Compléxité: Espace

Celle ci sera la même partout car le but dans tous les algorithmes est le même, le partage d'information, à la fin de l'exécution, tout processus aura une copie entière du vecteur.

Soit n le nombre de processus, et x la place en mémoire pour 1 vecteur. Alors la complexité en mémoire est:

$$O(n*x)$$

Compléxité: Temps

Ici nous aurons des complexités différentes pour chaque algorithme.

Simple: Pour cet algorithme on a une simple boucle *for* qui fait une itération par nombre de processus (-1). Disons que chaque itération prend t temps et qu'il y a n processus, alors la complexité est:

$$O((n-1)*t) \text{ , qu'on écrit } O(n*t)$$

Anneau: Maintenant c'est où nous voyons l'intérêt du parallélisme, on va faire le même nombre d'itérations, mais elles seront faites en parallèle par deux processus différents plutôt qu'un seul. Soit le temps de chaque itération temps t et qu'il y a n processus:

$$O(2*t + (n-3)/2 * t), \text{ qu'on écrit } O((n/2)*t)$$

Hypercube: Pour ce dernier le principe est exactement le même que avant, sauf que plutôt qu'avoir 2 processus qui envoient le vecteur, on en aura plus:

$$\begin{aligned} \text{cycle 1 -> } & \text{ on a 1 processus} \\ \text{cycle 2 -> } & \text{ on a 2 processus} \\ \text{cycle 3 -> } & \text{ on a 4 processus} \\ & \dots \\ \text{cycle n -> } & \text{ on a } 2^{(n-1)} \text{ processus} \end{aligned}$$

Pour n processus ils nous faudra donc $log_2(n)$ cycles. On a toujours temps t par cycle

$$O(log_2(n)*t)$$

Compléxité: Temps