Creating Artificial Societies

Francisco Grimaldo Moreno
Department of Computer Science
University of Valencia (Spain)
francisco.grimaldo@uv.es

December 10th, 2007

Outline

Introduction to Artificial Societies (AS)

Behavioural-Based Systems

- Examples of social emergence
- Creating AS in Jason

Knowledge-Based Systems

- Semantic information
- Social simulation: Coordination and Sociability

Future remarks

Exercise: Cooperative behaviour in Jason

Introduction to AS

Some definitions (1)

- An artificial society (AS) is a synthetic representation of a society.
- It simulates social phenomena: coordination, cooperation, competition, markets, social networks dynamics, etc.
- Types of social simulations: System Level vs Agent Based
- Agent Based Social Simulations (ABSS) are an amalgam of computer simulations, agent based modelling, and the social sciences. Provides the bridge between micro and macro levels
- “Artificial Society is the specific agent based computational model for computer simulation in social analysis”. [Wikipedia]

Introduction to AS

Some definitions (2)

- A society is a group of individuals exhibiting intelligence and interacting socially among them.
- An interaction is social when it involves an agent with a goal-oriented behaviour dealing with another agent considered as its similar.
- Types of social actions [Castelfranchi]:
  - A weak social action considers what another agent is doing, and might affect the considering agent.
  - A strong social action involves social goals. A social goal is directed towards another agent. This is, the social goal of one agent is to influence the mind or actions of another agent.

Introduction to AS

History

- Cellular automata [Von Neumann]: Mathematical model for self-replicating machines with very complicated rules on a rectangular grid.
- Game of Life [John Conway]: Simpler CA to observe the way that complex patterns can emerge from the implementation of very simple rules.
- Boids [Craig Reynolds]: Model the reality of lively biological agents (artificial life).
- Sugarscape [Epstein & Axtell]: explore social phenomena such as seasonal migration, pollution, reproduction, combat, transmission of disease, culture...
- Cognitive social simulation [Ron Sun]: Introducing models of human cognition in agent-based simulations.

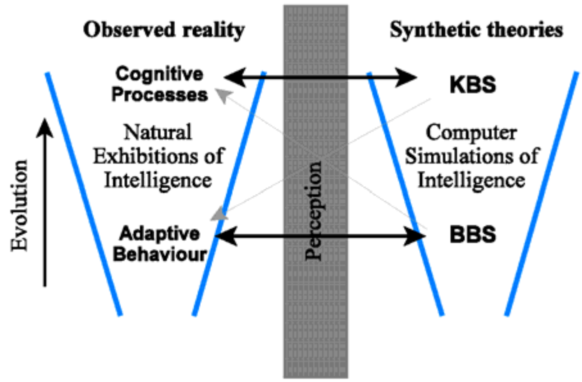
Introduction to AS

Application examples

- Computational sociology: AS are used to understand how societies work by synthetically creating them ([Gilbert], [Macy & Willer]).
- Economics: Market and consumer behaviour ([Jennings], [Shoham]).
- Virtual reality: Motion pictures, games, video simulations... ([Reynolds], [Terzopoulos], [Thalmann]).
- Robotics: Multi-robot environments (Robocup).

Introduction to AS

Behaviour-based vs Knowledge-based Systems



- KBS: Effective in solving “very difficult” problems (chess)
- BBS: Good for “simple” things (walk in crowded corridors)

Introduction to AS

Questions to answer

- How the agent perceives his environment and himself (how he obtains his beliefs and his goals)?
- How the agent selects which action to perform depending of his actual goals and beliefs? This is also known as the action selection problem.
- How the agents might cooperate to achieve individual or social goals?
- How the agents should compete to decide which goal should be achieved next?
- How the agents should avoid to interfere the achievement of other agents goals?

Outline

Introduction to Artificial Societies (AS)

Behavioural-Based Systems

Examples of social emergence

Creating AS in Jason

Knowledge-Based Systems

Semantic information

Social simulation: Coordination and Sociability

Future remarks

Exercise: Cooperative behaviour in Jason

Behavioural-Based Systems

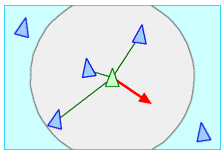
Introduction

- Inspired in the field of ethology, which is the part of biology studying animal behaviour.
- This is because many properties desirable in autonomous intelligent systems are present in animal behaviour: autonomy (self-control), adaptation to changes in the environment, situatedness, goal-directedness...
- Emergence refers to the way complex systems and patterns arise out of a multiplicity of relatively simple interactions

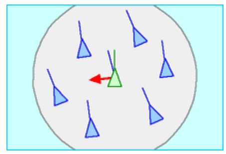
BBS: Social emergence

Boids [Reynolds, since 1986]

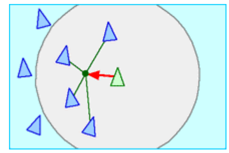
- Computer model of coordinated animal motion such as bird flocks and fish schools.
- Three basic steering behaviours:



Separation: steer to avoid crowding local flockmates



Alignment: steer towards the average heading of local flockmates

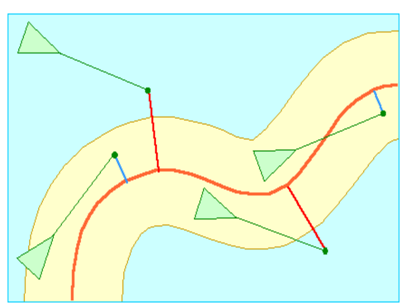


Cohesion: steer to move towards the average position of local flockmates

BBS: Social emergence

Boids [Reynolds, since 1986]

- Simple behaviours for individuals and pairs: s
  - Seek and flee
  - Pursue and evade
  - Wander
  - Obstacle avoidance
  - Wall/Path/Flow Following
- Combined/group behaviors:
  - Crowd Path Following
  - Leader Following
  - Unaligned Collision Avoidance
  - Queuing (at a doorway)



BBS: Social emergence

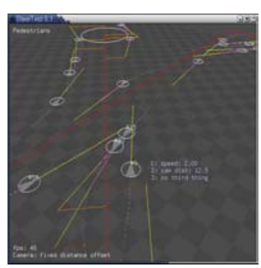
PSCrowd [Reynolds, 2006]

- PSCrowd is a library to support large crowd/flock (up to 15.000 individuals) simulations on PLAYSTATION®3 or other Cell-based systems.
- The crowds are modeled as interacting (Lagrangian) particle systems.

BBS: Social emergence

OpenSteer [Reynolds, 2004]

- OpenSteer is a C++ library to help construct steering behaviors for autonomous characters in games and animation
- PMFserv has used to build psychological models (emotions, stress...).



BBS: Social emergence

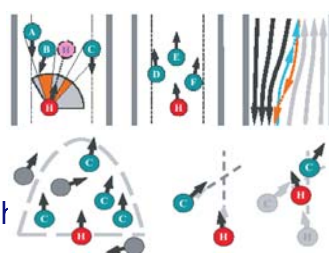
OpenSteer [Reynolds, 2004]

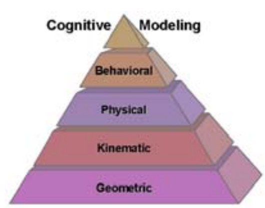
- OpenSteer basic commands:
  - c: change camera position.
  - s: select next agent.
  - r: restart current PlugIn.
  - Tab: select next PlugIn.
- Sample PlugIns:
  - Capture the flag.
  - Pedestrians.
  - Map drive.
  - Soccer.

BBS: Social emergence

Autonomous pedestrians [Terzopoulos, 2005]

- Emulate individual pedestrians.
- Reactive behaviour routines.
- Cognitive Modelling Language
- Behavioural controller can interrupt th execution of cognitive plans.
- No social model.
- Demo: Train Station.





BBS: Social emergence

Social forces [Helbing et al, 2000]

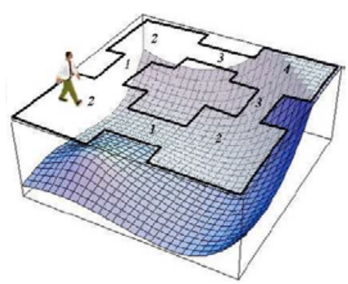
- Social force model: mixture of socio-psychological and physical forces.
- Basically agents adapt their velocity according to attractive and repulsive forces.

$$m_i \frac{d\mathbf{v}_i}{dt} = m_i \frac{v_i^0(t)\mathbf{e}_i^0(t) - \mathbf{v}_i(t)}{\tau_i} + \sum_{j(\neq i)} \mathbf{f}_{ij} + \sum_W \mathbf{f}_{iW} \qquad (1)$$

- Panic simulations in human crowds.
- E.g.: Individualism vs Herding

BBS: Social emergence

Continuum crowds [Treuille et al, 2005]

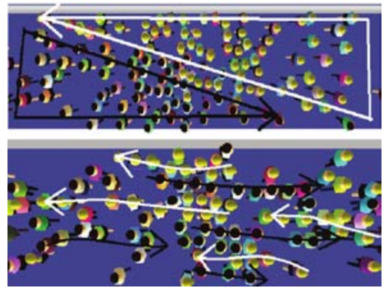


- Dynamic potential fields.
- Put a grid over the world and compute the value of each cell according to an aspect (goal, density, obstacles).
- Move agents following potential fields.
- Social outcomes:
  - People forming lanes to avoid collision.
  - Soft paths to avoid congestions compared to local models.
- Trade-off: Very time consuming.

BBS: Social emergence

Brand new approaches

[Thalmann et al, 2007]: Potential fields and Boids at different levels of detail



HiDAC [Pelechano, 2007]: Potential fields and Boids at different levels of detail

BBS: Creating AS in Jason

Jason overview

- Platform for developing multi-agent systems.
- Developed by Jomi F. Hübner and Rafael H. Bordini
- Jason implements the operational semantics of an extended variant of AgentSpeak.
- AgentSpeak: programming language for BDI agents
- Main language constructs: Believes, Goals & Plans.
- Quick review of the Agent Reasoning Cycle from Jason slides: “Programming Multi-Agent Systems in AgentSpeak using Jason”.

BBS: Creating AS in Jason

Game of life example

- Multi-agent implementation of the popular CA.
- On every step, each cell/agent decides whether to live or die according to its neighbours.
- The environment then updates agent's percepts.
- Aspects to pay attention to:
  - Life expectancy (compare low and high values for density).
  - Equilibrium situation.
  - Scalability --> pool of threads.
- Analyse step by step execution.

Outline

Introduction to Artificial Societies

Behavioural-based systems

Examples of social emergence

Creating AS in Jason

Knowledge-based systems

Semantic information

Social simulation: Coordination and Sociability

Future remarks

Exercise: Cooperative behaviour in Jason

Knowledge Based Systems
Introduction

- Previously introduced as “top-down” systems since they are mostly designed from the whole to the parts.
- But control can also be decentralized.
- Multi-agent systems: elegant and formal framework to develop artificial societies of cognitive agents.
- Include explicit knowledge representation:
  - World representation --> Agent – object interaction.
  - Relations representation --> Agent – agent interaction.
- Abstract & complex reasoning.

Knowledge Based Systems
Cognitive vs. reactive agents

Cognitive agents:

- Explicit representation.
- Direct communication.
- Deliberative control.
- Social organization.
- Low number of agents.

Reactive agents:

- Implicit representation.
- Indirect communication.
- Non-deliberative control.
- Ethological organization.
- High number of agents.

- Intelligent agents [Wooldridge & Jennings]: reactive, proactive and social skills.

KBS: Semantic information

World representation (1)

- Object Specific Reasoner [Levinson, 1996]: classifies objects into taxonomies (e.g. containers can be opened by hands)
- Task Definition Language [Vosinakis, 2003]: Parametrized action representation.
- Knowledge in the world [Doyle, 2002]: annotated environments to allow agent mobility.
- SeVEn platform [Otto, 2005]: Uses RDF to categorize objects using a type field.
- Not enough for dynamic environments.

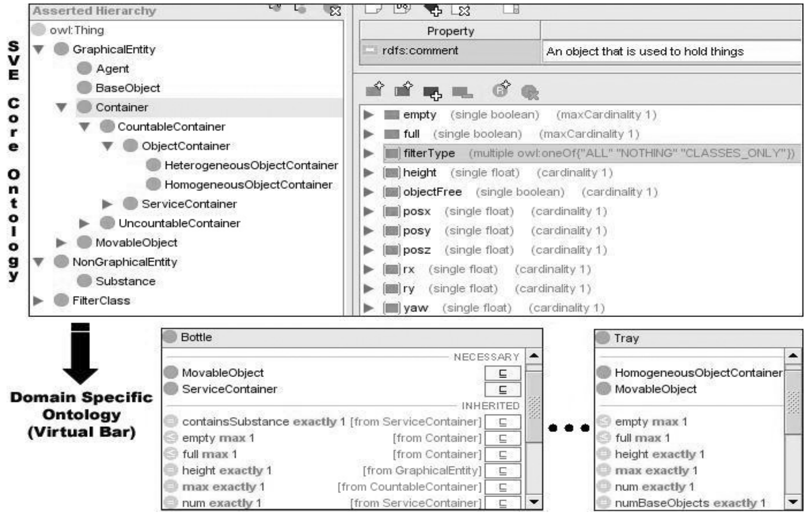
KBS: Semantic information

World representation (2)

- Ontology-based concept layer [Chang, 2005]: Annotation on the fly using. Flexible behaviour through ontological inference.
- Environment Description Language for Multi-Agent Simulation (ELMS) [Bordini, 2005]: Resource representation, agent perception and action, environment reactions.
- Ontology-based SVE to help sensorization and actuation of virtual agents in complex scenes [Grimaldo, 2006].

KBS: Semantic information

Ontology-based SVE [Grimaldo, 2006]



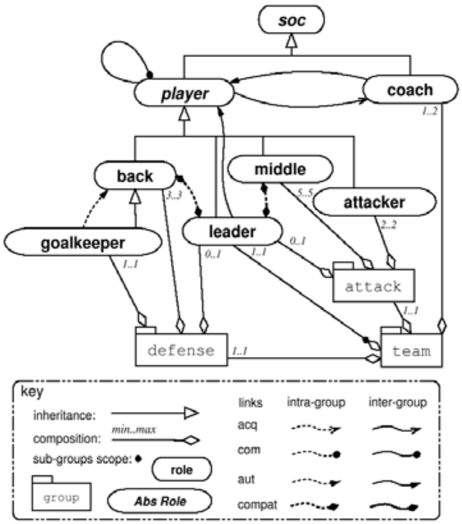
KBS: Semantic information

Relations representation

- From the interaction among agents emerge the notion of an artificial society.
- Ontologies to model social relations and support social reasoning (e.g. social constraints and conflict resolution). [Kao, 2005]
- Coordination artifacts [Viroli et al, 2006]: environment-based alternative for structuring interactions and provide collaboration.
- Organizational reasoning [Hübner & Sichman]: constrain agents' behaviour towards a global goal.

KBS: Semantic information

Organizations in MOISE+ (1)



- MOISE+: model to build organizations in MAS.
- Points of view:
  - Structural: roles, groups, communication and authority links...
  - Functional: goals, plans, missions, norms...
  - Deontic: Relates roles to missions.

KBS: Semantic information

Organizations in MOISE+ (2)

Functional specification

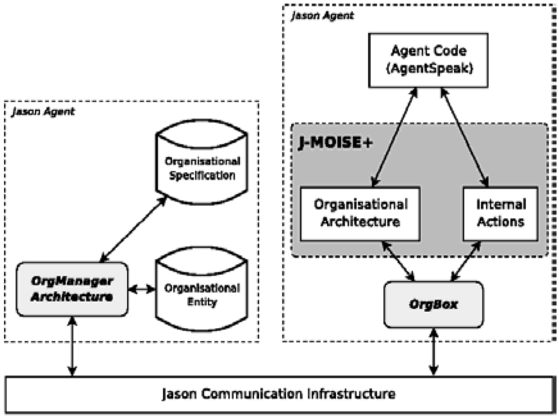
role	deontic relation	mission
back	*permission*	m_1
middle	*obligation*	m_2
attacker	*obligation*	m_3

Deontic specification

KBS: Semantic information

Organizations in MOISE+ (3)

- J-MOISE+: Integration in Jason.



- create/remove group
- add responsible group
- adopt/remove role
- set goal arguments
- set goal state
- commit mission
- remove mission

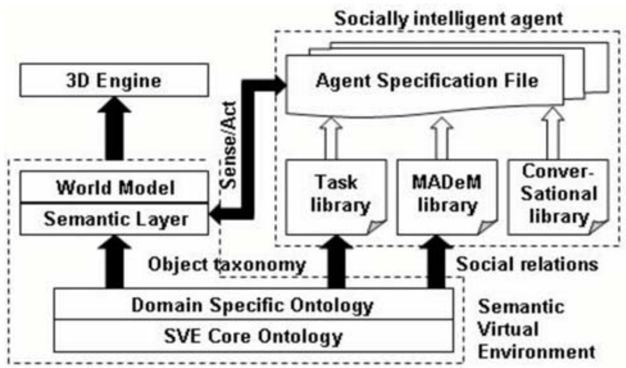
KBS: Social simulation

Social networks

- Interaction models have evolved to different kinds of social networks:
  - Dependence networks [Sichman]: Allow agents to cooperate or to perform social exchanges attending to their dependence relations (e.g. social dependence and social power).
  - Trust networks [Castelfranchi]: Define delegation strategies using contract net protocol and fuzzy cognitive representations of the other agents and the dynamic environment.
  - Preference networks [Grimaldo]: Agents express their preferences using utility functions which can be weighted to represent their attitude towards other agents.

KBS: Social simulation

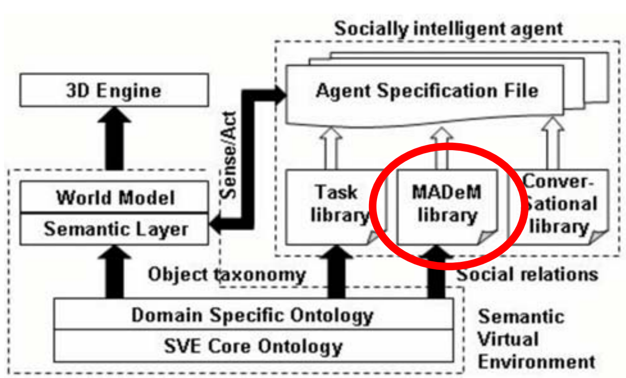
System architecture



- MAS platform: Jason → BDI Agents → AgentSpeak
- 3D Engine: Open SceneGraph

KBS: Social simulation

System architecture

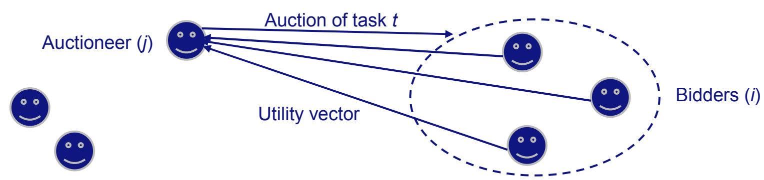


- MAS platform: Jason → BDI Agents → AgentSpeak
- 3D Engine: Open SceneGraph

KBS: Social simulation

Multi-agent Resource Allocation approach

- MARA *"…is the process of distributing a number of items (resources/tasks) amongst a number of agents…". [Chevaleyre et al]*
- What kind of resources is being distributed?
  - Tasks (that use objects in the environment)
  - Agents do not auction long sequences of actions but their next task
- Why are they being distributed?
  - Social agents interchange tasks/services between each other. [Piaget]
  - Allows the auctioning of tasks by any agent in order to reallocate them so that the global welfare can be increased
- How are they being distributed?
  - Allocation procedure: First-Price Sealed-Bid (FPSB) auction model



KBS: Social simulation

Coordination and Sociability (1)

- Utility vector: < U^i_{perf} , U^i_{int} , U^i_{ext} >
- Agent utility functions:
  - Performance utility function $U^i_{perf}(<i \leftarrow t>)$: Reflects the efficiency achieved when bidder *i* performs task *t*
  - Social utility functions: Represent the social interest in exchanging a task *t*. The aim is to promote social interactions with other agents
    - Internal social utility function $U^i_{int}(<i \leftarrow t, j \leftarrow t_{next}>)$: Utility to get the task
    - External social utility function $U^i_{ext}(<j \leftarrow t>)$: Utility given to the situation where the auctioneer executes the task

KBS: Social simulation

Coordination and Sociability (2)

Sociability

∈ [0,1]

$$winner_{perf}(t) = \left\{ k \in Agents \mid U^{k}_{perf}(t) = \max_{i \in Agents} \left\{ U^{i}_{perf}(<i \leftarrow t>) \right\} \right\} \quad (1)$$

$$winner_{soc}(t) = \begin{cases} j & U^{*}_{ext}(t) >= U^{*}_{int}(t) \\ i & U^{*}_{ext}(t) < U^{*}_{int}(t) \text{ and } U^{i}_{int}(t) = U^{*}_{int}(t) \end{cases} \quad (2)$$

Reciprocity

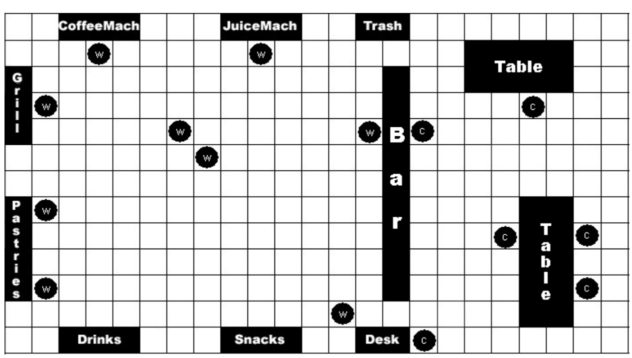
$$U^{*}_{int}(t) = \max_{i \in Agents} \left\{ U^{i}_{int}(<i \leftarrow t, j \leftarrow t_{next}>) * w_{ij} \right\} \quad (3)$$

$$U^{*}_{ext}(t) = \max_{i \in Agents} \left\{ U^{i}_{ext}(<j \leftarrow t>) * w_{ji} \right\} \quad (4)$$

$$w_{ij} = Favors_{ji} / Favors_{ij} \quad (5)$$

KBS: Social simulation

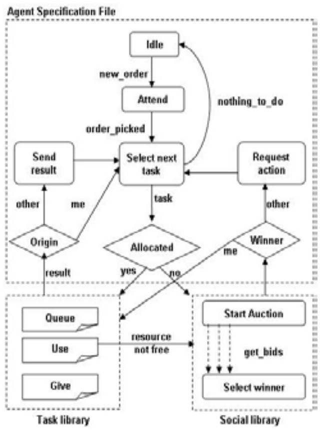
Application example: The virtual university bar



- Waiters: Serve orders and chat with their friends
- Customers: Place orders and consume with their friends
- Non shareable locations (e.g. the juice machine)

KBS: Social simulation

Application example: Waiters



- Performance utility function:

$$U^{i}_{perf}(i \leftarrow 'Use') = \begin{cases} 1 & \text{if } [(i = \text{Auctioneer}) \text{ and } (\text{IsFree(Resource)}] \text{ or } \\ & [\text{IsUsing(i, Resource) and not(IsComplete(Resource)}] \\ 0 & \text{Otherwise} \end{cases}$$

$$U^{i}_{perf}(i \leftarrow 'Give') = \begin{cases} 1 & \text{if } [(i = \text{Auctioneer}) \text{ and } (\text{nextAction} = \text{NULL})] \text{ or } \\ & [\text{currentTask} = '\text{Give' and not(handsBusy} < 2)] \\ 0 & \text{Otherwise} \end{cases}$$

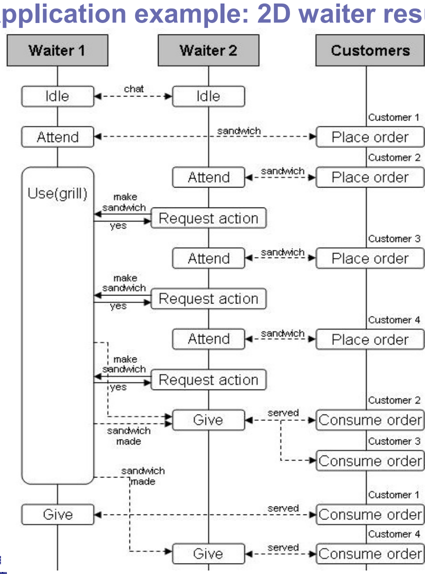
- Social utility functions:

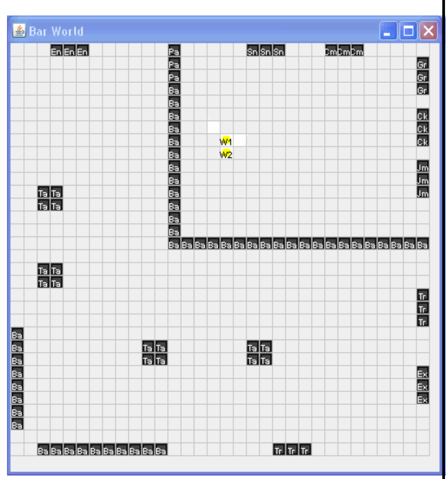
$$U^{i}_{\text{int}}(< i \leftarrow t, j \leftarrow t_{next} >) = \begin{cases} 1 & \text{if IsWorkFriend(i, j) and Near(t, } t_{next}\text{) and} \\ & \text{ExecTime(}t_{next}\text{)} > \text{RemainTime(currenTask)} \\ 0 & \text{Otherwise} \end{cases}$$

$$U^{i}_{ext}(j \leftarrow t) = \begin{cases} 1 & \text{if IsWorkFriend(i, j) and Near(currentTask, t)} \\ 0 & \text{Otherwise} \end{cases}$$

KBS: Social simulation

Application example: 2D waiter results

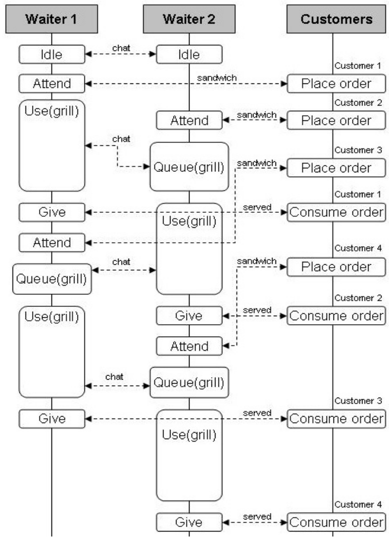


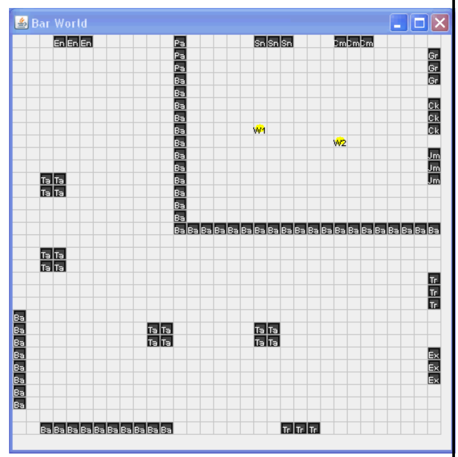


- Waiter sociability = 0

KBS: Social simulation

Application example: 2D waiter results

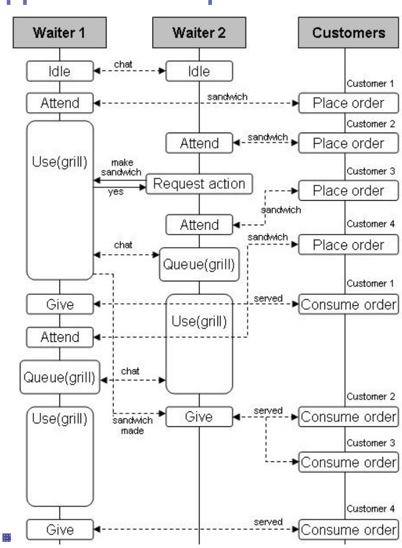


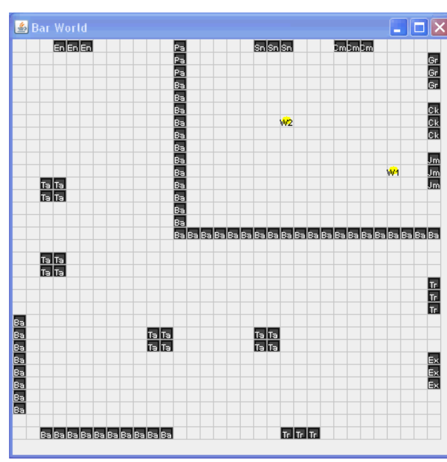


- Waiter sociability = 1

KBS: Social simulation

Application example: 2D waiter results





- Waiter sociability = 0.6

Future remarks

Consolider

- Our real world is becoming an artificial society!
- Multi-agent systems are an elegant and formal framework to create artificial societies.
- Use of semantic information to model the world and the society (ontologies, organizations, etc.).
- Social and organizational reasoning to enrich cognitive behaviours.
- Future work: Self-regulation, learning from experience...

Outline

Consolider

Introduction to Artificial Societies

Behavioural-based systems

Examples of social emergence

Creating AS in Jason

Knowledge-based systems

Semantic information

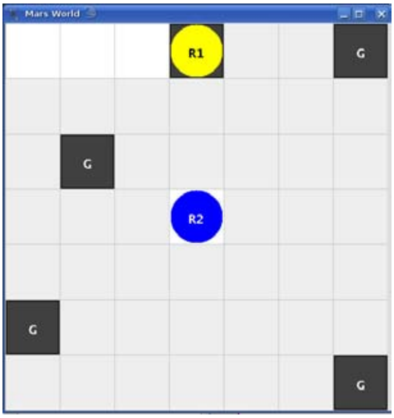
Social simulation: Coordination and Sociability

Future remarks

Exercise: Cooperative behaviour in Jason

Exercise: Cooperation in Jason

Cleaning robots example



- Cleaner agent (R1): explores the environment searching for garbage.
- Burner agent (R2): Receives and burns garbage.
- Explore AgentSpeak code.
- Explore Environment java code:
  - executeAction()
  - updatePercepts()
  - WorldModel class
  - WorldView class

Exercise: Cooperation in Jason

Extending the cleaning robots example

- Finalize MAS execution.
- Load problem: up to 20 random garbage.
- Continuous exploratory behaviour for the cleaner.
- Measure performance: execution time.
- How can a society of robots enhance global performance?
  - Increase the number of agents.
  - Assign zones of exploration.
  - Communicate where garbage is to other agents.

Exercise: Cooperation in Jason

Gold miners example

- Complex implementation with a leader that assigns quadrants to miners at start time.
- Negotiation for found gold: auctions to allocate gold the nearer miner.
- Miners can change a commitment to pick a gold if they found another one in the way.
- Better exploration algorithm: avoids sequential movement along the assigned quadrant.

Thank you

francisco.grimaldo@uv.es

December 10th, 2007