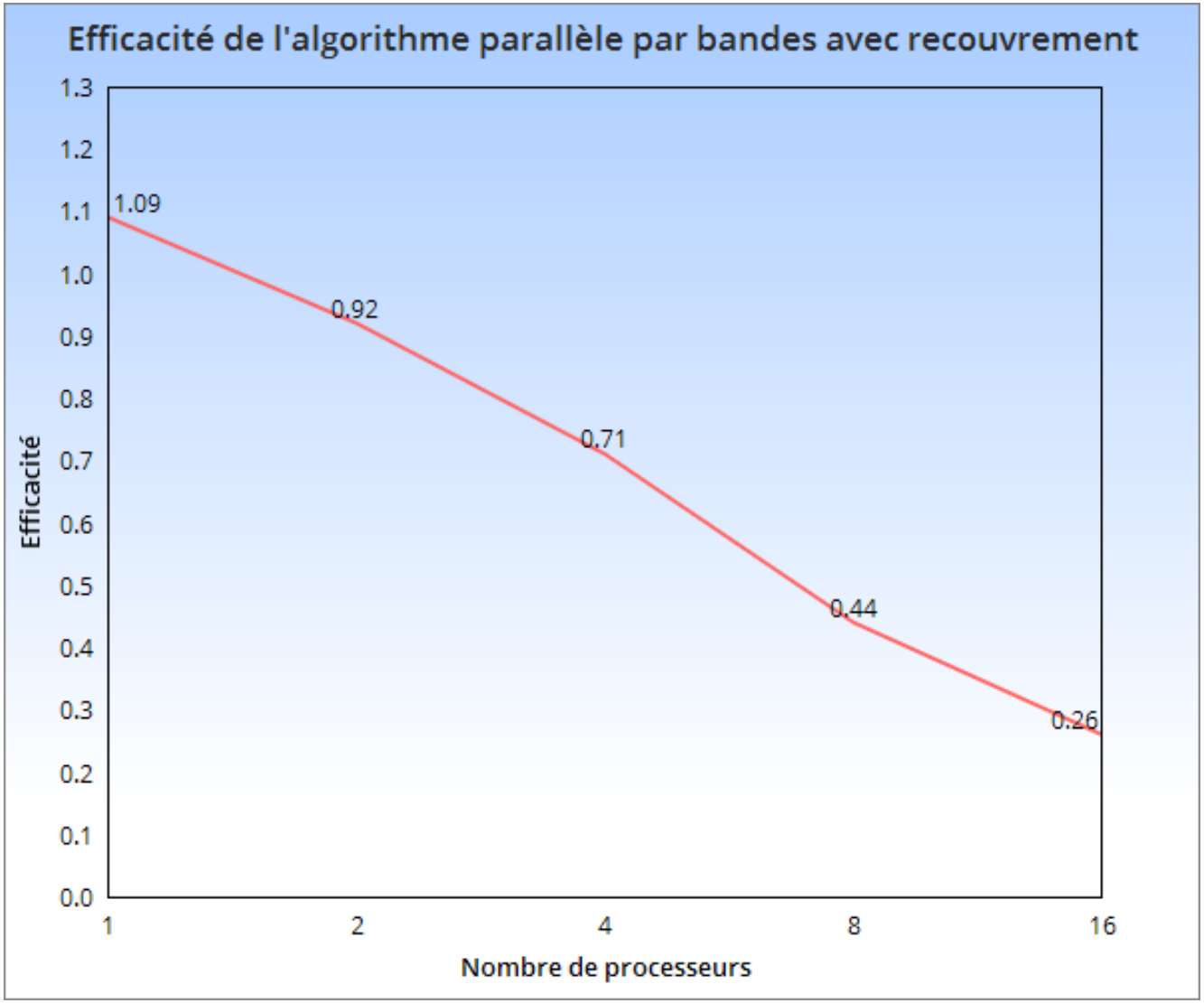
Les performances

Les courbes d'efficacité des algorithmes parallèles



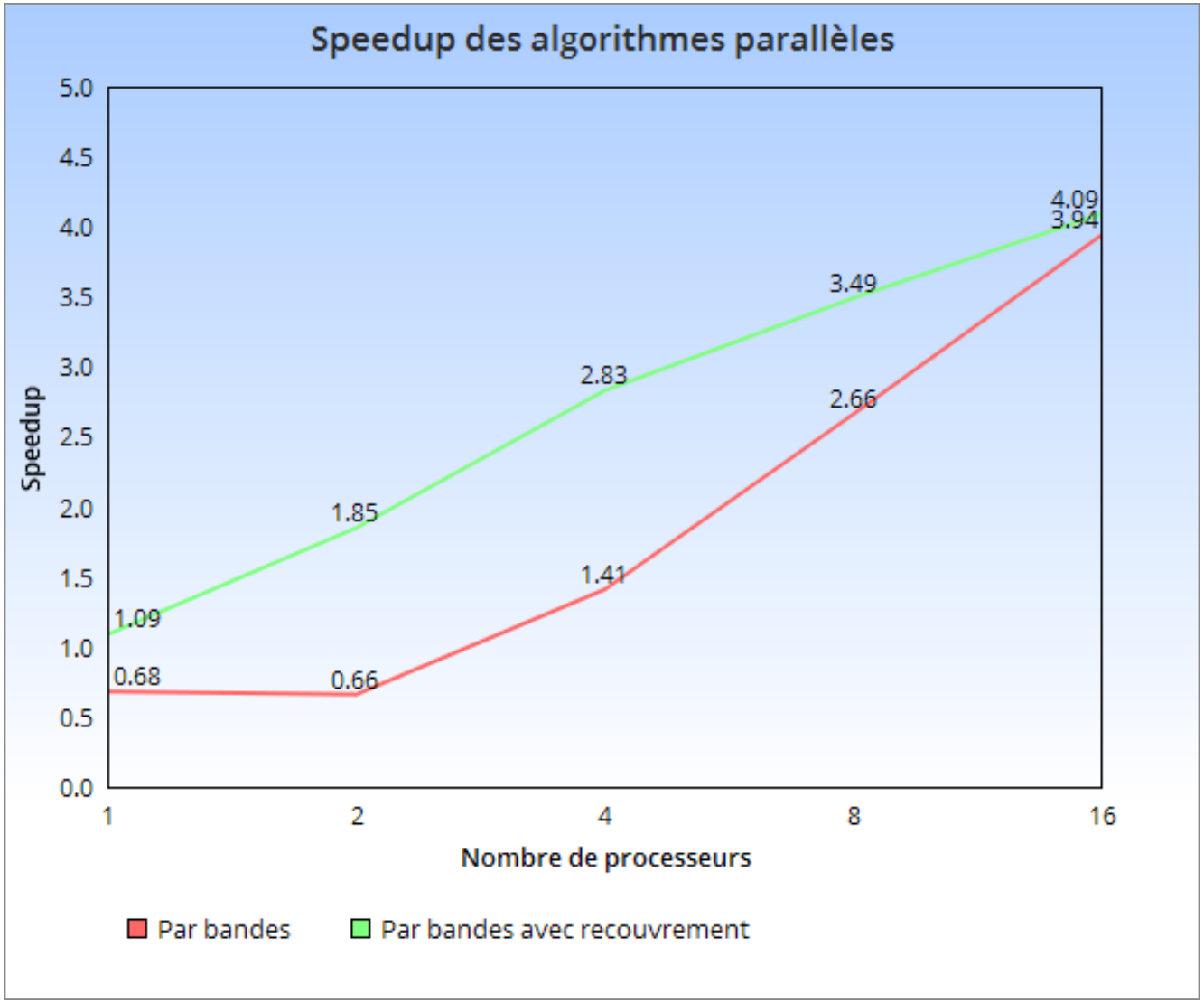
Les performances

Les courbes d'efficacité des algorithmes parallèles



Les performances

Les courbes d'accélération des algorithmes parallèles



Conclusion

Fait	Non fait
Paralléliser par bandes	Partie 2 avec OpenMP
Découper par blocs	Débuguer la parallélisation par blocs
Recouvrir les communications par le calcul	
Simuler des entrées/sorties parallèles (MPI-IO)	

Merci pour votre attention.