Université de Nice Sophia-Antipolis
Laboratoire I3S

D.E.A Informatique

Accès à PVM par l'intermédiaire du langage ADA dans un objectif de programmation conviviale d'applications parallèles

Nathalie FURMENTO

29 juin 1994

Responsables :

Mme. Françoise BAUDE

M. Daniel LAFAYE de MICHEAUX

Présentation du sujet

- Pourquoi PVM ?
  - Récupérer puissance non-utilisée dans les réseaux de station de travail,
  - S'adapter aux nouvelles tendances d'architecture parallèle : multi-processeurs Unix

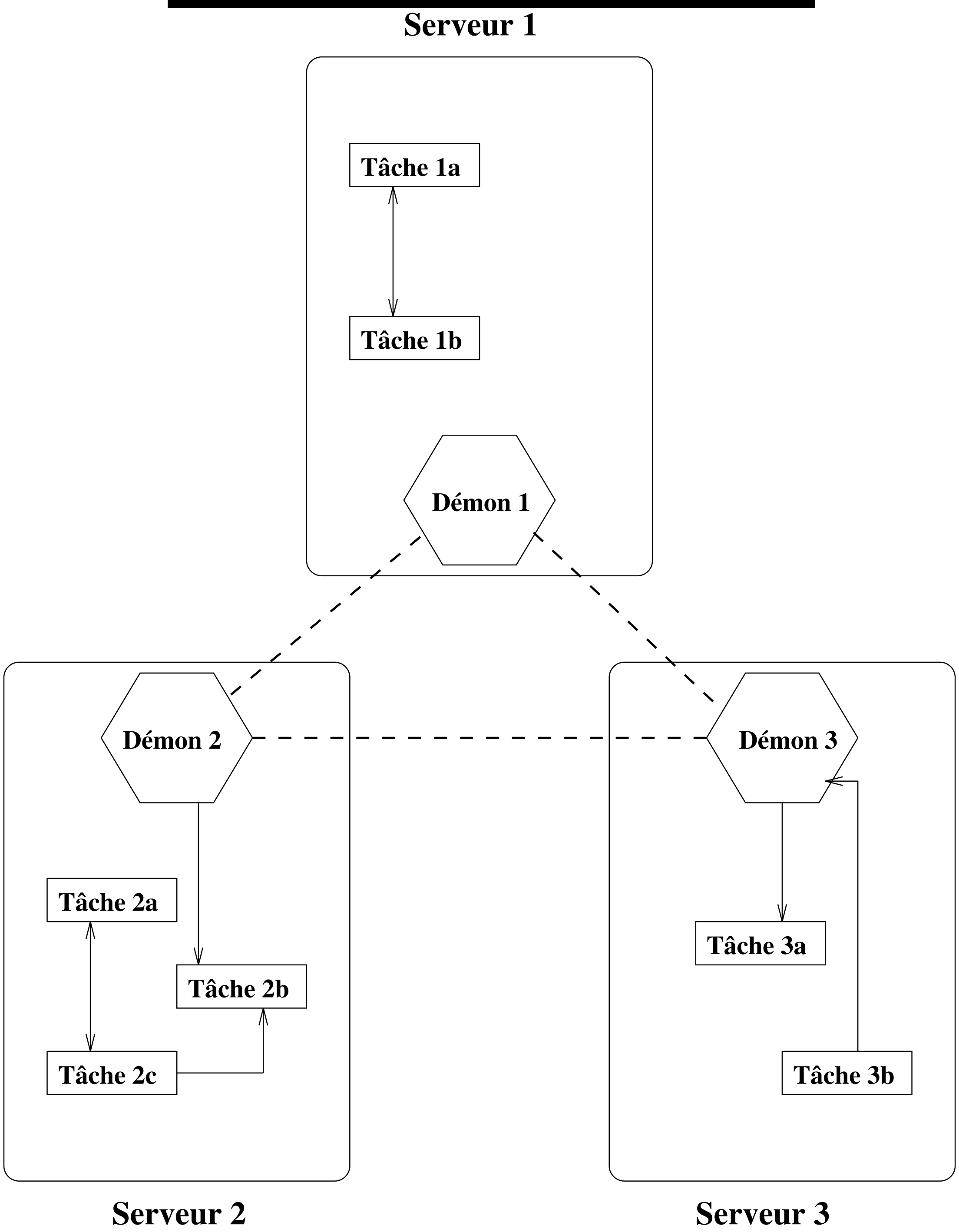
- Pourquoi PVM en ADA ?
  - PVM 3 est disponible en C et Fortran 77, langages de bas niveau mal adaptés au génie logiciel.

  ⇒ Interfacer PVM 3 en ADA

Plan de la soutenance

– Présentation de PVM 3,
– La librairie Ada parallèle,
– Les communications globales,
– Evaluation des temps de communication des primitives PVM 3,
– Un exemple d'application

Parallel **V**irtual **M**achine (1)



Parallel **V**irtual **M**achine (2)

Les fonctions de la librairie

- Gestion de la machine virtuelle
- Gestion des tâches
- Les communications (signaux ou messages)
- Gestion des groupes
- Gestion des erreurs

Message **P**assing **I**nterface

MPI : Norme d'un système d'échange de messages

- Efficace
- Portable sur plusieurs types de machines différentes
- Facilement utilisable

Inconvénients de MPI :

- Syntaxe trop proche de C
- Notion non conforme à la philosophie Ada
- Fonctionnalités trop nombreuses

L'interface ADA-PVM 3 (1)

Pourquoi ADA ?

– Modulaire
– Typage fort
– Générique
– Utilisé par beaucoup de développeurs dans le cadre numérique

⇒ Association d'une procédure ou fonction Ada à chaque primitive PVM 3.

L'interface ADA-PVM 3 (2)

Implémentation

- Caractéristiques :
  - Généricité et surcharge

C		Ada
une fonction par type de donnée	$\rightarrow$	Package générique

  - Gestion des erreurs

C		Ada
code retour	$\rightarrow$	exception

  - Contrôles avancés à la compilation
- Problèmes de transmission des paramètres :
  - Les chaînes,
  - Allocation mémoire,
  - Pointeur C $\neq$ access Ada,
  - Structures

Les communications globales

– Synchronisation : déjà implementée en PVM 3.
– Diffusion : solution de PVM 3 ou technique de l'arbre ou de l'étoile.
– Distribution : solution utilisée dans les architectures en forme d'étoile.
– Rassemblement : combinaison centralisée où l'opérateur est une fonction de concaténation.
– Combinaison : Algorithme d'élection de leader présent dans les systèmes distribués.

– Modèle de durée d'une communication élémentaire

$\Rightarrow$ Choix de l'algorithme

Evaluation des temps de communication élémentaire des primitives PVM 3 (1)

Communication entre deux processeurs

– Modèle classique :

$$t = \beta + L\tau$$

– Réseaux de stations :

$$\begin{array}{rrcl} & t & = & \beta_e + L\tau + \beta_r \\ & \beta_e & = & \beta_e^{'} + L\tau_e^{'} \\ & \beta_r & = & \beta_r^{'} + L\tau_r^{'} \\ \Rightarrow & t & = & (\beta_e^{'} + \beta_r^{'}) + (\tau + \tau_e^{'} + \tau_r^{'})L \end{array}$$

– Messages de petite taille (< 4096) :

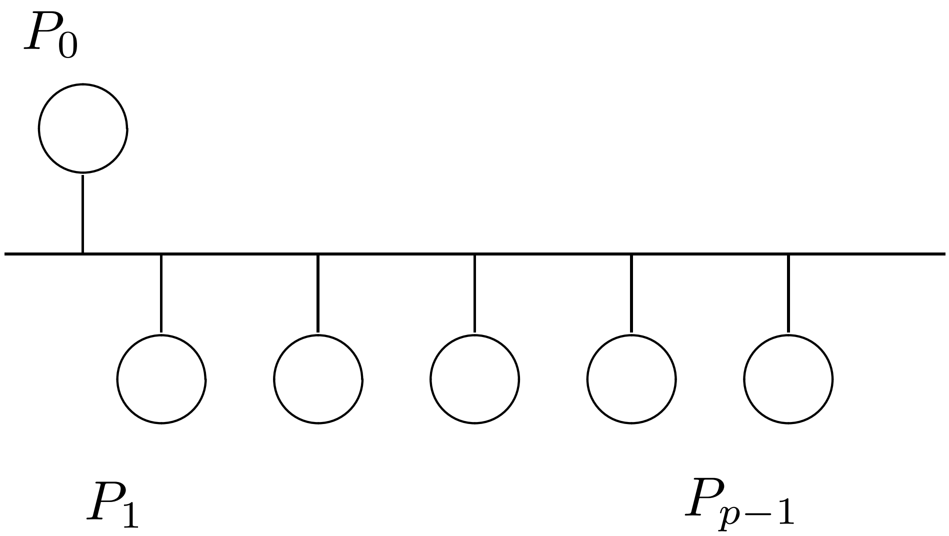
$$t = 0,01247 + 4,82.10^{-6}L$$

– Messages de grande taille :

$$t = 0,0202 + 5,2.10^{-6}L$$

Evaluation des temps de communication élémentaire des primitives PVM 3 (2)

Communication entre p processeurs



– Messages de petite taille :

$$T = 2\beta_r^{'} + 2(\tau + \tau_r^{'})L + \beta_e^{'}p + \tau_e^{'}Lp$$

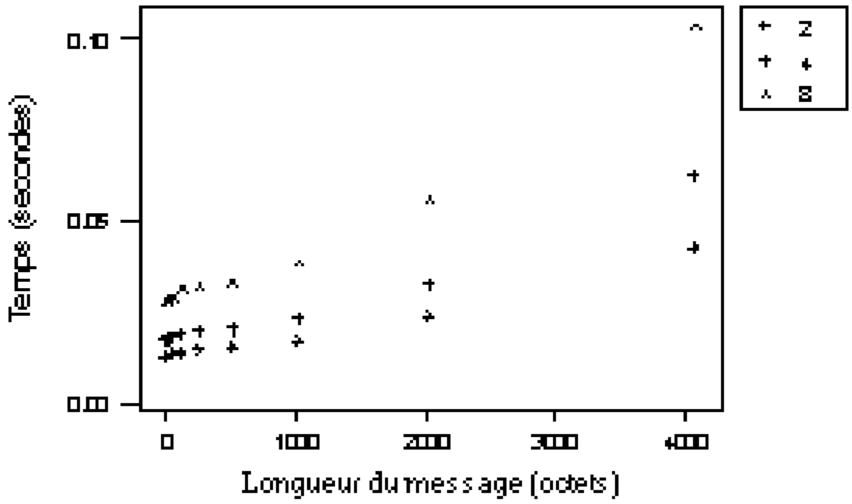
– Messages de grande taille :

$$T = 2(\beta_r^{'} + \beta_e^{'}) + 2(\tau_r^{'} + \tau_e^{'})L + \tau Lp$$

Evaluation des temps de communication élémentaire des primitives PVM 3 (3)

Résultats petits messages

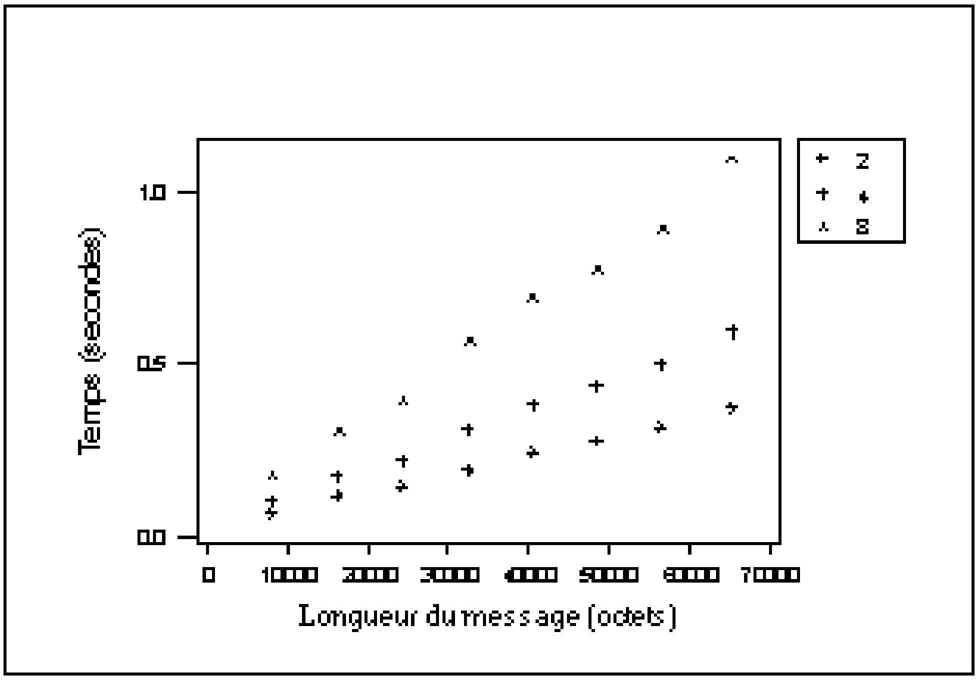
$$t \quad = \quad 2\beta_r^{'} + 2(\tau + \tau_r^{'})L + \beta_e^{'}p + \tau_e^{'}Lp$$



$$\begin{aligned} t \quad = \quad & 0,00721 + 0,00000212L + \\ & 0,00263p + 0,00000135Lp \end{aligned}$$

Evaluation des temps de communication élémentaire des primitives PVM 3 (4) Résultats messages longs

$$t \quad = \quad 2(\beta_r^{'} + \beta_e^{'}) + 2(\tau_r^{'} + \tau_e^{'})L + \tau Lp$$



$$t \quad = \quad 0,00292 + 0,00001L + 0,00002Lp$$

Un exemple d'application (1)

Objectif

- Reconstruction d'images astronomiques en utilisant les propriétés des *statistiques d'ordre supérieur*.
- K signaux ou images bruitées d'une étoile
  - Calcul des FFT de ces images
  - Calcul des bispectres et des modules

  $\Rightarrow$ Moyenne des résultats obtenus sur chaque image, et récursion de Weigelt.

  $\Rightarrow$ Reconstruction de l'image initiale en calculant la FFT inverse.

Un exemple d'application (2)

Réalisation

- Le problème : combinaison centralisée

$$T(F(x_1) \otimes \ldots \otimes F(x_i) \otimes \ldots \otimes F(x_K))$$

  - Image $(x_i) \rightarrow$ bispectre + module $(F(x_i))$
  - $\otimes$: addition sur (bispectre, module)
  - T : normalisation
- Choix d'implémentation :
  - Même travail sur chaque processeur mais disponibilité et/ou vitesse différente

  $\Rightarrow$ Volume de données à traiter différent selon les processeurs

  $\Rightarrow$ Equilibrage dynamique de charges

  $\Rightarrow$ Structure maître-esclave

Conclusion

- **Objectifs du stage atteints :**
  - Définition d'une librairie destinée à l'algorithmique parallèle, pour … l'écriture d'applications réparties
  - Démonstration de son utilité et de son efficacité
- **Travaux futurs :**
  - Amélioration des mesures de temps de communication,
  - Implémentation des communications globales (⇒ programmation SPMD)
  - Recouvrement temps de calcul / communications.