Agents Intelligents

Chapter 2/

From Russel&Norvig, Intelligence Artificielle, 3ème édition

http://aima.cs.berkeley.edu/

Plan

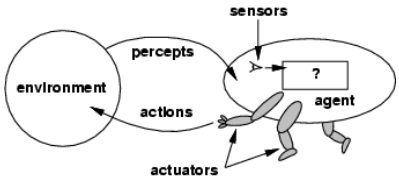
- Agents et environnements
- Le concept de Rationalité
- Spécification de l'environnement d'une tâche selon PEAS (Performance measure, Environment, Actuators, Sensors)
- Propriétés des environnements de tâche
- Types d'Agents: Structure des agents

Agents

- Un agent est toute entité qui peut être considérée comme percevant son environnement grâce à des capteurs(sensors) et qui agit sur cet environnement via des effecteurs (actuators)

- *Human agent:*
  - *eyes, ears, and other organs for sensors;*
  - *hands,legs, mouth, and other body parts for actuators*
- *Robotic agent:*
  - *cameras and infrared range finders for sensors;*
  - *various motors for actuators*

Agents et environments

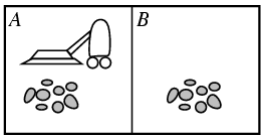


- La fonction agent fait correspondre une action à chaque *séquence de percepts* (historique de tout ce qu'il a perçu) :

$$[f: P^* \rightarrow A]$$

- Le programme agent s'exécute sur une architecture physique pour produire *f*
- agent = architecture + program
- Fonction agent = description mathématique abstraite
- Programme : imlémentation concrète

Agent aspirateur limité à deux emplacements



- Percepts: localisation et état , e.g., [A, Sale]
- Actions: *Gauche*, *Droite*, *Aspirer*, *NoOp*

Bons Comportements: le concept de Rationalité

- Un agent doit toujours effectuer “l’action appropriée", en fonction de ce qu’il perçoit et des actions qu’il peut effectuer. L’action appropriée est celle qui permet à l’agent de réussir le mieux.
- La mesure de Performance : Critère objectif pour le succès du comportement de l’agent

- *E.g., performance measure of a vacuum-cleaner agent could be*
  - *amount of dirt cleaned up,*
  - *amount of time taken,*
  - *amount of electricity consumed,*
  - *amount of noise generated, etc.*

Agent Rationnel

- Agent Rationnel : Pour chaque séquence de percepts possible, un agent rationnel doit sélectionner une action susceptible de *maximiser sa mesure de performance*, compte tenu des observations fournies par la séquence de percepts et de la connaissance dont il dispose.

Agent Rationnel

- Il faut distinguer la Rationalité de l'omniscience (all-knowing with infinite knowledge)
- La rationalité n'est pas synonyme de perfection
- Le choix rationnel ne dépend que de la séquence de percepts *à un moment déterminé*
- Les agents peuvent agir afin de *modifier les futurs* percepts et afin d'obtenir les informations utiles (*collecte d'informations, exploration*)
- Un agent est autonome si son comportement est déterminé par sa propre expérience (capable d'apprendre et de s'adapter)
- Un agent rationnel doit être autonome; il doit apprendre ce qu'il peut pour compenser ses connaissances a priori partielles ou incorrectes.

Spécification de l'environnement d'une tâche selon PEAS

- Environnement des tâches (task environment) = les « problèmes » dont les agents rationnels sont les « solutions »
- Lors de la conception d'un agent, la première étape doit toujours consister à spécifier cet environnement aussi complètement que possible

PEAS

- PEAS:
  - Performance measure = mesure de performance
  - Environment = Environnement
  - Actuators = Effecteurs
  - Sensors = Capteurs
- Exemple : Donner une description PEAS de l'environnement de tâche de conception d'un agent chauffeur de taxi
  → cf. page 44 pour un exemple de réponse

Exemple : description PEAS de l'environnement de la tâche de conception d'un chauffeur de taxi

Type d'agent = Chauffeur de taxi

Mesure de Performance	Environnement	Effecteurs	Capteurs
Minimiser la durée d'une course Maximiser ses profits Maximiser la satisfaction du client Meilleure réputation Sécurité	Routes passagers Voiture Clients Ville + lieux publics + Hotels Taxis concurrents	« système d'accélération » « système de freinage » klaxon Système de paiement	Caméra Capteurs sonores Capteurs de « distances » Capteur de vitesse Radar GPS Micro capteur

Exemple : la tâche de conception d'un chauffeur de taxi

Type d'agent = Chauffeur de taxi

Mesure de Performance	Environnement	Effecteurs	Capteurs
Minimiser le trajet Minimiser la durée du « trajet » Assurer la sécurité des clients Maximiser les gains Maximiser le Confort du client Augmenter la satisfaction des clients	Routes Autres véhicules Feux Passagers Piétons Police Les bagages Météo	Accélérateur Système de freins Climatisateur Poste radio	Radar Capteurs sonores Caméras thermomètre

Réponse Groupe 2

Type d'agent	Mesure de Performance	Environnement	Effecteurs	Capteurs
Chauffeur de Taxi	+ court chemin + rapide + Sécurité - la comsommation + revenu journalier Maximiser la satisfaction des clients			

PEAS

- e.g., the task of designing an automated taxi driver:
  - Performance measure: Safe, fast, legal, comfortable trip, maximize profits
  - Environment: Roads, other traffic, pedestrians, customers
  - Actuators: Steering wheel, accelerator, brake, signal, horn
  - Sensors: Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard

Description PEAS (2ème exemple) Agent: Medical diagnosis system

- Performance measure: Healthy patient, minimize costs, lawsuits
- Environment: Patient, hospital, staff
- Actuators: Screen display (questions, tests, diagnoses, treatments, referrals)
- Sensors: Keyboard (entry of symptoms, findings, patient's answers)

Description PEAS (3ème exemple)

- Agent: Part-picking robot
- Performance measure: Percentage of parts in correct bins
- Environment: Conveyor belt with parts, bins
- Actuators: Jointed arm and hand
- Sensors: Camera, joint angle sensors

Description PEAS (4ème exemple)

- Agent: Interactive English tutor
- Performance measure: Maximize student's score on test
- Environment: Set of students
- Actuators: Screen display (exercises, suggestions, corrections)
- Sensors: Keyboard

Environment types

- Fully observable (vs. partially observable): An agent's sensors give it access to the complete state of the environment at each point in time.
- Deterministic (vs. stochastic): The next state of the environment is completely determined by the current state and the action executed by the agent. (If the environment is deterministic except for the actions of other agents, then the environment is strategic)
- Episodic (vs. sequential): The agent's experience is divided into atomic "episodes" (each episode consists of the agent perceiving and then performing a single action), and the choice of action in each episode depends only on the episode itself.
-

Environment types

- Static (vs. dynamic): The environment is unchanged while an agent is deliberating. (The environment is semidynamic if the environment itself does not change with the passage of time but the agent's performance score does)
- Discrete (vs. continuous): A limited number of distinct, clearly defined percepts and actions.
- Single agent (vs. multiagent): An agent operating by itself in an environment.

Environment types

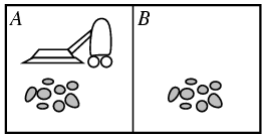
	Chess with a clock	Chess without a clock	Taxi driving
Fully observable	Yes	Yes	No
Deterministic	Strategic	Strategic	No
Episodic	No	No	No
Static	Semi	Yes	No
Discrete	Yes	Yes	No
Single agent	No	No	No

- The environment type largely determines the agent design
- The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

Fonctions et programmes Agent

- Un agent est complètement spécifié par sa fonction agent qui associe des percepts à des actions
- But : Trouver un moyen d'implémenter une fonction d'un agent rationnel de manière concise

Agent aspirateur limité à deux emplacements



- Percepts: localisation et état , e.g., [A, Sale]
- Actions: *Gauche*, *Droite*, *Aspirer*, *NoOp*

Construction d'un agent à partir d'une table

- Exemple : voir transparent suivant
- Limites:
  - Table de taille énorme
  - Temps de création de tables élevé
  - Pas d'autonomie
  - Difficultés pour l'apprentissage

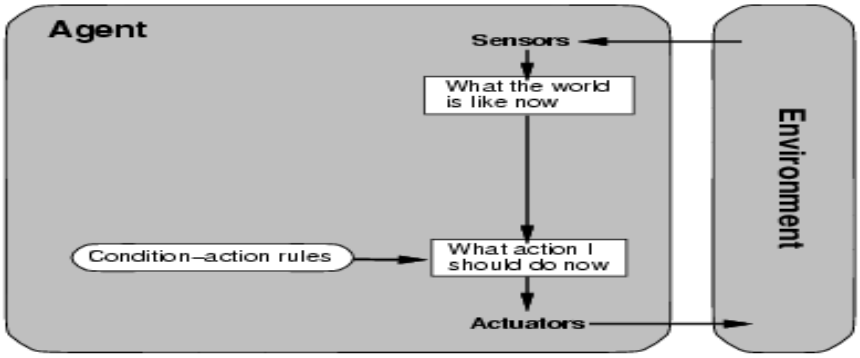
Séquence de perceptions	Action
[A,Propre]	Droite
[A,Sale]	Aspirer
[B,Propre]	Gauche
[B,Sale]	Aspirer
[A,Propre], [A,Propre]	Droite
[A,Propre], [A,Sale]	Aspirer
.	.
.	.
.	.
[A,Propre], [A,Propre],[A,Propre],	Droite
[A,Propre], [A,Propre], [A,Sale]	Aspirer
.	.
.	.
.	.

Fig2.3 (page 39): Tabulation Partielle d'une fonction agent simple pour le monde de l'agent aspirateur

Types d'agents

- Quatre types fondamentaux de programmes d'agents:
  - Agents réflexes simples
  - Agents réflexes fondés sur des modèles
  - Agents fondés sur des buts
  - Agents fondés sur l'utilité

Agents réflexes simples



function SIMPLE-REFLEX-AGENT(percept) returns an action

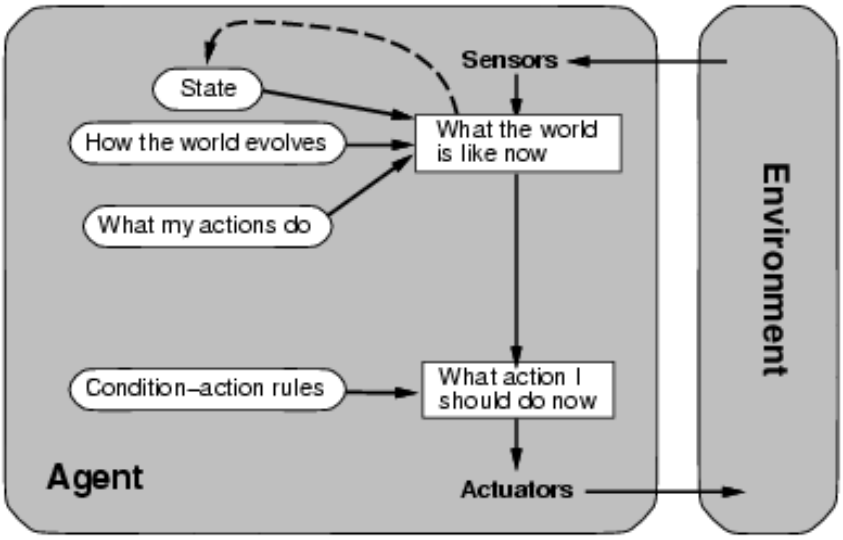
persistent: rules, a set of condition–action rules
state ←INTERPRET-INPUT(percept)
rule←RULE-MATCH(state, rules)
action←rule.ACTION
return action

Exemple

Programme agent pour un agent réflexe simple dans l'environnement de l'aspirateur à deux états

Function Agent-Aspirateur-Reflexe([location,status]) **returns** an action
if status = Dirty **then return** Suck
else if location = A **then return** Right
else if location = B **then return** Left

Model-based reflex agents



Model-based reflex agents

function MODEL-BASED-REFLEX-AGENT(percept) returns an action

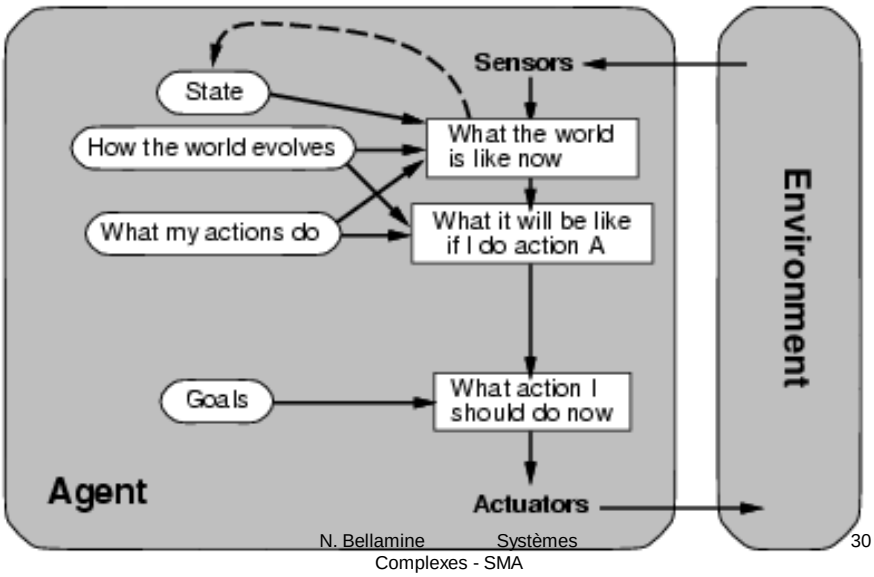
persistent:
state, the agent's current conception of the world state
model , a description of how the next state depends on current state and action
rules, a set of condition–action rules
action, the most recent action, initially none

state ←UPDATE-STATE(state, action, percept ,model)
rule←RULE-MATCH(state, rules)
action←rule.ACTION

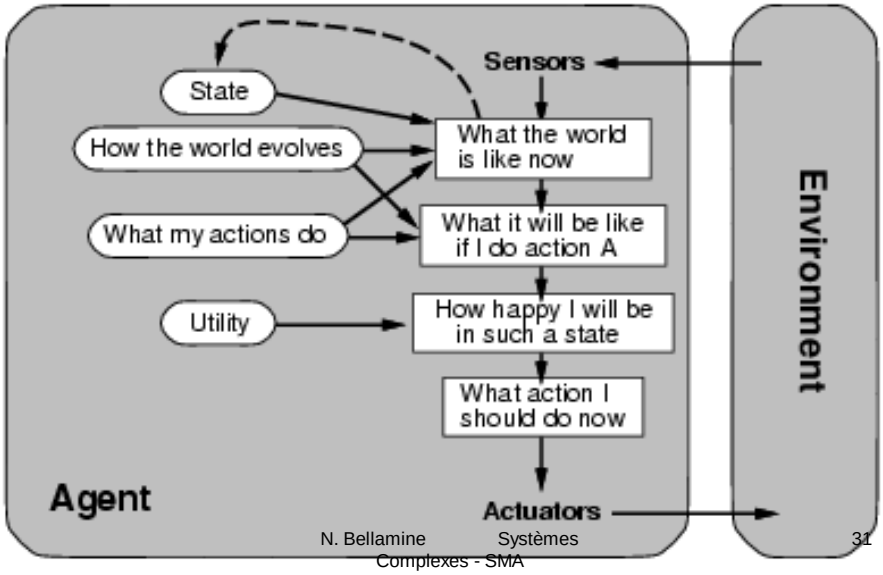
return action

A model-based reflex agent. It keeps track of the current state of the world, using an internal model. It then chooses an action in the same way as the reflex agent.

Goal-based agents



Utility-based agents



Model-based reflex agents

function MODEL-BASED-REFLEX-AGENT(percept) returns an action

persistent:
state, the agent's current conception of the world state
model , a description of how the next state depends on current state and action
rules, a set of condition–action rules
action, the most recent action, initially none

state ←UPDATE-STATE(state, action, percept ,model)
rule←RULE-MATCH(state, rules)
action←rule.ACTION

return action

Learning agents