Algorithmique et techniques de base du calcul parallèle

Important. L'examen est composé de deux problèmes indépendants, qui doivent être rédigés sur des feuilles séparées. Tous les documents sont autorisés.

Exercice I. Construction et analyse de programmes parallèles

Soit le problème :

- Entrée : $A = [a_0, \ldots, a_n]$ et $B = [b_0, \ldots, b_n]$ deux tableaux de réels (flottants)
- Sortie : le tableau $C = [c_0, \ldots, c_n]$ défini par $c_i = \bigoplus_{j=0}^{n} f(a_i, b_j) \quad \forall i = 0 \ldots n$

f est une fonction de coût constant α supposé connu (par exemple: $f(x, y) = \sqrt{x^2 + y^2}$).
$\oplus$ est une opération binaire, a priori non associative, implémentée par la fonction `OplusSeq(x,y)`, de coût $\tau_\oplus$ constant et négligeable devant α.
Ce problème est calculé par le programme séquentiel AllPairsSeq suivant, de coût $T_{seq}(n) \simeq \alpha.n^2$:

```
AllPairsSeq ( /* Input */ m , a[0 .. m ], b[ 0 .. m ],
              /* Output */    c[0 .. m ] )
   {
     for (int i = 0 ; i <= m; i++ )    /* Boucle séquentielle */
       {
         c[i] := f ( a[i] , b[0] ) ;
         for (int j = 1 ; j <= m; j++ )    /* Boucle séquentielle */
           c[i] := OplusSeq ( c[i],  f(a[i] , b[j] ) ;
       } ;
   } ;
```

Pour paralléliser ce programme, on partitionne A, B et C en K blocs de n/K éléments chacun. On note $A[i]$ le bloc i de A : il contient les éléments $a[i * K]..a[i * K + N/K - 1]$.
Le programme parallèle s'écrit alors :

```
AllBlocPairsPar ( /* Input */    n,   A[0] , .., A[K-1],     B[0] , .., A[K-1]
                  /* Output */        C[0] , .., C[K-1] )
   {
      for (int i = 0 ; i < K; i++ )    /* DOPAR :  Boucle parallele */
        {
          for (int j = 0 ; j < K; j++ )    /* Boucle parallèle */
             AllPairsSeq ( n/K, A[i], B[j], tmp[i,j] )  ;
          C[i] := tmp[i,0] ;
          for (int j = 1 ; j < K; j++ )    /* Boucle séquentielle */
             C[i] := OplusBlocSeq (C[i], tmp[i,j] ;
        } ;
   } ;
```

1. (2 points) Dessiner le graphe de flots de données macroscopique correspondant à l'exécution de `AllPairsPar` pour $K = 3$.
NB: on ne cherchera pas à paralléliser les appels à `OplusBlocSeq`, dont le coût pour un bloc de taille n/K est supposé $\frac{n}{K}\tau_\oplus$.

2. (2 points) Pour K donné, calculer $T_1(n)$ et $T_\infty(n)$. Comment choisir K pour obtenir une parallélisation efficace ?

3. (4 points) On se place maintenant sur une machine NUMA avec p processeurs identiques. Calculer les coûts de communication dans les 2 cas suivants :

1. **Parallélisation par distribution de données :** on choisit $K = p$; tous les calculs effectués sur un même bloc C_i sont effectués sur un même processeur.
   Calculer les coûts de communication $\mathcal{C}_1$ et $\mathcal{C}_\infty$.

2. **Parallélisation par distribution de calculs :** on choisit $p = K^2$ et on exécute chaque appel `AllPairsSeq ( n/K, A[i], B[j], tmp[i,j] )` sur un processeur différent.
   Calculer les coûts de communication $\mathcal{C}_1$ et $\mathcal{C}_\infty$.

3. En déduire la meilleure parallélisation.

4. (2 points) On suppose maintenant que $\oplus$ est associative; les opérations `OplusBlocSeq` entre les blocs `tmp[i,j]` pour un même bloc `C[i]` peuvent donc être faites en parallèle sous forme d'arbre binaire de réduction.

1. Quel choix de K minimise T_∞ tout en laissant T_1 proche de T_{seq} ?

2. Proposer pour chacune des deux distributions sur NUMA (questions 3.a et 3.b) un placement de l'arbre de réduction. Que deviennent, dans chacun de ces deux cas, $\mathcal{C}_1$ et $\mathcal{C}_\infty$?