**Projet de programmation du premier semestre**

**Organisation générale du programme :**

Le programme commence par une fonction *main* qui contient toute les fonction du programme. Dans ce main on commence par récupérer les données envoyées dans le terminal, les vérifier, puis les envoyer vers la fonction *contexte.* Cette fonction va faire varier les contextes de densité de population et de vaccination pour ensuite appeler la fonction *simulation*, qui elle effectue réellement les simulations. Le principe d’abstraction et de réutilisation est essentiellement utilisé dans la fonction *simulation.*

**Fonctionnement:**

La fonction *main* qui enveloppe tout le programme commence par appeler *lire\_verbose* qui s’occupe de retenir si les messages d’invitation doivent être affichés ou non tout au long du programme. Elle appelle ensuite *afficher\_grille* qui mémorise si la grille doit être affichée ou non lors des simulations.

La fonction *lire\_param* est ensuite appelée. Elle lit les paramètres d’entrée et initialise nos paramêtres *nbP, nbSim et taille\_monde.* Si les valeurs ne respectent pas les conditions, elle appelle la fonction d’erreur correspondante.

*Init\_tab\_personne* est la dernière fonction appelée par *main*; elle s’occupe de lire les données en entrée. Elle lit leur position initiale ainsi que leur but initial, les vérifie, les stocke dans un tableau de référence et appelle la fonction d’erreur correspondante.

*Contexte* est la dernière fonction appellée par main ; elle s’occupe de faire varier les contextes de densité de population et de taux de vaccination lors des simulations.

Cette fonction commence par attribuer un état de santé initial à nos personne grâce à la fonction *etat\_sante* puis contamine les personnes saines se trouvant initialement proche d’un contaminé en appelant *personne\_contaminée\_autour*. Elle remplit alors un tableau *tab\_pers* qui sera utilisé pour la simulation en copiant celui de référence. Elle effectue ensuite la *simulation* et remplit un tableau *resultat\_sim* avec la valeur du nombre de cycles renvoyée par la fonction *simulation*. Une fonction *tri* trie alors le tableau des résultats et la fonction *valeur\_mediane* nous renvoie la valeur médiane de la durée de contamination en *cycle*.

On affiche alors dans le terminal la valeur de la densité, celle du taux de vaccination et celle du nombre de cycles effectué pour un contexte.

La fontion *simulation* est elle aussi divisée en 6 sous fonctions. Certaines fonctions comme *incubation\_contamination,* qui passe les gens de l’état incubé à l’état contaminéou la fonction *personne\_contaminee\_autour,* qui cherche les personnes contaminéee autour de notre personne déplacée,sont même appellées deux fois. Elles sont donc réutilisées.

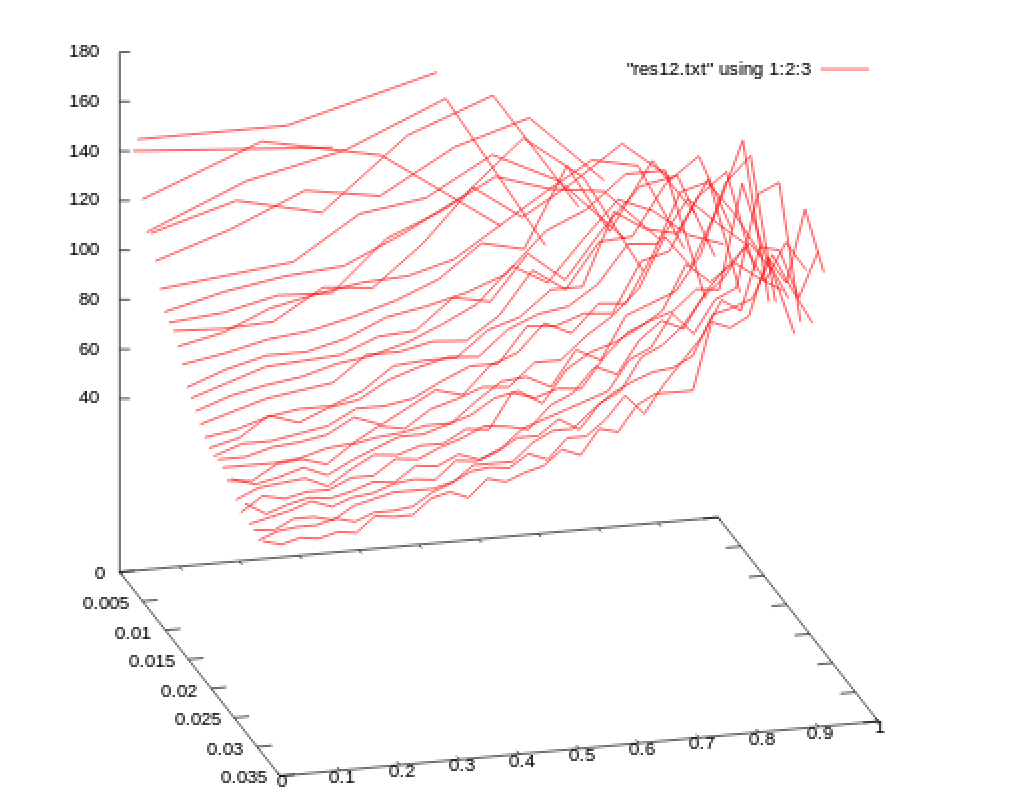
La partie déplacement est sous-traitée dans une fonction *deplacement* qui elle-même appelle 3 fonctions :

*sng* qui calcule le signe d’une opération, *case\_occupee* qui vérifie si la case est occupée et *nouveau\_but*, qui gènère un nouveau but aléatoire et qui est aussi appellée par *simulation.*

*contamine\_pers\_autour* s’occupe de contaminer les personnes saines lors du déplacement d’un contaminé.

*nb\_personne\_saine* compte le nombre de personnes encore saines.

Finalement *affichage\_grille* s’occupe de l’affichage de la grille et appelle la fonction *etat\_sante\_actuel* qui lui renvoie l’état de santé des personnes.

**Analyse du résultat du test 12 :**

On observe un résultat similaire à celui produit par le programme exemple.

La densité et le taux de vaccination influencent la durée des simulations.

On voit que plus la densité de personnes diminue plus la durée des simulations augmente car les personnes se rencontrent de moins en moins fréquemment.

Alors que pour le taux de vaccination c’est l’inverse. De surcroit, on observe qu’il faut un taux de vaccination élevé pour que la durée de contamination augmente. Seule la dernière valeur plonge vers le bas mais cela a été expliqué sur le forum.

**Pseudocode de la fonction contexte :**

**En entrée :** *n* : taille du monde, *nb\_pers*: nombre de personnes, *nb\_sim*: le nombre de simulation ,

*tab\_ref*: un tableau contenant les informations sur les personnes.

NB\_INFOS\_PERS 🡨 5

NB\_PERS\_MIN 🡨 2

resultat\_sim[nb\_sim]

densite

taux\_de\_vaccination

tab\_pers[nb\_pers][NB\_INFOS\_PERS]

Pour nbP\_prime allant de nb\_pers à NB\_PERS \_MIN par pas de -1

Pour nbP\_vac allant de 0 à nbP\_prime – NB\_PERS\_MIN

Pour i allant de 0 à nb\_sim

remplir\_tab\_pers(nb\_pers,tab\_ref, tab\_pers)

init\_etat\_sante(nbP\_vac, nb\_pers, tab\_pers)

pour j allant de 0 à nbP\_prime

posC = tab\_pers[j][POS\_C];

posL = tab\_pers[j][POS\_L];

etat\_sante = tab\_pers[j][SANTE];

si etat\_sante = SAIN

et si personne\_contaminee\_autour(posC, posL, n, nb\_pers,

tab\_pers, nbP\_prime) = 1

alors tab\_pers[j][SANTE] 🡨INCUBATION

resultat\_sim[i] = simulation(nb\_sim, n, nb\_pers, tab\_pers, nbP\_prime)

densite🡨nbP\_prime/(n\*n)

taux\_de\_vaccination🡨(nbP\_vac)/nbP\_prime

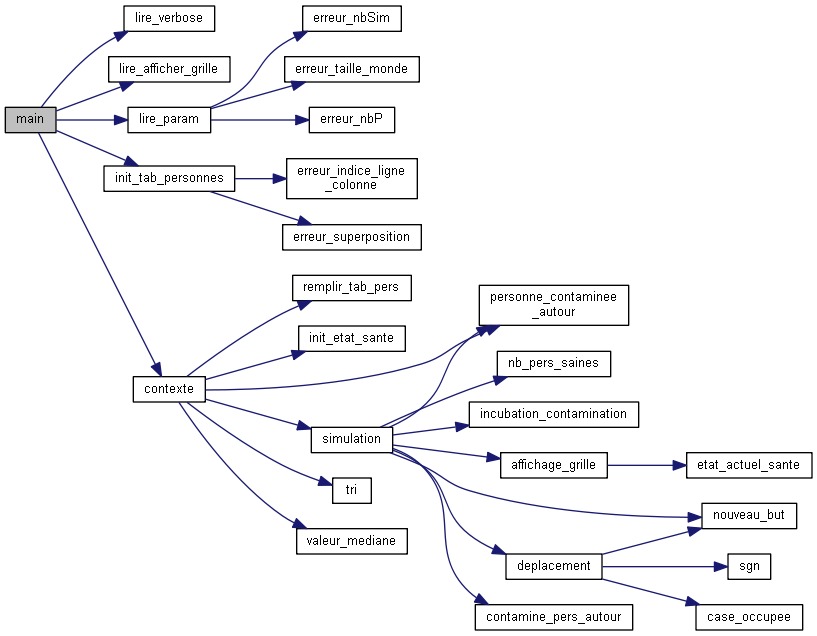
tri(resultat\_sim, nb\_sim)

Afficher densite , taux\_de\_vaccination, valeur\_mediane(resultat\_sim, nb\_sim));

Afficher « \n »

Complexité de l’algorithme en O(

**Graphique des appels de fonctions :**

**