Department of Computer Science and Engineering

IIIrd YEAR / VIth SEMESTER

CS8691 – ARTIFICIAL INTELLIGENCE

Mr. P.G. Siva Sharma Karthick M.E., MBA.,
Assistant Professor
Nadar Saraswathi College of Engineering & Technology,
Vadapudupatti, Annanji (po), Theni – 625531.
UNIT 04 – SOFTWARE AGENTS
Agent definitions

- "An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors."

- "Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed."

- “An autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to affect what it senses in the future.”
Basic abstract view of an agent
Reactivity

- An agent has to be able to react [adapt its behaviour] in an appropriate way to the dynamic changes in its “environment”
  - Other computational agents
  - Human agents/users
  - External information sources (e.g. sensors)
  - Physical objects (e.g. robots)
  - Internet
Kinds of environments (I)

- **Accessible vs inaccessible**

  - An accessible environment is one in which the agent can obtain complete, accurate, up-to-date information about the environment’s state.
  
  - Most moderately complex environments (including, for example, the everyday physical world and the Internet) are inaccessible.
  
  - The more accessible an environment is, the simpler it is to build agents to operate in it.
Kinds of environments (II)

- **Deterministic vs non-deterministic**
  - A deterministic environment is one in which any action has a single guaranteed effect — there is no uncertainty about the state that will result from performing an action.
  - The physical world can to all intents and purposes be regarded as non-deterministic.
  - Non-deterministic environments present greater problems for the agent designer.
Kinds of environments (III)

- Episodic vs non-episodic

- In an episodic environment, the performance of an agent is dependent on a number of discrete episodes, with no link between the performance of an agent in different scenarios.

- Episodic environments are simpler from the agent developer’s perspective because the agent can decide what action to perform based only on the current episode — it need not reason about the interactions between this and future episodes.
Kinds of environments (IV)

- **Static vs dynamic**

  - A static environment is one that can be assumed to remain unchanged except by the performance of actions by the agent.
  - A dynamic environment is one that has other processes operating on it, and which hence changes in ways beyond the agent’s control.
  - The physical world is a highly dynamic environment.
Kinds of environments (V)

- **Discrete vs continuous**
  - An environment is discrete if there are a fixed, finite number of actions and percepts in it.
  - The real world is a continuous environment.
Agent architectures

- An architecture proposes a particular methodology for building an autonomous agent
  - How the construction of the agent can be decomposed into the construction of a set of component modules
  - How these modules should be made to interact
  - These two aspects define how the sensor data and the current internal state of the agent determine the actions (effector outputs) and future internal state of the agent
From perception to action

\[ f = \text{state update function} \]
\[ s = \text{internal state} \]
\[ g = \text{output function} \]
Main kinds of agent architectures

- **Reactive architectures**
  - Focused on fast reactions/responses to changes detected in the environment

- **Deliberative architectures (symbolic)**
  - Focused on long-term planning of actions, centred on a set of basic goals

- **Hybrid architectures**
  - Combining a reactive side and a deliberative side
Reactive vs Deliberative: example

- Robot that has to reach a certain point
  - **Reactive**
    - Sensor in the front of the robot
    - Change movement right/left when sensor detects obstacle
      - Minimal computation based on current location and destination point
  - **Deliberative**
    - Explicit representation of the environment (map)
    - Planning procedure that finds the minimal route between the current position and the destination
      - High computational cost
      - Possible dynamic re-plannings needed
There are many unsolved (some would say insoluble) problems associated with symbolic AI:
- Computational cost, brute search
- Problems below the 100 ms threshold
  - For example, face recognition

These problems have led some researchers to question the viability of the whole paradigm, and to the development of reactive architectures.

Although united by a belief that the assumptions underpinning mainstream AI are in some sense wrong, reactive agent researchers use many different techniques.
Reactive agents – basic ideas

- Reactive agents have
  - at most a *very simple internal representation* of the world,
  - but provide *tight coupling of perception and action*
- Behaviour-based paradigm
- Intelligence is a product of the *interaction* between an agent and its environment
Classic example: ant colony

- A single ant has very little intelligence, computing power or reasoning abilities.
- The union of a set of ants and the interaction between them allows the formation of a highly complex, structured and efficient system.
Main characteristics (I)

- **Emergent functionality**
  - Simple agents
  - Simple interaction
  - Complex behaviour patterns appear as a result of the dynamic interactions
  - The global behaviour of the system is not specified a priori
    - Dynamic movement of robots, depending on obstacles
Main characteristics (II)

- **Task decomposition**
  - Agents composed of autonomous modules
  - Each module manages a given task
    - Sensor, control, computations
  - Minimal, low-level communication between modules
  - There isn’t any world global model
  - There isn’t any “planning/controller agent”
Main characteristics (III)

- **Raw data**
  - Basic data from sensors
  - There isn’t any complex symbolic management of data as in classical AI
    - Refusal of the *Hypothesis of the physic symbols system* [basic pillar of symbolic AI]
      - “Intelligent behaviour can only be obtained in symbol-processing systems”
Basic concept

- Each behaviour continually maps perceptual input to action output
- Reactive behaviour: action rules: $S \rightarrow A$
  where $S$ denotes the states of the environment, and $A$ the primitive actions the agent is capable of performing.
- Example:

  \[
  \text{action(s)} = \begin{cases} 
  \text{Heater off}, & \text{if temperature is OK in state s} \\
  \text{Heater on}, & \text{otherwise}
  \end{cases}
  \]
Basic schema of reactive architecture

Agent

Stimulus-response behaviours

State$_1$ → Action$_1$

State$_2$ → Action$_2$

::

State$_n$ → Action$_n$

Sensors

Effectors

Dept of CSE NSCET, Theni
Brooks refutal of symbolic AI

- Brooks has put forward three theses:
  1. Intelligent behaviour can be generated without explicit representations of the kind that symbolic AI proposes.
  2. Intelligent behaviour can be generated without explicit abstract reasoning of the kind that symbolic AI proposes.
     - Reduced computation on sensor-like data.
  3. Intelligence is an emergent property of certain complex systems.
Brooks – key ideas (I)

- **Situatedness**: ‘Real’ intelligence is situated in the world
  - The world is its best model
  - The world is always up-to-date
  - A model is an abstraction, a simplification of the world, considering a particular set of characteristics and disregarding others
Brooks – key ideas (II)

- **Embodiment**: ‘Real’ intelligence requires a physical body, and cannot be found in disembodied systems such as theorem provers or expert systems
  - Physical robots

- **Intelligence and emergence**: ‘Intelligent’ behavior arises as a result of an agent’s interaction with its environment. Also, intelligence is ‘in the eye of the beholder’; it is not an innate, isolated property
Brooks – behaviour languages

- To illustrate his ideas, Brooks built some systems based on his subsumption architecture.
- A subsumption architecture is a hierarchy of task-achieving behaviours.
- Each behaviour is a rather simple rule-like structure.
- Each behaviour ‘competes’ with others to exercise control over the agent, as different behaviours may be applicable at the same time.
Behaviour layers

- Lower layers represent more primitive kinds of behaviour (such as avoiding obstacles)
- Higher layers represent more complex behaviours (e.g. identifying an object)
- Lower layers have precedence over layers further up the hierarchy
- The resulting systems are, in terms of the amount of computation they do, extremely simple
- Some of the robots do tasks that would be impressive if they were accomplished by symbolic AI systems
Decomposition Based on Task Achieving Behaviours

Sensors → reason about behavior of objects → plan changes to the world → identify objects → monitor changes → build maps → explore → wander → avoid objects → Actuators
Situated Automata components

- An agent is specified in terms of two components: perception and action.
- Two programs are then used to synthesize agents:
  - RULER is used to specify the perception component of an agent.
  - GAPPS is used to specify the action component.
Circuit Model of a Finite-State Machine

Dept of CSE NSCET, Theni
RULER – Situated Automata

- RULER takes as its input three components
  - The semantics of the agent's inputs (‘whenever bit 1 is on, it is raining’)
  - A set of static facts (‘whenever it is raining, the ground is wet’)
  - A specification of the state transitions of the world (‘if the ground is wet, it stays wet until the sun comes out’).

- The programmer then specifies the desired semantics for the output (‘if this bit is on, the ground is wet’)

- The compiler designs a circuit whose output will have the correct semantics
GAPPS – Situated Automata

- The GAPPS program takes as its input
  - A set of *goal reduction rules*,
    - Rules that encode information about how goals can be achieved in a given state
  - A top level goal

- Then it generates a program that can be translated into a digital circuit in order to realize the goal

- The generated circuit does not represent or manipulate symbolic expressions; all symbolic manipulation is done at compile time
Advantages of Reactive Agents

- **Simplicity** of individual agents
- **Flexibility, adaptability**
  - Ideal in very dynamic and unpredictable environments
- **Computational tractability**
  - Avoiding complex planning/reasoning procedures
  - Avoiding continuous model update
- **Robustness** against failure
  - No central planning component (e.g. ant colony)
- **Elegance**
Limitations of Reactive Agents (I)

- Agents without environment models must have sufficient information available from local environment.
- If decisions are based on *local* environment, how can we take into account *non-local* information?
  - “Short-term” view
- No long-term planning capabilities
- Limited applicability
  - Games, simulations, basic robots (insects)
Limitations of Reactive Agents (II)

- Difficult to make reactive agents that learn
  - Dynamic evolution of rules?
- Since behaviour emerges from component interactions plus environment, it is hard to see how to engineer specific agents (no principled methodology exists)
- It is hard to engineer agents with large numbers of behaviours (dynamics of interactions become too complex to understand)
Deliberative agent architecture

- Explicit symbolic model of the world
- Decisions are made via logical reasoning, based on pattern matching and symbolic manipulation
- Sense-plan-act problem-solving paradigm of classical AI planning systems
Basic deliberative architecture
Practical reasoning

Reasoning directed towards actions — the process of figuring out what to do:

“Practical reasoning is a matter of weighing conflicting considerations for and against competing options, where the relevant considerations are provided by what the agent desires and what the agent believes.” (Bratman)

“We deliberate not about ends, but about means. We assume the end and consider how and by what means it is attained.” (Aristotle)
Human practical reasoning

- Human practical reasoning consists of two activities:
  - **Deliberation**, deciding *what* state of affairs we want to achieve
    - the outputs of deliberation are *intentions*
  - **Means-ends reasoning**, deciding *how* to achieve these states of affairs
    - the outputs of means-ends reasoning are *plans*
Belief-Desire-Intention paradigm

- **Beliefs:**
  - Agent's view of the environment/world.

- **Desires:**
  - Follow from the beliefs. Desires can be unrealistic and inconsistent.

- **Goals:**
  - A subset of the desires. Realistic and consistent. Determine potential processing.

- **Intentions:**
  - A subset of the goals. A goal becomes an intention when an agent decides to commit to it (e.g. by assigning priorities to goals)

- **Plans:**
  - Sequences of actions that are needed to achieve the intentions, given the agent's beliefs

Dept of CSE NSCET, Theni
BDI plans

In BDI implementations plans usually have:

- a *name*
- a *goal*
  - invocation condition that is the triggering event for the plan
- a *pre-condition list*
  - list of facts which must be true for plan to be executed
- a *delete list*
  - list of facts that are no longer true after plan is performed
- an *add list*
  - list of facts made true by executing the actions of the plan
- a *body*
  - list of actions
Belief-Desire-Intention architecture

Desires

Goals

Intentions

Plans

Knowledge

Beliefs
Intention is choice with commitment (Cohen & Levesque)

- There should be "rational balance" among the beliefs, goals, plans, intentions, commitments and actions of autonomous agents.
- Intentions play a big role in maintaining "rational balance"
- An autonomous agent should act on its intentions, not in spite of them
  - adopt intentions that are feasible
  - drop the ones that are not feasible
  - keep (or commit to) intentions, but not forever
  - discharge those intentions believed to have been satisfied
  - alter intentions when relevant beliefs change
Using plans to constrain reasoning

- An agent’s plans serve to frame its subsequent reasoning problems so as to constrain the amount of resources needed to solve them
  - Agents commit to their plans
  - Their plans tell them what to reason about, and what not to reason about
  - Plans can help reasoning in different levels of abstraction
Intention reconsideration

- Intentions (plans) enable the agent to be goal-driven rather than event-driven.
- By committing to intentions the agent can pursue long-term goals.
- However, it is necessary for a BDI agent to reconsider its intentions from time to time:
  - The agent should drop intentions that are no longer achievable.
  - The agent should adopt new intentions that are enabled by opportunities.
Problems in the deliberative approach

- **Dynamic world**
  - Update symbolic world model
  - World changes while planning is being done

- **Representation language**
  - Expressive enough to be useful in any domain
  - Limited enough to be computationally tractable

- **Classical planning => complete, optimal solutions**
  - High computational cost
  - Sometimes a sub-optimal low-cost fast reaction can be effective
Hybrid Approaches

- Many researchers have argued that neither a completely deliberative nor a completely reactive approach are suitable for building agents.
- They have suggested using hybrid systems, which attempt to marry classical and alternative approaches.
- An obvious approach is to build an agent out of two (or more) subsystems:
  - a deliberative one, containing a symbolic world model, which develops plans and makes decisions in the way proposed by symbolic AI.
  - a reactive one, which is capable of reacting quickly to events without complex reasoning.
Hybrid agent architecture

Agent

Deliberative component

World Model \rightarrow Planner \rightarrow Plan executor

Reactive component

State_1 \rightarrow Action_1

State_2 \rightarrow Action_2

\ldots

State_n \rightarrow Action_n

Observation \uparrow \text{modification} \downarrow

Sensors \rightarrow Effectors

Dept of CSE NSCET, Theni
Layered Architectures

- Often, the reactive component is given some kind of precedence over the deliberative one.
- This kind of structuring leads naturally to the idea of a *layered* architecture, of which TOURINGMACHINES and INTERRAP are examples.
- In such an architecture, an agent’s control subsystems are arranged into a hierarchy, with higher layers dealing with information at increasing levels of abstraction.
Layering techniques

- A key problem in such architectures is what kind of control framework to embed the agent’s subsystems in, to manage the interactions between the various layers.

- **Horizontal layering**
  Each layer is directly connected to the sensory input and action output.
  In effect, each layer itself acts like an agent, producing suggestions as to what action to perform.

- **Vertical layering**
  Sensory input and action output are dealt with by at most one layer each.
Horizontal layering

$m$ possible actions suggested by each layer, $n$ layers

$O(m^n)$ possible options to be considered

Introduces bottleneck in central control system

Dept of CSE NSCET, Theni
Vertical layering

$m$ possible actions suggested by each layer, $n$ layers

$O(mn)$ interactions between layers

Not fault tolerant to layer failure
Cooperative planning layer

- Works with the social model (beliefs on other agents of the system)
- Allows planning and cooperation with other agents
  - Global plans of action
  - Conflict resolution
Critiques to hybrid architectures

- Lack of general design guiding methodologies
- Very specific, application dependent
- Unsupported by formal theories
Collaborative Agents (I)

- A multi-agent system (MAS) may be seen as a collection of collaborative agents.
- They can communicate and cooperate with other agents, while keeping their autonomy.
- They usually negotiate with their peers to reach mutually acceptable agreements during cooperative problem solving.
They normally have limited learning capabilities

Collaborative agents are usually deliberative agents (e.g. BDI model), with some reasoning capabilities

- Reactive agents can hardly communicate and collaborate (only through actions that modify the common environment)

They are usually static, complex agents
Collaborative Agents: Motivations (I)

- To solve problems that are too large for a single centralised agent
  - Huge amount of knowledge to be considered
  - Many computational resources needed to solve the problem
  - Risk of having a centralised system
Collaborative Agents: Motivations (II)

- To create a system that functions beyond the capabilities of any of its members
  - Added value of a MAS

- To allow for the interconnection and inter-operation of existing legacy systems
  [ Recall agentification mechanisms seen on the previous lecture: translators, wrappers]
  - DBs, expert systems, electronic equipment, sensors

Example: organ transplant coordination
Collaborative Agents: Applications

- Provide solutions to **physically distributed problems**
  - Disaster in a city (police, firemen, ambulances)

- Provide solutions to problems with **distributed data sources**
  - Sensor network monitoring a given area

- Provide solutions that need **distributed expertise**
  - Health care provision (family doctors, nurses, specialists, laboratory analysis, …)
Benefits of Multi-Agent Systems (I)

- **Modularity**
  - Each agent is specialised in the solution of a particular kind of problems (leading also to **reusability**)
  - The complexity of the construction of agents is reduced
  - The process of solving a complex problem is reduced to solving easier subproblems
Benefits of Multi-Agent Systems (II)

- **Efficiency**
  - Problems can be solved more quickly, due to the inherent concurrency/parallelism
  - Different agents are working at the same time in different parts of a problem
    - These subproblems can be independent or (slightly) dependent
      - Share partial results
      - Coordinate the use of shared resources
Benefits of Multi-Agent Systems (III)

- **Reliability**
  - Avoid single point of failure in centralised systems
  - We can have redundancy
    - Different agents of the same type
    - Different agents that can do a certain task
  - If an individual agent fails, the other agents can take its work and re-distribute it dynamically
Benefits of Multi-Agent Systems (IV)

- **Flexibility**
  - Agents can be created/deleted dynamically, depending on the amount of work to be done, the available resources, etc.
  - Agents can dynamically generate subtasks and look for helping agents.
  - Agents with different skills may dynamically form teams/coalitions to work together.
Conflict management

- Examples
  - Two sub-solutions are incompatible
  - Conflicts in the use of shared resources
- Agents have to communicate with each other to solve these situations
- There may be different solutions to the same subproblem
Agent support in task execution (I)

Task sharing

- An agent can request the help of other agents to solve a particular task
  - Too complex / expensive for the agent to do individually
  - It can know that other agents have the appropriate knowledge/skills to solve that task
  - It can know that other agents already have to solve that task

Problem of task assignment

- Who can I ask for help?
- How do I know what tasks can other agents do?
Agent support in task execution (II)

- **Result sharing**
  - Use *intermediate results* obtained by other agents
    - [For example, traffic routes to certain points of the city]
  - Agents can provide intermediate subsolutions to help other agents in their work
  - That allows a fast recognition of
    - **Incorrect solutions**
      - An agent, on the basis of its knowledge, can detect an error on the results of other agents
    - **Conflictive solutions**
      - An agent can detect possible conflicts between its results and subsolutions of other agents
  - Cooperation/Negotiation to solve these problems
4-Result synthesis

- Put together the results of all agents to find the complete solution
- Who makes it?
- How is it made?
- If each subproblem has a unique solution, it is a relatively easy step
- Otherwise, there may be need of conflict detection, task re-assignment,
Why do we need Agent Communication?

- Multi agent systems allow distributed problem solving
- This requires the agents to coordinate their actions
- Agent communication facilitates this by allowing individual agents to interact
  - allows cooperation
  - allows information sharing
Communication Categories

- We can classify communication in a mechanistic manner
  - via the type of sendee-addressee link
  - via the nature of the medium
- or in a higher level meaning-based manner
  - via the type of intention
The Sendee - Addressee Link

- Communication can be
  - **Point to Point**
    - An agent talks directly to another agent
  - **Broadcast**
    - An agent sends some information to a group of agents
  - **Mediated**
    - The communication between two agents is mediated by a third party
    - Example: facilitators
1. Agent_2 tells the facilitator the services it provides
2. Agent_1 asks to the facilitator who can provide a certain service with some conditions
3. The facilitator requests the service to agent_2
4. Agent_2 provides the answer
5. The facilitator sends the answer to agent_1
1. Agent_2 tells the facilitator the services it provides
2. Agent_1 asks to the facilitator who can provide a certain service with some conditions
3. The facilitator tells agent_1 that agent_2 can do that service
4. Agent_1 requests the service from agent_2
5. Agent_2 sends the answer to agent_1
Locating other agents

- Unless we use some broadcast techniques (e.g. blackboard systems), agents must know the address of other agents - possible solutions are
  - Complete internal directory
  - Partial/hierarchical internal directory
  - Mediated (e.g. JADE’s DF)
Nature of the medium

- **Direct routing**
  - Message sent directly to other agent(s) with no interception or attenuation in strength

- **Signal propagation routing**
  - Commonly used by reactive agents
  - Agent sends signal whose intensity decreases according to distance (e.g. physical robots)

- **Public notice routing**
  - Blackboard systems
Agent communication

- Basic options used in MAS
  - Blackboard systems
  - Direct message passing
Blackboard Systems

![Diagram of Blackboard Systems with agents connected to a central blackboard]

Dept of CSE NSCET, Theni
Blackboard-based communication

- Each agent can put information/data/knowledge on the common information space
- Each agent can read from the blackboard at any moment
- There is no direct communication between agents
Information in blackboard

- **Data** of the common problem
- Current **state** of the solution
- Next **subproblems** to be solved
- Requests of **help**
- Present **task** of each agent
- Intermediate **results**
Uses of blackboard

- Detect **conflicts**
  - Different agents that want to perform the same task

- Notice **incompatible solutions**
  - Solutions using a shared resource at the same time

- Share results
  - Agents can use partial/complete results obtained by other agents

- Share tasks
  - Agents can request help in solving sub-tasks
Advanced blackboard systems (I)

- **Moderator agent**
  - Advertises in the blackboard the next problems to be solved
  - Checks which agents offer to solve them
  - Assigns the pending problems to the agents
    - It has internal domain and system knowledge to make this assignment
    - The blackboard is also used to communicate the assignments
Advanced blackboard systems (II)

- Dispatcher agent
  - Tells the agents registered in a blackboard about the changes produced on it that can be interesting/relevant for them
    - Example: new problem announcement => tell the agents that can be potentially interested in solving it
  - Agents do not need to be continuously checking the blackboard
Blackboard systems summary (I)

- **Positive aspects**
  - Flexible mechanism that allows communication/cooperation
    - E.g. $n$ blackboards
  - Independent of cooperation strategy
  - It does not place any restriction on the agents’ internal architecture
Message passing

- Information is passed from one agent to another. The nature of this information can be very varied. **Speech acts** provide one way to describe this variety.
Speech Acts

- Most treatments of communication in multi-agent systems borrow their inspiration from *speech act theory*
- Speech act theories are *pragmatic* theories of language, i.e., theories of language use: they attempt to account for how language is used by people every day to achieve their goals and intentions
- The origin of speech act theories is usually traced to Austin’s 1962 book, *How to Do Things with Words*
Speech Act Theory

- Austin noticed that some utterances are like ‘physical actions’ that appear to change the state of the world

- Paradigmatic examples would be:
  - Declaring war
  - ‘I now pronounce you man and wife’

- But more generally, everything we utter is uttered with the intention of satisfying some goal

- A theory of how utterances are used to achieve intentions is a speech act theory
Speech Acts

- A speech act is an act of communication
- Speech does not imply any particular communication media
- There are various types of speech act
- By using the various types of speech act, agents can interact effectively
Types of speech act

- **inform** other agents about some data
- **query** others about their current situation
- **answer** questions
- **request** others to act
- **promise** to do something
- **offer** deals
- **acknowledge** offers and requests
- …
Speech acts types – Searle (I)

- **Representative**
  - Communicate some state of affairs
    - Informing, asserting, claiming, describing, ...

- **Commissive**
  - Commit the speaker to some future course of action
    - Promising, agreeing, threatening, inviting, offering, swearing, volunteering, ...
Speech acts types – Searle (II)

- **Directive**
  - Intention to get the receiver to carry out some action
    - Requesting, commanding, daring, asking, begging, forbidding, advising, ...

- **Declaration**
  - Bring about a state of affairs
    - Arresting, marrying, declaring, ...

- **Expressive**
  - Indicate the speaker’s psychological state or mental attitude
    - Thanking, greeting, congratulating, apologizing, ...
Speech Act components

In general, a speech act can be seen to have two components:

- **a performative verb:**
  (e.g., request, inform, promise, ...)

- **propositional content:**
  (e.g., “the door is closed”)
Communication Standards

- Agents must understand each other even if running on different machines and/or different operating systems

- Standards
  - allow different groups to write cooperating agents
  - help abstract out communication, by defining high-level general languages and protocols
FIPA

- Foundation for Intelligent Physical Agents
- Beginning (1996): stand-alone non-profit organisation
- Now: IEEE Computer Society standards committee
- Mission: develop and promote agent standards
  - MAS architecture
  - Agent communication language (FIPA-ACL)
  - Communication protocols
Components of a FIPA-ACL message

ACL message

```
{inform
  :send agent1
  :receiver hpl-auction-server
  :content
    (price (bid good02 150)
    :in-reply-to round-4
    :reply-with bid04
    :ontology hpl-auction
    :language sl)
}
```
Parameters in a FIPA ACL message

- :sender - who sends the message
- :receiver - who is the recipient of the message
- :content - content of the message
- :reply-with - identifier of the message
- :reply-by - deadline for replying the message
- :in-reply-to - identifier of the message being replied
- :language – language in which the content is written
- :ontology - ontology used to represent the domain concepts
- :protocol - communication protocol to be followed
- :conversation-id - identifier of conversation
# FIPA-ACL performatives

<table>
<thead>
<tr>
<th>performative</th>
<th>passing info</th>
<th>requesting info</th>
<th>negotiation</th>
<th>performing actions</th>
<th>error handling</th>
</tr>
</thead>
<tbody>
<tr>
<td>accept-proposal</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>agree</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cancel</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cfp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>confirm</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>disconfirm</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>failure</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>inform</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>inform-if</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>inform-ref</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>not-understood</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>propose</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>query-if</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>query-ref</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>refuse</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>reject-proposal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>request-when</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>request-whenever</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>subscribe</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Dept of CSE NSCET, Theni
Inform

- Content: statement
- The sender informs the receiver that a given proposition is true
- The sending agent:
  - holds that some proposition is true
  - intends that the receiving agent also comes to believe that the proposition is true
  - does not already believe that the receiver has any knowledge of the truth of the proposition
Inform example

(inform

:sender  (agent-identifier :name i)
:receiver (agent-identifier :name j)
:content  "door( now, open )"
:language Prolog)

)
Request

- **Content:** action
- The sender requests the receiver to perform some action
- The sending agent:
  - intends the action content to be performed
  - believes recipient is capable of performing this action
  - does not believe that receiver already intends to perform action
Request example

(request
  :sender (agent-identifier :name i)
  :receiver (agent-identifier :name j)
  :content (action (agent-identifier :name j) open_the_door)
  :language fipa-sl
)

Dept of CSE NSCET, Theni
Communication protocols

- There are many situations in which agents engaged in a dialogue with a certain purpose exchange the same sequence of messages
  - When an agent makes a question to another
  - When an agent requests a service from another
  - When an agent looks for help from other agents

- To ease the management of this typical message interchanges we can use predefined protocols
FIPA-Request protocol
FIPA-Query protocol
FIPA- Contract Net protocol
FIPA Contract Net

1) cfp
2) propose
3) propose
4) propose
5) reject-proposal
6) inform
5) accept-proposal
1) cfp
2) propose
1) cfp
5) reject-proposal

Dept of CSE NSCET, Theni
What is Negotiation?

- **Negotiation** is a form of interaction in which a group of agents with conflicting interests try to come to a mutually acceptable agreement over some outcome.

- The outcome is typically represented in terms of the allocation of resources (commodities, services, time, money, CPU cycles, etc.)

- Agents' interests are conflicting in the sense that they cannot be simultaneously satisfied, either partially or fully (= trade-off)

- Automated negotiation would be negotiation that is automated with some computation support, e.g., fully automated negotiation among computational agents, partially automated negotiation with a computational mediator with human negotiators, etc.
bilateral negotiation

- We focus on “bilateral negotiations”, that is, negotiations involving two agents.

- Multi-party negotiations refers negotiations involving many of agents. In general, auctions and mechanisms can been seen as multi-party negotiations. Also, some researchers now focusing on heuristic-based one.
Main ingredients of Negotiation

1. The negotiation object, which defines the set of possible outcomes

2. The agents conducting the negotiation

3. The protocol according to which agents search for a specific agreement

4. The individual strategies that determine the agents’ behavior based on their preferences over the outcomes
Negotiation outcomes

- There are many ways to define the outcomes.
- Also called as agreements or deals.
- Characteristics
  - Continuous or discrete
  - Single issue or multiple issues
Example 1

- 1.0 litter milk between Alice and Bob
- The issue is (dividing) milk, that is single issue & continuous
- The possible outcome can be represented as a number in interval $[0, 1.0]$.
- One possible outcome is 0.2l for Alice and 0.8l for Bob.
Example 2

- Parking slot 1 and 2 for Charles and Daniel
- The issue is 3 parking slots, that is single issue & discrete
- The possible outcome can be represented as assignment of the parking slot
- One possible outcome is slot 1 for Charles and slot 2 for Bob
Example 3

- Buying a house between Seller and Buyer
  - The issues are price, design, and place (3 issues), that is multiple issue (& discrete and continuous)
  - The possible outcome can be represented as a tuple of values of the issues.
  - One possible outcome is ($150,000, modern, Dunedin)
Utility function

• One way to define preference relation for agent $i$ is to define a utility function $u_i : \Omega \to \mathbb{R}^+$ to assign real number to each possible outcome (cardinal utility $\leftrightarrow$ ordinal utility).

• The utility function $u_i(\cdot)$ represents the relation $\geq_i$ when we have $u_i(o_1) \geq u_i(o_2)$ if $o_1 \succeq_i o_2$.

• In multi-issue negotiation, it is possible to have a multi-attribute utility function which maps a vector of attribute values to a real number.

• A rational agent attempts to reach a deal that maximizes his/her utility.
Protocols

- Given a set of the agents and their preferences/utilities, we need a protocol.

- A protocol is rules of interaction for enabling the agents to search for an agreement.

- One-shot or repeated

- There are many protocols proposed so far.

- Example: Alternative-offer protocol (we will see this in the later section), auction, mediator, etc.
Strategy

- Given a set of agents, their preferences, and an agreed protocol, the final ingredient is the agent’s strategy.

- The strategy may specify what offer to make next or what information to reveal (truthfully or not).

- A rational agent’s strategy must aim to achieve the best possible outcome for him/her.

- Game-theory is analyzing agents’ strategic behavior.
Pareto Optimality

- At the Pareto optimal situation, without reducing another agent’s utility, there is no agent who can increase his/her utility.

- An outcome \( d \) is Pareto efficient (Pareto optimal) if there is no outcome that is better for at least one agent and not worse for the other agent.

- There is no game outcome \( d' \) for agents \( A \) and \( B \) s.t.
  \[
  [ u_A(d') \geq u_A(d) \quad \text{and} \quad u_B(d') \geq u_B(d) ]
  \]
  and \[
  [ u_A(d') > u_A(d) \quad \text{or} \quad u_B(d') > u_B(d) ]
  \]
Example: Cake division

- When dividing one cake, it is Pareto optimal if the entire cake is completely divided and allocated to members, and there is no remaining pieces.

- Pareto optimal does not mean fairness.

Blue or yellow can increase his/her cake without reducing opponent's cake!

Without reducing opponent's utility, there is no agent who can increase his/her cake.
Cooperative game or Non-cooperative game

- There are two ways to model bilateral negotiations: using cooperative game or using non-cooperative game.

  - In cooperative games, agreements are enforceable or binding, and it’s possible for the agents to negotiate outcomes that are mutually beneficial.

  - In non-cooperative game, the agents are self-interested and thus they have incentive to deviate from an agreement to improve his/her utility.

  - Thus, a same game would have the different outcome between cooperative games and non-cooperative games.
Bargaining based on non-cooperative game

• Usually, non-cooperative model of bargaining specifies a procedure of negotiation.
What is Argumentation?
What philosophers call it!

Arguing with Others
“A verbal and social activity of reason aimed at increasing (or decreasing) the acceptability of a controversial standpoint for the listener or reader, by putting forward a constellation of propositions (i.e. arguments) intended to justify (or refute) the standpoint before a rational judge” [van Eemeren et al]

“the giving of reasons to support or criticize a claim that is questionable, or open to doubt” [Walton]

Argumentation versus Reasoning
If you are the judge, argumentation becomes (nonmonotonic) reasoning.
Process of Argumentation

1. Constructing arguments (in favor of / against a “statement”) from available information.
   A: “Tweety is a bird, so it flies”
   B: “Tweety is just a cartoon!”

2. Determining the different conflicts among the arguments.
   “Since Tweety is a cartoon, it cannot fly!” (B attacks A)

3. Evaluating the acceptability of the different arguments.
   “Since we have no reason to believe otherwise, we’ll assume Tweety is a cartoon.” (accept B). “But then, this means despite being a bird he cannot fly.” (reject A).

4. Concluding, or defining the justified conclusions.
   “We conclude that Tweety cannot fly!”
Argumentation Scheme

Argumentation schemes are forms (or categories) of argument, representing stereotypical ways of drawing inferences from particular patterns of premises to conclusions in a particular domain (e.g. reasoning about action).

For each scheme, we list:

1. Premises
2. Conclusion
3. A set of *critical questions* that can be used to scrutinize the argument by questioning explicit or implicit premises.

Various formal and semi-formal models have been proposed.
Argumentation Scheme Example

Walton's "sufficient condition scheme for practical reasoning":

In the current circumstances R
We should perform action A
Which will result in new circumstances S
Which will realise goal G
Which will promote some value V.

Associated critical questions include:

CQ1: Are the believed circumstances true?
CQ2: Does the action have the stated consequences?
CQ3: Assuming the circumstances and that the action has
the stated consequences, will the action bring about the
desired goal?
CQ4: Does the goal realise the value stated?
CQ5: Are there alternative ways of realising the same
consequences?
Argument Labeling

A labelling specifies which arguments are:
- Accepted (labelled in)
- Rejected (labelled out)
- Undecided (labelled undec)

Labellings must satisfy the condition:
- An argument is in if and only if all of its defeating arguments are out.
- An argument is out if and only if at least one of its defeating arguments is in.
- Otherwise, it is undecided.

For any labelling satisfying the conditions above, those arguments that happen to be labelled in form a complete extension.
What is an Abstract Argument Game?

- Typically two agents:
  - PRO (the proponent)
  - OPP (the opponent)

- *Dialogue* begins with PRO asserting an argument.

- Then PRO and OPP take turns in a *dispute*, where each player makes an argument that attacks his counterpart’s last move.

- Agent wins a dispute if his counterpart cannot counter attack.

- But the counterpart may try a different line of attack, creating a new dispute.

- This results in a *dispute tree* structure.
The Power of Abstract Argument Games

- We can say that a player A has a *winning strategy* for an argument $x$ if, no matter what the other player does, player A wins.

- By adjusting the protocol, abstract argument games can correspond to different semantics:

  \[
  \text{I.e. PRO wins if and only if the argument in question belongs to the corresponding extension.}
  \]

- Adjustments include things like:
  - Preventing PRO or OPP from repeating their own moves or each others’ moves
  - Preventing players from presenting arguments that attack arguments they themselves stated previously
  - etc.
Strategic Behaviour in Argumentation

- So far, we focused on protocols (e.g. dialogue systems)
- Protocols specify the set of possible moves agents can make
- Agents may have many choices about what to say at a given time
- These choices, the agent’s *strategy*, significantly influence:
  1. the outcome of the dialogue
     (e.g. who wins)
  2. dialogue dynamics
     (e.g. whether it will terminate in a short number of moves).
Heuristic Argumentation Strategies

For example specify *attitudes* (due to Parsons et al, JLC):

- **Assertion attitudes:**
  - *confident* agent asserts any proposition for which he can construct an argument,
  - *careful* agent can do so only if he can construct such an argument and cannot construct a stronger counterargument
  - *thoughtful* agent can assert an a proposition only if he can construct an acceptable argument for the proposition.

- **Evaluation attitudes:**
  - *credulous* agent accepts a proposition if he can construct an argument for it
  - *cautious* agent does so only if he’s also unable to construct a stronger counterargument
  - *skeptical* agent accepts an argument only if he can construct an acceptable argument for the proposition
Why Argument Interchange Format?

- Common language for annotating argument structures
- Enable export / import between argumentation support tools
- Ideally:
  - Expressive enough (but not too much)
  - Extensible / customizable
  - Implementable with standard ontology languages (allows using common parsers, or even Web 3.0 reasoning engines)
Trust

A couple of definitions that I like:

“Trust begins where knowledge [certainty] ends: trust provides a basis dealing with uncertain, complex, and threatening images of the future.” (Luhmann, 1979)

“Trust is the outcome of observations leading to the belief that the actions of another may be relied upon, without explicit guarantee, to achieve a goal in a risky situation.” (Elofson, 2001)
Trust

“The subjective probability by which an individual, $A$, expects that another individual, $B$, performs a given action on which its welfare depends” [Gambetta]

“An expectation about an uncertain behaviour” [Marsh]

“The decision and the act of relying on, counting on, depending on [the trustee]” [Castelfranchi & Falcone]
**Reputation**

“**What a social entity says about a target regarding his/her behavior**”

- It is always associated to a specific behaviour/property
  - The social evaluation linked to the reputation is not necessarily a belief of the issuer.
  - Reputation cannot exist without communication.

Set of individuals plus a set of social relations among these individuals or properties that identify them as a group in front of its own members and the society at large.
What is reputation good for?

- Reputation is one of the elements that allows us to **build trust**.
- Reputation has also a social dimension. It is not only useful for the individual but also for the society as a mechanism for **social order**.
Characteristics of computational trust and reputation mechanisms

• Each agent is a norm enforcer and is also under surveillance by the others. No central authority needed.

• Their nature allows to arrive where laws and central authorities cannot.

• Punishment is based usually in ostracism. Therefore, exclusion must be a punishment for the outsider.
Characteristics of computational trust and reputation mechanisms

- Bootstrap problem.

- Not all kind of environments are suitable to apply these mechanisms. It is necessary a social environment.
Different approaches to control the interaction

Social approach

Institutional approach

Security approach

Trust and reputation mechanisms are at this level.

They are complementary and cover different aspects of interaction.

Dept of CSE NSCET, Theni
Classification dimensions

- **Paradigm type**
  - Mathematical approach
  - Cognitive approach

- **Information sources**
  - Direct experiences
  - Witness information
  - Sociological information
  - Prejudice

- **Visibility types**
  - Subjective
  - Global

- **Model’s granularity**
  - Single context
  - Multi context

- **Agent behaviour assumptions**
  - Cheating is not considered
  - Agents can hide or bias the information but they never lie

- **Type of exchanged information**
Subjective vs Global

• Global
  • The reputation is maintained as a centralized resource.
  • All the agents in that society have access to the same reputation values.

Advantages:
• Reputation information is available even if you are a newcomer and do not depend on how well connected or good informants you have.
• Agents can be simpler because they don’t need to calculate reputation values, just use them.

Disadvantages:
• Particular mental states of the agent or its singular situation are not taken into account when reputation is calculated. Therefore, a global view it is only possible when we can assume that all the agents think and behave similar.
• Not always is desirable for an agent to make public information about the direct experiences or submit that information to an external authority.
• Therefore, a high trust on the central institution managing reputation is essential.
Subjective vs Global

• **Subjective**
  • The reputation is maintained by each agent and is calculated according to its own direct experiences, information from its contacts, its social relations...

  **Advantages:**
  • Reputation values can be calculated taking into account the current state of the agent and its individual particularities.

  **Disadvantages:**
  • The **models are more complex**, usually because they can use extra sources of information.
  • Each agent has to worry about **getting the information** to build reputation values.
  • **Less information is available** so the models have to be more accurate to avoid noise.
Direct Trust

- Trust relationship calculated directly from an agent’s outcomes database.
Witness reputation

- Reputation that an agent builds on another agent based on the beliefs gathered from society members (witnesses).

Problems of witness information:

- Can be false.
- Can be incomplete.
- It may suffer from the “correlated evidence” problem.
Credibility model

- Two methods are used to evaluate the credibility of witnesses:

  - Social relations (socialCr)
  - Past history (infoCr)

Credibility (witnessCr)
Credibility model

- **socialCr**(a,w,b): credibility that agent a assigns to agent w when w is giving information about b and considering the social structure among w, b and himself.
Credibility model

**Regret** uses fuzzy rules to calculate how the structure of social relations influences the credibility on the information.

\[
\begin{align*}
\text{IF } & \text{ coop}(w,b) \text{ is } h \\
\text{THEN } & \text{ socialCr}(a,w,b) \text{ is } vl
\end{align*}
\]
Neighbourhood reputation

- The trust on the agents that are in the “neighbourhood” of the target agent and their relation with it are the elements used to calculate what we call the Neighbourhood reputation.
System reputation

■ The idea behind the System reputation is to use the common knowledge about social groups and the role that the agent is playing in the society as a mechanism to assign reputation values to other agents.

■ The knowledge necessary to calculate a system reputation is usually inherited from the group or groups to which the agent belongs to.
If the agent has a reliable *direct trust* value, it will use that as a measure of trust. If that value is not so reliable then it will use reputation.
What do you mean by “properly”?

Current models

Planner

Decision mechanism

Comm

Trust & Reputation system

Inputs

Agent

Black box

Reactive
What do you mean by “properly”?

Current models

- Planner
- Decision mechanism
- Trust & Reputation system
- Inputs
- Value
- Comm

Black box
Reactive

Agent
What do you mean by “properly”?

The next generation?
What do you mean by “properly”?

The next generation?

Not only reactive...
... proactive