Compte-rendu Projet INF 304

EL-BOUCH ISMAIL / JIANG Yilun / MAMADOU HASSIMIOU DIALLO

Tp 6

Tout d´abord on a complété terrain.c, après on utiliser la procédure affiche_terrain pour vérifier la lecture (test_terrain.c)

Ensuite nous avons créé 2 répertoires terrains_corrects (qui contient 2 teste correcte : terrain_7x5, terrain_11x9) et Terrains_incorrects :(qui contient 8 tests incorrectes : terrain_ERREUR_HAUTEUR_INCORRECTE, terrain_ERREUR_LARGEUR_INCORRECTE, terrain_ERREUR_LECTURE_HAUTEUR, terrain_ERREUR_LECTURE_LARGEUR, terrain_ERREUR_LIGNE_TROP_COURTE, terrain_ERREUR_LIGNE_TROP_LONGUE, terrain_ERREUR_LIGNES_MANQUANTES, terrain_ERREUR_POSITION_ROBOT_MANQUANTE)

Et à la fin de ce TP on a complété la fonction robot _peut_avancer.

TP7

Dans ce TP On a commencé par compiler le programme curiosity.c à l´aide du Makefile,

Ensuite on a créé 2 tests simple « (chemin.test=pour avancer à gauche et à droite jusqu´à que le programme soit terminer, par ex :

terrain_11x9.// nom de fichier terrain

txt simple.prg //nom de fichier programme

9//le nombre de pas maximum d'exécution

N //si le robot est sur une position normale à l'intérieur du terrain,

9 4 //position x et y

E//est)

, Et pour le 2eme test : c´est testretourner= pour avancer et retourner à la position initiale »

Puis on a créé un jeu de tests permettant de tester interprete*i*.c (les test interprete0,2,5,6,4,8 ns sont pas correct et les test interprete1,3,7,9,sont correcte)

Finalemet on a fait des exemple de programmes déclenchant une erreur à la **lecture** de programme C-A-D TEST DE ROBUSTESSE (lecture_du_programme1 « { A A » ici on a pas ferme l`accolade « } » pour indique la fin de boucle) (et lecture_du_programme2 « Z » ici la lettre Z n´est pas dans le programme.h))

Et à l´**exécution** du programme (execution_du_programme «3- »-→il manque un 2eme nombre pour faire la soustraction)

TP8

On commencer par compléter la fonction generation_aleatoire du paquetage generaton_terrains, puis on a complété le programme test_generation_terrains.c. Apres on a

Ensuite pour test de performance on a commencé par compléter le programme curiosity-pref.c

De plus on a élaboré plusieurs programmes permettant au robot de tenter de sortir de n´importe quel terrain (par ex : {1 M {1 M {D A} {G} ?} {A} ? C !} C ! Dans le fichier programme0.prg)

Et on a testé ces programmes sur différents terrains en indiquant si le programme permet au robot de sortir ou de rester bloqué.

Apres on a testé ces programme sur différents terrains (par ex pour le ***programme1.prg*** on trouve ***:*** ***./curiosity-perf test-tp8/programme1.prg 10000 9 9 0.5 test-tp8/grain.txt 100 test-tp8/restest1.txt,*** avec un densité d´obstacle de 0.5

```
Total des terrains : 10000
Nombre et pourcentage de sorties : 1389 (13.889999%)
Nombre et pourcentage de terrains bloqués : 3177 (31.770000%)
Nombre et pourcentage de terrains crashés : 5434 (54.340000%)
Nombre moyen de pas pour sortir : 54.573074
```

Et si on change la densité a 0.7 on trouve :

```
Total des terrains : 10000
Nombre et pourcentage de sorties : 154 (1.540000%)
Nombre et pourcentage de terrains bloqués : 8830 (88.300003%)
Nombre et pourcentage de terrains crashés : 1016 (10.160000%)
Nombre moyen de pas pour sortir : 51.344154
```

)

Puis on tester 2 programmes sur des terrains de divers densités d´obstacles : par exemple pour le programme2 : « ./curiosity-perf test-tp8/programme2.prg 10000 9 9 0.5 test-tp8/grain.txt 100 test-tp8/restest2.txt » avec la densité =0.5 on obtient

```
Total des terrains : 10000
Nombre et pourcentage de sorties : 1376 (13.760000%)
Nombre et pourcentage de terrains bloqués : 3311 (33.110001%)
Nombre et pourcentage de terrains crashés : 5313 (53.130001%)
Nombre moyen de pas pour sortir : 58.212208
```

Et si on garde le même programme et on change la densité a 0.7 on obtient

```
Total des terrains : 10000
Nombre et pourcentage de sorties : 173 (1.730000%)
Nombre et pourcentage de terrains bloqués : 8909 (89.090004%)
Nombre et pourcentage de terrains crashés : 918 (9.179999%)
Nombre moyen de pas pour sortir : 55.398846
```

Dans ce test on observe que le nombre de sorties avec la densité d´obstacle 0.5 est égale à 1376 et quand on change la densité à 0.7 on trouve que le nombre de sortie égale 173.

Apres avoir effectué plusieurs tests, on a constaté que plus la densité d´obstacles augmente, le nombre de sorties diminue.

Finalement on a testée programme1 dans un premier temps avec n=20, L=H=20 et occupation totale de *40 %* « ./curiosity-perf test-tp8/programme1.prg 20 25 25 0.4 test-tp8/grain.txt 100 test-tp8/restest1_1.txt »on obtient :

```
Total des terrains : 20
Nombre et pourcentage de sorties : 0 (0.000000%)
Nombre et pourcentage de terrains bloqués : 18 (90.000000%)
Nombre et pourcentage de terrains crashés : 2 (10.000000%)
Nombre moyen de pas pour sortir : nan
```

Puis avec *n=20*, *L = H = 9*, occupation totale de *70 % on trouve :* « ./curiosity-perf test-tp8/programme1.prg 20 9 9 0.7 test-tp8/grain.txt 100 test-tp8/restest1_1.txt »

```
Total des terrains : 20
Nombre et pourcentage de sorties : 2 (10.000000%)
Nombre et pourcentage de terrains bloqués : 16 (80.000000%)
Nombre et pourcentage de terrains crashés : 2 (10.000000%)
Nombre moyen de pas pour sortir : 48.500000
```

Ensuite le programme2 avec n=*20*, *L = H = 25*, occupation totale de *40 % on obtient « ./curiosity-perf test-tp8/programme2.prg 20 25 25 0.4 test-tp8/grain.txt 100 test-tp8/restest1_1.txt »*

```
Total des terrains : 20
Nombre et pourcentage de sorties : 0 (0.000000%)
Nombre et pourcentage de terrains bloqués : 9 (45.000000%)
Nombre et pourcentage de terrains crashés : 11 (55.000000%)
Nombre moyen de pas pour sortir : nan
```

Puis avec *n=20*, *L = H = 9*, occupation totale de *70 % on trouve « ./curiosity-perf test-tp8/programme2.prg 20 9 9 0.7 test-tp8/grain.txt 100 test-tp8/restest1_1.txt »*

```
Total des terrains : 20
Nombre et pourcentage de sorties : 1 (5.000000%)
Nombre et pourcentage de terrains bloqués : 19 (95.000000%)
Nombre et pourcentage de terrains crashés : 0 (0.000000%)
Nombre moyen de pas pour sortir : 45.000000
```

Tp9

On observe que dans le test programme-robot-correct-acceptes et le programme-robot-incorrect-acceptée on obtient un résultat d´observation qui est valide après l´exécution.

Traduction sous forme d´automate avec une fonction de transition totale :