Arnaud LEGRAND, Hélène RENARD, Yves ROBERT, Frédéric VIVIEN

LIP - École Normale Supérieure de Lyon

www.ens-lyon.fr/~hrenard

Helene.Renard@ens-lyon.fr

Placement et équilibrage de charge pour calculs itératifs sur grappes hétérogènes

projet *ReMap* : en coopération avec LIP, UMR CNRS - ENS Lyon - INRIA 5668

Plan de l'exposé

Plan de l'exposé

Plan de l'exposé

Plan de l'exposé

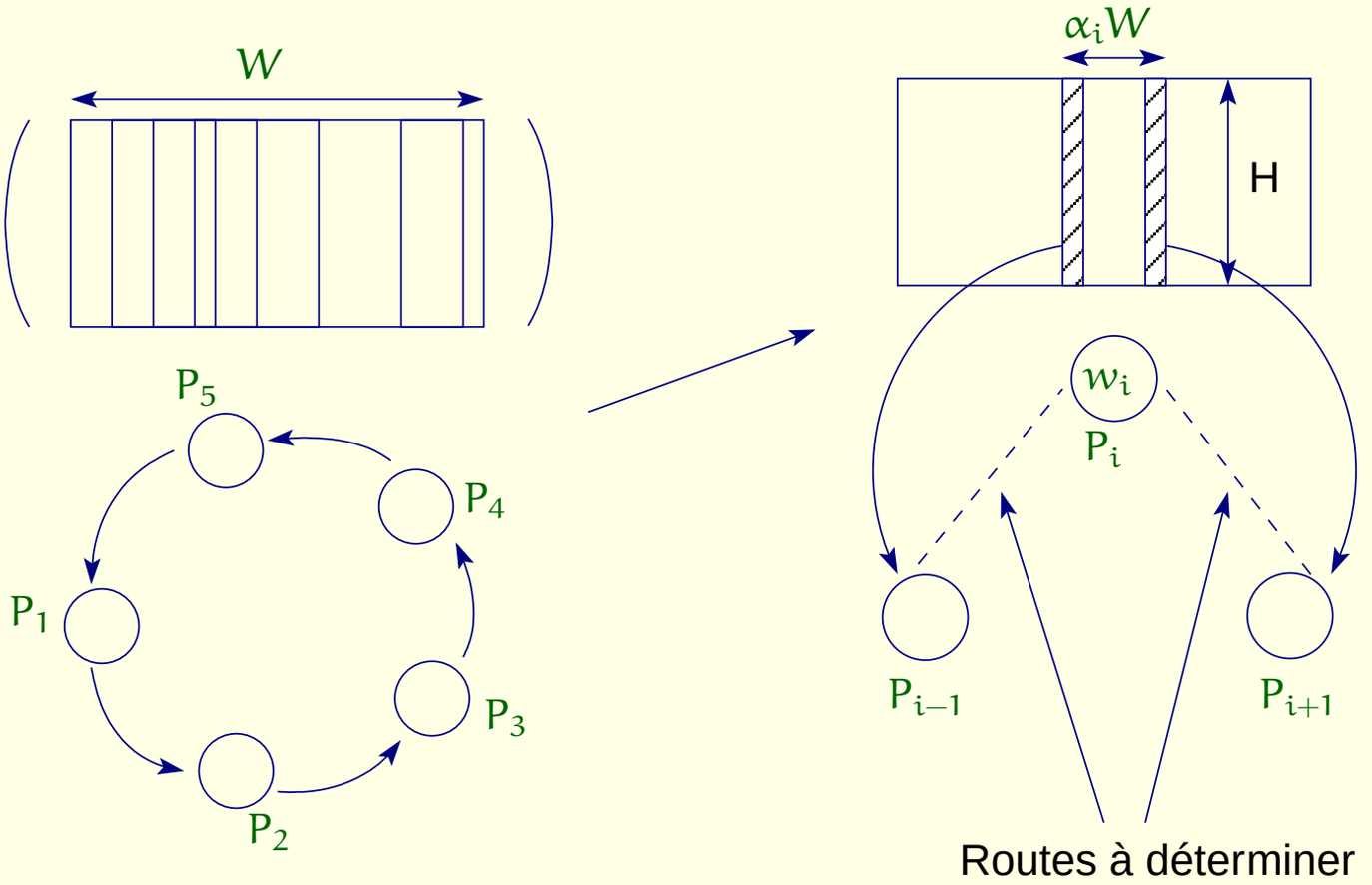
Plan de l'exposé

Plan de l'exposé

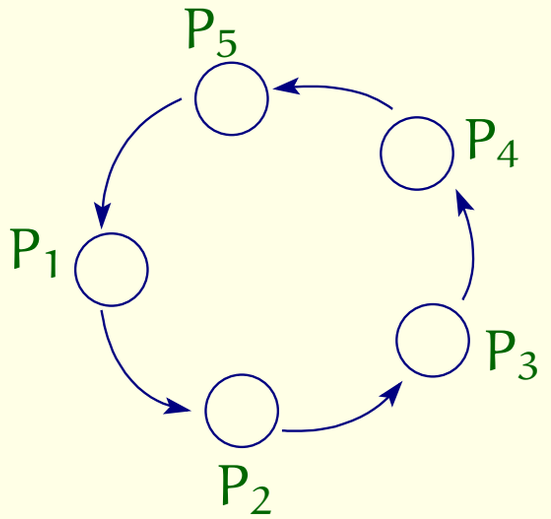
Cadre de travail

Problème cible (1/3)

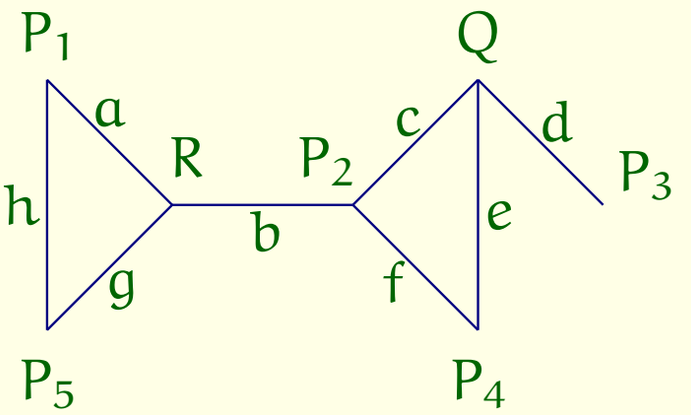
- Techniques d'équilibrage de charge pour algorithmes itératifs
- Matrice rectangulaire de données divisée en tranches verticales
- Recherche d'un anneau de processeurs (virtuel) à l'intérieur de la plate-forme hétérogène



Problème cible (2/3)



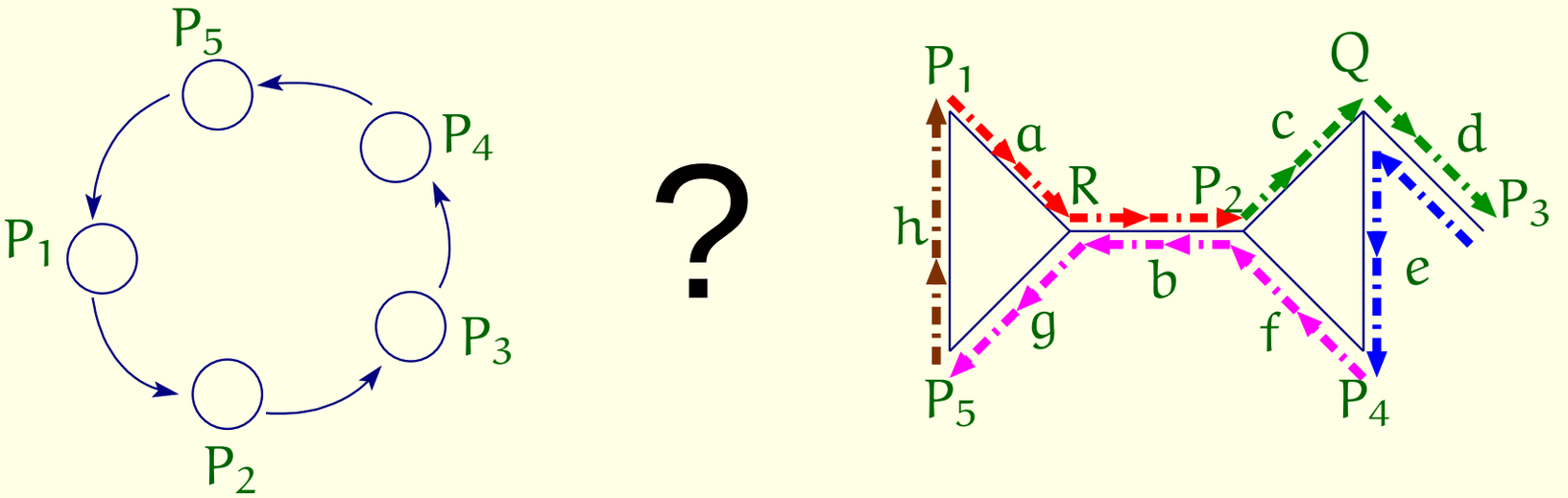
?



Questions

- Sélection des processeurs
- Organisation des processeurs

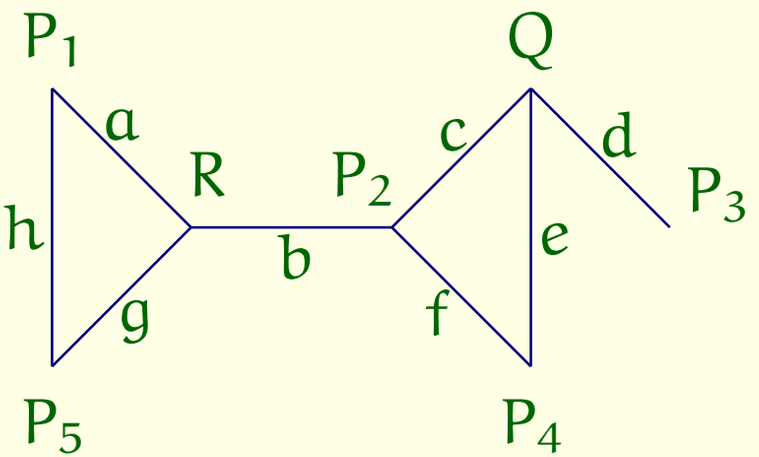
Problème cible (3/3)



Questions

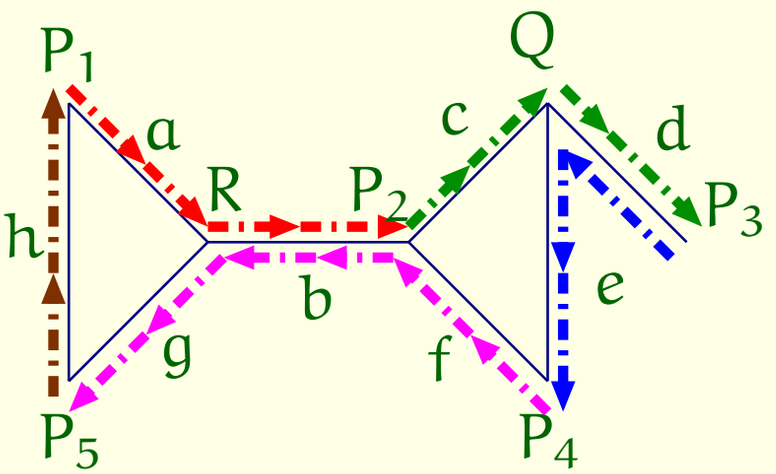
- Construction des chemins (pour les processeurs successifs dans l'anneau)
- Allocation des différentes bandes passantes aux chemins

Exemple : choix de l'anneau (1/4)



- 7 processeurs et 8 liens de communication bidirectionnels
- Anneau solution de taille 5 $P_1 \rightarrow P_2 \rightarrow P_3 \rightarrow P_4 \rightarrow P_5$, laissant de côté les processeurs Q et R
- Liens étiquetés par des lettres allant de a à h
- b_x désigne la bande passante du lien x

Exemple : choix des chemins (2/4)



De P_1 à P_2, nous utilisons les liens a et b d'où $\mathcal{S}_1 = \{a, b\}$. Mais de P_2 à P_1, nous utilisons plutôt les liens b, g et h, d'où $\mathcal{P}_2 = \{b, g, h\}$.

- De P_1 : à P_2, $\mathcal{S}_1 = \{a, b\}$ et à P_5, $\mathcal{P}_1 = \{h\}$
- De P_2 : à P_3, $\mathcal{S}_2 = \{c, d\}$ et à P_1, $\mathcal{P}_2 = \{b, g, h\}$
- De P_3 : à P_4, $\mathcal{S}_3 = \{d, e\}$ et à P_2, $\mathcal{P}_3 = \{d, e, f\}$
- De P_4 : à P_5, $\mathcal{S}_4 = \{f, b, g\}$ et à P_3, $\mathcal{P}_4 = \{e, d\}$
- De P_5 : à P_1, $\mathcal{S}_5 = \{h\}$ et à P_4, $\mathcal{P}_5 = \{g, b, f\}$

Exemple : coûts des chemins (3/4)

Pour P_1, puisque $\mathcal{S}_1 = \{a, b\}$, nous avons $c_{1,2} = \frac{1}{\min(s_{1,a}, s_{1,b})}$; et puisque $\mathcal{P}_1 = \{h\}$, nous avons $c_{1,5} = \frac{1}{p_{1,h}}$.

La liste de toutes les équations devant être satisfaites :

Lien a **:** $s_{1,a} \leq b_a$

Lien b **:** $s_{1,b} + s_{4,b} + p_{2,b} + p_{5,b} \leq b_b$

Lien c **:** $s_{2,c} \leq b_c$

Lien d **:** $s_{2,d} + s_{3,d} + p_{3,d} + p_{4,d} \leq b_d$

Lien e **:** $s_{3,e} + p_{3,e} + p_{4,e} \leq b_e$

Lien f **:** $s_{4,f} + p_{3,f} + p_{5,f} \leq b_f$

Lien g **:** $s_{4,g} + p_{2,g} + p_{5,g} \leq b_g$

Lien h **:** $s_{5,h} + p_{1,h} + p_{2,h} \leq b_h$

Exemple : programmation quadratique (4/4)

minimiser $\max_{1 \leq i \leq 5} (\alpha_i . W . w_i + H.(c_{i,i-1} + c_{i,i+1}))$ avec les contraintes suivantes

$$
\begin{cases}
\sum_{i=1}^{5} \alpha_i = 1 & & \\
s_{1,a} \leq b_a & s_{1,b} + s_{4,b} + p_{2,b} + p_{5,b} \leq b_b & s_{2,c} \leq b_c \\
s_{2,d} + s_{3,d} + p_{3,d} + p_{4,d} \leq b_d & s_{3,e} + p_{3,e} + p_{4,e} \leq b_e & s_{4,f} + p_{3,f} + p_{5,f} \leq b_f \\
s_{4,g} + p_{2,g} + p_{5,g} \leq b_g & s_{5,h} + p_{1,h} + p_{2,h} \leq b_h & \\
s_{1,a} . c_{1,2} \geq 1 & s_{1,b} . c_{1,2} \geq 1 & p_{1,h} . c_{1,5} \geq 1 \\
s_{2,c} . c_{2,3} \geq 1 & s_{2,d} . c_{2,3} \geq 1 & p_{2,b} . c_{2,1} \geq 1 \\
p_{2,g} . c_{2,1} \geq 1 & p_{2,h} . c_{2,1} \geq 1 & s_{3,d} . c_{3,4} \geq 1 \\
s_{3,e} . c_{3,4} \geq 1 & p_{3,d} . c_{3,2} \geq 1 & p_{3,e} . c_{3,2} \geq 1 \\
p_{3,f} . c_{3,2} \geq 1 & s_{4,f} . c_{4,5} \geq 1 & s_{4,b} . c_{4,5} \geq 1 \\
s_{4,g} . c_{4,5} \geq 1 & p_{4,e} . c_{4,3} \geq 1 & p_{4,d} . c_{4,3} \geq 1 \\
s_{5,h} . c_{5,1} \geq 1 & p_{5,g} . c_{5,4} \geq 1 & p_{5,b} . c_{5,4} \geq 1 \\
p_{5,f} . c_{5,4} \geq 1 & &
\end{cases}
$$

Présentation de la plate-forme (1/2)

1. **Coûts de calcul**
   - Graphe orienté $G = (P, E)$
   - P_i = ressource de calcul, pondérée par son temps de cycle w_i
2. **Coûts de communication**
   - $e \in E$ = arête du graphe = lien de communication, étiqueté par la bande passante b_e
   - L/b_e unités de temps pour transférer un message de taille L en utilisant l'arête e
   - Partage de bande passante :
     1er message utilise $2b_e/3 \Rightarrow$ 2e message utilise au plus $b_e/3$
3. **Routage**
   - Nous décidons comment les messages sont routés d'un processeur à un autre
   - Message passant par $p = (e_1, e_2, \ldots, e_m, \ldots, e_k)$:
     - fraction f_m de la bande passante disponible b_{e_m}
     - L/b unités de temps pour router le message, où $b = \min_{1 \leq m \leq k} f_m$
   - La bande passante totale de chaque lien ne doit pas être dépassée

Présentation de la plate-forme (2/2)

4. **Paramètres de l'application : calculs**
   - W = taille totale du travail
   - P_i effectue une partie $\alpha_i.W$ du travail où $\alpha_i \geq 0$ et $\sum_{i=1}^{p} \alpha_i = 1$
   - $\alpha_j = 0$ si P_j ne participe pas à la solution
5. **Paramètres de l'application : communications dans l'anneau**
   - P_i envoie un message de longueur H à son successeur $succ(i)$
   - P_i envoie un message de longueur H à son prédécesseur $pred(i)$
   - $\mathcal{S}_i$ chemin de communication de P_i à $P_{succ(i)}$
   - $s_{i,m}$ fraction de la bande passante b_{e_m} du lien physique $e_m \in \mathcal{S}_i$
   - $c_{i,succ(i)} = \frac{1}{\min_{e_m \in \mathcal{S}_i} s_{i,m}}$
   - P_i a besoin de $H.c_{i,succ(i)}$ unités de temps pour envoyer son message à $P_{succ(i)}$
6. **Fonction objectif**

$$T_{step} = \max_{1 \leq i \leq p} \mathbb{I}\{i\}[\alpha_i.W.w_i + H.(c_{i,pred(i)} + c_{i,succ(i)})]$$

Problème d'optimisation SHAREDRING (1/2)

Définition SHAREDRING(p, w_i, E, b_{e_m}, W, H) : minimiser

$$T_{step} = \min_{1 \leq q \leq p} \min_{\sigma \in \Theta_{q,p}} \max_{1 \leq i \leq q} \left(\alpha_{\sigma(i)} . W . w_{\sigma(i)} + H . \left(c_{\sigma(i),\sigma(i-1 \bmod q)} + c_{\sigma(i),\sigma(i+1 \bmod q)} \right) \right)$$

$$\sum_{i=1}^{q} \alpha_{\sigma(i)} = 1$$

- $\Theta_{q,p}$ = ensemble des fonctions injectives $\sigma : [1..q] \rightarrow [1..p]$ indexant les q processeurs sélectionnés pour l'anneau.
- $2q$ chemins de communication :
  - chemin $\mathcal{S}_i$ de $P_{\sigma(i)}$ à son successeur $P_{succ(\sigma(i))} = P_{\sigma(i+1 \bmod q)}$
  - chemin $\mathcal{P}_i$ de $P_{\sigma(i)}$ à son prédécesseur $P_{pred(\sigma(i))} = P_{\sigma(i-1 \bmod q)}$
- Pour chaque lien e_m
  - $s_{\sigma(i),m}$ fraction de bande passante b_{e_m} allouée au chemin $\mathcal{S}_{\sigma(i)}$
  - $p_{\sigma(i),m}$ fraction de bande passante b_{e_m} allouée au chemin $\mathcal{P}_{\sigma(i)}$

Problème d'optimisation SHAREDRING (2/2)

Équations :

$$\begin{cases} s_{\sigma(i),m} \geq 0,\ p_{\sigma(i),m} \geq 0, & 1 \leq i \leq q, \quad 1 \leq m \leq E \\ c_{\sigma(i),\mathrm{succ}(\sigma(i))} = \frac{1}{\min_{e_m \in \mathcal{S}_{\sigma(i)}} s_{\sigma(i),m}} & 1 \leq i \leq q \\ c_{\sigma(i),\mathrm{pred}(\sigma(i))} = \frac{1}{\min_{e_m \in \mathcal{P}_{\sigma(i)}} p_{\sigma(i),m}} & 1 \leq i \leq q \\ \sum_{i=1}^{q} (s_{\sigma(i),m} + p_{\sigma(i),m}) \leq b_{e_m} & 1 \leq m \leq E \end{cases}$$

Remarque Plus le ratio $\frac{W}{H}$ sera grand, plus le nombre de processeurs augmentera dans la solution optimale

Doute ☹ Extraire le « meilleur » anneau = problème combinatoire difficile

Variantes

1 Surcoûts d'initialisation

Seules des applications importantes sont susceptibles d'être déployées

$\Rightarrow$ surcoûts d'initialisation peuvent être négligés

2 Liens bidirectionnels

Modéliser un lien bidirectionnel de bande passante b :

- attribuer une fraction de bande passante f_{path} à chaque chemin de communication nécessitant un lien bidirectionnel de bande passante b, sans tenir compte de l'orientation du chemin
- contrainte $\sum f_{path} \leq b$

$\Rightarrow$ liens uni / bi - directionnels peuvent exister simultanément dans le réseau

3 Liens multiples

Modéliser G comme un multi-graphe plutôt que comme un graphe simple.

4 Liens centraux

Remplacer la contrainte $\sum f_{path} \leq b$ par $f_{path} \leq b$ pour chaque chemin utilisant le lien

Variantes

1 Surcoûts d'initialisation

Seules des applications importantes sont susceptibles d'être déployées

$\Rightarrow$ surcoûts d'initialisation peuvent être négligés

2 Liens bidirectionnels

Modéliser un lien bidirectionnel de bande passante b :

- attribuer une fraction de bande passante f_{path} à chaque chemin de communication nécessitant un lien bidirectionnel de bande passante b, sans tenir compte de l'orientation du chemin
- contrainte $\sum f_{path} \leq b$

$\Rightarrow$ liens uni / bi - directionnels peuvent exister simultanément dans le réseau

3 Liens multiples

Modéliser G comme un multi-graphe plutôt que comme un graphe simple.

4 Liens centraux

Remplacer la contrainte $\sum f_{path} \leq b$ par $f_{path} \leq b$ pour chaque chemin utilisant le lien

Variantes

1 Surcoûts d'initialisation

Seules des applications importantes sont susceptibles d'être déployées

$\Rightarrow$ surcoûts d'initialisation peuvent être négligés

2 Liens bidirectionnels

Modéliser un lien bidirectionnel de bande passante b :

- attribuer une fraction de bande passante f_{path} à chaque chemin de communication nécessitant un lien bidirectionnel de bande passante b, sans tenir compte de l'orientation du chemin
- contrainte $\sum f_{path} \leq b$

$\Rightarrow$ liens uni / bi - directionnels peuvent exister simultanément dans le réseau

3 Liens multiples

Modéliser G comme un multi-graphe plutôt que comme un graphe simple.

4 Liens centraux

Remplacer la contrainte $\sum f_{path} \leq b$ par $f_{path} \leq b$ pour chaque chemin utilisant le lien

Variantes

1 Surcoûts d'initialisation

Seules des applications importantes sont susceptibles d'être déployées

$\Rightarrow$ surcoûts d'initialisation peuvent être négligés

2 Liens bidirectionnels

Modéliser un lien bidirectionnel de bande passante b :

- attribuer une fraction de bande passante f_{path} à chaque chemin de communication nécessitant un lien bidirectionnel de bande passante b, sans tenir compte de l'orientation du chemin
- contrainte $\sum f_{path} \leq b$

$\Rightarrow$ liens uni / bi - directionnels peuvent exister simultanément dans le réseau

3 Liens multiples

Modéliser G comme un multi-graphe plutôt que comme un graphe simple.

4 Liens centraux

Remplacer la contrainte $\sum f_{path} \leq b$ par $f_{path} \leq b$ pour chaque chemin utilisant le lien

Le problème d'optimisation SLICERING

Définition SLICERING

Même problème que précédemment, mais avec routage et bande passante fixés

$\Leftrightarrow$ graphe complet (virtuel) de liens dédiés

Simulation

Étant donné un réseau « réel », construisons le problème simplifié de la manière suivante :

- prenons le réseau actuel en entrée
- calculons les plus courts chemins (en terme de bande passante) entre toutes les paires de processeurs

$\Rightarrow$ (faux) réseau totalement connecté

Complexité

Complexité

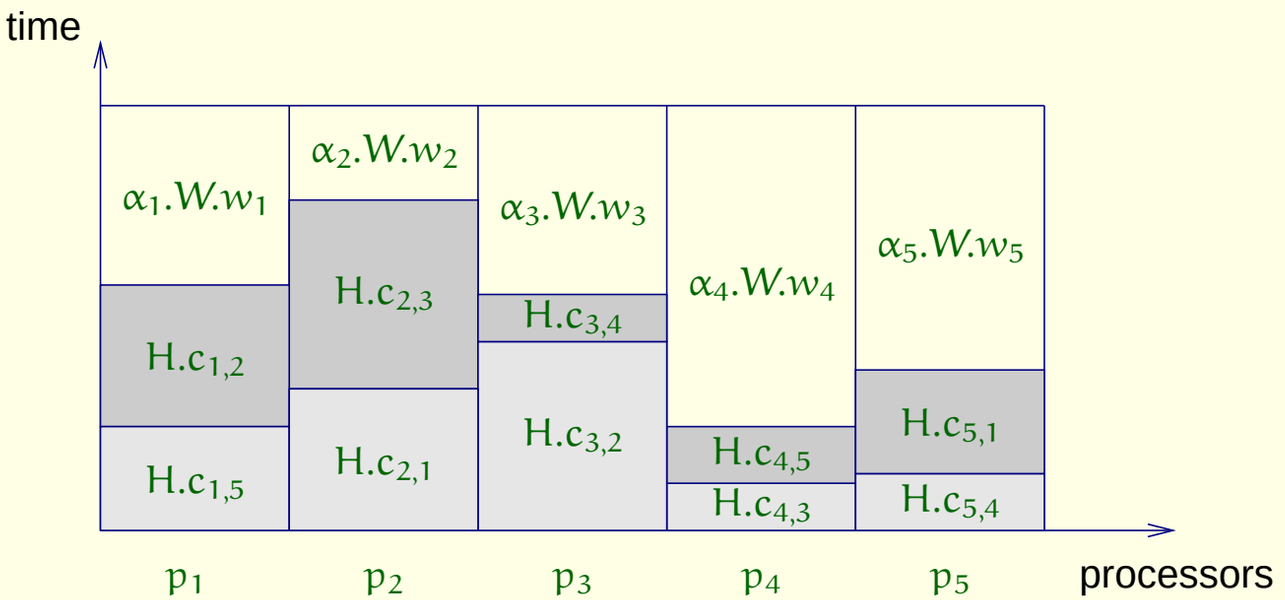
Proposition Le problème de décision associé au problème d'optimisation SHAREDRING est NP-complet

– réduction à un cycle Hamiltonien
– même résultat pour SLICERING (avec simplification)

Heuristiques

Construction de l'anneau (1/2)

$$T_{step} = \alpha_i.W.w_i + H.(c_{i,i-1} + c_{i,i+1})$$



Résumé des temps de calcul et de communication avec $q = 5$ processeurs

- $\frac{T_{step}}{W.w_{cumul}} = 1 + \frac{H}{W} \sum_{i=1}^{q} \frac{c_{i,i-1}+c_{i,i+1}}{w_i}$ où $w_{cumul} = \frac{1}{\sum_{i=1}^{q} \frac{1}{w_i}}$
- T_{step} minimal quand $\sum_{i=1}^{q} \frac{c_{i,i-1}+c_{i,i+1}}{w_i}$ minimal (avec tous les processeurs)

Construction de l'anneau (2/2)

1. Sélectionner la meilleure paire de processeurs
2. Inclure itérativement un nouveau processeur dans l'anneau solution actuel :
   - pour chaque paire de processeurs consécutifs (P_j, P_k) dans l'anneau
   - pour chaque processeur restant P_i
     - calculer le coût d'insertion de P_i entre P_j et P_k :
       i. mettre à jour w_{cumul} en ajoutant le nouveau processeur P_k, ce qui diminue w_{cumul}
       ii. dans $\sum_{s=1}^{r} \frac{c_{\sigma(s),\sigma(s-1)} + c_{\sigma(s),\sigma(s+1)}}{w_{\sigma(s)}}$, supprimer les 2 anciens chemins puis additionner $\frac{c_{k,j}+c_{k,i}}{w_k}$, $\frac{c_{j,k}}{w_j}$ et $\frac{c_{i,k}}{w_i}$
     - retenir la paire et le processeur minimisant le coût d'insertion
   - retourner la nouvelle valeur de T_{step}

Allocation de bande passante

– nous avons construit $2r$ chemins de communication
– une certaine fraction de la bande passante initiale leur a été allouée

Pour construire les quatre nouveaux chemins impliquant P_k :
– $G = (V, E, b)$, arête étiquetée par la bande passante disponible restante
– $b(e_m)$ la bande passante laissée par les $2r$ chemins
– Réinjecter les fractions de bande passante utilisées par les deux anciens chemins de communication entre P_i et P_j
– Algorithme de calcul du plus court chemin (en termes de bande passante) pour déterminer les quatre chemins, de P_k à P_i et P_j et vice-versa

Stratégie :
– Calculer indépendamment quatre chemins de bande passante maximale, à l'aide d'un algorithme standard de calcul du plus court chemin dans G.
– Si certains chemins partagent des liens, ne pas changer les chemins, mais employer une méthode (analytique) brutale pour calculer les fractions de bande passante

Optimisations

- nous construisons un anneau de façon gloutonne en fractionnant les bandes passantes pour insérer de nouveaux processeurs
- nous ne recalculons jamais les fractions de bande passante ayant été attribuées aux précédents chemins de communication

1. **Max-min fairness :**
   algorithme traditionnel de partage de bande passante, conçu pour maximiser la bande passante minimum allouée à un chemin
2. **Résolution quadratique :**
   à l'aide du logiciel KINSOL. Une fois obtenu un anneau ainsi que tous les chemins de communication, le programme pour minimiser T_{step} est quadratique en les inconnues α_i, $s_{i,j}$ et $p_{i,j}$

Heuristique pour SLICERING

- Heuristique gloutonne similaire :
  - calcule les bandes passantes du réseau (virtuel) totalement connecté
  - augmente l'anneau jusqu'à avoir p processeurs et retourne la valeur minimale obtenue pour T_{step}

- **Évaluation de l'impact du partage des liens**
  - l'heuristique de SLICERING retourne la même solution que l'heuristique de SHAREDRING si le réseau totalement connecté est donné en entrée
  - garder cet anneau ainsi que les chemins de communication, et re-calculer les bandes passantes afin d'obtenir une solution réelle
  - comparer cette dernière avec la solution retournée par SHAREDRING

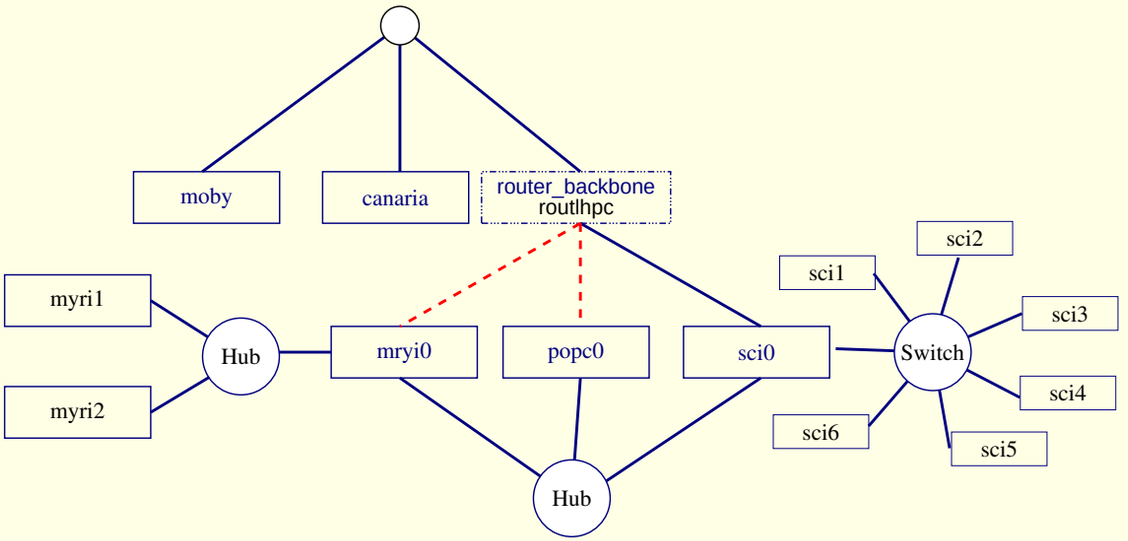
Heuristique pour SLICERING

- Heuristique gloutonne similaire :
  - calcule les bandes passantes du réseau (virtuel) totalement connecté
  - augmente l'anneau jusqu'à avoir p processeurs et retourne la valeur minimale obtenue pour T_{step}

- **Évaluation de l'impact du partage des liens**
  - l'heuristique de SLICERING retourne la même solution que l'heuristique de SHAREDRING si le réseau totalement connecté est donné en entrée
  - garder cet anneau ainsi que les chemins de communication, et re-calculer les bandes passantes afin d'obtenir une solution réelle
  - comparer cette dernière avec la solution retournée par SHAREDRING

Résultats expérimentaux

Description de la plate-forme (Lyon)

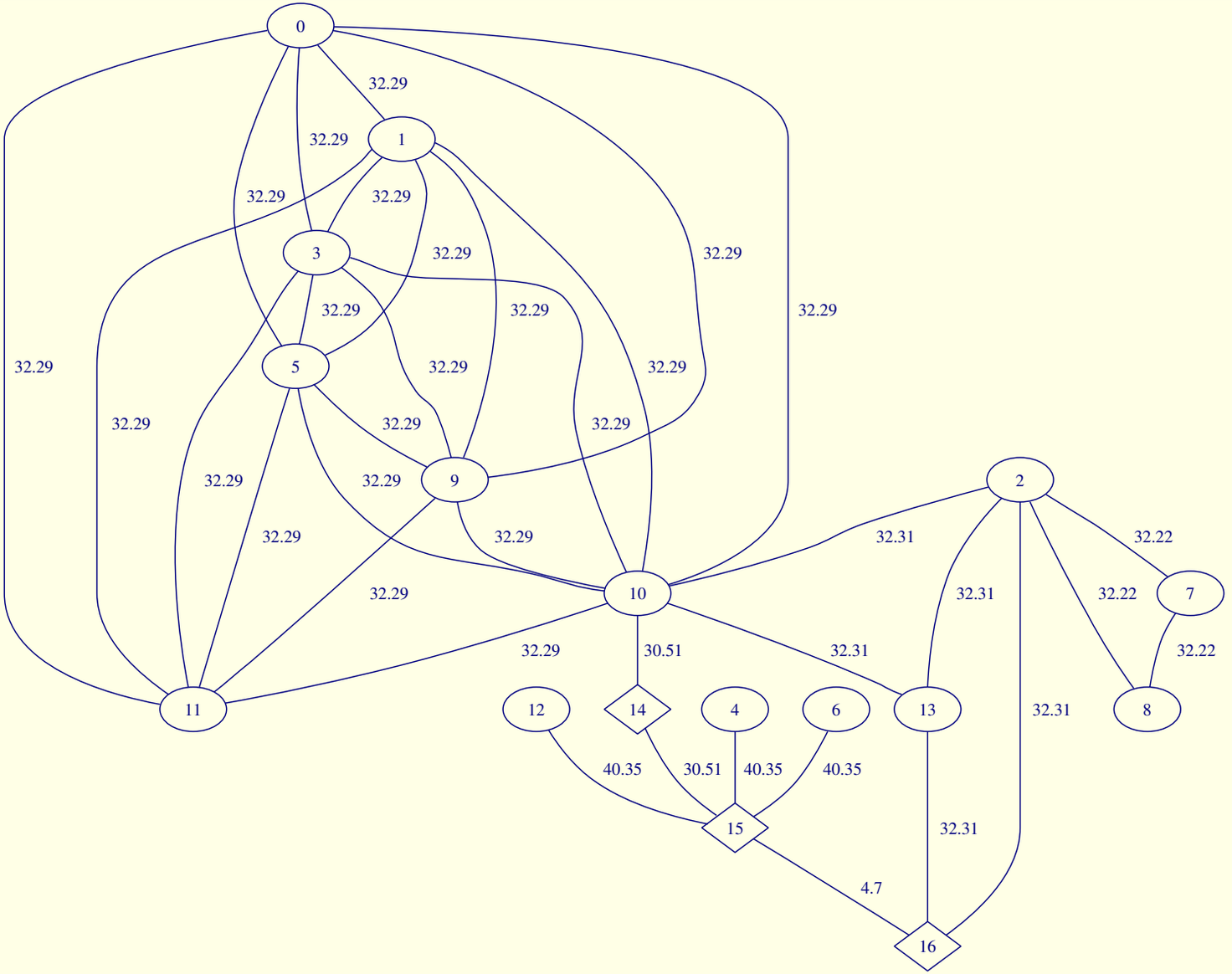


Topologie de la plate-forme de Lyon

P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}
0.0206	0.0206	0.0206	0.0206	0.0291	0.0206	0.0087	0.0206	0.0206	0.0206	0.0206	0.0206	0.0291	0.0451	0	0	0

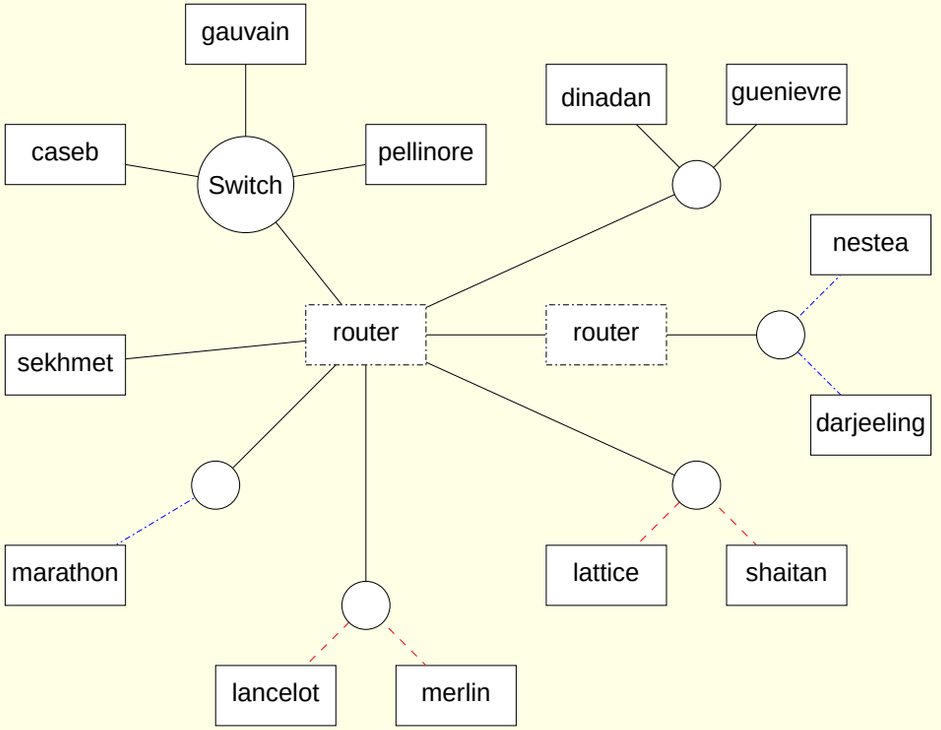
Temps de cycles des processeurs (en secondes par megaflop)
pour la plate-forme de Lyon

Description de la plate-forme (Lyon)



Vue abstraite de la plate-forme de Lyon.

Description de la plate-forme (Strasbourg)

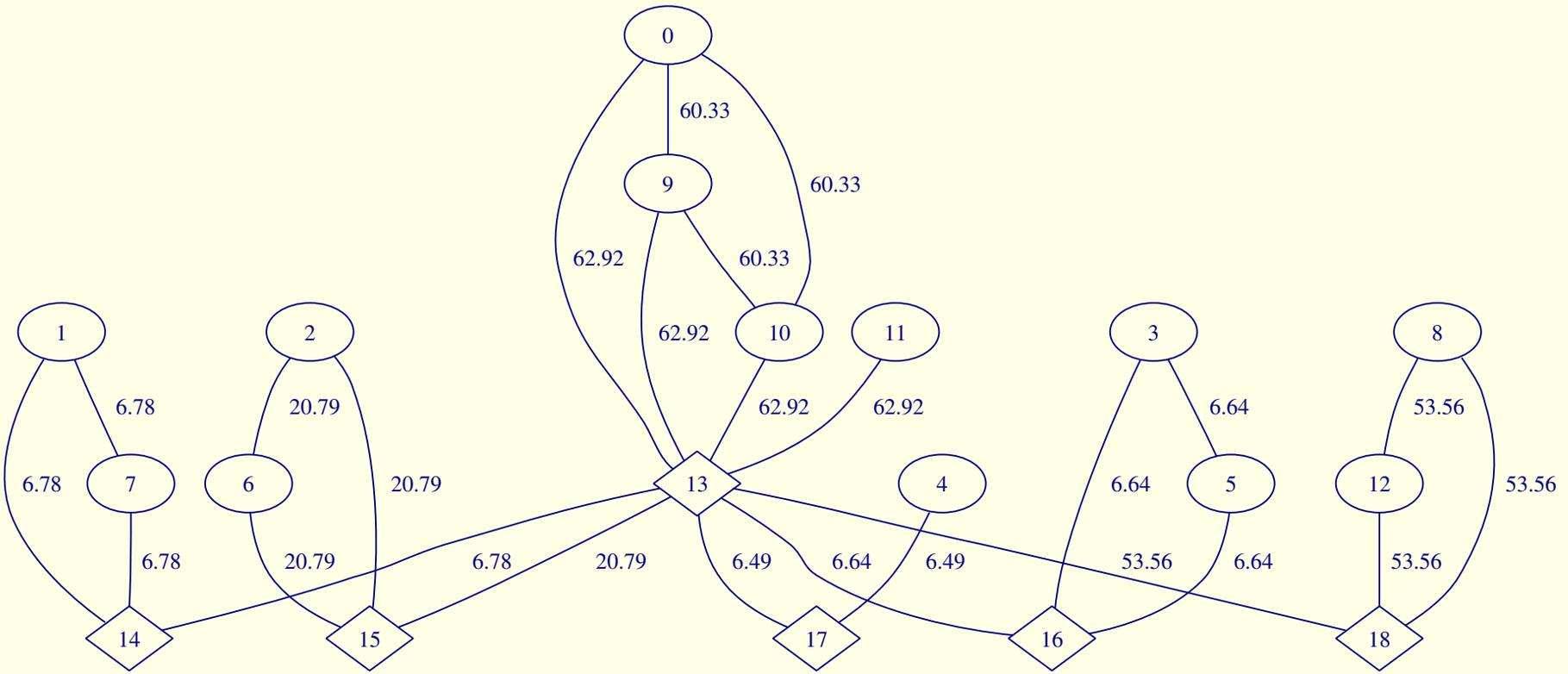


Topologie de la plate-forme de Strasbourg

P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}	P_{17}	P_{18}
0.0087	0.0072	0.0087	0.0131	0.016	0.0058	0.0087	0.0262	0.0102	0.0131	0.0072	0.0058	0.0072	0	0	0	0	0	0

Temps de cycles des processeurs (en secondes par megaflop)
pour la plate-forme de Strasbourg

Description de la plate-forme (Strasbourg)



Vue abstraite de la plate-forme de Strasbourg.

Résultats (1/2)

Première heuristique slice-ring modifiée afin d'obtenir une solution réelle

Seconde heuristique gloutonne pour SHAREDRING, avec l'optimisation de programmation quadratique

Ratio H/W	H1 : slice-ring	H2 : shared-ring	Amélioration
0.1	3.17	3.15	0.63%
1	3.46	3.22	6.94%
10	10.39	3.9	62.46%

T_{step}/W pour chaque heuristique sur la plate-forme de Lyon

Ratio H/W	H1 : slice-ring	H2 : shared-ring	Amélioration
0.1	7.32	7.26	0.82%
1	9.65	7.53	21.97%
10	19.24	10.26	46.67%

T_{step}/W pour chaque heuristique sur la plate-forme de Strasbourg

Résultats (2/2)

1 Lorsque l'impact du coût de communication est bas :

- but principal = équilibrer les calculs
- les deux heuristiques sont équivalentes

2 Lorsque le rapport H/W devient plus important :

- l'effet de la contention des liens devient évident
- la solution retournée par la seconde heuristique est bien meilleure

3 ☺ Une modélisation précise des communications a un impact important sur la performance des stratégies d'équilibrage de charge

Résultats (2/2)

1 Lorsque l'impact du coût de communication est bas :
- but principal = équilibrer les calculs
- les deux heuristiques sont équivalentes

2 Lorsque le rapport H/W devient plus important :
- l'effet de la contention des liens devient évident
- la solution retournée par la seconde heuristique est bien meilleure

3 ☺ Une modélisation précise des communications a un impact important sur la performance des stratégies d'équilibrage de charge

Résultats (2/2)

1 Lorsque l'impact du coût de communication est bas :

- but principal = équilibrer les calculs
- les deux heuristiques sont équivalentes

2 Lorsque le rapport H/W devient plus important :

- l'effet de la contention des liens devient évident
- la solution retournée par la seconde heuristique est bien meilleure

3 ☺ Une modélisation précise des communications a un impact important sur la performance des stratégies d'équilibrage de charge

Travaux antérieurs

Travaux antérieurs

- Stratégies d'équilibrage de charge sur plate-formes homogènes / hétérogènes

- Grande majorité : stratégies dynamiques
  - « utiliser le passé pour prédire le futur »
  - données sont échangées uniquement entre processeurs voisins
  - difficile dans le contexte d'une approche orientée bibliothèque logicielle

- Stratégies statiques = moins générales
  - Multiplication de matrices (LU et QR)

$\Rightarrow$ Distribution de tâches indépendantes :
- sur un réseau linéaire de processeurs hétérogènes : facile
- sur une grille bidimensionnelle de processeurs : difficile

$\Rightarrow$ Ne pas prendre en compte les contraintes géométriques induites par les deux dimensions de la grille conduit à :
- partitionnements irréguliers
- bon équilibrage de charge mais sont bien plus difficile à mettre en œuvre

Travaux antérieurs

- Stratégies d'équilibrage de charge sur plate-formes homogènes / hétérogènes
- Grande majorité : stratégies dynamiques
  - « utiliser le passé pour prédire le futur »
  - données sont échangées uniquement entre processeurs voisins
  - difficile dans le contexte d'une approche orientée bibliothèque logicielle
- Stratégies statiques = moins générales
  - Multiplication de matrices (LU et QR)

$\Rightarrow$ Distribution de tâches indépendantes :

- sur un réseau linéaire de processeurs hétérogènes : facile
- sur une grille bidimensionnelle de processeurs : difficile

$\Rightarrow$ Ne pas prendre en compte les contraintes géométriques induites par les deux dimensions de la grille conduit à :

- partitionnements irréguliers
- bon équilibrage de charge mais sont bien plus difficile à mettre en œuvre

Travaux antérieurs

- Stratégies d'équilibrage de charge sur plate-formes homogènes / hétérogènes
- Grande majorité : stratégies dynamiques
  - « utiliser le passé pour prédire le futur »
  - données sont échangées uniquement entre processeurs voisins
  - difficile dans le contexte d'une approche orientée bibliothèque logicielle
- Stratégies statiques = moins générales
  - Multiplication de matrices (LU et QR)

$\Rightarrow$ Distribution de tâches indépendantes :
- sur un réseau linéaire de processeurs hétérogènes : facile
- sur une grille bidimensionnelle de processeurs : difficile

$\Rightarrow$ Ne pas prendre en compte les contraintes géométriques induites par les deux dimensions de la grille conduit à :
- partitionnements irréguliers
- bon équilibrage de charge mais sont bien plus difficile à mettre en œuvre

Travaux antérieurs

- Stratégies d'équilibrage de charge sur plate-formes homogènes / hétérogènes
- Grande majorité : stratégies dynamiques
  - « utiliser le passé pour prédire le futur »
  - données sont échangées uniquement entre processeurs voisins
  - difficile dans le contexte d'une approche orientée bibliothèque logicielle
- Stratégies statiques = moins générales
  - Multiplication de matrices (LU et QR)

$\Rightarrow$ Distribution de tâches indépendantes :

- sur un réseau linéaire de processeurs hétérogènes : facile
- sur une grille bidimensionnelle de processeurs : difficile

$\Rightarrow$ Ne pas prendre en compte les contraintes géométriques induites par les deux dimensions de la grille conduit à :

- partitionnements irréguliers
- bon équilibrage de charge mais sont bien plus difficile à mettre en œuvre

Travaux antérieurs

- Stratégies d'équilibrage de charge sur plate-formes homogènes / hétérogènes
- Grande majorité : stratégies dynamiques
  - « utiliser le passé pour prédire le futur »
  - données sont échangées uniquement entre processeurs voisins
  - difficile dans le contexte d'une approche orientée bibliothèque logicielle
- Stratégies statiques = moins générales
  - Multiplication de matrices (LU et QR)

$\Rightarrow$ Distribution de tâches indépendantes :
- sur un réseau linéaire de processeurs hétérogènes : facile
- sur une grille bidimensionnelle de processeurs : difficile

$\Rightarrow$ Ne pas prendre en compte les contraintes géométriques induites par les deux dimensions de la grille conduit à :
- partitionnements irréguliers
- bon équilibrage de charge mais sont bien plus difficile à mettre en œuvre

Conclusion

– Difficultés sur les plate-formes hétérogènes : ☹
  - difficulté supplémentaire à équilibrer la charge
  - données et calculs ne sont pas répartis équitablement entre les processeurs
  - minimiser les surcoûts de communication devient alors une tâche ardue

– Principaux résultats : ☺
  - NP-complétude du problème d'optimisation SHAREDRING
    $\Rightarrow$ fournit une nouvelle preuve de la difficulté intrinsèque de la conception d'algorithmes hétérogènes
  - la conception d'une heuristique efficace, qui fournit une ligne de conduite pragmatique au concepteur de calculs scientifiques itératifs

– Travaux futurs : ☹☺
  - modèles de plate-formes plus précis
  - adaptation aux changements de capacité des ressources
    $\Rightarrow$ reste valable lorsque l'on injecte de la connaissance dynamique dans les ordonnanceurs statiques ?

Max-min fairness

Goal : maximize $\min_i d_i$

Motivation

Each path receives the same bandwidth through bottleneck links

$$\forall p_i \in \mathcal{P}, \quad \exists e \in p_i, \qquad \sum_{p_j \ni e} d_j = u(e) \text{ and } d_i = \max\{d_j, p_j \ni e\}$$