Rapport de CSPC

BURNOT Jean-Christophe

REMILIEUX Maximilien

**Jeu de la vie:**

1) Méthode grille:

La méthode "grille" permet de générer une grille de cellules de taille 15 par 15 appelé "self.t", avec des cellules initialement toutes mortes.

Les cellules mortes ont pour valeur "False" et sont représentées par des croix rouges, alors que les cellules vivantes ont pour valeur "True" et sont représentées par des ronds bleus.

De plus cette fonction appelle la fonction "plac\_cel".

2) Méthode place\_cel:

La méthode "place\_cel" permet de placer aléatoirement des cellules vivantes dans le tableau de cellule "self.t", c’est-à-dire changer les valeurs des cases correspondant aux cellules par "True".

De plus cette fonction appelle la fonction "affichage".

3) La méthode Voisins:

La méthode "Voisins", à pour but de remplir un tableau "self.voisins", en comptant le nombre de voisins que possèdes chaque cellule. Pour cela on parcourt la grille de cellule "self.t" et on regarde les 8 casses autours. On ajoute 1 dans le tableau "self.voisins" si la cellule est vivante (c’est-à-dire à l’état "True").

De plus cette fonction appelle la fonction "survivre".

4) La méthode survivre:

La méthode survivre est la méthode qui applique les règles du jeu de la vie à chacune des cellules que l’on a à placer dans le tableau "self.t". Pour cela on parcourt le tableau "self.t" et on regarde le nombre de voisins dans le tableau "self.voisins":

* Si la valeur est "True" et que le nombre de voisins n’est pas compris entre [2,3] alors la cellule meurt, c’est-à-dire que l’on change sa valeur par "False".
* Si la valeur est "False" et que le nombre de voisin est supérieur à 3 alors la cellule vie, c’est-à-dire que l’on change sa valeur par "True".

De plus cette fonction appelle la fonction "affichage".

5) La méthode affichage:

La méthode affichage permet d’afficher basiquement dans la console les cellules. Pour cela elle parcourt le tableau de cellule "self.t":

* Si la valeur est "True" elle affiche un cercle bleu
* Si la valeur est "False" elle affiche une croix rouge

6) Programme principal:

Le programme principal sert simplement à créer un objet de la classe "GameOfLive" et d’appeler la fonction "grille" une fois pour initialiser et créer le tableau de cellules. Puis il appelle plusieurs fois la fonction "Voisins" pour appliquer les règles du jeu de la vie.

Remarque: Je crois avoir encore quelques petits problèmes en mode "multiprocess", donc j’ai préféré laisser en commentaire la partie en séquentiel. Pour passer du mode "multiprocess", il faut inverser la partie commentée avec la partie décommentée.

**Calcul de Pi par la méthode de Hits-Miss de Monte Carlo:**

Pour cet exercice nous avons calculé la valeur de Pi avec la méthode de Hits-Miss de Monte Carlo en séquentiel et en mulitiprocess pour pouvoir comparer les temps de calcul.

Pour cela on a initialisé une variable partager "nb\_hits", et demandé à chaque processus d’appliquer la méthode "frequence\_de\_hits\_pour\_n\_essais" avant de regrouper toute ces valeurs dans la liste "list\_pros" pour estimer une valeur approcher de Pi.

On compare ensuite les temps de calculs et on les affiche dans la console.

**Restaurant:**

Tous le code repose sur la gestion par sémaphores des tampons et des listes partagées. Les Matrices importantes son "tampon" et "etatServeur", ces deux listes seront modifiées par les processus serveur et client à l'aide des fonctions adéquates. La matrice "tampon" contient les commandes avec en premier paramètre un lettre associé à la commande et en second le client qui a passé commande

1)clients: Processus qui crée toutes les secondes des commandes si le tampon n'est pas plein. Il attribue aux commandes crées une lettre et un client de manière aléatoire.

2)serveur: Fonction qui va être exécutée par plusieurs processus simultanément et qui va traité les commande. Le processus décale le tampon si il détecte une commande (action protégée par le sémaphore "semTampon").

En dehors de ce sémaphore on refait le test pour savoir si il y a une commande et on modifie l'état du serveur qui va traiter la commande, puis au bout d'un certain temps on repasse sont état à 0 et on incrémente la valeur "pService".

3)patron: C'est la fonction d'affichage du code, il récupère un ensemble d'information sous la contrainte des 3 sémaphores et les affiche. C'est lui qui traduira la première colonne de tampon en lettre

4)Autres fonctions: Les autres fonctions servent a:

* fArrayToList": Convertir les Array en listes
* "fIntListToAlphabet", "fIntToAlphabet": Convertir un entier ou une liste d'entiers en lettres.
* "fDecaleurListe", "fResetListe": Permettent de faire de opérations sur les listes ici un décalage et une mise à 0.

**Robot:**

Le code n'est pas fonctionnel et s'apparente en l'état à une fork bomb.