TD 1 : PRAM

1 Systèmes Parallèles et Distribués

A.1 Rappeler les différentes catégories de systèmes distribués et donner pour chacune d'elle un exemple (si possible ne provenant pas du cours).

A.2 Rappeler les différents modes de fonctionnement d'une PRAM. Montrer que certains d'entre eux peuvent également s'interpréter en terme d'architecture distribuée.

2 Tri et Réseaux de Tri

On dispose de p processeurs organisés linairement. C'est-à-dire que chaque processeur dispose d'une zone de mémoire propre qui ne peut être écrite que par lui-même.

On souhaite trier n entiers, répartis régulièrement sur chacun des processeurs.

2.1 Tri par insertion

```
for i from 1 to n do
  p = 1
  for j from 1 to n do
    if (E(j)<E(i)) then
      p = p + 1
    end if
  end for
  T(p) = E(i)
end for
```

B.1 Proposez un algorithme pour PRAM.

B.2 Calculez la complexité.

2.2 Tri à bulle

Pour cet exercice chaque PRAM ne peut lire que ces voisins.

B.3 Proposer un programme PRAM de tri pour $p = n$.

B.4 Quelle est sa complexité ? Son efficacité ?

B.5 Pourrait-on être aussi efficace en EREW ?

On dispose désormais le tableau de n entiers en tableaux de $\frac{n}{p}$ entiers sur chaque processeur.

B.6 Proposez une adaptation du programme précédent.

B.7 Quelle est la complexité ? L'efficacité ?

B.8 Question optionnelle : pour quelles valeurs de p le tri est-il optimal ?

2.3 Tri par fusion

B.9 Proposez un algorithme de tri par fusion.

B.10 Vérifiez que vous avez une complexité de $O(log_2(n))$.

B.11 Pouvez vous l'adapter pour un PRAM de type EREW ?

3 Sommes partielles

Voici l'algorithme vu en cours :

```
pourchaque proc i en parallele {
  y[i] = x[i]
  tantque ( il existe proc i tq next[i] != NULL ) {
    si ( next[i] != NULL ) {
      y[next[i]] = op(y[i], y[next[i]])
      next[i] = next[next[i]] // saut de pointeur
    }
  }
}
```

C.1 Prouvez l'algorithme

C.2 Calculez la complexité.