L3 informatique, 2024-2025, Projet de LFC

Création d'un langage de script pour un système multi-agents

Description générale du projet

Le but du projet est d'analyser un programme écrit dans une ébauche de langage de script décrivant un système multi-agents, puis de le traduire dans un autre langage. Le langage dans lequel le programme doit être traduit vous sera donné ultérieurement.

1 Système multi-agents

En général

Un système multi-agents est constitué d'un ensemble d'agents (processus, robots, avatars, etc.) qui évoluent dans un environnement. Les agents sont les éléments actifs du système et interagissent entre eux et avec l'environnement. Dans l'environnement se trouvent également un ensemble d'objets que les agents peuvent percevoir, créer, détruire et modifier.
Les systèmes multi-agents peuvent être utilisés dans des domaines tels que l'intelligence artificielle ou l'intelligence distribuée pour la résolution de problèmes complexes, ainsi que comme outils de simulation en animation comportementale.

Dans le cadre du projet

Le système multi-agent que l'on souhaite définir avec le langage de script que vous avez à analyser est constitué d'agents, pouvant être de types différents (par exemple des proies et des prédateurs), et de contextes (par exemple, un point d'eau, une zone ombragée, la pluie).

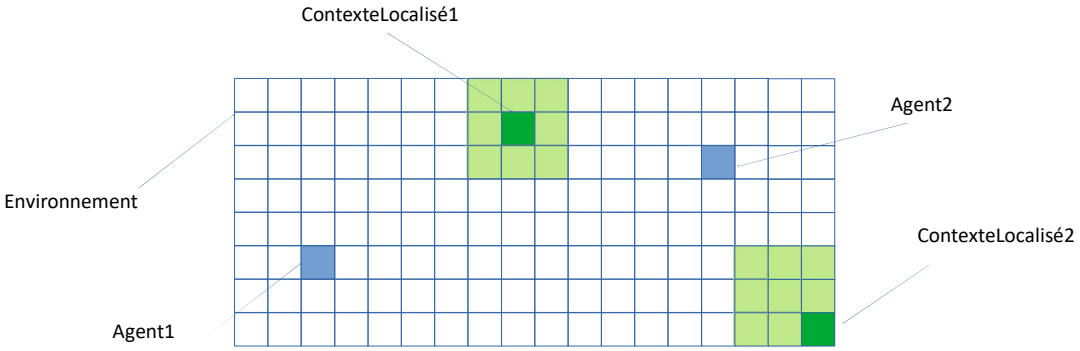
L'environnement est une grille régulière représentée par un tableau à deux dimensions. Pour définir l'environnement, on doit donner le nom et la taille du tableau.

Chaque agent est caractérisé par un certain nombre d'attributs, dépendant du type d'agent auquel il appartient.
Pour définir un type d'agent, on donne le nom de ce type, ainsi que la liste des attributs qui le caractérisent en indiquant, pour chaque attribut, le type de valeur qu'on peut lui affecter.
Pour définir un agent, on donne le nom de celui-ci, le type d'agent auquel il appartient, ainsi qu'une valeur pour chaque attribut correspondant à ce type d'agent. On indique également la position de l'agent dans l'environnement.

Dans l'environnement, les agents sont soumis à des contextes. Ces derniers représentent des objets situés dans l'environnement ou des événements agissant sur tout l'environnement.
On a donc deux sortes de contextes :

- Les contextes dits localisés (un arbre, un point d'eau), qui ont une position précise dans l'environnement et auxquels sont associés une zone d'influence,
- Les contextes dits non localisés (la pluie), qui agissent sur tout l'environnement.

Chaque contexte est défini par son nom et sa force. La force d'un contexte sert à pondérer l'influence du contexte sur le comportement des agents qui se trouvent dans sa zone d'influence. Plus un contexte a une force élevée, plus il a d'influence sur le comportement des agents qui sont dans sa zone d'influence.
Pour les contextes localisés, on doit également indiquer leur position dans l'environnement et le rayon de leur zone d'influence (ici un disque centré sur la position du contexte).



2 Modalités

Vous devrez rendre le projet par étapes (travaux à rendre toutes les 1, 2 ou 3 semaines selon les étapes). Un sujet spécifique à chaque étape sera mis en ligne sur plubel avec indication de la date limite pour rendre celle-ci.
Lorsque vous rendez une étape, mentionnez le numéro de groupe de projet dans le document principal.
Pour les étapes où il faut rendre un programme, vous devez fournir les explications nécessaires à la création de l'exécutable et ne pas oublier d'ajouter tous les modules indispensables à la compilation et l'exécution de votre programme (.h, .c, etc.). Afin que nous puissions tester vos programmes, merci de faire en sorte que ceux-ci puissent être compilés et exécutés sur les machines des salles d'enseignement. Inutile de fournir un exécutable ; il ne fonctionnera pas.
Après chaque étape des analyses lexicale et syntaxique, il vous sera communiqué un corrigé de celle-ci afin que vous puissiez réaliser l'étape suivante sur une base correcte.

3 Fichiers à analyser

Les fichiers à analyser contiennent des *Instructions* permettant de décrire un système multi-agents simple.
Le programme à analyser contiendra uniquement des instructions permettant de créer l'environnement, les types d'agents, les agents et les contextes.
Ces instructions sont décrites ci-dessous. Les mots en gras sont des mots-clés.

Création de l'environnement

Environnement *NomEnvironnement***[***NbLigne***,***NbColonnes***]**

Où

- *NomEnvironnement* est un identificateur (le nom que l'on choisit pour l'environnement)
- *NbLignes* et *NbColonnes* sont des nombres entiers qui donnent la taille du tableau représentant l'environnement.

Création d'un type d'agent

NewTypeAgent *NomType*
{
Déclaration_des_attributs
}

Où

- *NomType* est un identificateur (le nom que l'on choisit pour ce type d'agent)
- *Déclaration_des_attributs* est une suite de déclarations d'attributs de la forme *NomAttribut* **:** *TypeAttribut*

Un type d'agent possède au moins un attribut. S'il en a plusieurs, les déclarations d'attributs sont séparées par des virgules. *NomAttribut* est un identificateur et *TypeAttribut* est un type parmi {**int**, **double**, **char**, **string**, **boolean**}

Création d'un agent

NewAgent *NomAgent* **:** *NomTypeAgent* **[***NoLigne***,** *NoColonne***]**
{
Affectation_des_attributs
}

Où

- *NomAgent* et *NomTypeAgent* sont des identificateurs (*NomTypeAgent* est le nom du type d'agent auquel l'agent appartient),
- *NoLigne* et *NoColonne* sont des entiers donnant la position de l'agent dans l'environnement,
- *Affectation_des_attributs* est une suite d'affectations de la forme
  NomAttribut **=** *ValeurAttribut*

Les affectations de valeurs aux attributs sont séparées par des virgules. *NomAttribut* est un identificateur et *ValeurAttribut* est une constante entière, réelle, caractère, chaîne de caractères ou booléenne (**TRUE** ou **FALSE**). Les noms d'attributs sont ceux que l'on trouve dans la déclaration du type d'agent auquel l'agent appartient.

Création d'un contexte

On définit un contexte non localisé avec :

NewContexte *NomContexte***[***Force***]**

Où

- *NomContexte* est un identificateur (le nom que l'on donne au contexte)
- *Force* est un entier entre 0 et 100 inclus

Et un contexte localisé avec :

NewContexte *NomContexte***[***Force***,** *NoLigne***,** *NoColonne***,** *RayonInfluence***]**

Où

- *NoLigne*, *NoColonne* et *RayonInfluence* sont des nombres entiers

Informations complémentaires utiles pour les premières étapes du projet :

- Les identificateurs sont des mots constitués de lettres non accentuées majuscules ou minuscules, de chiffres et de « _ ». Ils commencent obligatoirement par une lettre.
- Les réels s'écrivent avec une virgule séparatrice.
- Les constantes chaînes de caractères commencent et terminent par un guillemet. Entre ces 2 guillemets, on peut trouver n'importe quel suite de caractères excepté des guillemets.
- Les constantes caractères sont constituées d'un caractère quelconque entre cotes (ex : 'a').
- Les espaces n'ont aucune importance. On peut en ajouter autant que l'on veut dans le programme pour séparer les mots sans que cela ne change quoi que ce soit.
- Les retours à la ligne peuvent également être utilisés pour améliorer la lisibilité des programmes mais ils n'ont pas de signification particulière.
- Dans un programme, on ne peut créer qu'un seul environnement mais on peut créer plusieurs types d'agent, plusieurs agents et plusieurs contextes.

4 Etapes du projet

Analyse lexicale

La première étape consistera à établir la liste des unités lexicales du langage source et donner la description de celles-ci.
Cette liste sera utilisée lors de la seconde étape pour l'écriture du programme lex qui permettra de générer l'analyseur lexical de votre compilateur.

Analyse syntaxique

Les étapes suivantes consisteront à décrire de façon formelle la syntaxe d'un fichier source et écrire le programme yacc à partir duquel sera généré l'analyseur syntaxique du compilateur.

Analyse sémantique/traduction

Les dernières étapes consisteront à insérer dans les programmes lex et yacc les instructions permettant de

- vérifier que les instructions du fichier source ont un sens,
- traduire le programme dans un autre langage, qui vous sera indiqué ultérieurement.

Table des symboles

La construction de la table des symboles commencera après l'écriture du programme lex. Dans cette structure seront notamment stockées toutes les informations concernant les identificateurs mentionnés dans le fichier source.