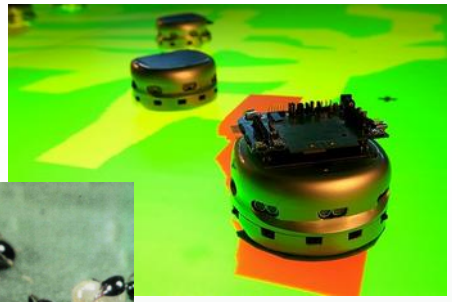
CentraleSupélec

Mineure, 3ème année

Vie artificielle (TP 3)

Recherche de ressources (modèle fourmis)



Alexis Scheuer & *Olivier Simonin*

Maître de conférences UL (FST) / Loria

UNIVERSITÉ DE LORRAINE

Loria Laboratoire lorrain de recherche en informatique et ses applications

Objectifs

- Définir le comportement de fourragement
  - architecture de subsomption
- Utiliser un simulateur (java)
  - compléter le comportement des agent
- Faire varier les paramètres et analyser

Le simulateur : classes graphiques

- `ForAnt`
  - `main()` : création de la fenêtre (`MainFrame`) et lancement (`begin`)
  - `begin()` : boucle de simulation et tempo (`sleep(n)`)
- `MainFrame`
  - dimension fenêtre
  - création du terrain et lancement des fourmis : `action()`

Le simulateur : classe du terrain

`Terrain`

- Définition (constructeur)
  - Env. (taille, zoom), obstacles (aléa.), nourriture (aléa. et qté)
  - Phéromone (`Gradient`)
  - Colonie de fourmis (`Colony base`)
- `action()`
  - diffusion/évaporation phéromones : `spreadFoodPheromone()`
  - `base.bouge()`
- Coefficients
  - `Classe : DENSITY_OBSTACLE (0.05)`
  - `spreadFoodPheromone() : diffusion = 0.05, evaporation = 0.005`

Le simulateur : classes des fourmis

- `Colony`
  - `bouge()`
    - active toutes les fourmis :
      `ArrayList<Worker> ants` → `ants.get(i).bouge()`
    - gère la fin de la simulation
      (ressources consommées ou Time Out)
- `Worker` (une fourmi)
  - `enum Etat { RETOUR, RECHERCHE, MORTE }`
    → var. `etat`
  - `enum Direction { NORD, EST, SUD, OUEST }`
  - bouge() : **à définir !**

Concevoir le comportement

2 comportements distincts (subsomption)

- pour la **recherche** de ressource
- pour le **retour** à la base

Intégrer le changement de comportement

Perceptions et actions possibles ?

Perceptions et actions disponibles

Perceptions

(bool.) arrivée nid ?

(bool.) arrivée sur nourriture ?

(dir.) direction nid

(dir.) nourriture ds. voisinage direct ?

(dir.) phéromone ds. voisinage qui éloigne du nid ?

Actions

déposer nourriture au nid

déposer phéromone

(dir) direction aléatoire

prendre une direction (dir)

récupérer nourriture

Programmer les comportements

Compléter la fonction `bouge()` dans la classe `Worker`

```
if (etat == Etat.RECHERCHE) {
   ...
}
else if (etat == Etat.RETOUR) {
   ...
}
```

Méthodes à utiliser

arrivée nid ? :	`boolean terrain.isColonyAtPoint(currentPos)`
arrivée sur nourriture ? :	`boolean terrain.isFoodAtPoint(currentPos)`
direction nid :	`Direction direction(colony.getPosition())`
nourriture ds. voisinage direct ? :	`Direction searchForFood()`
Phéromone ← nid ? :	`Direction searchForfoodPheromone(null)`

déposer nourriture :	`dropFood()`
déposer phéromone :	`dropFoodPheromone()`
direction aléatoire :	`Direction randomDirection()`
prendre une direction d :	`newDirection = d`
récupérer nourriture :	`terrain.updateFood(currentPos)`

Tester et expérimenter

- Compiler et vérifier

  `javac *.java` et `java ForAnt`

- Définir le comportement des fourmis

- Faire varier les paramètres :
  - le taux d'évaporation et/ou de diffusion
  - la densité des obstacles
  - le nombre d'agents

Améliorer

- Mettre en œuvre l'algorithme distribué de la vague :
  - Remplace la connaissance systématique de la direction du nid par une descente de gradient
  - Ajoute une tâche aux fourmis = mise à jour de la distance à l'endroit courant, en fonction de son voisinage