Compte-Rendu du Mini-Projet INF304

Anaïs Belal - Feth-Ellah Boudellal

INM-01 2023/2024

Lien GitHub : https://github.com/GovDz/INF304/ Si jamais le ZIP ne marche pas

I. TP07:

Pour le TP7, nous avons adopté une approche systématique en créant 24 fichiers tests pour explorer divers aspects du robot. Ces tests englobent des cas tels que l'utilisation de commandes :

- Boucles imbriquées.
- Groupes de commandes.
- Sorties de terrains.
- Commandes d'ignorance et de rotation.
- La division par zéro.
- Rencontres avec des obstacles (eau, rocher).
- Programmes trop grands.

Inspirés par l'APP1 de INF301, nous avons varié les terrains avec différents types d'obstacles pour simuler des conditions réelles. Chaque test est conçu pour évaluer les fonctionnalités du robot dans des scénarios spécifiques, contribuant ainsi à une évaluation complète et à la robustesse du programme dans des situations diverses.

II. TP08:

Parmi les sept programmes développés, seuls les programmes N°3 et N°7 ont affiché des résultats acceptables, démontrant une réussite limitée dans la conception des robots-programmes.

Ainsi nous avons créé un fichier en Python nommé "MassTestTP8.py" qui effectue l'ensemble des tests et affiche les statistiques résultantes dans un fichier appelé "stats.txt".

- Programme-Robot N°3 : `{1M{7M{D}{G}?}{A}?C!}C!`

Description supposée :

Le programme se déplace en ligne droite jusqu'à ce qu'il détecte un obstacle. Si un obstacle est en face, il vérifie la présence d'un obstacle à sa gauche. S'il n'y a pas d'obstacle à gauche, il tourne dans cette direction, sinon il tourne à droite. Ce cycle se répète continuellement.

Stats: Dans le Fichier (Stats.txt et Stats2.txt)

```
Test 3:
Tests total : 20
Réussites : 10
Échecs : 10
  - Chutes dans l'eau : 0
  - Écrasements contre un rocher : 0
  - Perdus : 10
Nombre moyen de pas pour les réussites : 465.60
Somme des pas : 4656
```

Et

```
Test 3:
Tests total : 20
Réussites : 15
Échecs : 5
  - Chutes dans l'eau : 0
  - Écrasements contre un rocher : 0
  - Perdus : 5
Nombre moyen de pas pour les réussites : 91.60
Somme des pas : 1374
```

- Programme-Robot N°7 : `{1M{7M{DA}{GA}?}{A}?C!}C!`

Description supposée :
Pareillement que pour le précédent programme, celui-ci avance tout droit constamment et puis s'il trouve un obstacle, il fait la mesure à gauche puis s'il y trouve un obstacle il tourne directement à droite et avance, sinon il continue sa démarche à gauche.

Stats: Dans le Fichier (Stats.txt et Stats2.txt)

```
Test 7:
Tests total : 20
Réussites : 2
Échecs : 18
  - Chutes dans l'eau : 8
  - Écrasements contre un rocher : 7
  - Perdus : 3
Nombre moyen de pas pour les réussites : 255.50
Somme des pas : 511
```

ET

```
Test 7:
Tests total : 20
Réussites : 14
Échecs : 6
  - Chutes dans l'eau : 0
  - Écrasements contre un rocher : 3
  - Perdus : 3
Nombre moyen de pas pour les réussites : 69.14
Somme des pas : 968
```

Remarque : Le second programme est plus performant en comparaison avec le premier car même si le nombre d'échecs est relativement plus élevé pour le deuxième, si on prend en compte le nombre de pas effectués on conclut que le deuxième programme exécute beaucoup moins de pas et donc prend beaucoup moins de temps pour lire tout le programme.

III. *TP09:*

Description de l'observateur et sa traduction en automate à états finis:

Le fichier observateur.c comporte 3 fonctions dont : initial, transitions, BoolFinal

Etats initial(): elle se charge d'initialiser l'état.

Etats transition(Etats e, Alphabets c) : retourne erreur si l'état actuel est un état initial et que le commande est 'A' (parce qu'il n'y a pas de mesure avant). Retourne initial si l'état est G ou D et mesure si l'état est M. Ensuite, si c'est M, on vérifie que l'état est A pour repartir à Initial sinon reste sur mesure toujours si commande=G ou B ou M. Enfin, après avoir testé tous ces cas; la fonction renvoie erreur car le programme devient erroné systématiquement.

int BoolFinal(Etats e): renvoie true si l'état actuel n'est pas une erreur.

Globalement, observateur vérifie qu'à l'exécution d'un programme-robot, la fonction «avancer» de l'environnement soit toujours précédée d'une «mesure» par le robot de la case devant lui.

Conclusion:

En conclusion, le mini-projet INF304 a adopté une approche systématique avec plusieurs tests variés, révélant des succès limités dans la conception des robots-programmes. Les programmes N°3 et N°7 ont affiché des résultats acceptables, mais des défis subsistent. Le TP09 a introduit un observateur vérifiant la précédence de la mesure avant l'avancement, renforçant la robustesse du projet.