Vers un solveur direct hiérarchique hautes performances

COMPAS 2024

4 juillet 2024

Abel Calluaud [1,2]

[1] Commissariat à l'énergie atomique et aux énergies alternatives
[2] Univ. Bordeaux, CNRS, Bordeaux INP, INRIA, LaBRI, UMR 5800

Sommaire

Problématique

État de l'art

Feuille de route

Résultats

Problématique

Sujet

Résolution directe de systèmes linéaires à grande échelle.

→ **Calcul parallèle hautes performances**

→ **Techniques de compression**

Applications cibles

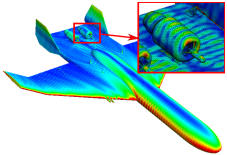


Figure: Électromagnétisme (CEA CESTA)

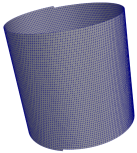


Figure: Aéroacoustique (Airbus)

Passage à l'échelle de la résolution directe de systèmes linéaires

1: Factorization : $A = LU$
$O(n^3)$ opérations

2: Résolution : $Ly = b$ et
$Ux = y$ $O(n^2)$ opérations

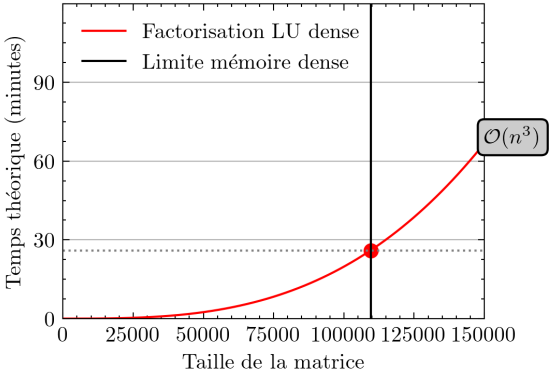


Figure: Performance théoriques de la **factorisation LU dense** en *double complexe* par rapport aux **limites mémoires et calculatoires** du matériel (2 x 18-core Intel CascadeLake, 3.9 GHz, 512-bit SIMD, 192GB RAM)

Passage à l'échelle de la résolution directe de systèmes linéaires

1: Factorization : $A = LU$

$O(n^3)$ opérations

2: Résolution : $Ly = b$ et $Ux = y$ $O(n^2)$ opérations

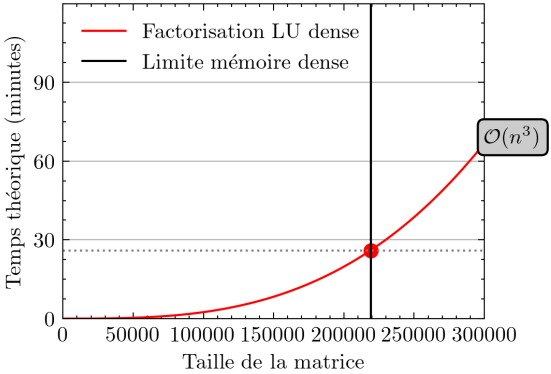


Figure: Performance théoriques de la **factorisation LU dense** en *simple précision* par rapport aux **limites mémoires et calculatoires** du matériel (2 x 18-core Intel CascadeLake, 3.9 GHz, 512-bit SIMD, 192GB RAM)

Approximation de rang faible

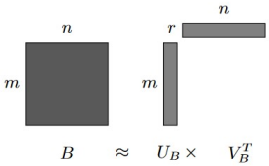


Figure: Approximation de rang faible

- ✚ Représentation d'une matrice B par une matrice de rang plus faible.
- ✚ Décomposition obtenue par la SVD, les variantes QR ou ACA.

$\Rightarrow$ **Réduction du coût de stockage et de calcul.**

Figure: Exemple en compression d'images

Format Tile Low Rank (TLR)

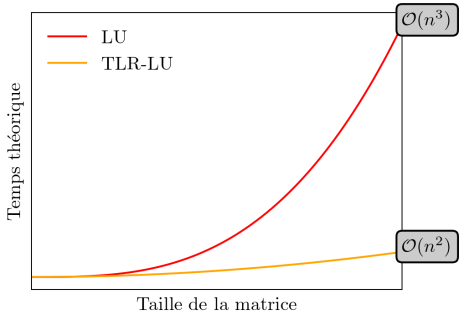


Figure: Temps de factorisation LU en fonction de la taille de matrice

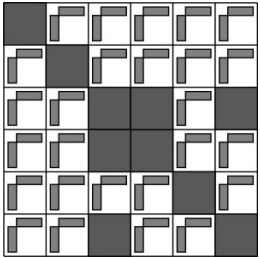


Figure: Format Tile Low Rank (BLR)

Stockage: $\mathcal{O}(n^{1.5})$

Calcul: $\mathcal{O}(n^2)$

Format Hiérarchical Low Rank (HLR)

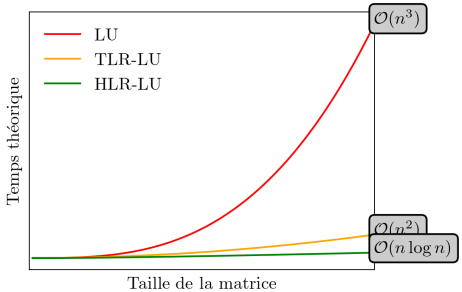


Figure: Temps de factorisation LU en fonction de la taille de matrice

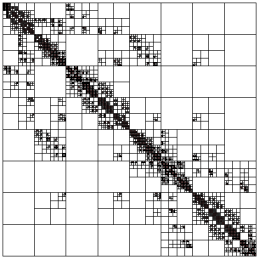


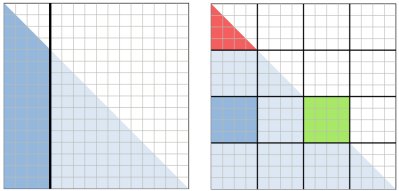
Figure: Format hiérarchique

Stockage: $\mathcal{O}(n \log(n))$

Calcul: $\mathcal{O}(n \log(n))$

Etat de l'art des solveurs directes

Algorithme panel vs tuile



- ✚ exprimer du **parallélisme de tâche** à grain fin.
- ✚ **librairies** d'agèbre linéaire **optimisées** sur des tuiles **locales**.
- ✚ **support d'exécution** basé sur un **paradigme de tâches**.

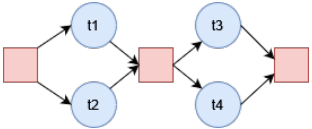


Figure: Paradigme fork-join

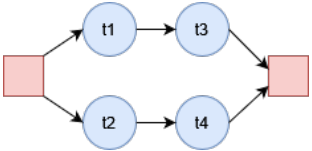


Figure: Paradigme de tâches

Modèle de programmation par les tâches

- Soumission de tâches avec des annotations pour les d'accès aux données : lecture seule (R), écriture (RW).
- Ordonnancement délégué à un support d'exécution basé sur le modèle Sequential Task Flow

```
for (k=0;k<NB;k++){
  POTRF(A_kk^RW)
  for (j=k+1;j<NB;j++){
    TRSM(A_kk^R, A_jk^RW)
    SYRK(A_jk^R, A_jj^RW)
    for (i=k+1;i<j;i++)
      GEMM(A_jk^R, A_ik^R, A_ij^RW)
  }
}
```

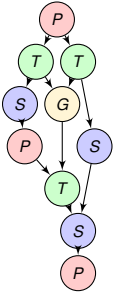


Figure: Factorisation de Cholesky via le modèle STF

Vers un solveur direct hiérarchique hautes performances

Objectif

Concevoir un solveur direct scalable pour l'algèbre linéaire dense avec compression de rang faible.

Briques de construction

- ✚ Moteur de tâches asynchrone scalable
- ✚ Décomposition fine des calculs
- ✚ Algorithmes d'algèbre linéaire tuilés
- ✚ Noyaux d'algèbre de rang faible

Vers un solveur direct hiérarchique hautes performance

Objectif

Concevoir un solveur direct scalable pour l'algèbre linéaire dense avec compression de rang faible.

Briques de construction

- Moteur de tâches asynchrone scalable : StarPU
- Décomposition fine des calculs : tâches récursives de StarPU
- Algorithmes d'algèbre linéaire tuilés : CHAMELEON
- Noyaux de compression de rang faible : RAPACK = BLAS + LAPACK + PASTIX

Vers un solveur direct hiérarchique hautes performances

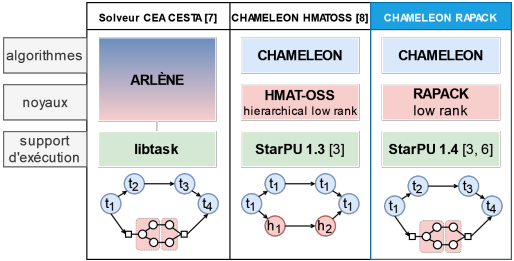


Figure: Comparaison du projet de recherche avec les logiciels existants

Atouts de l'approche proposée

- ✓ Modèle de programmation par tâches récursives
  - ➢ Décomposition fine des calculs
  - ➢ Granularité des tâches
  - ➢ Scalabilité
- ✓ Noyaux de compressions alternatifs : PQRCP, RQRCP, TQRCP, RQRRT

Performances en mémoire partagée

Configuration logiciel

Opération : Cholesky

Générateur : Aléatoire

Matrice : Matrice de rang 2%

Taille de tuile : 512

Précision à 10^{-8}

Configuration matériel

Arithmétique : double complexe

Noeud : bora 2 x *Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz*.

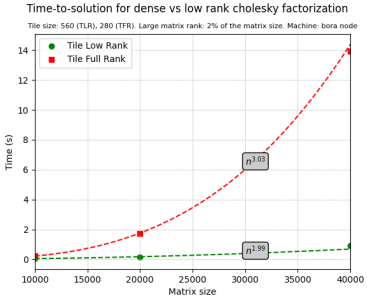


Figure: Performance de la factorisation de Cholesky en format Tuile et TLR

Performance en mémoire distribuée

Configuration matériel

Arithmétique : double complexe

Noeud : *bora* 2 x 18-core Intel CascadeLake, 3.9 GHz, 512-bit SIMD, 192GB RAM

Parallélisme noeud : 35 threads de calcul et 1 thread de communication par noeud.

Parallélisme distribué : 4 noeuds, 1 processus MPI par noeud.

Paramètres numériques

- Interaction $K(x_i, x_j) = exp(ikd)/d$ ou $d = |x_i - x_j|$.
- Précision : 10^{-4}

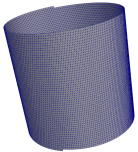


Figure: Nuage de points autour d'un cylindre (Airbus)

Reproductible via Guix : `compas24.calluaud.net`

Choix de la taille de tuile

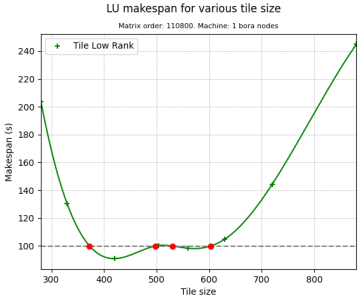


Figure: Temps de factorisation LU en fonction de la taille de tuile en mémoire partagée (1 noeuds `bora`)

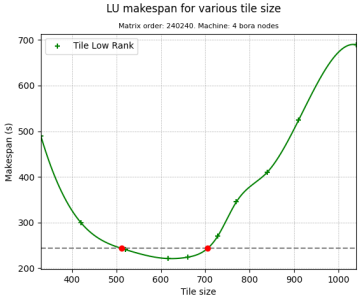


Figure: Temps de factorisation LU en fonction de la taille de tuile en mémoire distribuée (4 noeuds `bora`)

Choix d'un compromis pour les expériences suivantes : 512

Scalabilité en mémoire distribuée

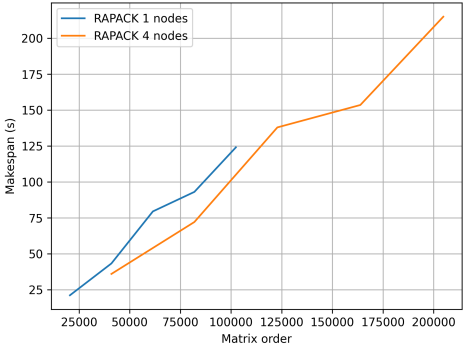


Figure: Performances de la factorisation LU en mémoire distribuée

Performances avec les noyaux de compressions alternatif

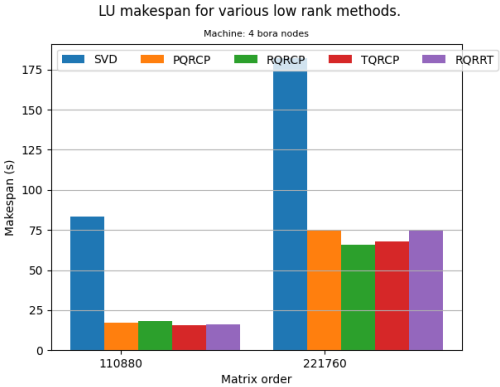


Figure: Performances de la factorisation LU en mémoire distribuée en fonction de la méthode de compression de rang faible

Conclusion

Résultats

- **Performances** : Scalabilité en mémoire partagée et distribuée du en mono-niveau : Factorisation LU, Cholesky, Multiplication de matrices
- **Performances des noyaux de compression** : PQRCP, RQRCP, TQRCP intéressants.

Perspectives

- Validation des performances sur d'autres matrices.
- Etude de la précision selon le noyaux de compression utilisé.

→ **Extension au cas hiérarchique.**