Rapport Projet

Gestion de la Concurrence

SI4

MOLINIER Julien & LERAS Maxime

4/11/2019

Table des Matières

1 Prérequis

Les librairies suivantes doivent impérativement être installer avant le lancement du programme :

- pygame : permet d'afficher un visuel pour verifier le fonctionnement de l'algorithme

2 Scénario à 1 seul thread

2.1 Principe de fonctionnement de l'algorithme

L'algorithme commence par trier la liste des personnes. Pour cela les personnes sont triées par distance euclidienne d_i décroissante avec la sortie.

$$d_i = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2}$$

Il fait ensuite avancer les personnes en parcourant cette liste triée. Dans le cas idéal il fait avancer une personne en diagonale vers la sortie car il s'agit du chemin le plus court. Si une personne ou un obstacle bloque se déplacement il bouge la personne dans le sens des x décroissants ou bien dans un second choix vers les y décroissants. Dans le cas limite où tous les déplacements seraient bloqués la personne ne bouge pas et attend le prochain tour. L'algorithme boucle ainsi et supprime de la liste chaque personne se présentant devant la sortie. Lorsque la liste est vide l'algorithme s'arrête car toutes les personnes ont réussi à sortir.

Voici le pseudo-code :

```
Fonction algorithme0() :
Début :
    Tant que longueur(people) !=0 :
        people = sort(people)
        Pour chaque p dans people :
            Si p[0]==0 et p[1]==0
                Supprimer p de people
            Si canGoHere(p[0] - 1, p[1] - 1) :
                Deplacer p
            Si canGoHere(p[0] - 1, p[1]) :
                Deplacer p
```

```
                    Si canGoHere(p[0], p[1] - 1) :
                        Deplacer p
                Fin Pour
        Fin Tant que
    Fin
```

2.2 Performance de l'algorithme

Afin de mesurer les performances de l'algorithme, le paramètre -m donne le temps moyen que met la foule à sortir du terrain. On reproduit 5 fois l'algorithme et on supprime le temps le plus court et le temps le plus long pour obtenir un résultat précis qui n'est pas influencé par l'échauffement du processeur avant d'obtenir des performances stables.

Voici un tableau représentant les durées moyennes de l'algorithme en fonction de la taille de la foule.

Taille de la foule	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9
Durée (s)	0.001	0.003	0.005	0.010	0.023	0.057	0.154	0.524	2.078