



Avancées du logiciel *elsA* pour le calcul haute performance

Michel Gazaix (ONERA)

Pierre-Marie Brunet, Cédric Trivino (C-S)

Journée Scientifique ONERA, 9 Octobre 2009



Plan de l'exposé

- **Présentation du logiciel *e/sA***
- **Vers le massivement parallèle**
 - ▷ Scalabilité (ou "passage à l'échelle") faible et forte
- **Résultats acquis**
 - ▷ **Trois exemples**
 - ⇒ *Avion civil (AIRBUS)*
 - ⇒ *DPW4 : 4th AIAA CFD Drag Prediction Workshop*
 - ⇒ *Compresseur (SNECMA)*
- **Récentes évolutions du calcul haute performance**
- **Quelle(s) stratégie(s) pour *e/sA* ?**

***elsA* : résout les équations de Navier-Stokes compressible**

Développé par l'ONERA et le CERFACS depuis 1997

- **Maillage multibloc structuré**
 - ▷ Raccords entre blocs "non structurés"
 - ⇒ 'match', 'nearmatch', 'nomatch'
 - ⇒ *Chimère*
- **Extension aux maillages hybrides (coexistence blocs structurés et non structurés) en cours de développement**
 - ▷ Non abordé ici
- **Cell-center**
- **Simulation d'écoulements internes et externes**
- **Multi-Physique**
 - ▷ Aéro-élasticité, Aéro-acoustique, Aéro-thermique
- **Grande diversité de modèles de turbulence**
 - ▷ (U)RANS
 - ▷ DES, LES

e/sA : Architecture Orienté Objet

- **Langage de programmation : C++ / Fortran**
 - ▷ Interface utilisateur : Python
- **Parallélisme MPI : SPMD**
 - ▷ Équilibrage de charge : répartition des blocs sur les processeurs
- **Utilisation sur ordinateurs vectoriels (NEC SX8+, SX9) et scalaires**
 - ▷ Pas d'incompatibilité entre optimisation scalaire et vectoriel

Le parallélisme dans e/sA

elsA parallèle : Principes généraux

Utilisation de MPI

□ Apports du parallélisme

- ▷ Diminution du temps de restitution
- ▷ Calcul de grosses configurations

□ Paradigme gros grain

- ▷ *elsA* : pas (presque pas) de support OpenMP (thread)

□ Communications MPI

- ▷ Échange de données frontières aux raccords inter-blocs
 - *multiplicité des raccords dans elsA*
 - *point important pour être compétitif avec le non-structuré*
 - *Coût de développement et de maintenance non négligeable*
- ▷ Chimère : échange de données volumiques
- ▷ Conditions limites "globales" (débit par ex.)

Peut devenir complexe à programmer

(ex : chimère chorochronique)

***e/sA* parallèle : Spécificités du multibloc structuré**

Équilibrage de charge (load balancing)

- **Allouer Nb blocs sur P processeurs**
- **Problème vraiment difficile (NP- complet)**
 - ▷ Plus facile en non structuré
Metis, Scotch, Chaco, . . .
- **Pas de solutions miracles en multibloc structuré**
 - ▷ On ne connaît pas d'algorithme "optimum"
(pas de calcul de l'optimum en temps fini)
- **Algorithme "heuristique"**
 - ▷ En général rapide, peu coûteux en mémoire
*contrairement au non structuré (ParMetis),
pas nécessaire de paralléliser la phase de découpage
ex : découpage maillage $1.7 \cdot 10^9$ points pour 8192 procs sur un seul
nœud*

elsA parallèle : Spécificités du multibloc structuré

Si le découpage de blocs n'est pas nécessaire :

- **Problème nettement moins compliqué**
 - ▷ Pondération calcul \leftrightarrow communication : assez complexe
 - *Particulièrement pour raccord 'nomatch' et chimère*
 - ▷ Attention à la loi d'Amdhal
 - *Équilibrage de plus en plus difficile pour P grand même un faible déséquilibre entraîne une forte dégradation*

elsA parallèle : Spécificités du multibloc structuré

Si le découpage de blocs est nécessaire :

□ **Contrainte forte du multigrille**

▷ Si n_g grilles grossières :

$$\Rightarrow \text{indice}(\text{split}) = 0 \pmod{2^{n_g} + 1}$$

▷ Contrainte similaire si on veut conserver le "mesh sequencing"

(calcul 1 point sur 4, puis 1 sur 2, enfin tous les points)

▷ cette contrainte n'existe pas en non structuré

□ **Dégradation potentielle de l'efficacité numérique**

▷ Augmentation mémoire et temps CPU (points fictifs)

▷ Dégradation du caractère implicite

- on se limite aujourd'hui à un implicite par bloc

(comme la quasi-totalité des logiciels CFD structurés)

- comportement identique en non structuré
- en pratique : le découpage dégrade peu la convergence

Vers le massivement parallèle (≥ 1000 processeurs)

Vers le massivement parallèle

Scalabilité faible et forte

- **Le but recherché est la conservation de performances acceptables d'une application pour une suite croissante de taille de données et de calculateurs.**
- **Scalabilité forte**
 - ▷ L'efficacité reste bonne quand le nombre de processeur augmente, en conservant une taille de problème fixe
 - ▷ Le parallélisme permet de diminuer le temps de restitution
 - ⇒ *par ex. calcul instationnaire (LES)*
 - ▷ L'efficacité forte est en pratique très difficile à obtenir pour un logiciel CFD
- **Scalabilité faible**
 - ▷ L'efficacité reste bonne quand le nombre de processeur augmente, en conservant une taille de problème fixe par processeur
 - ▷ Le parallélisme permet d'augmenter la taille du problème : situation très fréquente

Vers le massivement parallèle

Scalabilité forte : difficile !

- **Découpage des blocs existants :**
 - ▷ Augmente le nombre de cellules fictives (calculs supplémentaires)
 - ▷ Plus de données à échanger par le réseau d'interconnexion
- **Les communications globales deviennent plus coûteuses ($\text{Log}(Np)$ étapes)**
- **Exemple concret (AIRBUS CADWIE)**
 - ▷ Calcul instationnaire, LES ou DES, maillage fixé, (presque) pas de limites sur le nombre de processeurs disponibles
 - ▷ L'utilisateur souhaite obtenir le temps de restitution le plus faible possible
 - ▷ À partir d'un certain nombre de processeurs, Np_{max} , le temps de restitution ne diminuera presque plus
 - ⇒ Np_{max} dépend de la machine et de la taille du problème
 - ⇒ Sur IBM BlueGene, pour un maillage de 15 millions de points, Np_{max} est voisin de 128

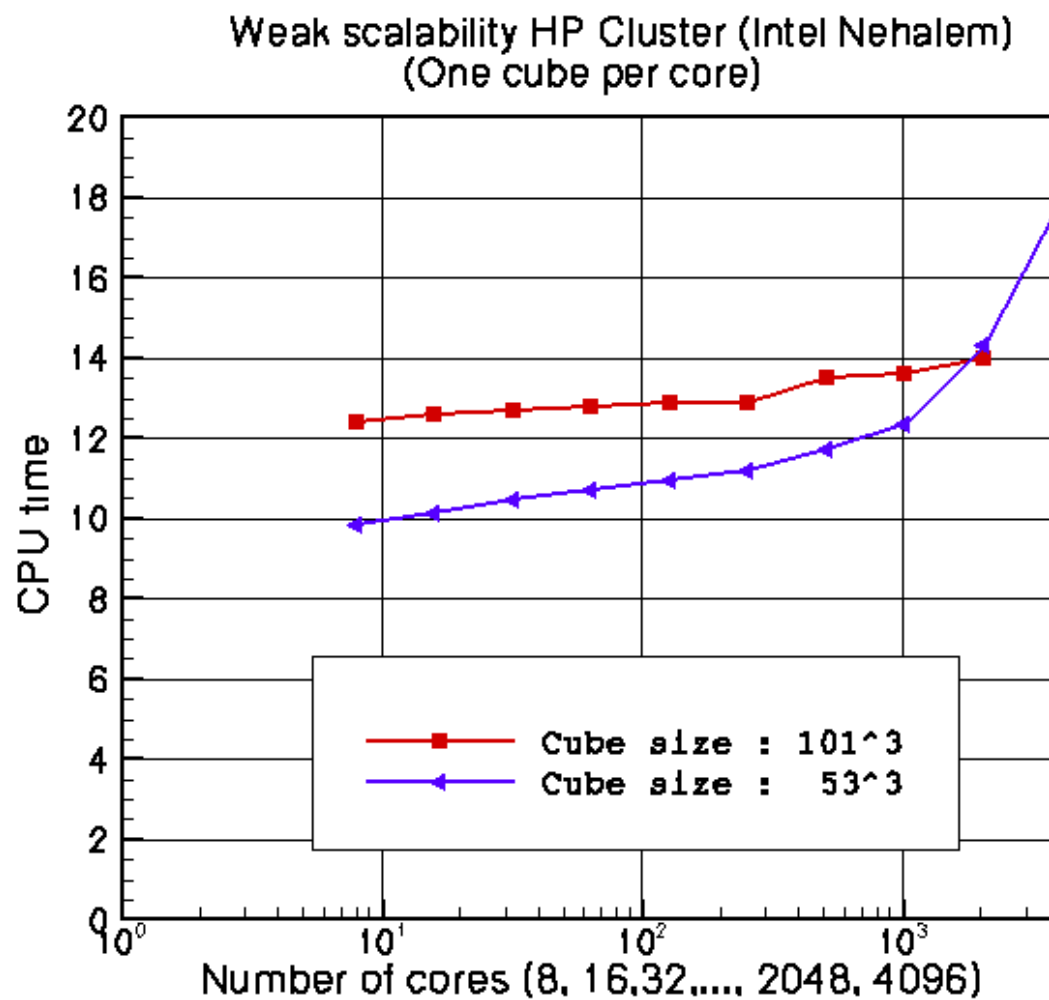
Grand nombre de machines évaluées

Consommateur en temps...

- NEC SX8+ (ONERA iseran), NEC SX-9 (METEO)**
- IBM PowerPC (BSC MareNostrum)**
- IBM BlueGene/L (CERFACS) et BlueGene/P (EDF)**
- SUN, processeur AMD Opteron (C²A²S²E cluster)**
- BULL, processeur Intel Itanium 2 (ONERA galibier)**
- BULL, processeur Intel Nehalem (CCRT titane)**
- HP, processeur Intel Nehalem (AIRBUS)**
- CRAY XT5, processor Istanbul (12 cores / node)**

Scalabilité faible (HP Intel Nehalem)

AIRBUS HPC3



Grand nombre de machines évaluées

Consommateur en temps...

- ❑ **Nombre maximum de cœurs : 8192**
- ❑ **Nombre maximum de blocs : 17000**
- ❑ **Jusqu'à 12 Milliards de points**
- ❑ **Trop grande variabilité des temps mesurés (CPU, Wall clock,...)**
 - ▷ Problème souvent présent si NPROC grand (sauf IBM BlueGene)
 - ▷ Peu compatible avec une démarche scientifique rationnelle
 - ▷ Rend très difficile (et frustrant) le travail d'optimisation
 - Ex : "tuning" de l'algorithme de découpage des blocs*
- ❑ **Attention : Swapping possible sur certaines machines**
 - ▷ Pb aggravé par l'augmentation de la mémoire nécessaire avec NPROC
- ❑ **Instabilité des plateformes**
 - ▷ `ex : MPI_Sendrecv_replace : Internal MPI error`
- ❑ **Outils d'analyse complexes, demandant un temps de formation important**
 - ▷ Souvent dépendant de la plateforme, de la librairie MPI

Algorithmique et Calcul Massivement Parallèle

- Certains algorithmes prennent l'avantage lorsque NPROC croît
- Pas toujours facile à identifier *a priori*
 - ▷ ex : Chimère vs Hybride (Structuré / non structuré)
- Cas important : Time Spectral Method vs URANS
 - ▷ Simulation instationnaire d'écoulements périodiques
 - ▷ URANS : scalabilité forte difficile à maintenir
 - ▷ TSM : $2N_{harm} + 1$ instances d'e/sA couplées (faiblement)
 - ⇒ "embarrassingly parallel"

Quelques exemples de résultats acquis

Adaptation du logiciel pour NPROC grand

Trois exemples

□ Amélioration de la phase d'initialisation

▷ Version initiale : > une heure pour 8192 cœurs

⇒ *Recodage de l'algorithme de tri des objets conditions limites*

□ Modification du format de définition de la topologie

▷ Script Python lu par chaque processeur

▷ Nécessite une mémoire trop importante

□ Modification de l'ordonnancement des échanges MPI

▷ Collaboration ONERA / CERFACS / C-S / BSC

▷ Algorithme de type "Coloriage de graphe"

▷ Temps de restitution divisé par un facteur voisin de 2

pour des configurations industrielles typiques

▷ Extension aux raccords 'nomatch' en cours

Optimisation du calcul de la distance à la paroi

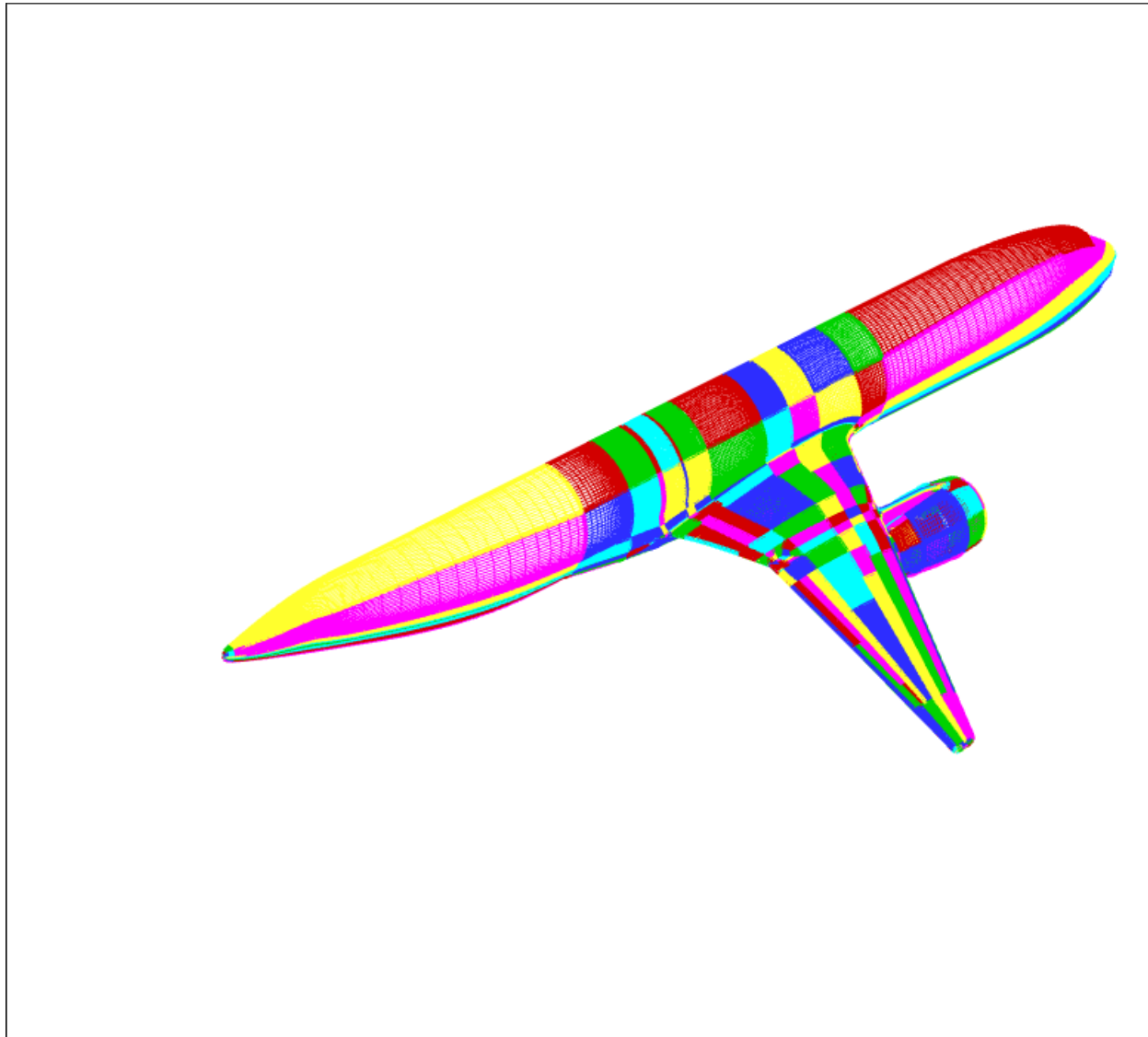
- **Nécessaire pour la plupart des modèles de turbulence**
- **Algorithme "naïf" en N^3**
 - ▷ huit heures sur 2048 cœurs
 - ⇒ *maillage DPW4 (1.7 10⁹ points)*
- **Adaptation de l'algorithme**
 - ▷ Utilisation d'un arbre ADT
- **Étude préliminaire d'accélération par GPU**
 - ▷ Facteur d'accélération voisin de 5

Simulation d'un avion civil

Projet FUSIM (AIRBUS)

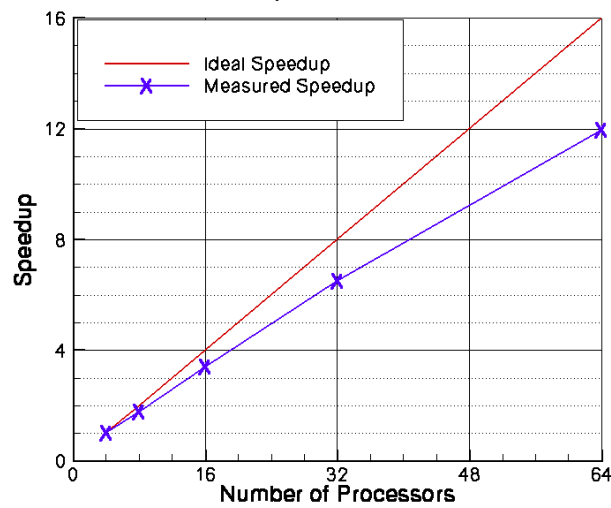
- 1074 blocs, maillage $28 \cdot 10^6$ points
- Étude sur différentes plate-formes de calcul
 - ▷ NEC SX8+
 - ▷ SGI Altix Xeon bi-processeur
 - ▷ BULL Novascale Itanium-2
 - ▷ HP AMD Opteron
 - ▷ IBM BlueGene/L
- Communication ParallelCFD 2008

Configuration Avion (AIRBUS)

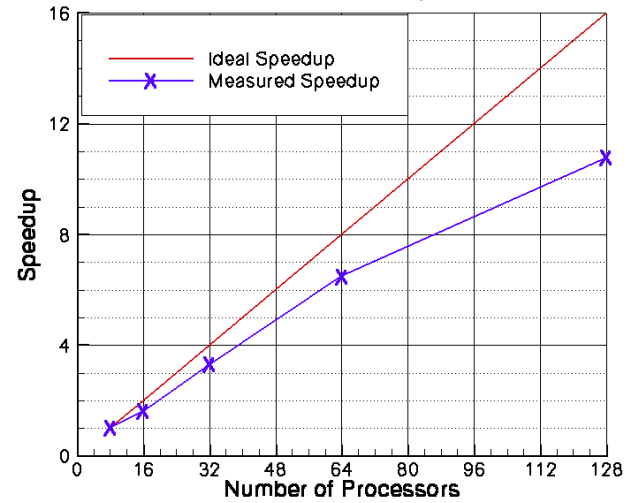


Speedup parallèle

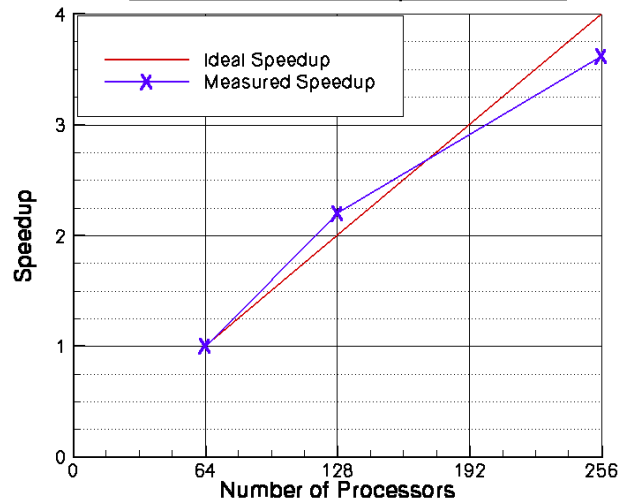
**ONERA naruto (SGI Altix XE Xeon 3 GHz dual core)
(normalized with Nproc=4)**



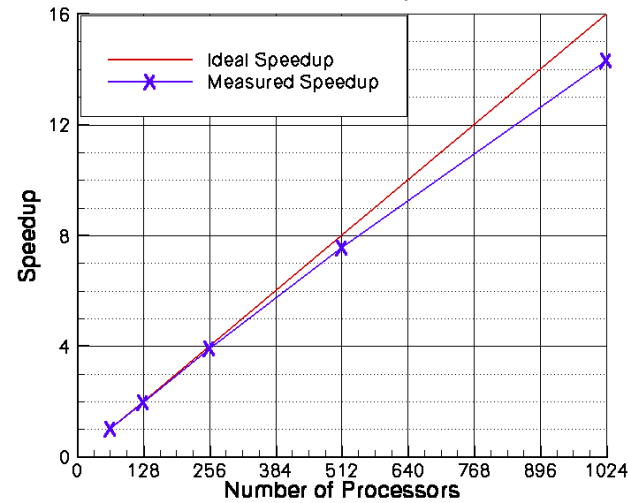
**CCRT-platine (BULL Novascale, Infiniband)
(normalized with Nproc=8)**



**CCRT-tantale (HP Opteron 2.4 GHz)
(normalized with Nproc=64)**



**BlueGene/L (Cerfacs)
(normalized with Nproc=64)**



Comparaison des performances *e/sA*

1074 blocs, maillage $28 \cdot 10^6$ points

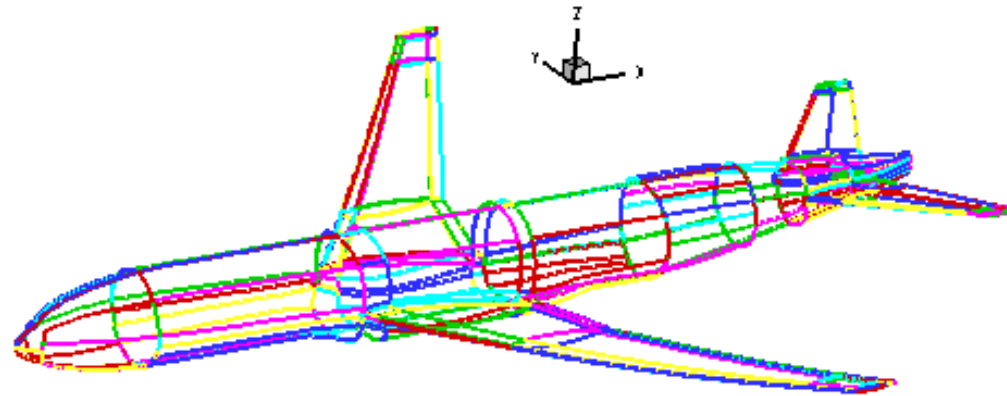
- **NEC SX8+, 1 processeur : 89.0 secondes/cycle**
 - ▷ 2.7 GigaFlops, $3.5 \mu s$ / cycle/ cell
- **BULL Novascale Itanium2 (onera :galibier), 32 proc : 22.8 secondes/cycle**
- **IBM BlueGene/L, 128 proc : 21.2 secondes/cycle**
- **IBM BlueGene/L, 1024 proc : 2.91 secondes/cycle**
 - ▷ Speedup entre 128 et 1024 proc : 7.28
 - ▷ 82 GigaFlops, approx. 30 proc NEC SX8
- **Plus le processeur est "lent", plus il est "facile" d'obtenir un bon speedup (à réseau donné)**

Drag Prediction Workshop (DPW IV)

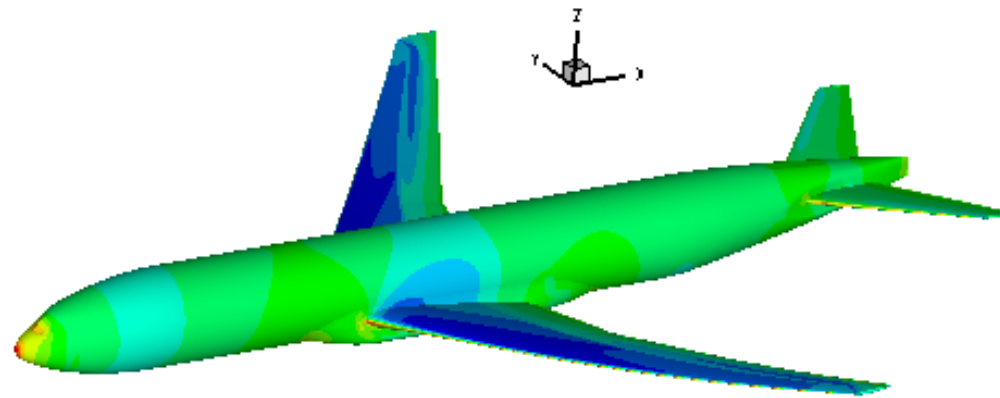
Collaboration AIRBUS / ONERA (DAAP/DSNA) / C-S

- **Voilure - fuselage - HTP**
- **Une partie des calculs a été effectuée dans le cadre du projet POPS :**
 - ▷ Bull et partenaires du pôle de compétitivité mondial SYSTEM@TIC
 - ▷ Développer une nouvelle génération d'applications à l'échelle du Petaflops
- **Plusieurs maillages disponibles**
 - ▷ 200 millions de cellules
 - ▷ **1.6 milliard de cellules**
- **Calculs effectués sur plusieurs machines**
 - ▷ IBM BlueGene/L
 - ▷ NEC SX-9
 - ▷ HP et BULL (Processeur Intel Nehalem)

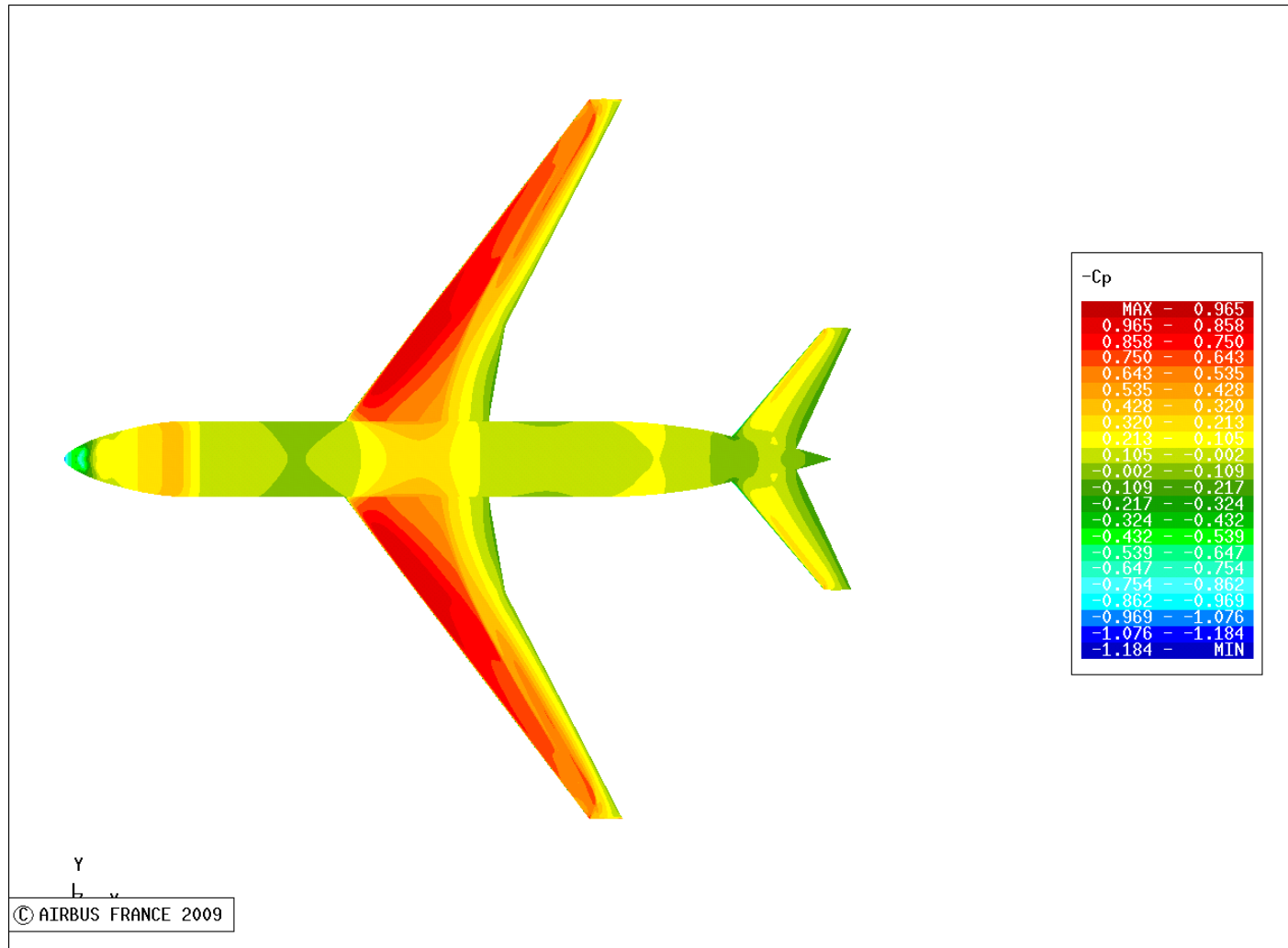
DPW4



DPW4

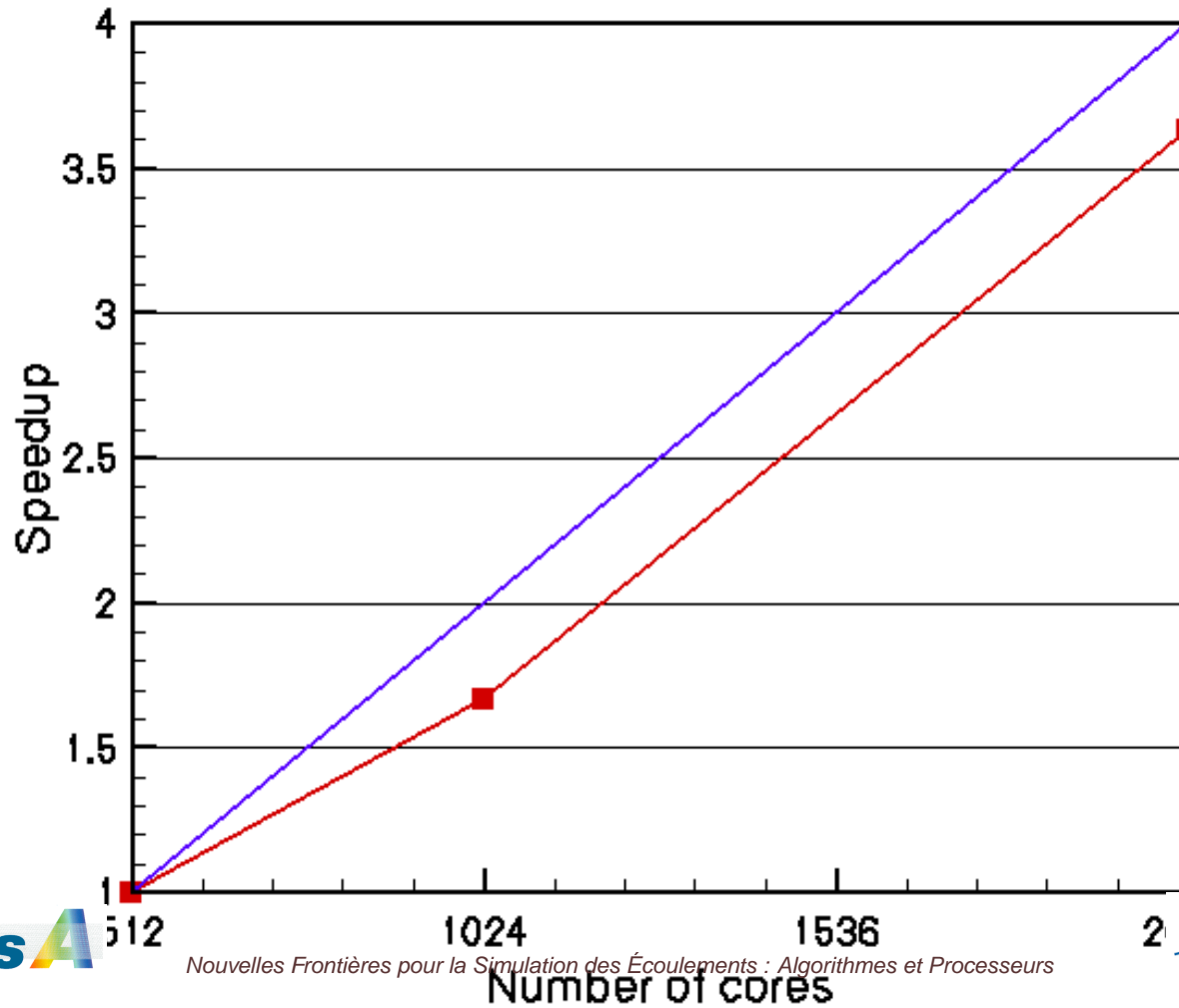


DPW4



DPW4 : Scalabilité forte (Très gros maillage, bon équilibrage de charge)

DPW4 - $1.7 \cdot 10^9$ mesh point
(no multigrid)

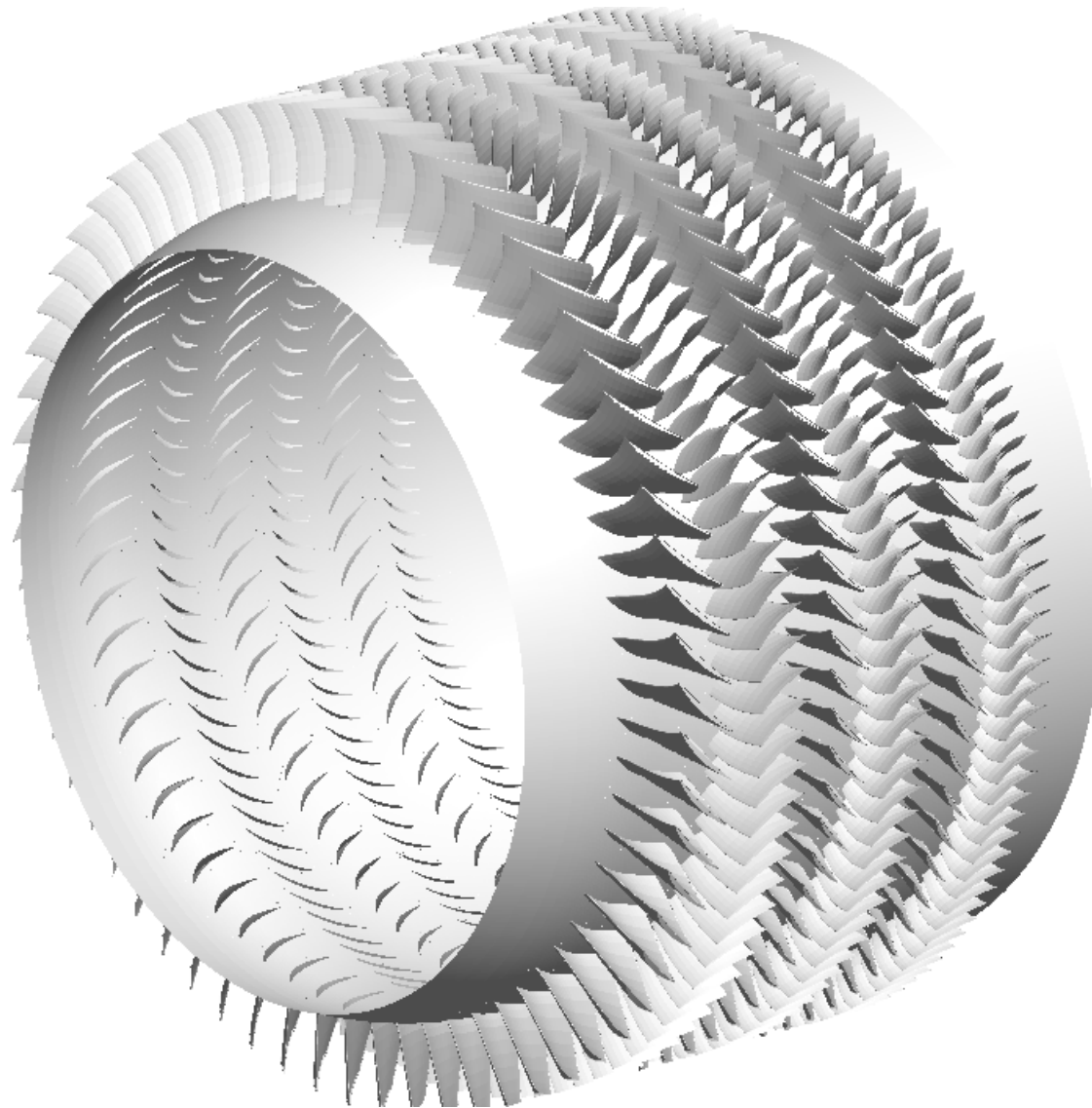


Simulation instationnaire d'un compresseur complet

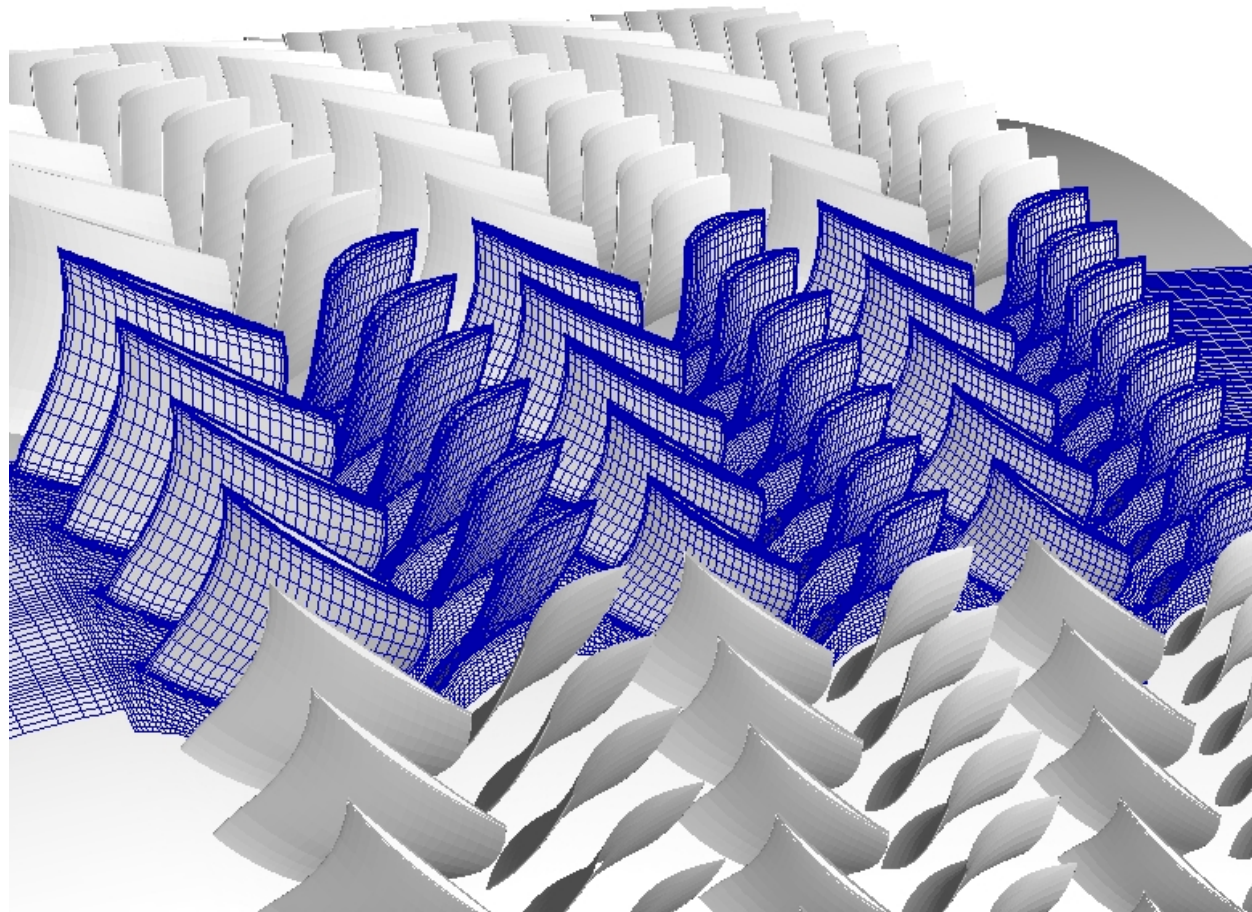
Collaboration CERFACS (Nicolas Gourdain) - ONERA

- **Compresseur trois étages (6 roues)**
 - ▷ Configuration CREATE (SNECMA / LMFA)
 - ⇒ *2848 blocs, 134 10⁶ points*
- **Calcul instationnaire complet**
 - ▷ 26 jours, 512 processeurs IBM BlueGene/L (prêt EDF): 320 000 heures
- **Turbulence : $k - \omega$ avec loi de paroi (sans calcul de distance)**
- **Pas d'approximation pour le couplage rotor / stator "Sliding Mesh nomatch"**
 - ▷ Étude des mécanismes physiques complexes (décollement tournant. . .)
 - ▷ Essentiel pour les nouvelles générations de moteurs
- **Jusqu'à 4096 processeurs**
 - ▷ Pour 8192 processeurs : mémoire insuffisante
 - ⇒ *Interprétation (par Python) du script devient trop gourmande*
 - ⇒ *Format de définition de la topologie à modifier*
- **À paraître dans J. of Parallel and Distributed Computing**

Simulation instationnaire d'un compresseur complet

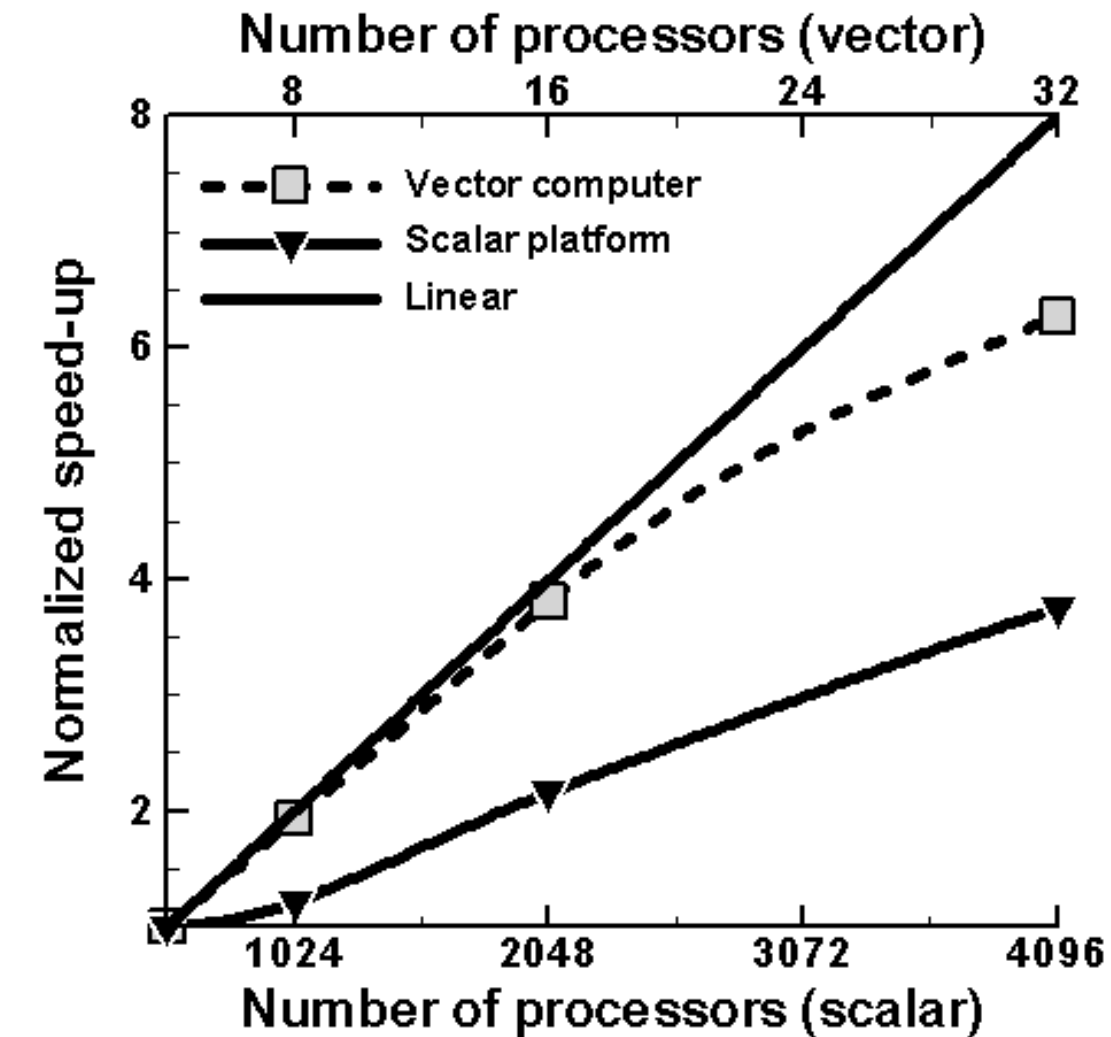


Simulation instationnaire d'un compresseur complet



Simulation instationnaire d'un compresseur complet

Speedup moyen



Efficacité MPI elsA

- **Efficacité relativement satisfaisante**
- **Possibilités d'amélioration :**
 - ▷ Recouvrement calculs / communications (non bloquantes)
 - ▷ Concaténation des messages (en option)
 - ▷ Amélioration de l'algorithme d'équilibrage de charge
 - ⇒ *Pondération actuelle : uniquement le nombre de cellules*
 - Possibilité de prendre en compte le coût des communications*
 - demande un paramétrage pour chaque couple CPU/réseau*

Croissance du calcul scientifique haute performance

Croissance de l'impact du HPC en CFD (1)

Plus de calculs, plus gros, plus précis

□ **Augmentation de la taille des simulations**

▷ Workshops *elsA*: augmentation continue de la taille des simulations

▷ Prise en compte détaillée de la géométrie

⇒ *"Effets technologiques"*

▷ Besoin de précision accrue

⇒ *Les maillages actuels sont probablement insuffisants*

Conclusion DPW III

(<http://aaac.larc.nasa.gov/tsab/cfdlarc/aiaa-dpw/Workshop3>)

□ **Montée en ordre des méthodes numériques**

▷ WENO, Galerkin, Spectral Volume...

□ **Augmentation de la complexité du modèle physique**

▷ Passage de RANS à LES...

□ **Temps physique simulé plus grand (meilleure statistique en LES par ex.)**

□ **Quantification des incertitudes (Polynomial chaos, Monte Carlo...)**

Croissance de l'impact du HPC en CFD (2)

□ Augmentation des simulation instationnaires

- ▷ Capturer les fréquences intéressantes souvent très coûteux

⇒ *Simulation instationnaire d'une turbomachine multi-étage*

- ▷ LES, DES, RANS/DES

- ▷ Couplage avec la mécanique du vol : "Flying by the equations"

⇒ *Environ 25 fois plus coûteux*

que l'obtention d'une solution stationnaire

M.D. Salas, J. Scientific Computing, Vol. 28, Sept. 2006

□ Multi-physique

- ▷ Aérodynamique / Structure / Thermique / Acoustique

□ Production de base de données ("Flying through the database")

- ▷ Très grand nombre de calculs nécessaire

⇒ *AIAA 2008-712 (Rossow / Kroll): 10^5 simulations*

□ Calculs multi-échelles

□ Augmentation de la simulation par rapport à l'expérience

□ Boucle d'optimisation

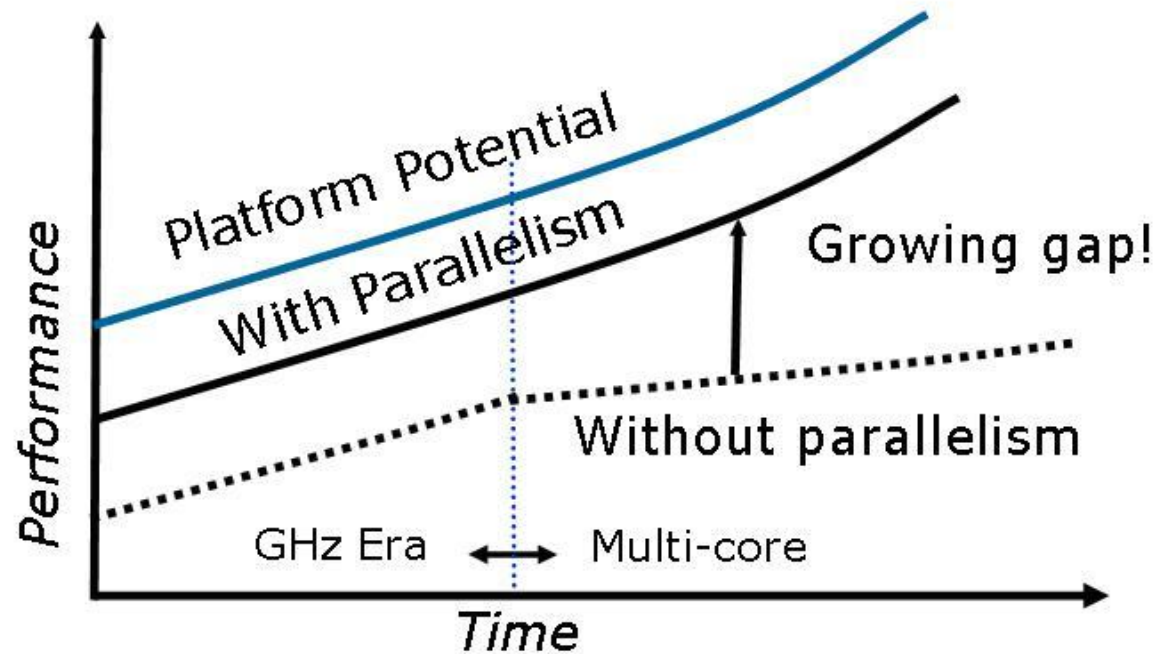
Croissance du calcul scientifique haute performance

Ce n'est plus une affaire de spécialistes

- Capacité de simulation : Clef pour rester compétitif**
- Fin de l'augmentation des fréquences des CPU**
 - ▷ Les logiciels de calcul doivent évoluer
- Prise de conscience de la complexité inhérente au calcul massivement parallèle**

Changement de paradigme : 2005

Need for Parallelism



Croissance du calcul scientifique haute performance

Initiatives Françaises et Européennes

- **GENCI**
- **PRACE (<http://www.prace-project.eu/>)**
 - ▷ "Partnership for Advanced Computing in Europe"
- **DEISA (Distributed European Infrastructure for Supercomputing Applications)**
- **C²A²S²E**
 - ▷ Center for Computer Applications in AeroSpace Science and Engineering
 - ⇒ AIRBUS / DLR / Niedersachsen (6144 cœurs, update en 2010)
 - ⇒ "The virtual airplane"
 - ⇒ elsA est utilisé (AIRBUS) sur C²A²S²E
- **Objectif AIRBUS : doublement puissance disponible chaque année**
 - ▷ Machine HP : 10 000 cœurs Intel Nehalem (09/2009)

Quelle(s) stratégie(s) pour e/sA ?

Quelle stratégie pour les années à venir?

Peu de certitudes...

- **Fin de l'augmentation de la fréquence d'horloge**
 - ▷ Évacuation de la chaleur, consommation électrique
- **Parallélisme**
 - ▷ Quelles architectures ?
 - ▷ Quel modèle de programmation ?
- **Processeurs spécialisés**
 - ▷ Lesquels ?
- **"Memory wall problem"**
 - ▷ Le transfert des données progresse moins vite que le CPU
⇒ *accès mémoire et réseaux d'interconnexion trop lent*
- **Parallélisation des Entrées/Sortie (I/O)**
 - ▷ Essentiel pour l'instationnaire
 - ▷ Potentiel de MPI 2 ?
 - ▷ Quid de HDF5 ? NetCDF ?

Quelle stratégie pour les années à venir?

Peu de certitudes...

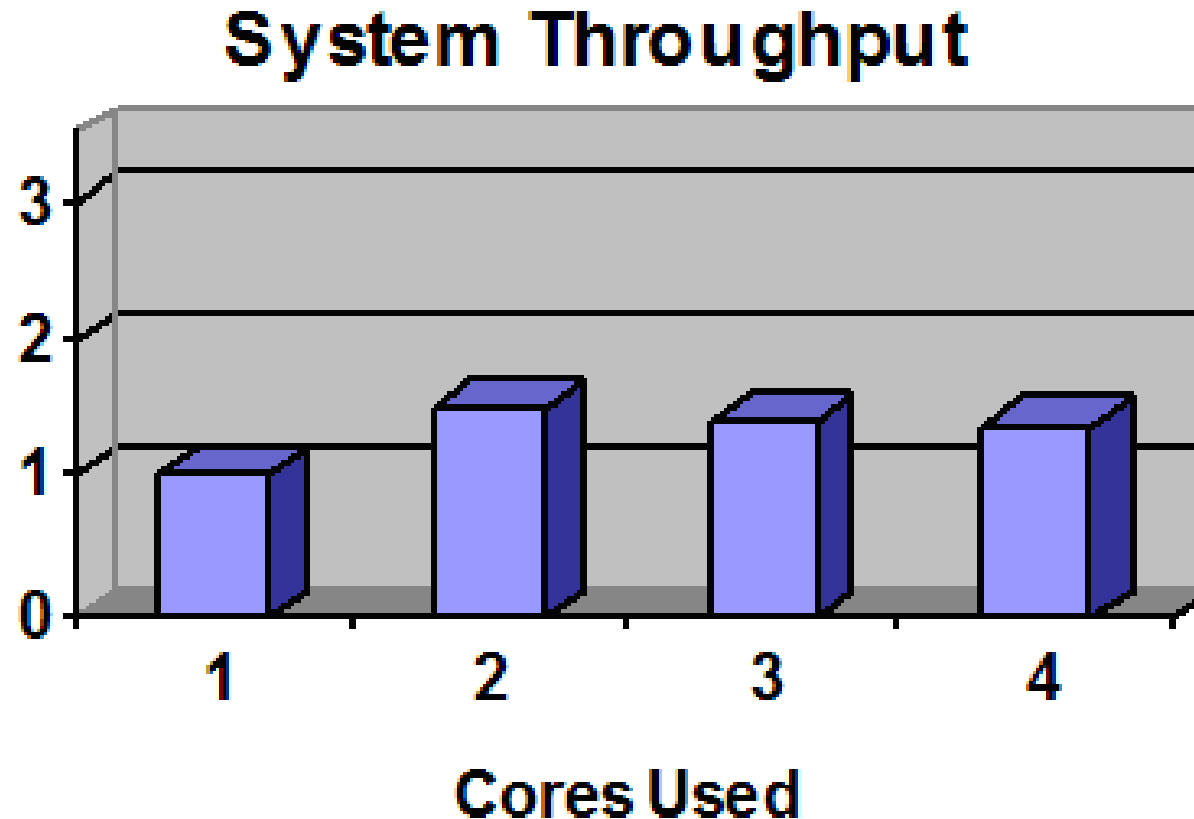
- **Impact de l'architecture multi-cœur**
 - ▷ Bande passante mémoire insuffisante ?
- **Importance accrue du rapport Flop/watt**
- **Langage de programmation parallèle**
 - ▷ Programmer avec MPI ressemble à programmer en assembleur (en pire?)
 - ⇒ *Programmation trop bas niveau*
 - ▷ Il existe des tentatives pour augmenter l'abstraction
 - ⇒ *PGAS(Partitioned Global Address Space) langages*
 - UPC (Unified Parallel C)*
 - <http://www.alphaworks.ibm.com/tech/upccompiler>*
 - Co-Array Fortran(<http://www.co-array.org/>)*
 - ⇒ *Langages HPC : Chapel (Cray) , X10 (IBM) Fortress (SUN)*
 - <http://chapel.cs.washington.edu/>*
 - ⇒ *Mini-expérience sur Processeur Cell avec elsA*

Quelle stratégie pour les années à venir?

Architecture multi-cœur

- Prévision : doublement du nombre de cœurs à chaque génération de processeur**
- Certains éléments hardware sont dupliqués : unité de calcul, cache L1,...**
- D'autres éléments sont partagés : cache L2/L3, Bande passante mémoire (DRAM)**
- Augmentation du nombre de cœurs : problématique pour les ressources partagées :**
 - ▷ 1. Pas de progrès importants attendues pour la bande passante (et latence) mémoire (DDR3)
 - ▷ 2. La taille du cache partagé risque de croître moins vite que le nombre de cœurs
- Le risque de perte de performances existe même pour les applications non parallèles !**
 - ▷ "Throughput" du centre de calcul

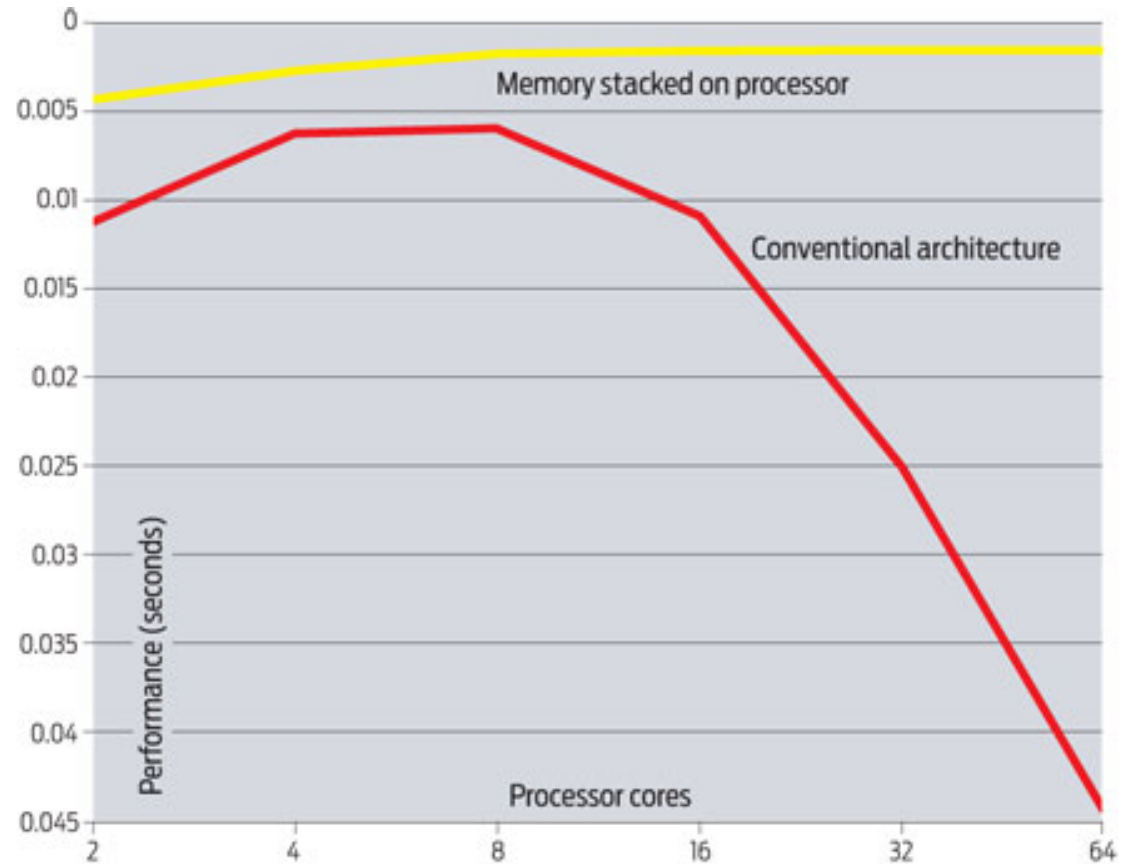
Dégradation de l'efficacité globale (applications séquentielles)



Graph 1. The amount of work (throughput) performed by a multicore chip while varying the number of cores utilized, normalized to the work performed by a single core.

Simulation (Sandia) de l'impact du nombre de cœurs en HPC

<http://www.spectrum.ieee.org/nov08/6912>



Simulation (Sandia) de l'impact du nombre de cœurs en HPC

Multicore Is Bad News For Supercomputers

- Sandia, Samuel K. Moore, Novembre 2008
- More cores per chip will slow some programs [red] unless there's a big boost in memory bandwidth [yellow].
- More cores doesn't mean better performance
- "The key to solving this bottleneck is tighter, and maybe smarter, integration of memory and processors (...). For its part, Sandia is trying to improve memory bandwidth."

Quelle stratégie pour les années à venir?

S'adapter au multi-cœur

- **La bande passante mémoire ne croîtra pas aussi vite que le nombre de cœurs**
- **Optimiser elsA en conséquence**

- ▷ **Augmenter la localité spatiale**

- ⇒ *Par ex : boucle interne selon la plus grande dimension (IJK)*

- ⇒ *Pas évident en formulation structurée*

- ▷ **Augmenter la localité temporelle**

- ⇒ *Réutiliser les données*

pas facile dans un logiciel généraliste comme elsA, avec beaucoup d'options (pour satisfaire plusieurs types d'utilisateurs) et un codage modulaire (pour réduire les coûts de maintenance)

Quelle stratégie pour les années à venir?

Accélérateurs de calculs

- Très gros effort de portage pour un logiciel multi-finalité tel que *elsA*
- Difficulté : évolution très rapide de la technologie
 - ▷ Beaucoup plus rapide que pour les codes scientifiques
 - ⇒ *elsA* : né en 1997...
- GPU : Utilisations de processeurs graphiques (Nvidia, ...)
- Architecture Intel Larrabee
- FPGA
- IBM Cell

GPU versus Larrabee

Un exemple de choix difficile...

□ GPU Computing

- ▷ Grande effervescence dans la communauté HPC
- ▷ Annonce par Nvidia de l'architecture Fermi (remplace Tesla)
 - ⇒ *Meilleure performance en double précision*
 - ⇒ *Error Correcting code*
 - ⇒ *IEEE-754*

□ Intel LArrabee

- ▷ Hybride entre CPU multi-cœur et GPU
- ▷ Modèle de programmation mieux adapté à *e/sA* ?

Stratégie pour les années à venir

- **Améliorer la scalabilité**
- **Défi de l'équilibrage de charge automatique :**
 - ▷ Maillage composite (HyperFlex)
 - ⇒ *Structuré / Non structuré, Cartésien, Adaptatif*
 - ▷ Calcul couplé multi-physique
- **S'adapter au multi-cœur**
 - ▷ Améliorer l'utilisation du cache mémoire partagé
 - ▷ Programmation hybride OpenMP - MPI
 - ⇒ *OpenMP à l'intérieur d'un nœud SMP,*
MPI entre les nœuds
Diminue la complexité de l'équilibrage de charge
- **Parallélisation des I/O**
- **Expérimenter l'accélération par processeur graphique (GPU)**
 - ▷ Nécessite la réécriture d'une partie du noyau de calcul
 - ▷ Nouveau langage à maîtriser

Conclusion

- **Ne pas manquer le passage au massivement parallèle**
- **La taille d' *e/sA* impose une planification des efforts**
 - ▷ Réussir l'articulation entre :
 - ⇒ *Prototype innovant (GPU, ...)* : "veille technologique"
 - ⇒ *Logiciel validé utilisé en production industrielle sur plate-formes variées*
- **Besoin en organisation et ressources humaines**
 - ▷ Travail collectif indispensable
 - ▷ Formation au massivement parallèle nécessaire
 - ⇒ *Architecture informatique, programmation (langages), expertise CFD, analyse numérique, algorithmique parallèle. . .*
- **Développer les relations de l'ONERA avec la communauté scientifique HPC**
 - ▷ Augmenter l'implication dans les collaborations européennes
 - ▷ Dossier GENCI en cours