Programmation et exploitation des architectures de calcul hybride

Retours d'expériences menées avec le code de nanosimulation BigDFT

25ème Forum ORAP, EDF Clamart

Luigi Genovese (ESRF)
Jean-François Méhaut (UJF-LIG-INRIA)

Matthieu Ospici (Bull-CEA-LIG-INRIA-UJF)
Thierry Deutsh (CEA-INAC-L_Sim)
Stéphane Goedecker (Univ. Bâle)

Bull Échirolles (Grenoble)
Laboratoire d'informatique de Grenoble
INRIA Grenoble Rhône-Alpes, EPI Mescal
Laboratoire de simulation atomistique (L_Sim)
European Synchrotron Radiation Facility (ESRF)

Plan de la première partie

- Contexte et objectifs
- Programmation des grappes hybrides
- Exploitation des grappes hybrides
- Conclusion et perspectives

INRIA Grenoble Rhône-Alpes

Laboratoire d'Informatique de Grenoble (LIG)

Equipe-Projet INRIA Mescal

Middleware Efficiently SCALable

Directeur LIG: Brigitte Plateau (PR INPG/ENSIMAG)

Responsable Mescal: Bruno Gaujal (DR INRIA)

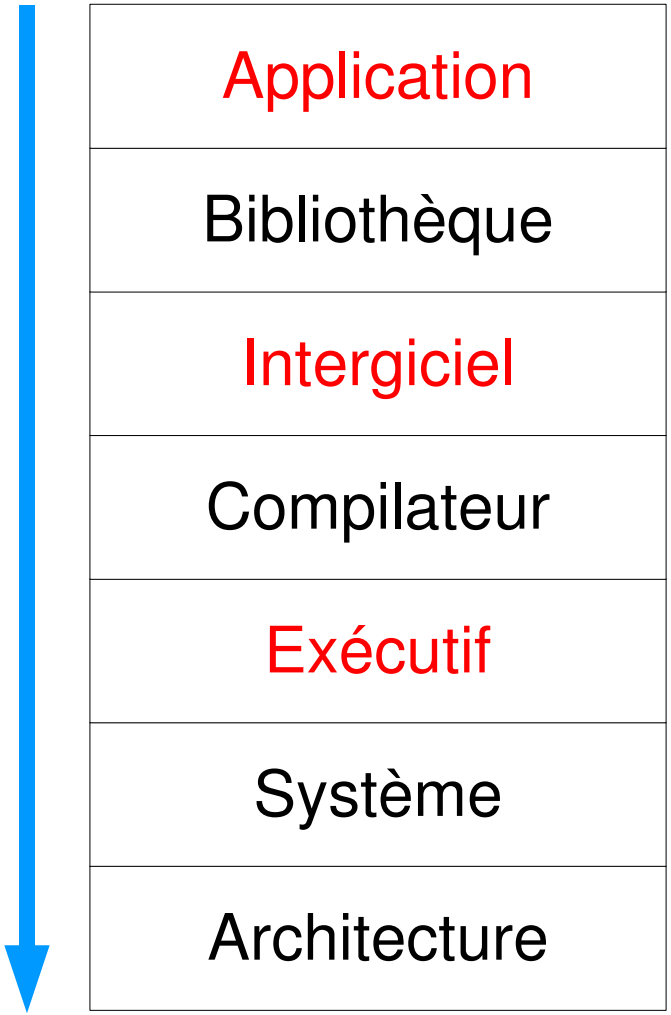
- Thématique scientifique: passage à l'échelle
  - Evaluation de performance
    - Analyse, modélisation, observation
    - Prédiction, dimensionnement
  - Architectures, systèmes, intergiciels et applications
    - Grilles de calcul
      - déploiement d'applications, transferts de données
      - plates-formes Mésocentre CIMENT, Grid'5 000
    - Grappes de calcul
      - gestion de données, gestionnaire de ressources, NUMA, hybride,...
- Collaborations
  - Bull, CEA-INAC, ST Microelectronics, BRGM, C/S, edXact, Infiniscale, INRIAs...

Architectures de calcul hybride

- Grappes de calcul hybride
  - Noeuds de calcul avec processeurs multi-coeurs généralistes (Intel, AMD)
  - Réseaux à haut débit (type Infiniband)
  - Noeuds de calculs avec accélérateurs (GPU NVIDIA, Cell, FPGA,...)
- Quelques exemples...
  - Roadrunner (Processeurs AMD et Cell), Los Alamos
  - Accelerator Cluster (Processeurs AMD, GPU, FPGA), Urbana, UIUC, NCSA, laboratoire commun UIUC INRIA
  - Titane-Bull (Intel Nehalem, GPU NVIDIA Tesla), Genci-CCRT
  - Iblis-Bull (Intel Xeon, GPU NVIDIA Tesla), Genci-Cines
  - Digitalis-Bull (Intel Nehalem, GPU), Mésocentre CIMENT, INRIA RA
- Programmation de ces architectures
  - Comment hybrider les applications?

Quelles démarches pour hybrider?

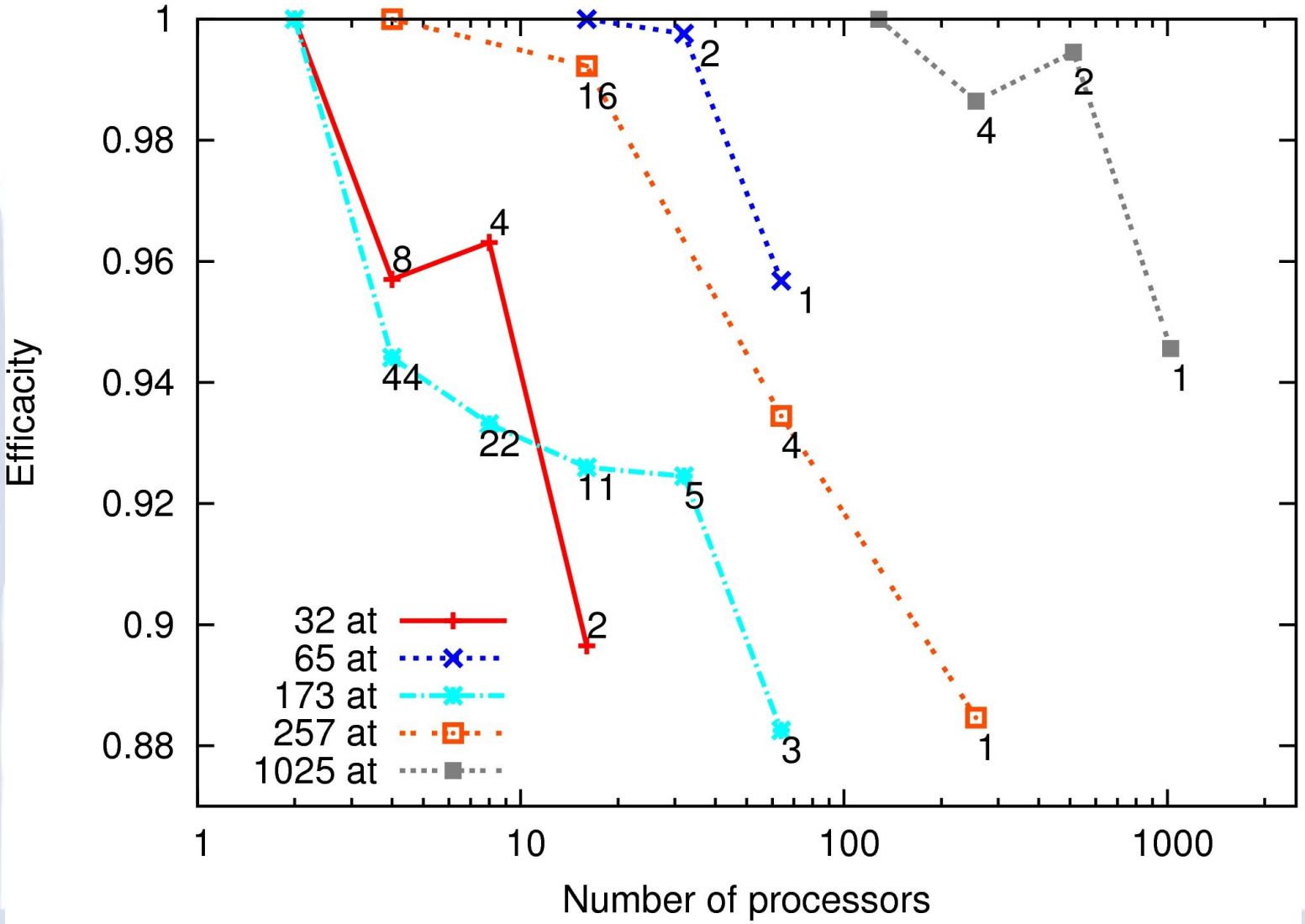
- Descendante (Mescal, Grenoble)
  1) Guidée par les applications (BigDFT, Ondes3D, JivaroD,...)
  2) Extension d'intergiciels et d'exécutifs (S_GPU, Mai, Minas)
- Ascendante (Runtime, Bordeaux)
  1) Guidée par les bonnes abstractions pour la performance dans les intergiciels et exécutifs (Marcel, PM2, StarPU,...)
  2) Adaptation des bibliothèques et applications



BigDFT: une application à hybrider

- Caractéristiques «informatiques »
  - Passage à l'échelle sur des grappes homogènes
    - Efficacité de 90% sur des grappes à 1000 coeurs
  - Code Fortran-90 (environ 70000 lignes), MPI
  - Analyse de performance, profiling du code (gprof, vtune,...)
    - Produit de convolution (~70% du temps de calcul)
    - Fonctions BLAS (~10% du temps de calclul)
  - Développé dans le cadre d'un projet européen
    - Plusieurs développeurs dans différents pays
  - Utilisé en mode production
    - Par des utilisateurs (chercheurs en physique)

BigDFT: passage à l'échelle sur des architectures homogènes



Objectifs pour BigDFT

- Passage à l'échelle sur des architectures de grappes hybrides
  - Va t'on conserver les 90% sur 1000 coeurs avec des accélérateurs?
  - Quels gains apportent les accélérateurs avec de telles plates-formes?
- Consommation énergétique et grappes hybrides
  - Mesure de consommation
  - Prédiction de consommation

BigDFT hybride: contraintes

- Minimiser l'intrusion dans le code source
  - Ne pas réécrire BigDFT en C
    - 70000 lignes...
    - Plusieurs développeurs physiciens dans différents pays
- Version hybride
  - Aujourd'hui: Fortran+MPI+GPU
  - Demain: Fortran+MPI+Threads+GPU

Programmation des grappes hybrides

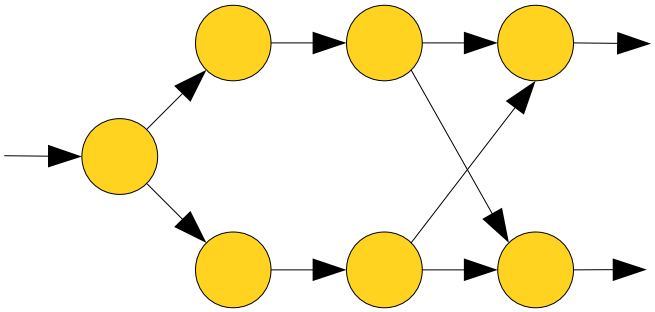
Programmation parallèle hybride

- Grappes de machines multiprocesseurs
  - Problème connu depuis le milieu des années 90
  - Faire cohabiter du passage de message (MPI) avec des threads en mémoire partagée (Posix, OpenMP, TBB,..)
  - De nombreux projets de recherche (Nexus, Chant, PM2, Athapascan, MPC,...)
  - Aujourd'hui, pas de modèle et standard clairement adopté
- Grappes de calcul hybrides (avec accélérateurs)
  - Technologies très changeantes (GPU, Intel Larrabee,...)
  - Espaces mémoire disjoints
  - Interface spécifiques (Cuda, OpenCL, Cell SDK,...)

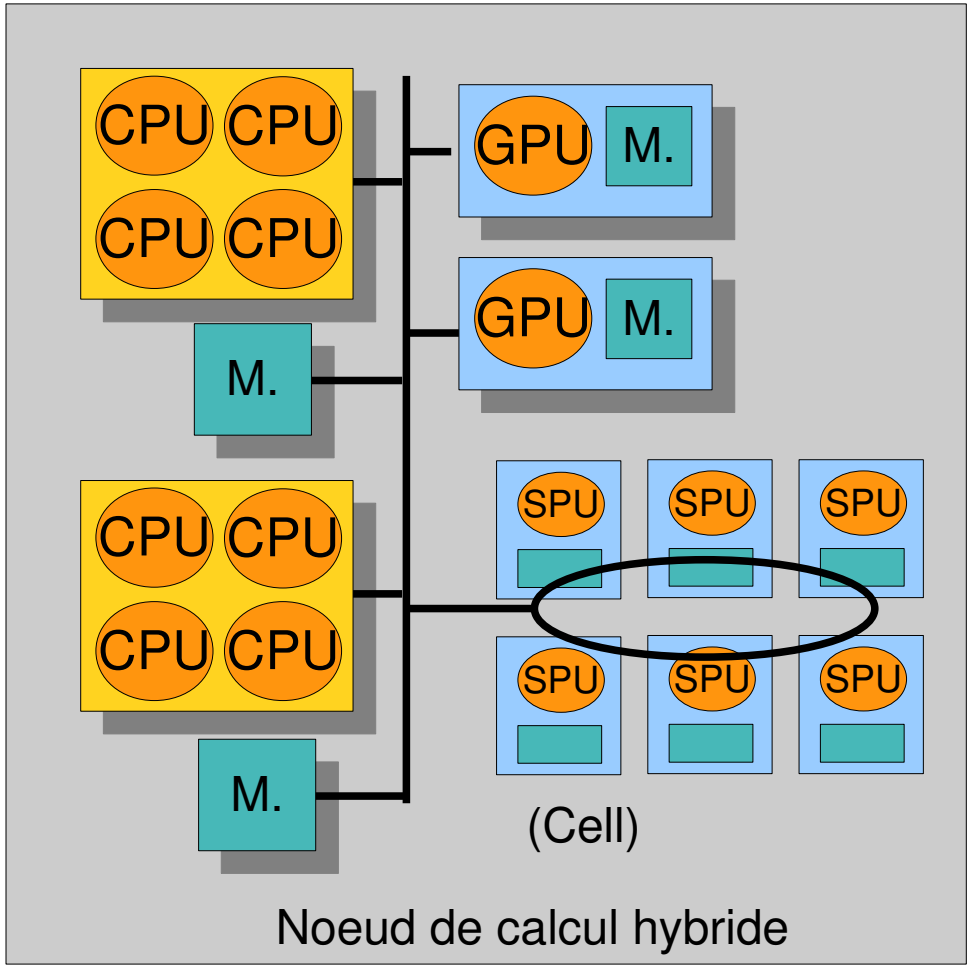
Le modèle d'exécution StarPU (1)

EP INRIA Runtime (Bordeaux Sud Ouest)

Application: ensemble de tâches avec des dépendances



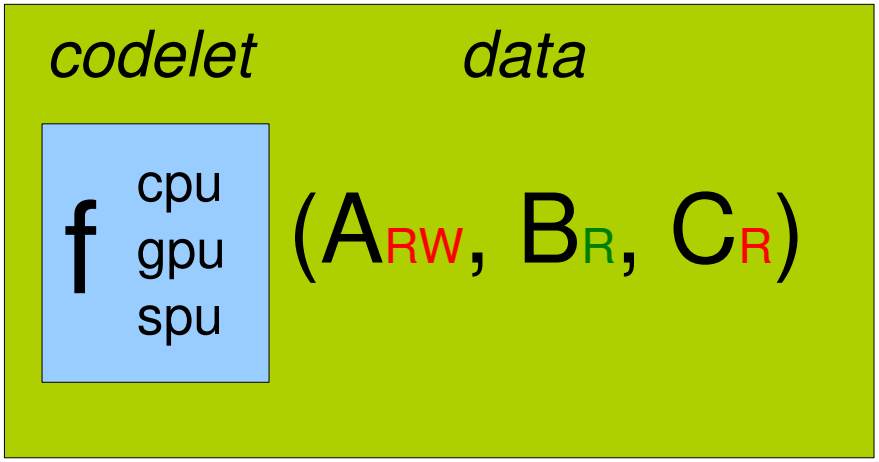
- Comment déployer cette application?
  - ✗ Différents types de processeurs (CPU Intel, GPU NVIDIA; IBM Cell)
  - ✗ Interfaces de programmation spécifiques à l'architecture
- StarPU; un modèle unifié d'exécution
  - ✔ Programmabilité
  - ✔ Portabilité des performances



Le modèle d'exécution StarPU (2)

EP INRIA Runtime (Bordeaux Sud Ouest)

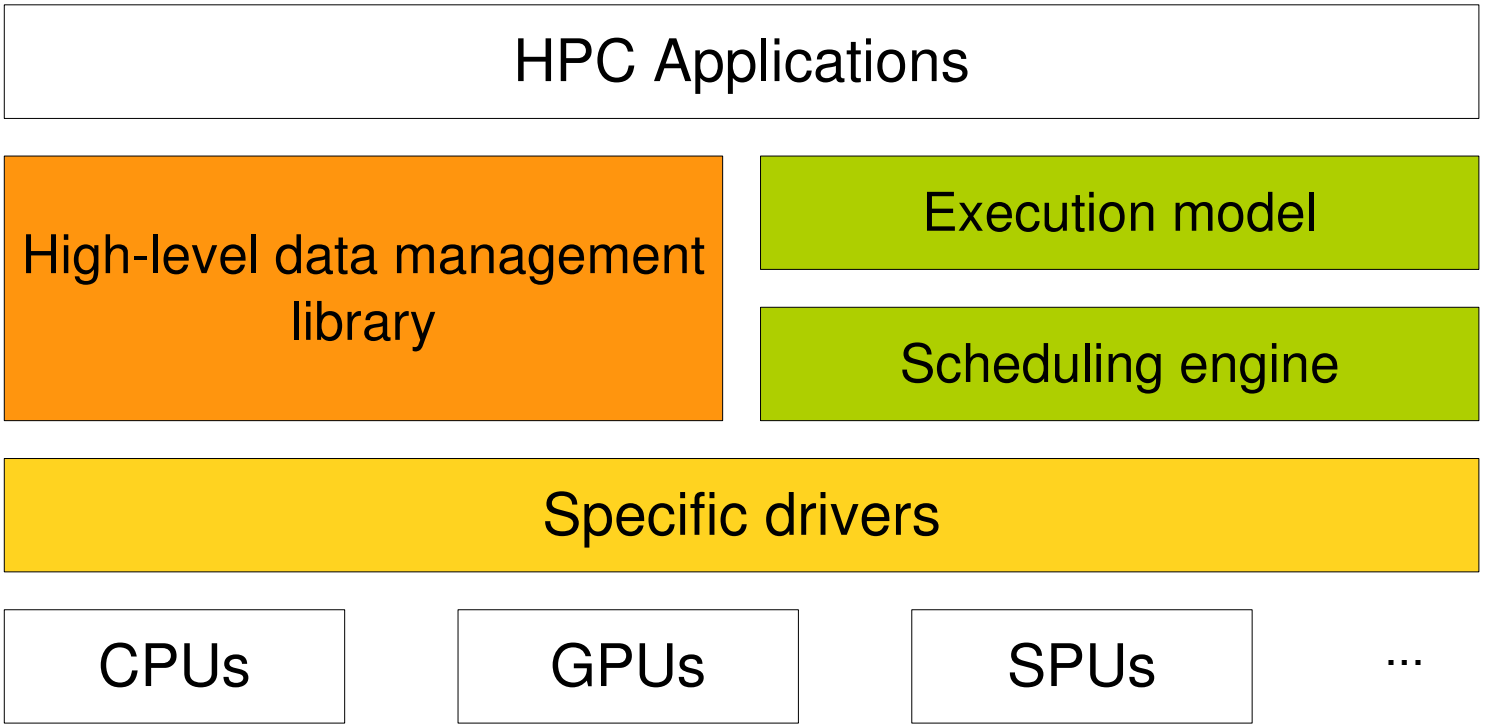
- Structure d'une tâche StarPU
  - Asynchrone
  - Codelet
    - Une implémentation par type de processeur
  - Fonction d'accès explicite aux données
- Gestion des données de l'application
  - Maintien de la cohérence
  - Interface de haut
    - Structures de données élémentaires



- Qui génère les codelets?
  - Pas StarPU !
  - Environnements compilants
    - CellSs (Barcelone, BSC)
    - HMPP (CAPS)
    - ...

Le modèle d'exécution StarPU (3)

EP INRIA Runtime (Bordeaux Sud Ouest)



Mastering CPUs, GPUs, SPUs … ***PUs**

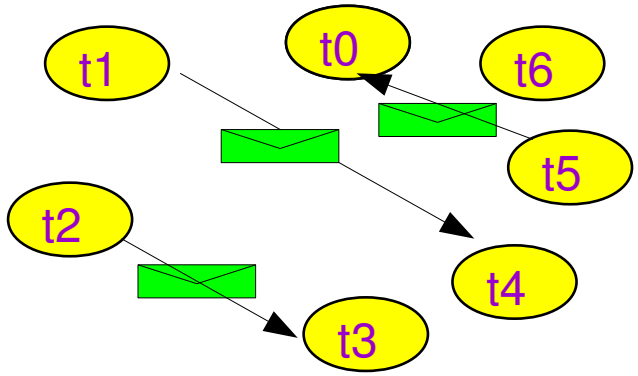
Premières conclusions sur StarPU

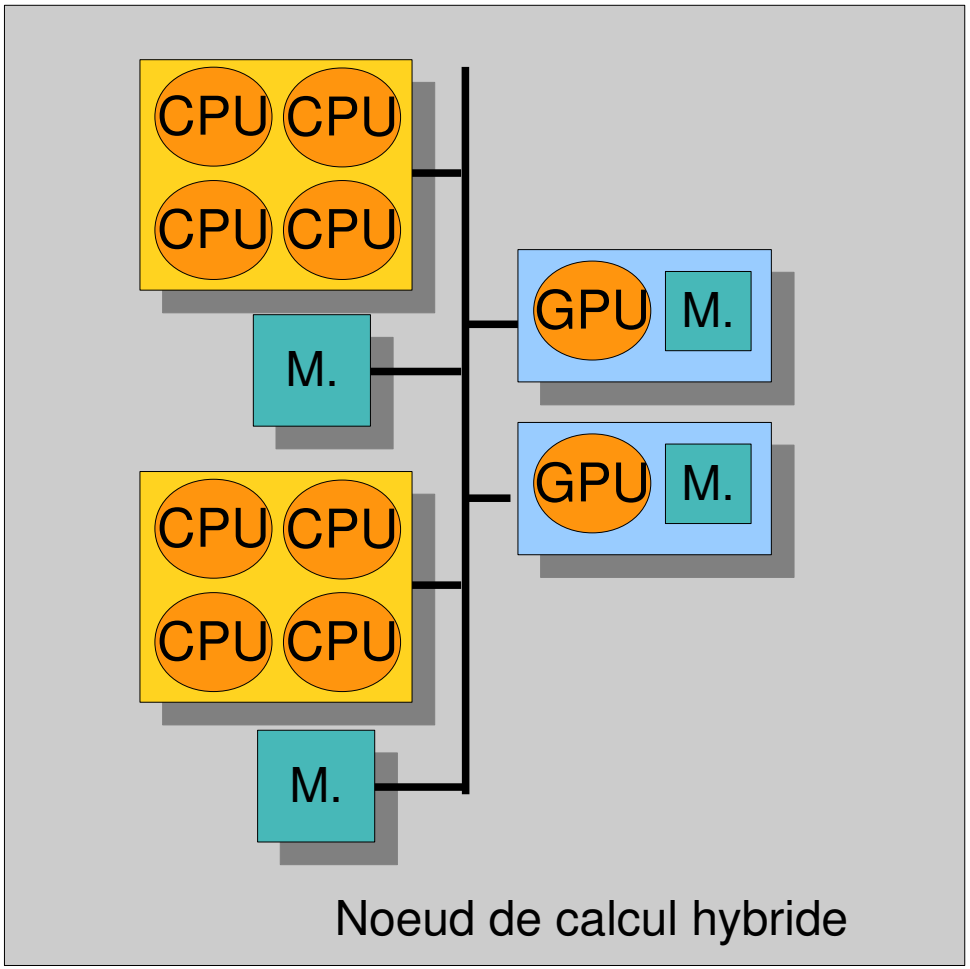
- Modèle d'exécution très intéressant
  - Exécution hybride
  - Banalisation des ressources de calcul
- Exécution asynchrone des tâches
  - Programmeur ne contrôle pas
- Effort assez important au niveau des codes (bibliothèques, applications)
- Pas de support réseau rapide
  - A l'intérieur d'un noeud
  - MPI+StarPU?

Le modèle d'exécution S_GPU (1)

Bull, CEA INAC, EP INRIA Mescal (Grenoble Rhône-Alpes)

Applications: Programme MPI (tâches communicantes)





- Comment les GPU vont être utilisés par les tâches MPI?

✗ Deux types de processeurs (CPU Intel, GPU NVIDIA)

✗ Interfaces de programmation spécifiques à l'architecture

- S_GPU: GPU virtuels

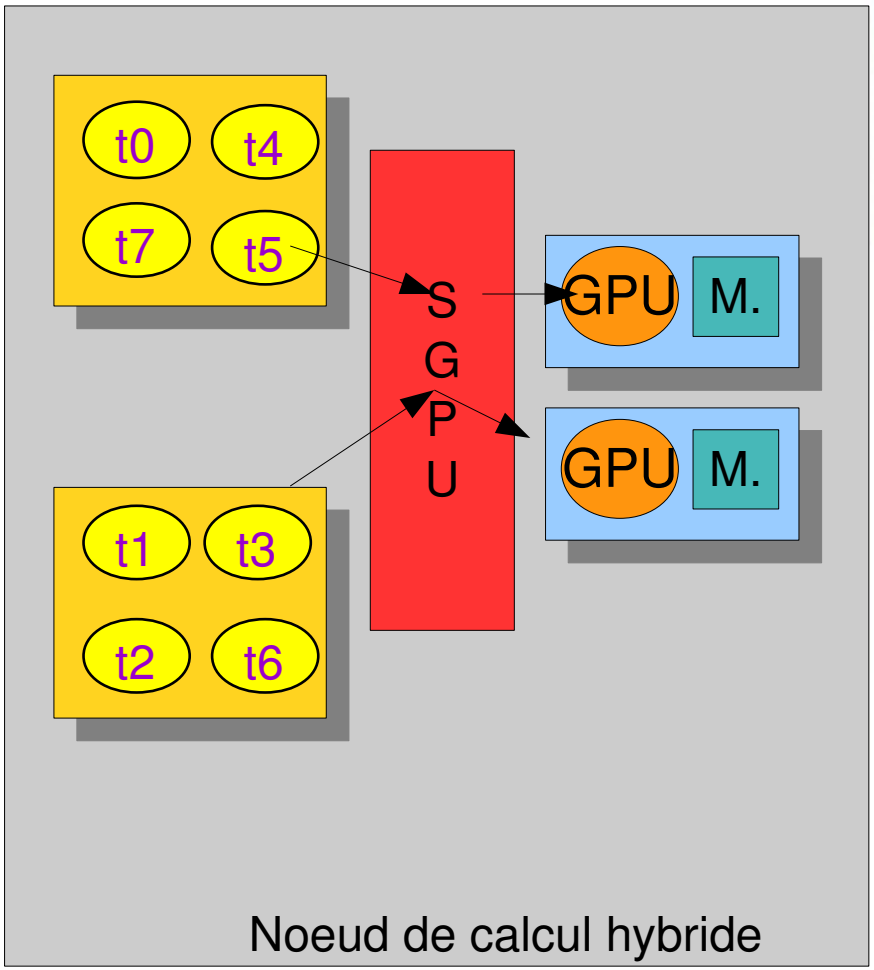
✔ Partager l'utilisation des GPUs

✔ Recouvrir les transferts mémoire

Le modèle d'exécution S_GPU (2)

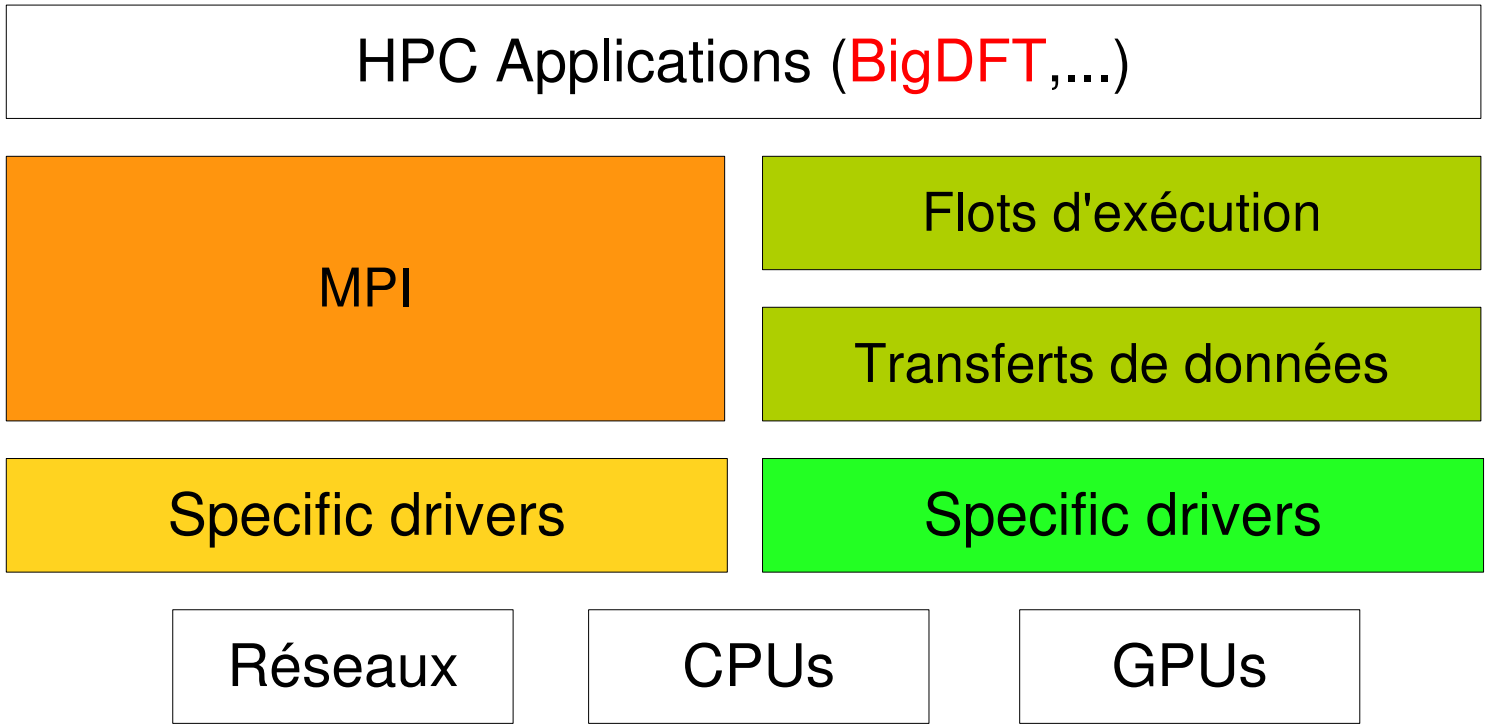
Bull, CEA INAC, EP INRIA Mescal (Grenoble Rhône-Alpes)

- Flots (processus)
  - Synchrone
  - Ensemble de données
  - Noyau de calcul
  - Exécution prise en charge par S_GPU
- Gestion des données de l'application
  - Explicite (programmeur)
  - Recouvrement transfert mémoire et calcusl sur GPU



Le modèle d'exécution S_GPU (3)

Bull, CEA INAC, EP INRIA Mescal (Grenoble Rhône-Alpes)



Mastering GPUs from MPI Applications

Premières conclusions sur S_GPU

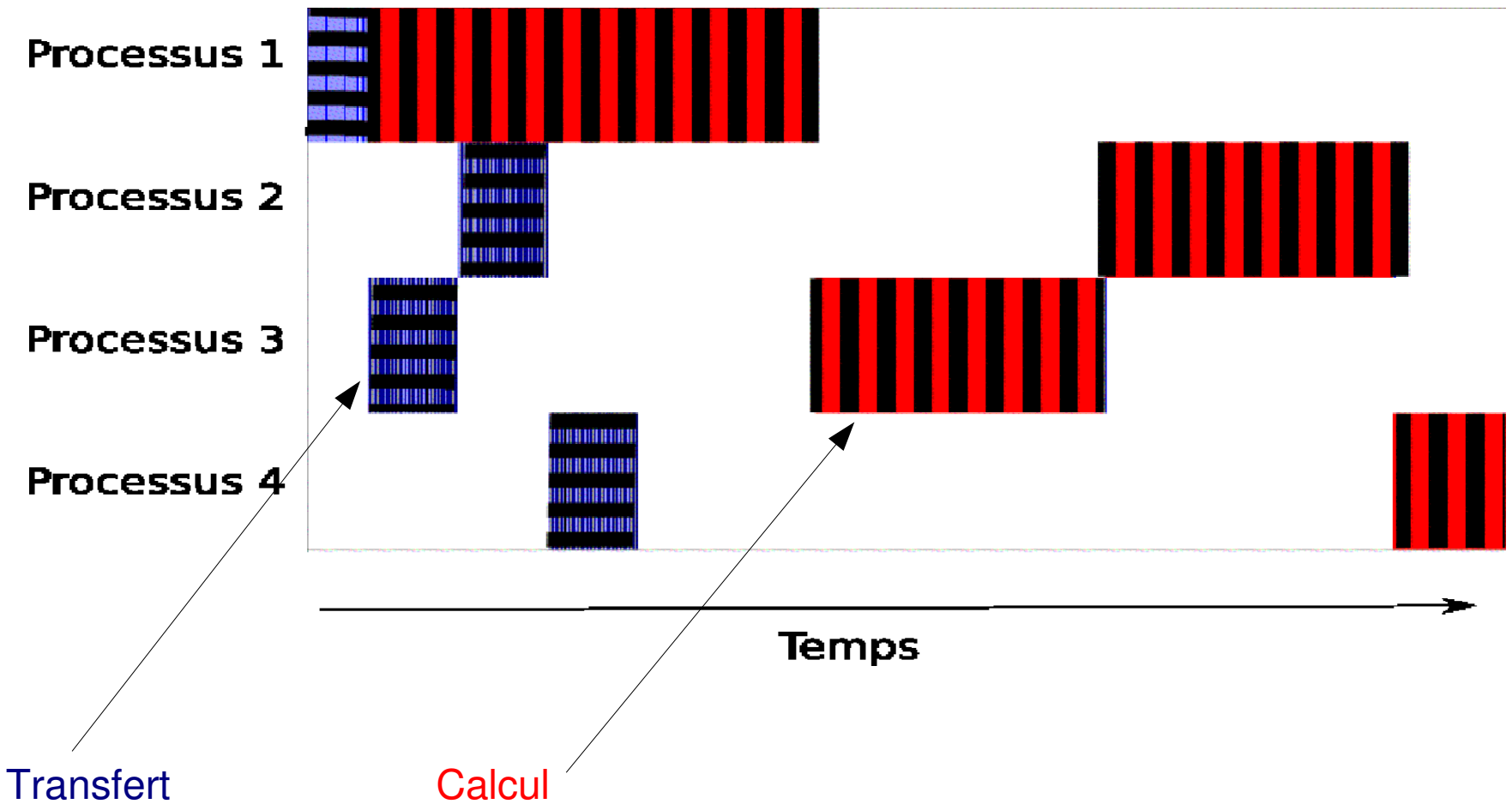
- Modèle d'exécution très simple
  - Synchrone
  - Accélérateur vu comme un co-processeur
- Exécution synchrone des flots
  - Programmeur contrôle tout ce qui se passe
- Intégration assez facile dans les codesparallèles (bibliothèques, applications)
  - Intrusion limitée
- Gestion des données un peu fastidieuse
  - StarPU, ...?

Exploitation des grappes hybrides

Exploitation des grappes hybrides

- Coopérations entre les CPU et les GPU
  - Intégration à S_GPU de mécanismes de traçage
  - Visualisation des traces en post-mortem (next slide)
  - Recouvrement Transferts calcul
- Consommation énergétique
  - Mesure de la consommation énergétique sur BigDFT/S_GPU
  - Comparaison sans GPU et avec GPU
- Gestionnaire de ressources
  - Intégration de la ressource GPU dans OAR (CIMENT, Grid5k)
  - Prédiction de performance et de consommation

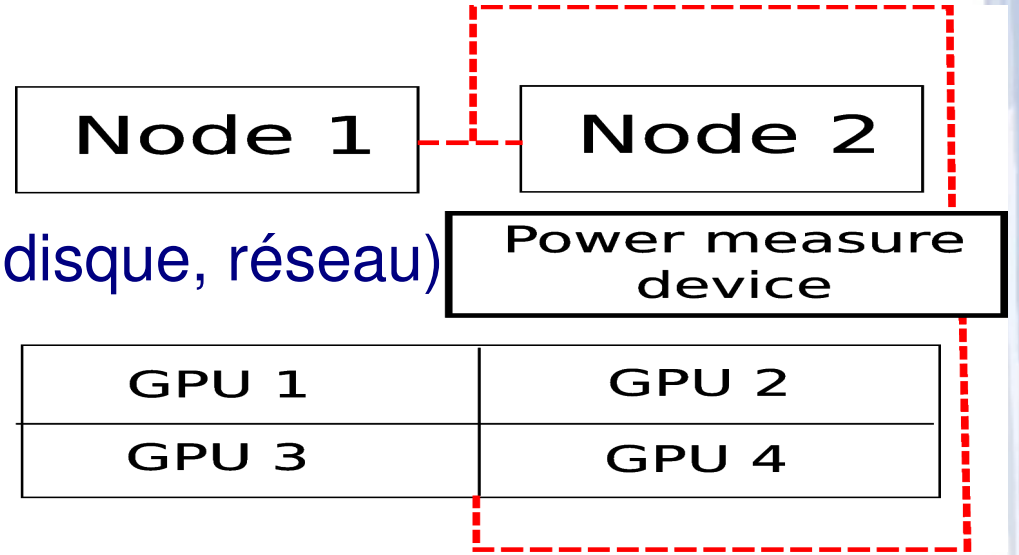
Trace d'exécution de BigDFT et S_GPU



1 GPU partagé entre 4 tâches processus MPI (cœurs)

Mesure de la consommation énergétique

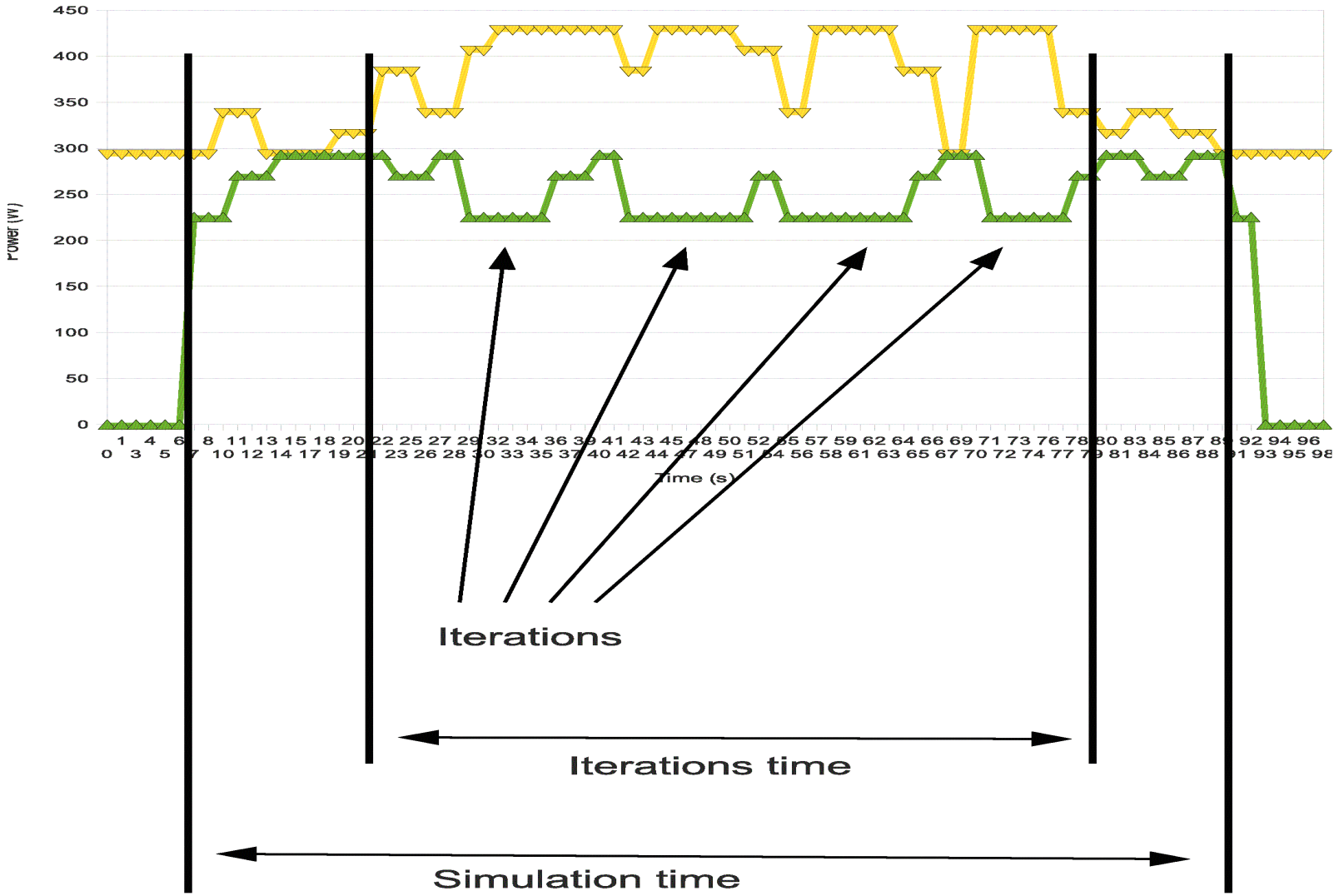
- Plate-forme de mesure
  - Bull Echirolles, X. Vigouroux
  - Cosommation Noeud (CPU, disque, réseau)
  - Consommation GPU



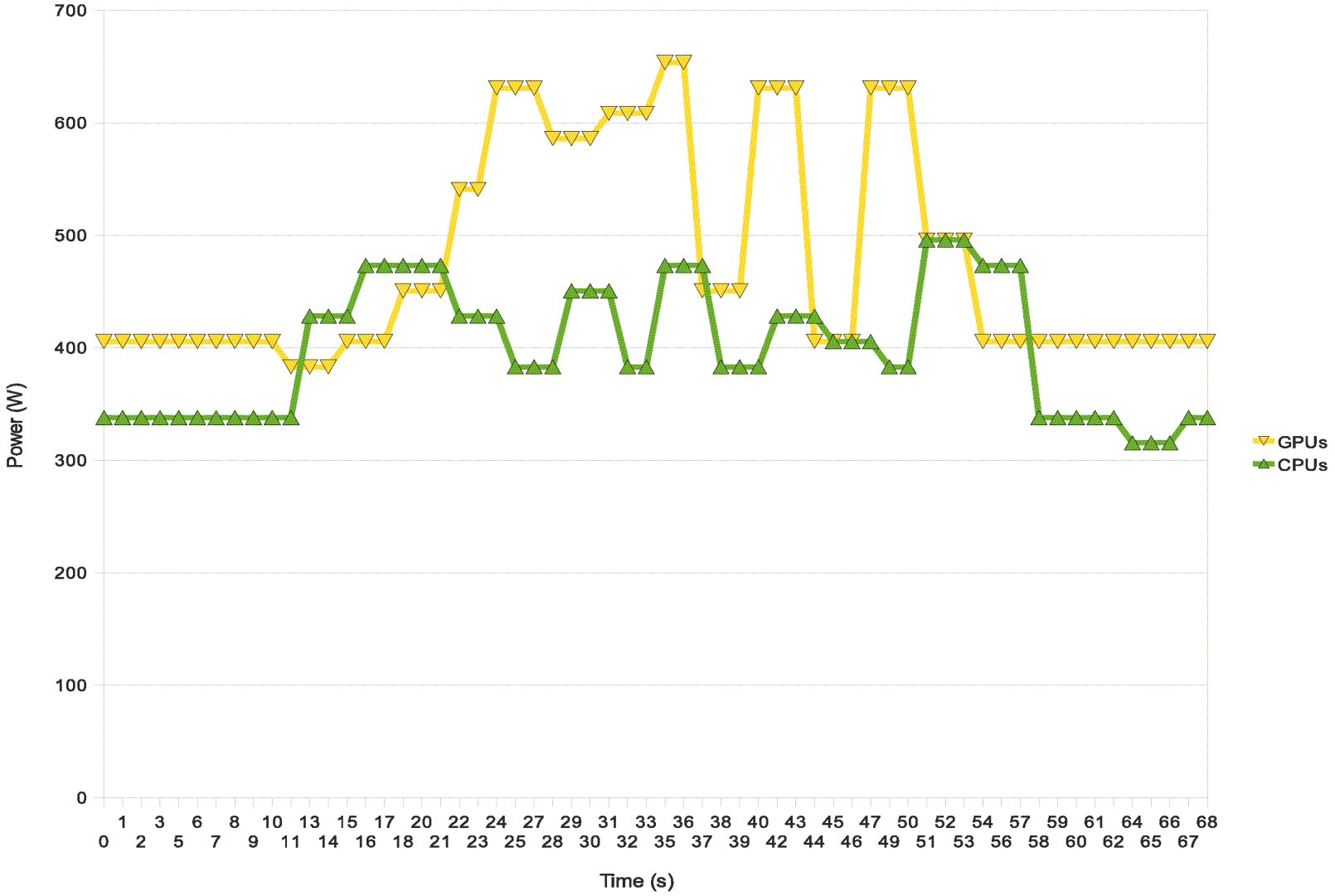
- Comportement énergétique de BigDFT et BigDFT/S_GPU
  - Avec le même jeu de données

Appareil de mesure de la consommation énergétique

Consommation énergétique 8 cœurs / 2 GPUs



Consommation énergétique 16 cœurs / 4 GPUs



Consommation (kJ)

	GPU	CPU	TOT
8 MPIs, 0 GPU (kJ)	60	59	119
8 MPIs, 2 GPUs (kJ)	28	19	47
16 MPIs, 4 GPUs (kJ)	28	23	51

- L'utilisation efficace des GPUs permet une diminution de la consommation énergétique
- Un GPU inutilisé augmente significativement la consommation totale

Conclusion

- S_GPU : Virtualisation et partage des GPU entre plusieurs cœurs généralistes
- Intégration de S_GPU à une application « réelle » de calcul scientifique
- Comparaison à d'autres approches de partage des GPUs
- Utilisation avec succès de S_GPU sur des grappes hybrides (Titane et Iblis)
- Évaluation de l'impact des GPUs sur la consommation énergétique

Perspectives

- Utilisation de S_GPU sur d'autres codes applicatifs
- Opérations asynchrones de S_GPU pour une utilisation simultanée des GPUs et CPUs
- Modélisation et prédiction des performances et de la consommation énergétique
- Intégration de modèles prédictifs dans des gestionnaires de ressources de grappes hybrides

Projets collaboratifs

- Projet ANR ProHMPT 2009-2011
  - INRIA Runtime, Bull, Caps Entreprise, CEA Inac, Cea Dam/Cesta, UVSQ, INRIA Mescal
- Projet RTRA Nanosciences 2010-2012
  - D'autres applications des nanosciences
  - CEA Inac, Institut Néel, INPG, INRIA Mescal
- Projet européen Prace
  - BigDFT: application benchmark (WP8?)

QUESTIONS ?

- Prix Bull Joseph Fourier Genci (juin 2009)
  - L. Genovese + BigDFT + GPU