Mise en œuvre et étude de performance d'algorithmes itératifs parallèles sur diverses architectures Application à l'optimisation et la commande

Mohamed JARRAYA
LAAS-CNRS, Toulouse

Plan de l'exposé

I. Introduction

II. Algorithmes itératifs parallèles

III. Mise en œuvre

IV. Applications et étude de performance

V. Conclusions et perspectives

I. Introduction

- **Applications industrielles et grands projets scientifiques nécessitent des volumes de calcul très importants.**
- **Utilisation des calculateurs parallèles à hautes performances.**
- **CRAY T3E et Origin 2000 à 256 processeurs.**
- **Projets Européens et Américains à 1024 processeurs.**
- **Autres solutions : les réseaux hauts débits ou ”les clusters”.**

Introduction

- **Méthodes numériques parallèles performantes.**
- **Maximiser l'efficacité de la mise en œuvre.**
- **Equilibrage de charge ?**
- **Synchronisation ou non du calcul ?**
- **Fréquence optimale des échanges de données ?**

II. Algorithmes itératifs parallèles

Les itérations asynchrones standards

Soit F une application de R^n dans R^n. Considérons le problème de point fixe:

$$x^* = F(x^*). \tag{1}$$

Définition 1 : (cf. Chazan et Miranker 69, Miellou 75 et Baudet 78) Une itération **asynchrone** associée à $F : \prod_{i=1}^{\alpha} R^{n_i} \to \prod_{i=1}^{\alpha} R^{n_i}$ et à $x(0) \in \prod_{i=1}^{\alpha} R^{n_i}$ est une **suite** $\{x(k)\}$ **de vecteurs** de $\prod_{i=1}^{\alpha} R^{n_i}$ définie par :

$$\begin{cases} x_i(k) = F_i(x_1(\tau_1(k)), ..., x_\alpha(\tau_\alpha(k))), \text{ si } i \in S(k), \\ x_i(k) = x_i(k-1), \text{ si } i \notin S(k). \end{cases} \tag{2}$$

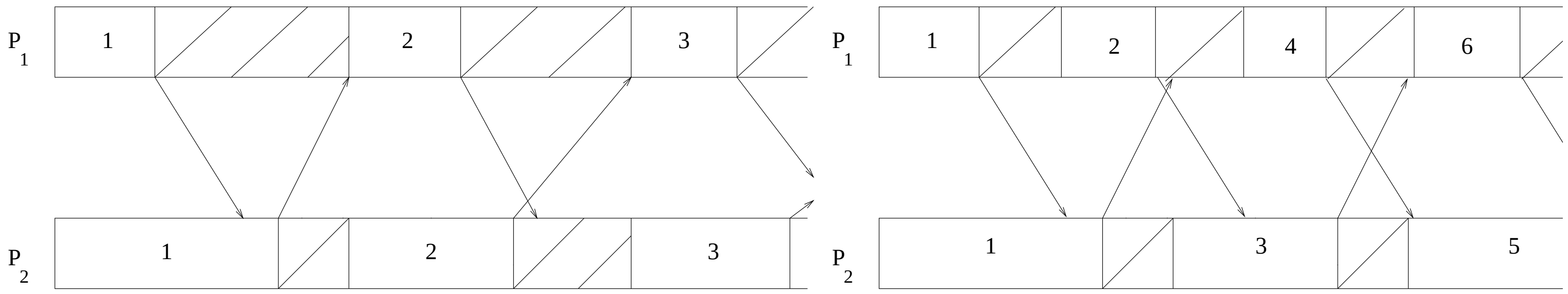
Algorithmes itératifs parallèles

(a) $S(k) \in \{1, ..., \alpha\}$ **et** $S(k)$ **non vide,**

(b) i **apparaît un nombre infini de fois dans** $\{S(k)\}$**,**

(c) $0 \leq \tau_i(k) \leq k-1, k = 1, 2, ...,$

(d) $\tau_i(k)$ **tend vers l'infini lorsque** k **tend vers l'infini.**



Itérations synchrones Itérations asynchones

Algorithmes itératifs parallèles

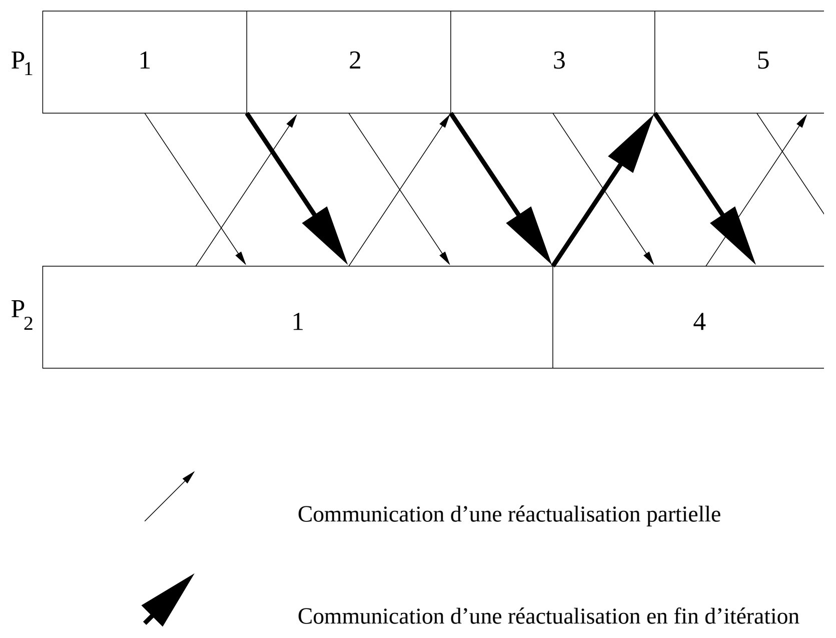
Les itérations asynchrones avec communication flexible

Définition 2 : (cf. Miellou, El Baz et Spiteri 98) Une **itération asynchrone avec communication flexible** relative à une application de point fixe **monotone croissante** sur $\prod_{i=1}^{\alpha} R^{n_i}$ et à une **sur solution** $x(0)$ satisfaisant : $F(x(0)) \geq x(0)$, est une **suite** $\{x(k)\}$, $k = 0, 1, ...,$ **de vecteurs** de $\prod_{i=1}^{\alpha} R^{n_i}$ telle que pour $i = 1, ..., \alpha$:

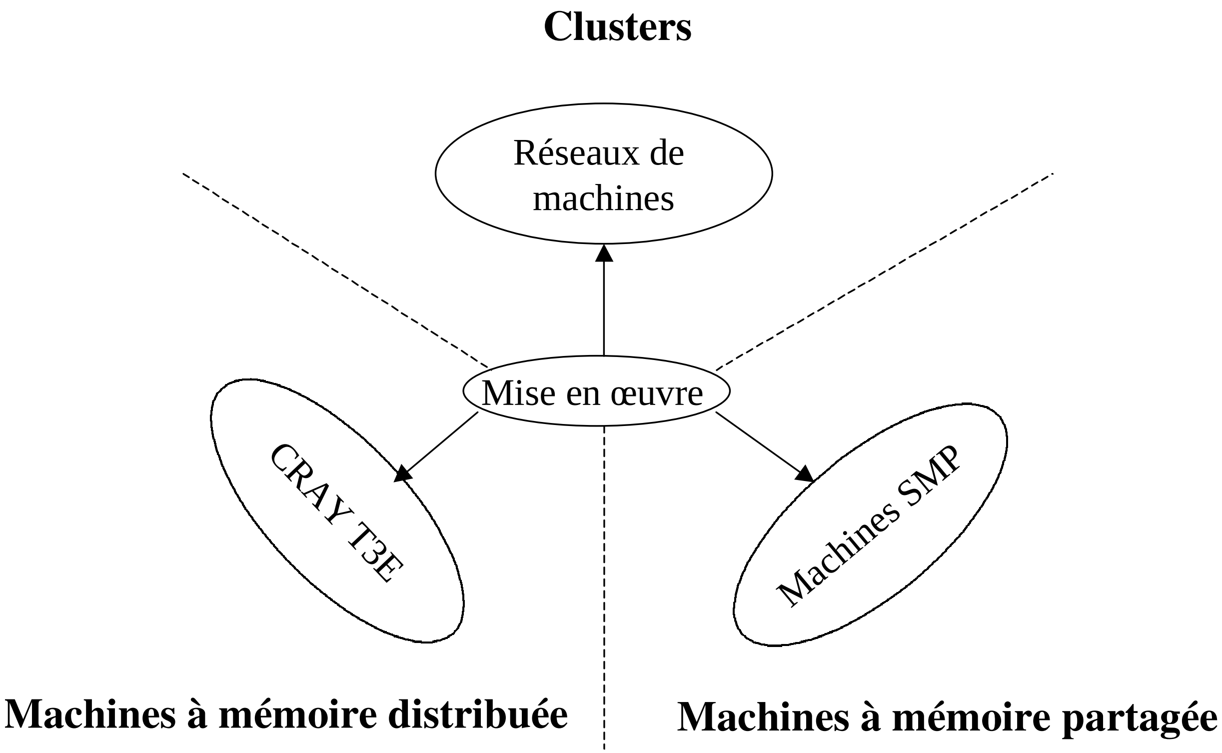
$$\begin{cases} x_i(k) = F_i(\tilde{x}(k)), \text{ si } i \in S(k), \\ x_i(k) = x_i(k-1), \text{ si } i \notin S(k). \end{cases} \tag{3}$$

$$\tilde{x}_l(k) \in \langle x_l(k-1), \min\{x_l(\tau_l(k)), \tilde{x}_l(q)\} \rangle_l, \;\; l = 1, ..., \alpha. \tag{4}$$

Algorithmes itératifs parallèles



III. Mise en œuvre



Mise en œuvre

Sur CRAY T3E

- **Utilisation du CRAY T3E de l'IDRIS-CNRS.**
- **Un supercalculateur à 256 processeurs $\alpha EV5$ dont chacun a une puissance de crête de 600Mfops/s.**
- **Son architecture est un tore 3D.**
- **Le réseau d'interconnexion est à très faible latence ($1\mu s$) et à forte bande passante (480 MOctets/s).**
- **Bibliothèques d'échange de données MPI et SHMEM.**

Mise en œuvre

- **MPI est un standard pour les applications parallèles.**
- **Communications point à point.**
- **Communications collectives.**
- **Définition de types dérivés.**
- **Echange de données synchrone (MPI_Send() et MPI_Recv()).**
- **Echange de données asynchrone (MPI_Isend() et MPI_Irecv()).**

Mise en œuvre

- **SHMEM propre à l'architecture CRAY.**
- **Communications point à point.**
- **Communications collectives.**
- **Copie mémoire à mémoire (shmem_put() ou shmem_get()).**
- **Mise en œuvre asynchrone naturelle.**

Mise en œuvre

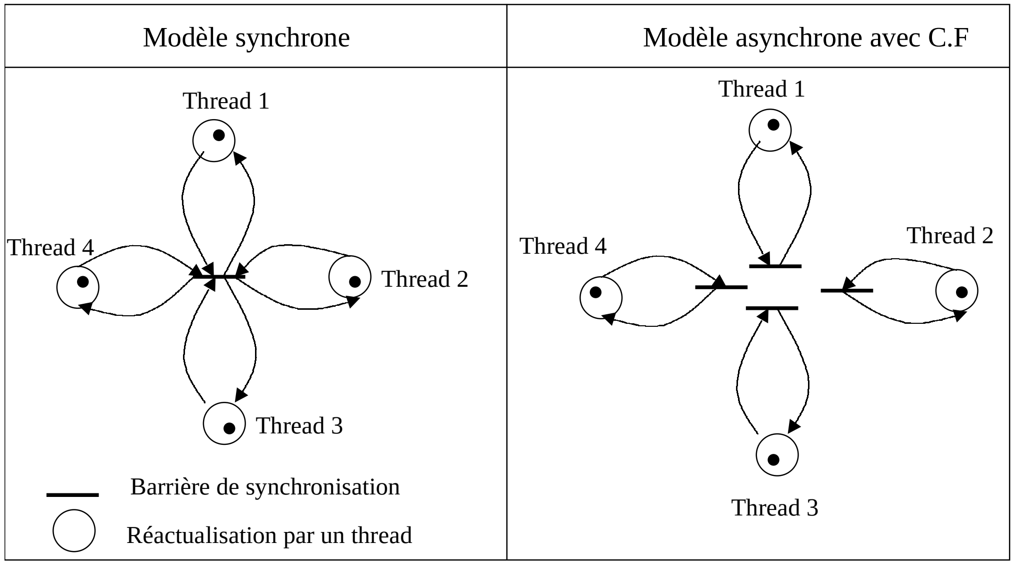
Sur machines SMP

- **Un PC quadriprocesseur Pentium Pro à 4x200MHz partageant une mémoire de 64MOctets.**
- **Utilisation de la bibliothèque de threads POSIX ou processus légers.**
- **Accès direct des processeurs à la mémoire commune par lecture et écriture.**

⇒ Contrôle de l'accès par des variables d'exclusion mutuelle.

Mise en œuvre

- **Mise en œuvre synchrone et asynchrone :**



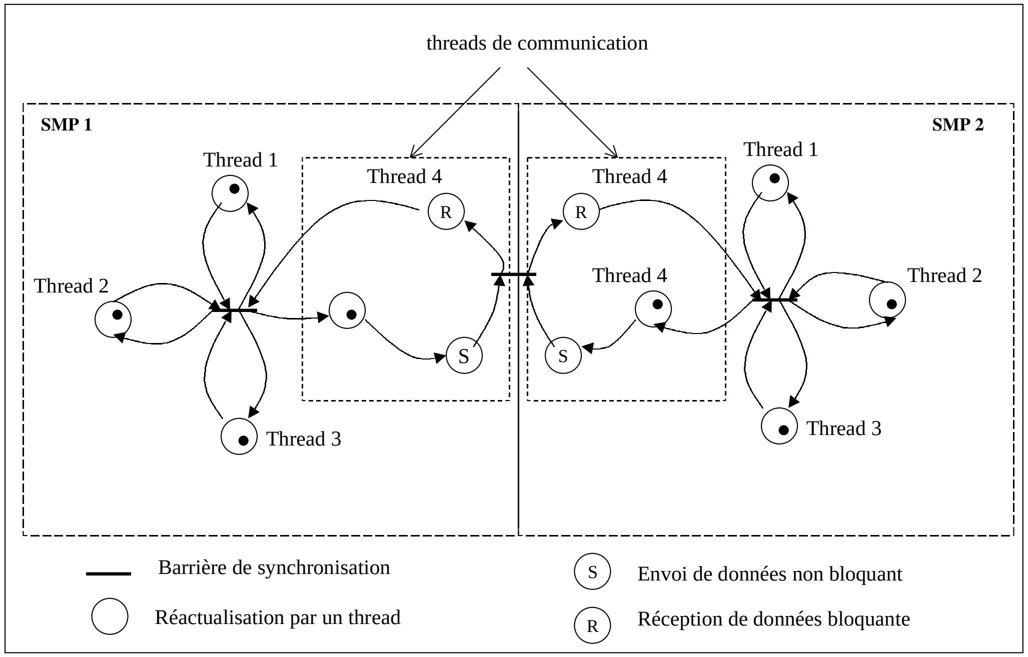
Mise en œuvre

Sur réseaux de machines

- **Un réseau de 4 PC bi-processeur Pentium II à 2x200MHz.**
- **un réseau Ethernet à 100MOctets/s et un réseau Myrinet à 1GOctets/s.**
- **Utilisation de MPICH et threads POSIX sur le réseau Ethernet.**
- **Utilisation de MPICH-gmm et threads POSIX sur le réseau Myrinet.**

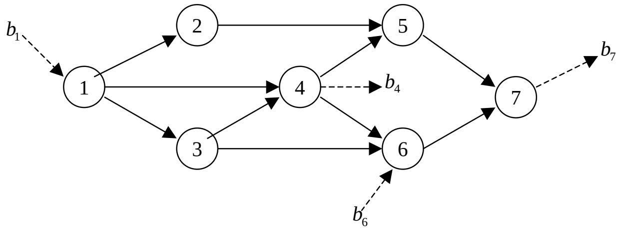
Mise en œuvre

- **Mise en œuvre synchrone :**



IV. Applications

Problèmes de flot convexe dans les réseaux



N est l'ensemble des sommets et $A \subset N \times N$ est l'ensemble des arcs.
b_i est la production ou la consommation de produit au sommet i ;

$$\sum_{i \in N} b_i = 0.$$

Applications

$$\begin{cases} min \sum_{(i,l)\in A} c_{il}(f_{il}), \\ \text{sous} \sum_{(i,l)\in A} f_{il} - \sum_{(m,i)\in A} f_{mi} = b_i, \forall i \in N. \end{cases} \tag{5}$$

- **Le problème admet une solution admissible.**
- c_{il} **est strictement convexe et semi-continue inférieurement.**
- **La fonction convexe conjuguée de** c_{il} **:** $c^*_{il}(t_{il}) = \sup_{f_{il}} \{t_{il}.f_{il} - c_{il}(f_{il})\}$, **est à valeur réelle.**

  $\Rightarrow$ **Le problème (5) possède une solution optimale unique (cf. Rockafellar 70).**

Applications

- **Problème Dual :**

$$\min_{x \in R^n} q(x) \tag{6}$$

- **La fonction duale :** $q(x) = \sum_{(i,l)\in A} c_{il}^*(x_i - x_l) - \sum_{i\in N} b_i.x_i$

- **Admissibilité Régulière :**

  ⇒ Le problème (6) possède une solution optimale
  (cf. Rockafellar 84).

- **Le vecteur gradient :**

$$g_i(x) = \frac{\partial q(x)}{\partial x_i} = \sum_{(i,l)\in A} \nabla c_{il}^*(x_i - x_l) - \sum_{(m,i)\in A} \nabla c_{mi}^*(x_m - x_i) - b_i, i \in N.$$

est une fonction continue monotone non décroissante de x_i**.**

Applications

- **Considérons l' application de gradient** $F'_i(x) = x_i - \frac{1}{\alpha} g_i(x), \forall i \in N - \{d\}$,

- F' **est monotone croissante et continue pour** $\alpha = \beta . \max_{i \in N} a_i$.

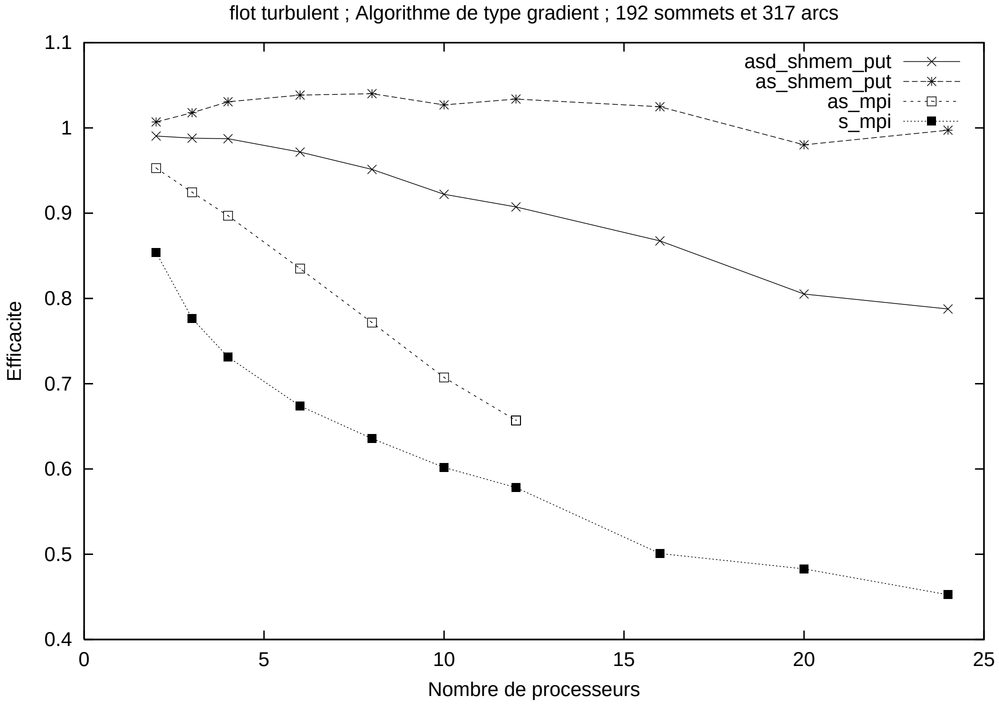
  $\Rightarrow$ **Convergence monotone des itérations asynchrones et asynchrones avec communication flexible associées à** F' **ou à une méthode de type gradient (cf. El Baz 96 et El Baz, Spiteri, Miellou 98).**

- **Test de terminaison :** $(|g_d(x)| \leq \epsilon)$ **pour un problème simple flot.**

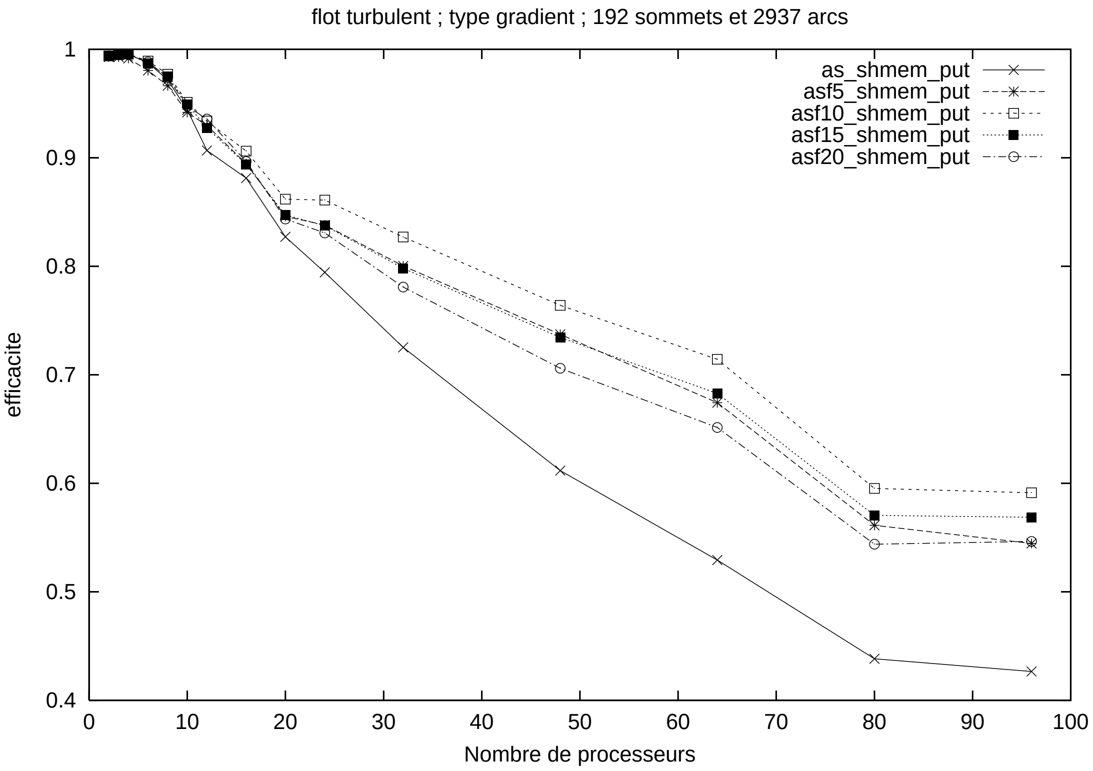
- **Envoi d'un message de terminaison globale par le processeur qui réactualise la composante** x_d.

Applications

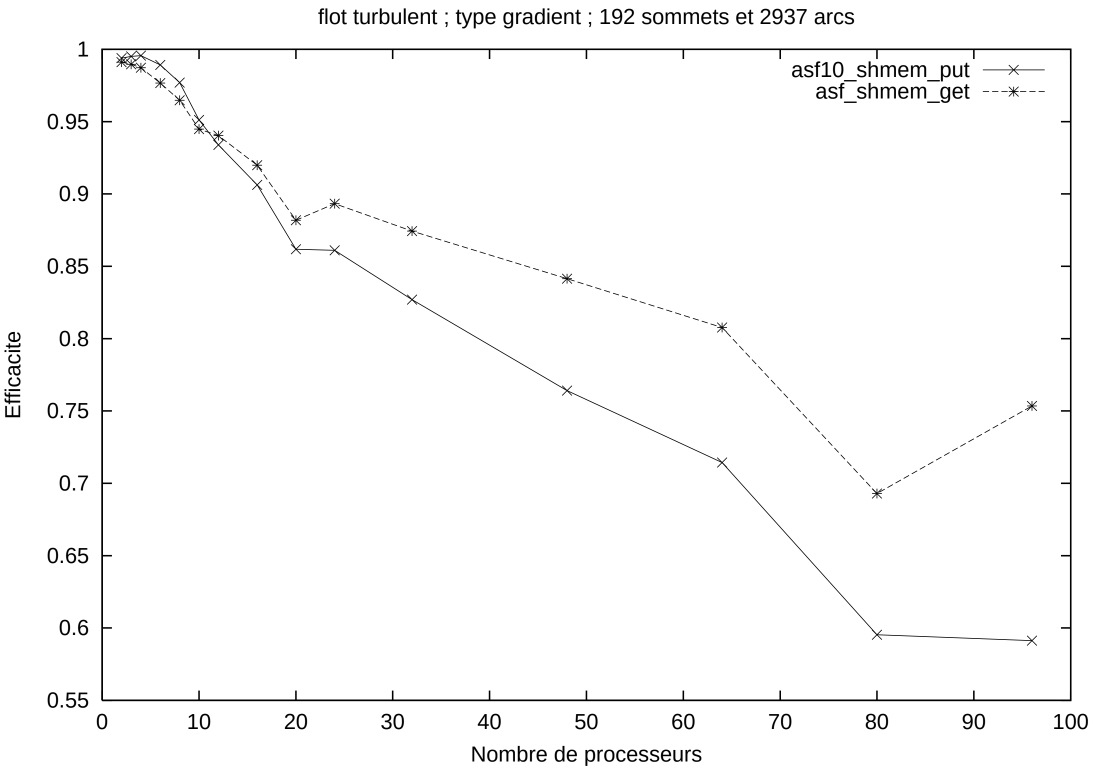
Résultats expérimentaux sur CRAY T3E



Applications



Applications



 Soutenance de thèse de Mohamed JARRAYA

Applications

Résultats expérimentaux sur SMP

	Synchrone		Asynchrone avec c.f	
Threads	Temps(sec)	Efficacité	Temps (sec)	Efficacité
1	205.61	1	205.61	1
2	116.92	0.879	103.66	0.991
3	81.25	0.843	69.22	0.990
4	62.45	0.823	52.16	0.985

Applications

Résultats expérimentaux sur réseaux de SMP

	Sur Ethernet				sur Myrinet			
	Synchrone		Asynchrone c.f		Synchrone		Asynchrone c.f	
Threads	T(sec)	E	T(sec)	E	T(sec)	E	T(sec)	E
(1,1,0,0)	118.38	0.845	107.96	0.926	118.45	0.844	105.65	0.946
(2,1,0,0)	80.93	0.824	70.22	0.949	81.18	0.821	69.22	0.963
(2,2,0,0)	63.29	0.789	52.82	0.946	63.09	0.791	51.95	0.962
(1,1,1,0)	85.55	0.779	77.21	0.863	84.94	0.784	73.45	0.907
(2,1,1,0)	64.19	0.779	55.76	0.897	63.94	0.782	54.16	0.923
(2,2,1,0)	52.94	0.754	43.80	0.913	52.46	0.762	42.64	0.938
(2,2,2,0)	45.50	0.731	37.31	0.893	45.03	0.740	36.31	0.918
(1,1,1,1)	68.19	0.731	62.32	0.802	67.55	0.744	57.18	0.870
(2,1,1,1)	54.32	0.735	47.48	0.824	53.97	0.739	44.90	0.890
(2,2,1,1)	45.86	0.726	38.92	0.856	45.34	0.735	37.25	0.895
(2,2,2,1)	40.38	0.707	33.26	0.859	39.82	0.717	31.49	0.907
(2,2,2,2)	36.21	0.699	30.07	0.831	35.60	0.702	28.51	0.876

Applications

Problèmes de commande optimale

- **Régulation de la température d'un four à n zones de chauffage (cf. El Awtani, Jacquemard, Lhote, Lang, Miellou et Spiteri).**

$$\begin{cases} \dfrac{dy(t)}{dt} + Ay(t) = Bu(t) \ , \forall t \in [t_0, T], \\ z(t) = Cy(t), \\ \textbf{min } J = \dfrac{1}{2} \displaystyle\int_{t_0}^{T} (|Cy(t) - z_d|_2^2 + k|u(t) - u_d|_2^2) dt. \end{cases} \tag{7}$$

$A(2n, 2n)$ **est une M-matrice, (**$a_{ij} \leq 0, \forall i \neq j$ **et** $A^{-1} \geq 0$**),** $B(2n, m) \geq 0$ **et** $C(n, 2n) \geq 0$**,** u **est la commande et** y **est le vecteur d'état.**

Applications

- **L'étude de la commande optimale sans contrainte amène à un système dont la discrétisation sur l'intervalle $[t_0, T]$ par la méthode d'approximation d'Euler donne :**

$$\begin{cases} \dfrac{y(t+h)-y(t)}{h} + Ay(t+h) + \dfrac{1}{k}BB^*p(t+h) = Bu_d, \\ \dfrac{p(t)-p(t+h)}{h} + A^*p(t) - C^*Cy(t) = -C^*z_d. \end{cases} \tag{8}$$

- **Le Système (8) est un système linéaire par bloc qui peut être représenté de la manière suivante :**

$$\begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{11}^* \end{pmatrix} \begin{pmatrix} Y \\ P \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \end{pmatrix}.$$

Applications

- **Application d'un algorithme itératif à deux niveaux :**
  (cf. O'Leary et White 85, Frommer et Szyld 94 et Bahi, Miellou et Rhofir 98).

$$\textbf{première décomposition :} \begin{cases} M_{11} \rightarrow M_{11}^1 - M_{11}^2 \\ M_{22} \rightarrow M_{22}^1 - M_{22}^2 \\ M_{11}^1 = diag(hA + I, ..., hA + I) \\ M_{22}^2 = diag(hA^* + I, ..., hA^* + I) \end{cases}$$

$$\textbf{Deuxième décomposition :} \begin{cases} (hA + I) \rightarrow R - S \\ (hA^* + I) \rightarrow R - S^* \\ R = diag(ha_{11} + 1, ..., ha_{(2n)(2n)} + 1) \end{cases}$$

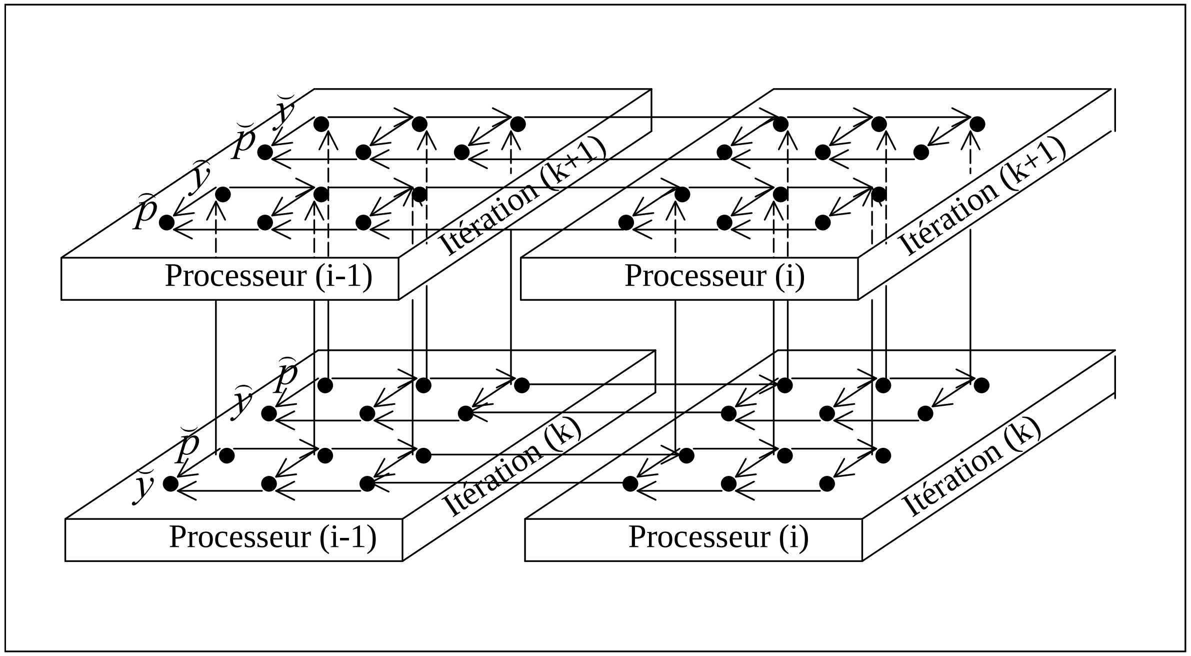
Applications

- **Schéma itératif à deux niveaux pour le calcul du vecteur état :**

Premier niveau
répéter $Z^{(j,0)} = Y^{(j)}$ Deuxième niveau: **répéter** $RZ^{(j,l+1)} = SZ^{(j,l)} + M_{11}^{2} Y^{(j)} - M_{12} P + E_1$ $l = l + 1$ **Jusqu 'à** $\left\| Z^{(j,l)} - Z^{(j,l-1)} \right\|_{\infty} < \varepsilon_2$ $Y^{(j+1)} = Z^{(j,l)}$ $j = j + 1$ **Jusqu 'à** $\left\| Y^{(j)} - Y^{(j-1)} \right\|_{\infty} < \varepsilon_1$

Applications

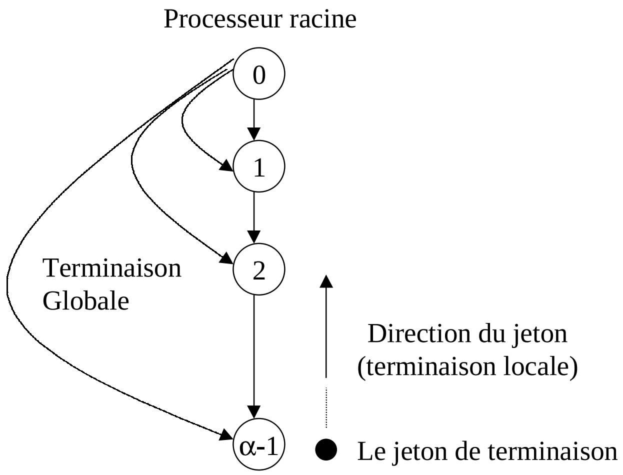
- **Construction de deux séquences monotones de sous-solutions et de sur solutions.**



Applications

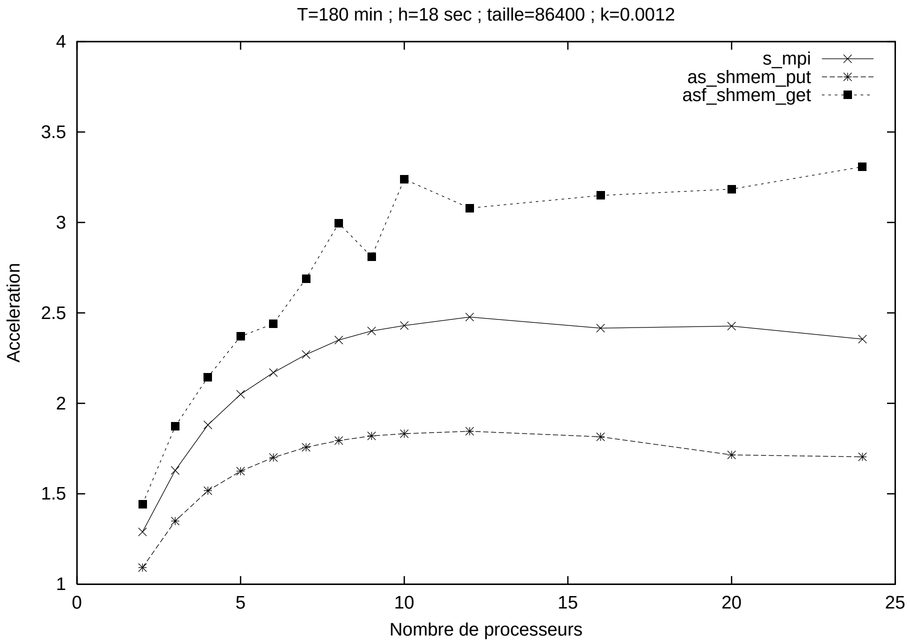
⇒ **Convergence monotone des itérations asynchrones avec communication flexible dans le cas de l'algorithme à un niveau.**

- **Terminaison des itérations asynchrones :**



Applications

Résultats expérimentaux sur CRAY T3E



Applications

Résultats expérimentaux sur SMP

	Synchrone		Asynchrone avec c.f	
Threads	Temps(sec)	Accélération	Temps (sec)	Accélération
1	677.13	1	677.13	1
2	512.36	1.321	465.00	1.455
3	401.90	1.684	347.51	1.948
4	347.96	1.946	313.72	2.158

Résultats expérimentaux sur réseaux Myrinet de SMP

	Synchrone		Asynchrone avec c.f	
Threads	Temps(sec)	Accélération	Temps(sec)	Accélération
(1,1,0)	275.24	1.314	326	1.109
(2,1,0)	220.50	1.641	242.63	1.491
(2,2,0)	191.75	1.887	205.15	1.764
(1,1,1)	213.79	1.692	259.69	1.393
(2,1,1)	183.45	1.972	212.70	1.701
(2,2,1)	171.25	2.113	185.31	1.953
(2,2,2)	159.57	2.268	160.51	2.254

Applications

Systèmes Markoviens

- **Nous considérons un système Markovien qui modélise un réseau téléphonique à commutation de circuit (cf. Le Gall 87) :**

$$Qp = 0. \tag{9}$$

- $p = [p_0, ..., p_{n-1}]$ **est le vecteur probabilité stationnaire ayant** n **blocs** $p_k = [\pi_{k,0}, ..., \pi_{k,z_k-1}]$, $k = 0, 1, ..., n-1$.

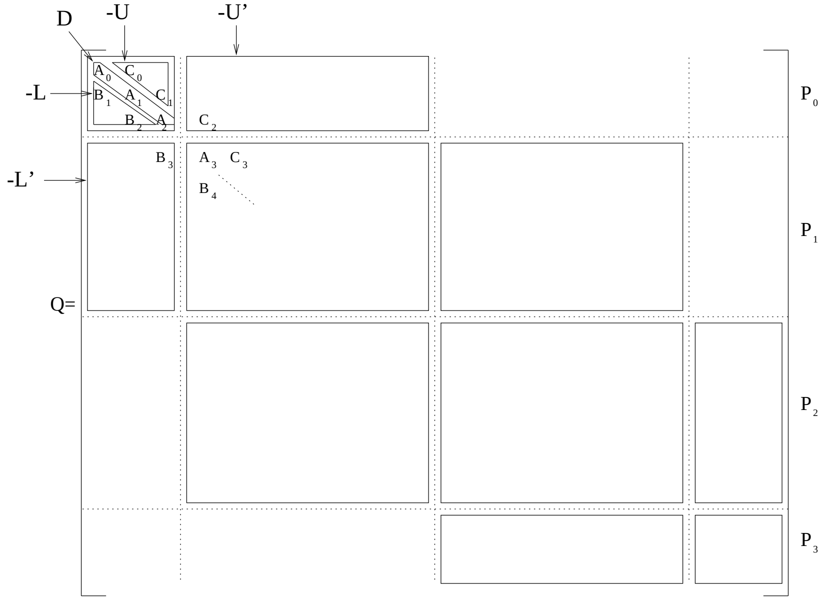
- Q **est le générateur infinitésimal qui est une matrice tridiagonale par bloc, irréductible, apériodique et une Z-matrice.**

  $\Rightarrow$ **Il existe une solution unique** p^* **de (9) vérifiant :** $\sum_{k=0}^{n-1} \sum_{i=0}^{z_k-1} \pi^*_{k,i} = 1$.

Applications

- **Méthodes de résolution :** $Q = D - L - L' - U - U'$

  - **Gauss-Seidel**
  - **SOR**
  - **Agrégation-désagrégation**



- **Mise en œuvre parallèle :**

  - **Amodio et Mazia 95**
  - **Coloration en rouge et noir**

- **Convergence des itérations asynchrones avec retards bornés (cf. Lubachevsky et Mitra 86, Bertsekas et Tsitsiklis 89 et Bahi 98) ou non bornés (cf. El Baz 94).**

Applications

- **Méthode de terminaison de Bertsekas et Tsitsiklis (cf. Bertsekas et Tsitsiklis 89).**

  - Une donnée communiquée est correctement reçue avec un retard fini non spécifié.

- **Terminaison sans acquittement systématique (cf. El Baz 96).**

  - Le temps de transmission des messages entre toute paire de processus adjacents est borné par une constante δ.

  - Les messages entre toute paire de processus adjacents sont reçus suivant leur ordre d'émission.

Applications

Résultats expérimentaux sur CRAY T3E

- **Système de dimension** 6110×6110, $\epsilon = 1O^{-4}$ **et** $\omega = 1.4$.
- **Cas synchrone avec la méthode d'Amodio et Mazia.**

Gauss-Seidel									
processeurs	1	2	3	4	6	8	10	12	16
itérations	197	197	197	198	198	214	215	254	312
temps	48.01	27.92	23.87	18.80	12.80	11.52	11.29	10.34	8.64
efficacité		0.859	0.670	0.638	0.625	0.520	0.425	0.386	0.347
SOR									
itérations	89	88	88	87	87	126	171	DIV	DIV
temps	22.02	12.67	10.81	8.39	5.78	6.96	9.14	DIV	DIV
efficacité		0.868	0.678	0.655	0.634	0.394	0.240	DIV	DIV
Méthode d'agrégation-désagrégation									
itérations	80	79	78	78	76	91	116	DIV	DIV
temps	20.00	11.49	9.67	7.61	5.09	5.12	6.32	DIV	DIV
efficacité		0.870	0.688	0.656	0.654	0.487	0.316	DIV	DIV

Applications

- **Cas synchrone avec coloration en rouge et noir :**

Gauss-Seidel									
processeurs	1	2	3	4	6	8	10	12	16
itérations	197	198	198	198	198	198	198	198	198
temps	48.01	28.05	23.95	18.78	12.06	10.37	10.43	10.45	10.55
efficacité		0.855	0.667	0.638	0.663	0.578	0.460	0.382	0.284
SOR									
itérations	89	87	87	87	87	87	87	87	87
temps	22.02	12.55	10.71	8.41	5.44	4.74	4.80	4.81	4.90
efficacité		0.875	0.685	0.653	0.673	0.579	0.458	0.380	0.280
Agrégation-désagrégation									
itérations	80	75	75	75	75	75	75	75	75
temps	20.00	10.92	9.32	7.32	4.71	4.10	4.19	4.20	4.28
efficacité		0.915	0.714	0.682	0.702	0.608	0.477	0.396	0.291

Applications

- **Cas asynchrone :**

Gauss-Seidel et terminaison avec acquittement									
processeurs	1	2	3	4	6	8	10	12	16
itérations	197	265	408	431	565	822	1115	997	1700
temps	48.01	27.24	23.09	16.92	12.41	11.78	9.74	8.51	8.49
efficacité		0.881	0.692	0.709	0.644	0.509	0.492	0.386	0.353
SOR et terminaison avec acquittement									
itérations	89	180	262	299	426	576	3822	DIV	DIV
temps	22.02	18.76	15.13	11.94	9.49	8.19	35.19	DIV	DIV
efficacité		0.586	0.485	0.460	0.386	0.335	0.062	DIV	DIV
Gauss-Seidel et terminaison sans acquittement									
processeurs	1	2	3	4	6	8	10	12	16
itérations	197	244	369	399	560	853	1301	1127	1734
temps	48.01	25.05	20.85	15.63	12.19	11.74	10.10	9.08	7.87
efficacité		0.958	0.767	0.767	0.655	0.511	0.475	0.440	0.380
SOR et terminaison sans acquittement									
itérations	89	149	237	268	362	635	3484	DIV	DIV
temps	22.02	15.52	13.65	10.69	8.26	8.71	25.16	DIV	DIV
efficacité		0.709	0.537	0.514	0.444	0.315	0.087	DIV	DIV

Applications

Résultats expérimentaux sur SMP

	Gauss-Seidel					
	Synchrone			Asynchrone		
threads	itérations	temps	efficacité	itérations	temps	efficacité
1	197	374.67		197	374.67	
2	197	225.01	0.832	217	183.77	1.019
3	197	188.39	0.662	286	135.83	0.919
4	197	149.19	0.627	495	106.26	0.881
	SOR					
1	89	169.04		89	169.04	
2	88	109.88	0.769	97	83.11	1.016
3	88	92.21	0.611	133	63.56	0.886
4	87	71.94	0.587	236	51.24	0.824

Applications

Résultats expérimentaux sur résaux Myrinet de SMP

	Gauss-Seidel					
	Synchrone			Asynchrone (terminaison sans aquit)		
threads	itérations	temps	efficacité	itérations	temps	efficacité
(1,0)	197	311.27		197	311.27	
(1,1)	197	179.63	0.866	304	209.51	0.742
(2,1)	198	168.48	0.615	350	165.40	0.627
(2,2)	197	103.87	0.749	579	105.73	0.735
	SOR					
(1,0)	89	142.19		89	142.19	
(1,1)	88	81.27	0.874	229	158.12	0.449
(2,1)	88	74.81	0.633	234	113.24	0.418
(2,2)	87	46.23	0.768	369	67.73	0.524

V. Conclusions

- **Résultat de convergence des itérations asynchrones avec communication flexible pour le problème de commande optimale.**
- **Les performances varient selon le type d'application et d'architecture.**
- **Le choix de l'environnement de communication est très important.**
- **Les itérations asynchrones sont plus performantes que les itérations synchrones quand la charge est déséquilibrée.**
- **Performances optimales pour un accès direct en mémoire.**

⇒ Intérêt des itérations asynchrones avec communication flexible.

Perspectives

- **Problèmes de flot : partitionnement optimal des sommets du graphe sur l'ensemble des processeurs.**
- **Problèmes de commande optimale : Etude de la convergence des itérations asynchrones avec communication flexible pour l'algorithme à deux niveaux.**
- **Systèmes Markoviens : Etude des itérations asynchrones pour la méthode d'agrégation-désagrégation et les méthodes de projection.**
- **Mise en œuvre sur d'autres architectures comme l'ORIGIN2000 et la machine multiprocesseur vectorielle NEC SX5.**