Simulation de collapse d'une bulle à l'aide d'un supercalculateur hybride

Remy Dubois*, Eric Goncalves da Silva † and Philippe Parnaudeau†

* IDRIS, CNRS, UPS 851
e-mail: remy.dubois@idris.fr, web page: http://www.idris.fr

† Institut Pprime, CNRS, UPR 3346
e-mail: eric.goncalves@ensma.fr, philippe.parnaudeau@cnrs.pprime.fr - Web page: https://www.pprime.fr

Le supercalculateur hybride

Figure: Jean Zay

- La puissance crête cumulée est de **28 Pflop/s**
- **2 696** Nvidia Tesla V100
- **85 000** cœurs Intel Cascade Lake
- Intel Omni-PAth 100 Gb/s

Contexte général

La cavitation est un phénomène qui peut avoir des conséquences négatives (érosion), mais également positives (soin par lithotripsie.)

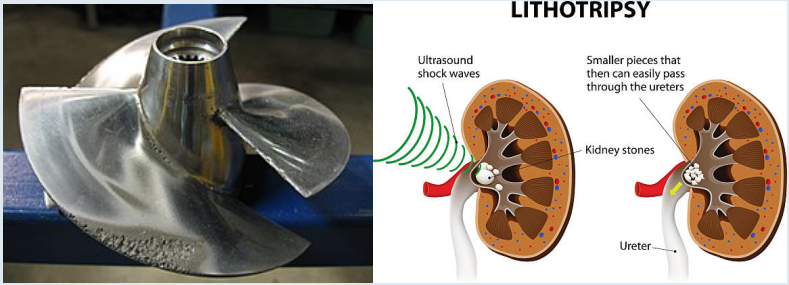


Figure: dommages dû à de la cavitation sur une hélice, traitement par onde de choc de calculs rénaux

Collapse d'une bulle

Le collapse d'une bulle induit par une onde de choc est un événement violent, d'une grande intensité (Giga Pascal) et très rapide (μ seconde).

Notre code multiphasique (**SCB**) permet de réaliser des simulations numériques 3D réalistes de tels phénomènes.

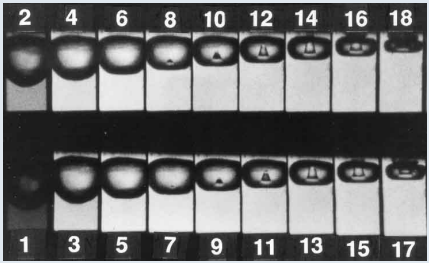


Figure: différents instants du collapse d'une bulle, d'après Y. Tomita et A. Shima, A. (1990) : High-speed photographic observations of laser-induced cavitation bubbles in water, Acustica,71, No. 3

Modèle à 4 équations

Principales difficultés :

- Ecoulement diphasique, compressible, onde de choc
- Intensité et brièveté du phénomène
- Coût élevé des simulations numériques 3D ($> 1.10^9$ cellules)

Hypothèses :

- Effets de tension de surface et visqueux : négligés
- Pas de transfert de masse
- Maillage structuré cartésien

Solutions retenues :

$\Rightarrow$ Modèle à 4 équations : bonne précision et coût de calcul plus faible
$\Rightarrow$ Approche hybride MPI-ACC pour usage sur supercalculateur hétérogène

Équations de base

Description du modèle

• Le 4-Eq est une réduction du modèle à 5-Eq proposé par Kapila.
3 équations conservatives : masse, momentum et énergie totale + équation de transport pour le taux de vide (α)

Expression numérique du modèle

$$\frac{\partial \mathbf{W}}{\partial t} + \nabla \cdot \mathbf{A} + \mathbf{S}\nabla \cdot \mathbf{u} = \mathbf{0}$$

$\mathbf{W} = (\rho, \rho\mathbf{u}, E, \alpha)^\intercal$: vecteur d'état
$\mathbf{A} = (\rho\mathbf{u}, \rho\mathbf{u} \otimes \mathbf{u} + P\mathbf{1}, \alpha\mathbf{u})^\intercal$: vecteur flux
$\mathbf{S} = (0, \mathbf{0}, 0, -(K + \alpha))^\intercal$: terme source

• L'équation de Wallis : vitesse du son
• En supposant l'équilibre mécanique et thermique du système : l'équation des gaz raides

Stratégie

Remarques :

- Coût des simulations 3D ⇒ Utilisation des supercalculateurs
- Supercalculateur hétérogène (CPU+ accélérateur) ⇒ 2 paradigmes (MPI+xxx)
- GPU majoritaire ⇒ OpenCL, CUDA, OpenMP, OpenACC, etc. : lequel choisir ?

Choix des paradigmes

⇒ MPI+OpenMP - Au départ, pour un usage uniquement CPU-CPU

⇒ MPI+OpenACC - A présent pour usage CPU-CPU et CPU-GPU

Objectifs :

- Performance
- Portabilité
- Développement et maintenance

Parallélisation ”mémoire distribuée” (MPI)

- Code ”stencil” (5 pts / direction)
- Points fantômes
- Echange P2P

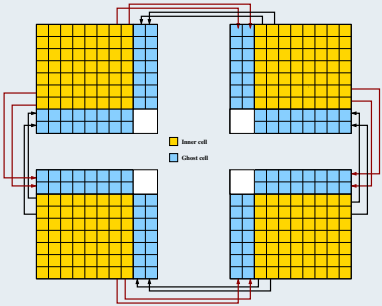


Figure: vue schématique des communications

Parallélisation ”mémoire partagée”

- Programmation ”fine-grain” $\Longleftrightarrow$ directive *OMP DO*
- Minimiser l'usage de la directive *OMP PARALLEL*
- *ACC DATA PRESENT*

```
1  DO ndt=1,ndtmax
2
3    !$OMP PARALLEL IF(ijmax.gt.256) default(none)
4    !$OMP DO SCHEDULE (runtime) PRIVATE (i,j,k) COLLAPSE(2)
5     DO k=kmin,kmax
6     DO j=jmin,jmax
7     DO i=imin,imax
8        RI1=w1(i,j,k) - w1(i-1,j,k)
9        sl=dmax(0.0,dmin(Ri1,1.0))+dmin(0,dmax(1,Ri1))
10       W1(i,j,k)= W1(i-1,j,k)+1/4*sl*(W1(i-1,j,k)-W1(i-2,j,k))+1/4*sl*(W1(i ,j,
                  k)-W1(i-1,j,k))
11    ENDDO
12    ENDDO
13    ENDDO
14   !$OMP END DO
15    CALL BOUNDARY (W1)
16   !$OMP END PARALLEL
17    CALL MPI_SENDRECV(W1, imax*kmax, MPI_DOUBLE_PRECISION,&neib_mpi(
            N),tag, W1, imax*kmax, MPI_DOUBLE_PRECISION,&neib_mpi(S) ,tag,
            comm, status, err_mpi)
18
19 ENDDO
```

Listing 1: OpenMP

```
1  !$ACC DATA COPY (W1)
2  DO ndt=1,ndtmax
3    !$ACC KERNELS DATA PRESENT(W1)
4     DO k=kmin,kmax
5     DO j=jmin,jmax
6     DO i=imin,imax
7        RI1=w1(i,j,k) - w1(i-1,j,k)
8        sl=dmax(0.0,dmin(Ri1,1.0))+dmin(0,dmax(1,Ri1))
9        W1(i,j,k)= W1(i-1,j,k)+1/4*sl*(W1(i-1,j,k)-W1(i-2,j,k))+1/4*sl*(W1(i ,j,
                  k)-W1(i-1,j,k))
10    ENDDO
11    ENDDO
12    ENDDO
13   !$ACC END KERNELS
14    CALL BOUNDARY (W1)
15   !$ACC UPDATE HOST(W1)
16    CALL MPI_SENDRECV(W1, imax*kmax, MPI_DOUBLE_PRECISION,&neib_mpi(
            N),tag,W1, imax*kmax, MPI_DOUBLE_PRECISION,&neib_mpi(S) ,tag,
            comm, statut, err_mpi)
17   !$ACC UPDATE DEVICE(W1)
18 ENDDO
19 !$ACC END DATA
```

Listing 2: OpenACC

Un coeur Intel Xeon 6248

Résumé de l'anayse (Roofline) :

- Intensité arithmétique $\simeq 0.3 \iff$ Mémoire- dépendant donc limité par le débit mémoire
- $\simeq 7\%$ du pic théorique d'un seul coeur, un taux de vectorisation de l'ordre $\simeq 85\%$
- $\simeq 20\%$ de la bande passante mémoire maximale

Remarques :

- Merci Intel-Advisor et la librairie PAPI
- Intensité arithmétique $=$ FLOPs $/$ Octet manipulé (calcul/mémoire)
- Intensité arithmétique faible commun avec ce type de code (stencil)

OpenMP vs OpenACC : CPU

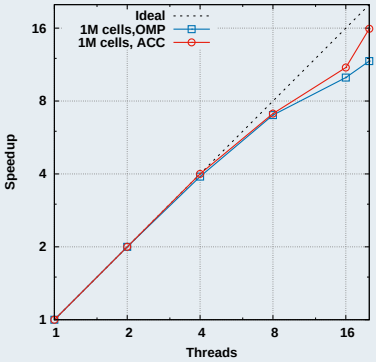
- Strong scaling (passage à l'échelle "fort"): $[1 : 20]$ coeurs d'un processeur Intel Xeon 6248

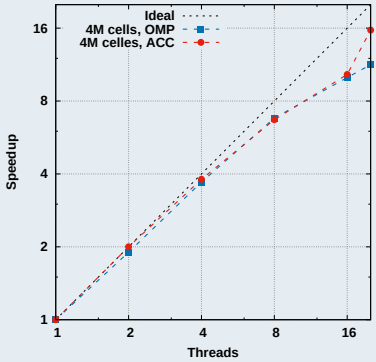
	1 million de cellules			4 millions de cellules		
	Temps de calcul		Accélération	Temps de calcul		Accélération
Threads	**OMP**	**ACC**		**OMP**	**ACC**	
1	691.00	1243.85	1.8	2770.12	4047.05	1.46
2	348.77	603.18	1.73	1399.92	2016.77	1.44
4	183.43	309.99	1.69	733.03	1047.72	1.43
8	102.73	176.96	1.72	408.13	600.99	1.47
16	69.67	113.77	1.63	268.62	389.09	1.44
20	62.10	78.60	1.26	244.35	255.85	1.04

Table: OpenMP (**OMP**) vs OpenACC (**ACC**)

OpenMP vs OpenACC : CPU

- Strong scaling: $[1 : 20]$ coeurs d'un processeur Intel Xeon 6248





- La programmation grain fin explique sans doute le manque de performance > 10 coeurs.

CPU vs GPU

- Strong scaling : 1 et 20 coeurs Intel Xeon 6248 vs 1 NVidia Tesla V100 card

	1 million de cellules			4 millions de cellules		
	Temps de calcul		Accélération	Temps de calcul		Accélération
Threads	CPU-ACC	GPU-ACC		CPU-ACC	GPU-ACC	
1	1243.85	34.60	35.94	4047.05	102.79	39.37
20	78.60		2.27	255.85		2.48

Table: **CPU** Versus **GPU** with OpenACC

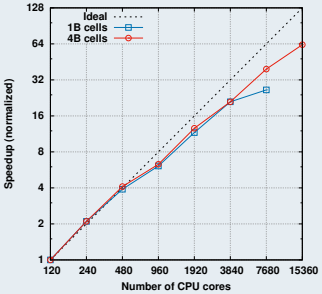
	1 million de Cellules			4 millions de Cellules		
	Temps de calcul		Accélération	Temps de calcul		Accélération
Threads	CPU-OMP	GPU-ACC		CPU-OMP	GPU-ACC	
1	691.00	34.60	19.97	2770.12	102.79	26.94
20	62.10		1.79	244.35		2.37

Table: **CPU** avec OpenMP Versus **GPU** avec OpenACC

MPI-seul (CPU)

Accélération

- Strong scaling : $[120 : 15\,360]$ Intel Xeon Gold 6248
- Simulation 3D avec 1 milliard et 4 milliards de cellules

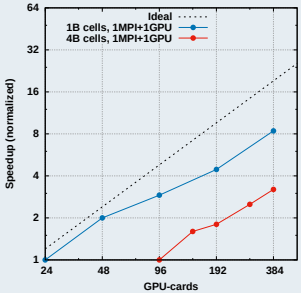


- Efficacité $\simeq 50\%$ avec 15 360 cores
- Efficacité $> 80\% \iff 10^6$ cellules par sous-domaine MPI est un optimum

MPI-OpenACC (GPU)

Accélération

- Strong scaling : $[24 : 384]$ NVidia Tesla V100 card
- Simulation 3D avec 1 milliard et 4 milliards de cellules



- Efficacité $\simeq 80\%$ avec 384 GPU
- Efficacité $> 80\% \iff 10^7$ cellules par sous-domaine MPI est un optimum

Multi-CPU vs multi-GPU

1 milliard de cellules				4 milliards de cellules			
CPU		GPU		CPU		GPU	
Coeurs	Temps	Cartes	Temps	Coeurs	Temps	Cartes	Temps
240	1400	24	1917	960	1001	96	1030
480	730	48	955	1920	530	192	580
960	470	96	658	3840	318	384	320
1920	245	192	432				
3840	136	384	228				

Table: MPI-Alone (**CPU**) versus MPI-OpenACC (MPI+GPU) (**GPU**)

- Performance intéressante en multi-GPU
- OpenACC est un choix mature

Collapse d'une bulle en proche paroi dû à une onde incidente

Description

- Une bulle d'air proche d'une paroi dans une liquide, avec un rayon initial : R_0
- Onde incidente se déplaçant à Mach $= 1.024$

$$(\rho, \mathbf{u}, p) = \begin{cases} (998 \text{ kg/m}^3, \mathbf{0} \text{ m/s}, 10^5 \text{ Pa}) & \text{Dans le liquide (eau)} \\ (1 \text{ kg/m}^3, \mathbf{0} \text{ m/s}, 10^5 \text{ Pa}) & \text{Dans la bulle (air)} \end{cases}$$

- Symétrie en y et $z \rightarrow$ limite la simulation à un quart du domaine
- $[L_x \times L_y \times L_z] = [8R_0 \times 12R_0 \times 12R_0]$
- 2 maillages utilisés $N_{r_{(b,0)}} = [55, 110]$
- Etude comparative avec Wermelinger *et al.*(J. Comput. Sci., 2018)

Evolution de la pression

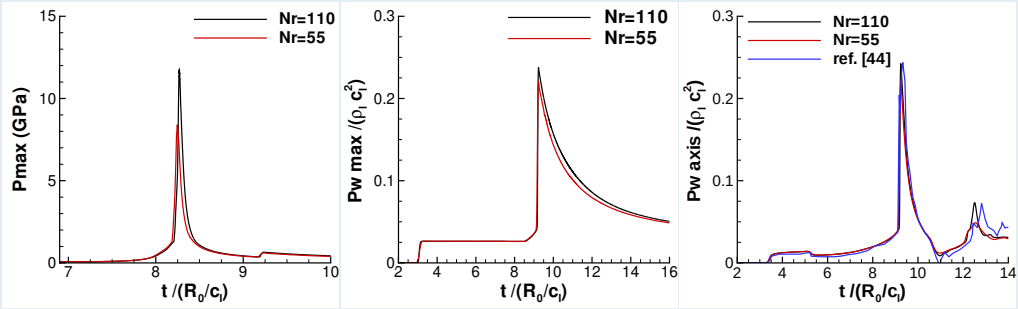


Figure: Pression dans le liquide (Gauche), à la paroi (Centre) et le long d'un axe sur la paroi (Droite)

- Bon accord avec les résultats de Wermelinger *et al.*(ref. [44]).
- L'intensité maximale de pression dans le fluide est $340\times$ intensité pression onde incidente.
- L'influence du maillage est forte pour capter le pic de pression dans le fluide, cela l'est moins à la paroi.

Visualisations

Composante longitudinale de la vitesse, pression à la paroi, gradient de la densité et taux de vide

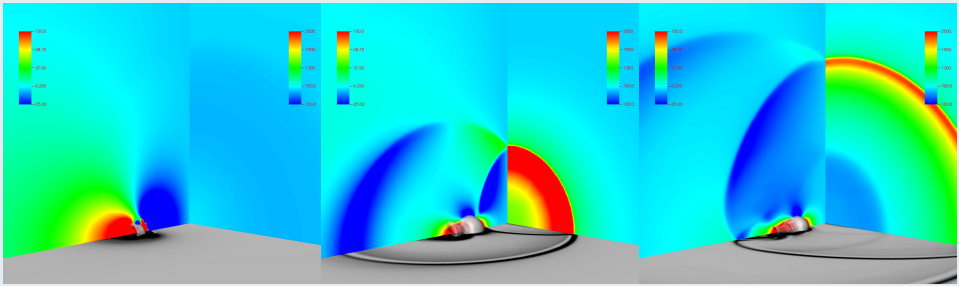


Figure: De gauche à droite $t^* = (8.4, 10.1, 12.1)$

• A $t^* = 8.4$: la bulle a une forme toroïdale + jet rapide à l'intérieur $\rightarrow$ "Water-hammer", premier collapse $\rightarrow$ fragments de bulle

• A $t^* = 10.1$: l'onde de choc du "Water-hammer" arrive à la paroi $\rightarrow$ Intense peak de pression à la paroi (érosion) $\rightarrow$ onde réfléchie dans le fluide

• A $t^* = 12.1$: l'onde réfléchie collisionne les fragments de bulle $\rightarrow$ Re-collapse

Conclusion

Résultats

- OpenACC : performances acceptables sur un ou plusieurs accélérateurs (GPU)
- Le modèle 4 équations est efficace et précis
- ”High performance computing of stiff bubble collapse on CPU-GPU heterogeneous platform”, Computers & Mathematics with Applications, 2021

Perspectives

- MPI-OpenMP versus MPI-OpenACC sur GPU
- J’espère une version commune entre OpenACC and OpenMP avant ma retraite ?!

Remerciements

Ce travail a été financé par l’Agence Nationale de la Recherche ANR (project 18-CE46-009). Les calculs ont été réalisés sur le supercalculateur Jean Zay du GENCI (IDRIS, CNRS) sous l’allocation A0072A10981.