Introduction: Intelligent Agents

(Cognitive) Agents

- Software / hardware entities that display a certain degree of autonomy / taking initiative, are proactive/goal-directed
- Mostly described in terms of having 'mental states' ('strong' notion of agency) - 'cognitive' agents
- Display informational and motivational attitudes

Agent metaphor

- From an engineering perspective, the agent's metaphor (i.e. using agent concepts metaphorically) helps to design and construct complicated (distributed) systems!!
- Example: (multi) robotic systems
  - Cognitive robotics
Ideas behind agents

- Stemming from philosophy
  - Practical reasoning, reasoning about actions
  - Characterization of rational decision-making
  - Balancing desires and beliefs
  - Interplay between beliefs, desires, intentions (Bratman)
  - Intentional stance (Dennett)

Practical Reasoning

- (Bona fide) practical syllogism

  Exercise would be good for me.
  Jogging is exercise.

  Therefore, jogging would be good for me.

  • ’Just’ deductive reasoning

Practical Reasoning

- More interesting practical syllogism

  Would that I exercise.
  Jogging is exercise.

  Therefore, I shall go jogging

  • No deduction, rather specification of selection of action / decision of the agent

Dennett’s intentional stance

- Anthropomorphic instance of the design (functionality) stance, contra the physical stance
- Instrumental / operational use of beliefs and desires of human beings: no causally active inner states of people, just calculational devices

Bratman: the role of intentions

- Rational behavior needs, besides beliefs and desires, also intentions
- Two justifications for this:
  1. (Resource-bounded) agents need to settle on some desire(s) and commit themselves
  2. Co-ordination of future actions after commitment(s)

Bratman

- Intentions, unlike mere desires, play the following functional roles:
  1. Intentions normally pose problems for the agent; the agent needs to determine a way to achieve them
  2. focus on solving concrete problems
  3. Intentions provide a “screen of admissibility” for adopting other intentions
  4. Agents “track” the success of their attempts to achieve their intentions -- may give rise to replanning
Agent logics

- Logics for specifying intelligent/rational agents inspired by Bratman’s philosophy:
  - BDI logic
  - Cohen & Levesque
  - KARO logic
  - BDI model/architecture (Rao & Georgeff)

Agent logics

- Philosophical logic
- A formal treatment of intensional notions
- Various ‘flavours’:
  - Epistemic / doxastic
  - Temporal / dynamic (action logic)
  - Deontic
  - Combinations (BDI, KARO)

Cohen & Levesque

- Achievement goals
  \[ A\text{-GOAL} i[] = \text{GOAL} i(\text{LATER} []) \land \text{BEL} i \neg[] \]
- No deferral forever assumption
  \[ \vdash \neg(\text{GOAL} i(\text{LATER} [])) \]
- Agents eventually drop all achievement goals!

Cohen & Levesque

- Persistent goals
  \[ P\text{-GOAL} i[] = \text{GOAL} i(\text{LATER} []) \land \neg\text{BEL} i[] \]
  \[ \vdash \text{intend-to-do} \]

Cohen & Levesque

- Intention (‘intend-to-do’)
Rao & Georgeff: BDI theory
- "Rational agent possesses mental attitudes of beliefs, desires and intentions, representing the information, motivational, and deliberative states of an agent, respectively"
- "These mental attitudes determine the system’s behaviour and are critical for achieving adequate or optimal performance when deliberation is subject to resource bounds" --- computational perspective!

Rao & Georgeff’s BDI Logic
- Commitment strategies in BDI logic
  - INTEND(inevitable(\(\neg\)INTEND(\(\neg\)INTEND(\(\neg\)BEL(j)))))
  - "no infinite deferral"
  - INTEND(inevitable(\(\neg\)INTEND(\(\neg\)BEL(j)))))
    - inevitable(INTEND(\(\neg\)BEL(j))))
  - "blindly committed agent"

BDI Architecture

Agent design & programming
BDI architecture: ‘deliberation cycle’

Agent-oriented programming (Y. Shoham)

Agent-oriented programming

The language 3APL

Mental attitudes in 3APL

Mental attitudes in 3APL

3APL agent (original version)
3APL program

- A set of capabilities: basic actions:
  - e.g. gripper_up, pickup, move_left, move_right, sense
- An initial belief base: simple propositions:
  - e.g. block_on_table
- A set of initial plans: imperative programs:
  - e.g. gripper_up; pickup

3APL control architecture

- The control architecture implements the deliberation or (Sense)-Update-Act cycle:
  - Rule application phase (plan generation / updating):
  - Execution Phase (belief updating by plan execution)

3APL program (ctd)

- A set of plan revision rules: guarded clauses of the form $[] \cdot [\cdot]$, where
  - $[]$ is a plan,
  - $[\cdot]$ is a guard and
  - $[\cdot]$ is a (revised) plan
- e.g. gripper_up; pickup | no_block | find_block; gripper_up; pickup
- If the guard is implied by the agent's belief base
- The rule becomes applicable and may be applied.

3APL control loop (‘deliberation cycle’)

1. Find rules matching plans (= commitments)
2. Select rules from (1) matching the beliefs.
3. Select rule from (2) and fire it on plan base
4. Select plans that can be executed
5. Select one plan from (4) and execute it

3APL: extensions

- Extend language, e.g.
  - Declarative goals
  - Plan generation and goal revision rules
    - $[] \cdot [\cdot]$ (plan generation)
    - $[\cdot] \cdot [\cdot]$ (goal revision)
  - Communication primitives
  - Programmable control (deliberation) loop
  - Nested modalities, e.g. $B\diamond G$

Extended 3APL agent

- A complex mental state incorporating
  - Beliefs about the agent's environment
  - Plans, describing actions to achieve the goals
  - Goals, representing the states of affairs to be achieved
  - Set of mechanisms working on mental state
    - To execute plans (controlling the environment)
    - For decision-making or practical reasoning (plan revision, plan generation)
- A set of capabilities, i.e. basic actions
**Extended Deliberation Cycle**

1. Find Plan Generation Rules that Match Goals
2. Remove Plan Generation Rules with atoms in head that exist in Belief Base
3. Find Plan Generation (PG) Rules that Match Beliefs
4. Select a Plan Generation (PG) Rule to Apply
5. Apply the Plan Generation (PG) Rule, thus adding a plan to the planbase
6. Find Plan Revision (PR) Rules that Match Plans
7. Find Plan Revision (PR) Rules that Match Beliefs
8. Select a Plan Revision (PR) Rule to Apply to a Plan
9. Apply the Plan Revision (PR) Rule to the Plan
10. Find Plans To Execute
11. Select a Plan To Execute
12. Execute the (first basic action of the) Plan

---

**Formal Semantics**

- Designing a programming language
  - Define the constructs of the language
  - Define the semantics of a program in this language
  - Define "what happens" if the program is executed

- Formal semantics
  - Give meaning to programs formally and precisely
  - Many advantages
    - Problems become clear
    - Comparison with other languages
    - Basis for verification

---

**Towards program verification**

- Dynamic logic for 3APL
  - In PDL: \([\tau_1 \vdash \tau_2]_\phi \iff [\tau_1][\tau_2]_\phi\)
  - This is not a validity for 3APL plans
  - Dynamic logic for restricted plans
    - Sound and complete axiomatization
  - Extendable to logic for arbitrary plans
    - But... infinitary axiomatization
    - Not (yet) really a practical method for program verification

---

**Programming emotional agents**

- Main ideas:
  - Inspired by emotions of humans
  - Emotional states organize ready repertoires of action (Damasio)
  - Our perspective: emotions as a ‘designing tool’ for agents
Emotions in KARO and 3APL

Multi-Agent 3APL: communication

Heterogeneous MAS: Ontology negotiation

AO Methodology

OperA
Consequence of OperA

- Separation between the individual (agent) and collective (society) level
  - For the construction of individual agents you can use what you want, e.g. BDI model!!
  - Link between the individual and organisation via interaction structure, roles, norms and contracts!

From analysis via design to implementation

- Roles from analysis [] agent types in 3APL
- Agent type:
  - specification of deliberation process +
  - set of roles (characterized in terms of beliefs, goals, plans, capabilities, messages, PR/PG/GR rules)
- Norms may be implemented in various ways:
  - as goals
  - in social or interaction structure
    - obligations, protocols
  - in environment
    - norm enforcement

Extending 3APL with obligations? Towards BDI+ (eg BOID theory)

- One might also consider augmenting 3APL by rules dealing with obligations directly:
  - (goal generation by obligations)
  - (goal generation by obligations + desires)

- Are these sensible in practice?
Protocols in Normative Agent Systems

- Norms tend to be vague and ambiguous
- Protocols in (electronic) institutions:
  - Protocols can help agents follow procedure
  - Protocols can help obtain results
  - Protocols should be norm-compliant
  - Following the protocol does not violate any of the norms
- How to prove norm compliance of protocols?

Verifying Norm Compliance of Protocols

- Formal methods based on program verification
- Translate protocol to program, norms to LTL formulas
  - Intermediate states are important! use temp. logic
- Need to connect the abstract level to the concrete level
- Norm compliance is a safety/invariance property of the protocol
- Liveness check for checking effectiveness

Formal Aspects of Institutions and Organisations

- Development of logical frameworks for grounding the specification of multi-agent institutions and organisations
- Formal analysis of constitutive norms (counts-as)
- Formal analysis of the notion of social structure within groups of agents
- Use of modal logics and description logic

Applications: our projects

- ANITA "Administrative Normative Information Transaction Agents" (ANITA) with UM, RUG and UL
- STW project "Distributed Model-Based Diagnosis and Repair" with TUD, NLR and UM
- PhD project together with the company Emotional Brain in Almere and RUG on multi-agent expert systems
- BBK/IrG project "Adaptive Support Systems" with TNO
- Intelligent companions (with DECIS, RUG, Philips, Berchet)
- Ontologies for MAS (with Information Sciences & IBM)
- Agent programming for mobile devices (with Melbourne)
- Adaptive Support Systems (with TNO)
- AIBO Soccer (with DECIS, TUD, UvA)
- Virtual Characters in Games (with GIVE group and TNO)

(Multi) Robot Systems

- Traffic & transport
- Space robots
- Rescue robots
- Robot soccer
- Robot companions
- …


**NASA explorer robots**

**Autonomous vehicles**

Autonomous Unmanned Aerial Vehicle - Linköping

**Robot soccer**

**AIBO programming**

**Intelligent Robot Companions**

- Companions of human users
  - Personal assistants
    - PSA’s for ISS (NASA)
    - Intelligent user interface (Philips)
  - Playmates / Mentors
    - Toy robot (‘boon companion’, Berchet)

**Philips iCat**
**NASA’s Personal Satellite Assistant (PSA)**

- **‘Boon companion’ project**
  - **Aim:** devising an intelligent companion
    - toy robot companion (Berchet)
    - intelligent interface (iCat, Philips)
  - **UU part(s) in the project:**
    - Reasoning (‘deliberation’) module
      - Personal reasoning (BDI, emotions, perceptions)
      - Social reasoning (roles, norms, obligations, interaction)
    - Communication / dialogue module
  - Integrated use of cognitive / BDI model(s) (extended with emotions), agent programming, learning techniques, NLP

---

**Airport Traffic Plan Repair**

- **Mechanism Design for Airport Traffic Planning**
  - **Aim:** implementing multiagent techniques in the domain of airport traffic tactical planning.
    - In the last stage of planning, just before execution, various disrupting events can occur that disrupt the planning at an airport.
    - Determining the best way to solve these disruptions is a typical multiagent resource allocation problem, in the sense that it has to satisfy certain criteria: efficiency, fairness, incentive-compatibility.

---

**Multi-agent expert systems**

- **Motivation:**
  - Expert system for multi-disciplinary domains
    - Case: company Emotional Brain wants an expert system for group (dys)functioning diagnosis, based on diverse medical, psychological, sociological data and background knowledge
  - Idea: use a multi-agent system that can coordinate / negotiate the opinions of the various agents (experts) and make a decision

---

**Multi-agent decision-making**

- Formalism to prescribe the activity of making decisions by agents and activity of communication between agents.
  - Sets of rules define when agents may make decisions and when to utter communicative acts; in addition, these rules define how the agent’s cognitive state is updated afterwards.
  - Sets of rules make up either decision games or dialogue games.
Conclusion

Over the last decade we have been engaged in:

- Logic of agency/agents
- Agent-oriented programming
- Development of programming language 3APL
- Agent-oriented software engineering
- Applications
  - Ontologies for MAS (with IBM)
  - Multi-agent expert systems (with EB)
  - Airport Traffic Planning (with NLR, TUD and UM)
  - Adaptive Support Systems (with TNO)
  - Virtual Characters in Games (with GIVE group and TNO)
  - Intelligent companions (with DECIS, Philips, Berchet)
  - ...

Credits: the UU-IS Group

Frank Dignum
Frank de Boer
Jan Broersen
Martin Caminada
Mehdi Dastani
Henry Prakken
Gerard Vreeswijk
Marco Wiering
Hub Aldewereld
Susan van den Braak
Jurriaan van Diggelen
Davide Grossi
Geert Jonker
Fernando Koch
Henk Jan Lebbink
Cees Pierik
Birna van Riemsdijk

Software & More Information

http://www.cs.uu.nl/3apl/

Thank you for your attention!
(And for your support during the last 10 years!!)