Description haut niveau de l’implémentation :

On utilise akka. Une méthode main créé N process, et leur donne la liste de ces N process. La méthode main envoie ensuite launch aux process qui choisissent aléatoirement chacun 0 ou 1. Ensuite ils proposent la valeur choisie.

L’implémentation est ensuite vraiment très proche de celle déjà donnée dans le cours. Différentes classes codes les différents types de messages que peuvent s’envoyer les process (AckMsg, DecideMsg, etc…). A la réception d’un message, un process traite le message reçu en fonction de la classe de ce message (il y a une méthode pour chaque type de message : ackReceived, decideReceived, etc.).

Les petites astuces non précisées dans le cours sont :

* Lors de l’exécution d’une méthode suite à la réception de certains messages réponse, le process vérifie que :
  + Le process n’a pas crash
  + L’opération qu’il faisait n’a pas déjà aborté (boolean)
  + le message qui lui a été envoyé a un ballot correspondant au ballot de l’opération en cours
* il y a un compteur pour compter le nombre de réponse Ack ou Gather
* lorsque qu’un process aborte, si il n’est pas un non learder (il est en hold) et si il n’a pas encore décidé, il va reproposer la valeur qu’il avait choisie aléatoirement au début
* lorsque qu’un acteur reçoit un message Decide, il ne le renvoit pas à tout le monde parce que c’est inutile comme tous le monde connait tout le monde (l’émetteur de ce message s’en est déjà chargé). C’est plus simple algorithmiquement comme ça.

Voilà les résultats obtenus pour les différents paramètres demandés :

|  |  |  |  |
| --- | --- | --- | --- |
| **N** | **f** | **tle** | **Moyenne temps** |
| 3 | 1 | 0,5 | 0,0122 |
| 3 | 1 | 1 | 0,0106 |
| 3 | 1 | 1,5 | 0,011 |
| 3 | 1 | 2 | 0,0058 |
| 10 | 4 | 0,5 | 0,0248 |
| 10 | 4 | 1 | 0,015 |
| 10 | 4 | 1,5 | 0,025 |
| 10 | 4 | 2 | 0,0242 |
| 100 | 49 | 0,5 | 0,5548 |
| 100 | 49 | 1 | 1,0382 |
| 100 | 49 | 1,5 | 1,5418 |
| 100 | 49 | 2 | 2,0466 |

Par peur de perdre du temps inutilement à chercher une solution peut-être difficile ou même inexistante, le relevé de ces 60 points s’est fait à la main avec des copié-collé des valeurs de la fenêtre de commande jusqu’à un Excel.

On remarque que plus N augmente, plus le temps de Decide est long.

Pour N=3 et N=10, les process décident suffisamment vite (avant 0.5 secondes) pour que la valeur de tle n’ait aucun impact sur la moyenne du temps pour décider.

En revanche pour N=100, c’est quasiment tle qui détermine le temps pour décider. En essayant avec une valeur tle=50 secondes, on se rend compte que les process mettent au moins 50 secondes pour se décider, comme quoi avec un nombre N>=100 et f proche de N/2 c’est déjà très dur de se décider sans leader !

Pour N=100, tle=50 s et f=0, il faut environ 13 secondes (moyenne sur 4 tests, mais ça varie en réalité beaucoup) pour décider. L’impact de f est grand sur le temps de réponse.