EVROPSKÁ UNIE

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

OP Vzdělávání pro konkurenceschopnost

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Introduction to Programming Autonomous Agents and Multi-agent Systems (short tutorial)

Peter Novák

Agent Technology Center, Department of Cybernetics

Czech Technical University in Prague

22$^{\text{nd}}$ March 2010

Tato prezentace je spolufinancována Evropským sociálním fondem a státním rozpočtem České republiky.

Agenda

Motivation & Scope

How to program intelligent agents?

Theoretical foundations

Agent oriented programming languages

Research outlook

Conclusion

Motivation & Scope

Definition 1 (intelligent agents). are artificial entities assumed to be *autonomous*, *proactive*, *reactive*, as well as *socially able*.

- **autonomy:** the agent acts without its user's intervention, *agent is not an object* (in OOP sense)
- **proactiveness:** the agent acts *purposefully*, i.e., towards reaching goals
- **reactivity:** the agent is responsice to changes of the environment
- **social ability:** it is capable to communicate, coordinate and cooperate with its users and peers

Inteligent agents

Definition 2 (intelligent agents)**.** are artificial entities assumed to be *autonomous*, *proactive*, *reactive*, as well as *socially able*.

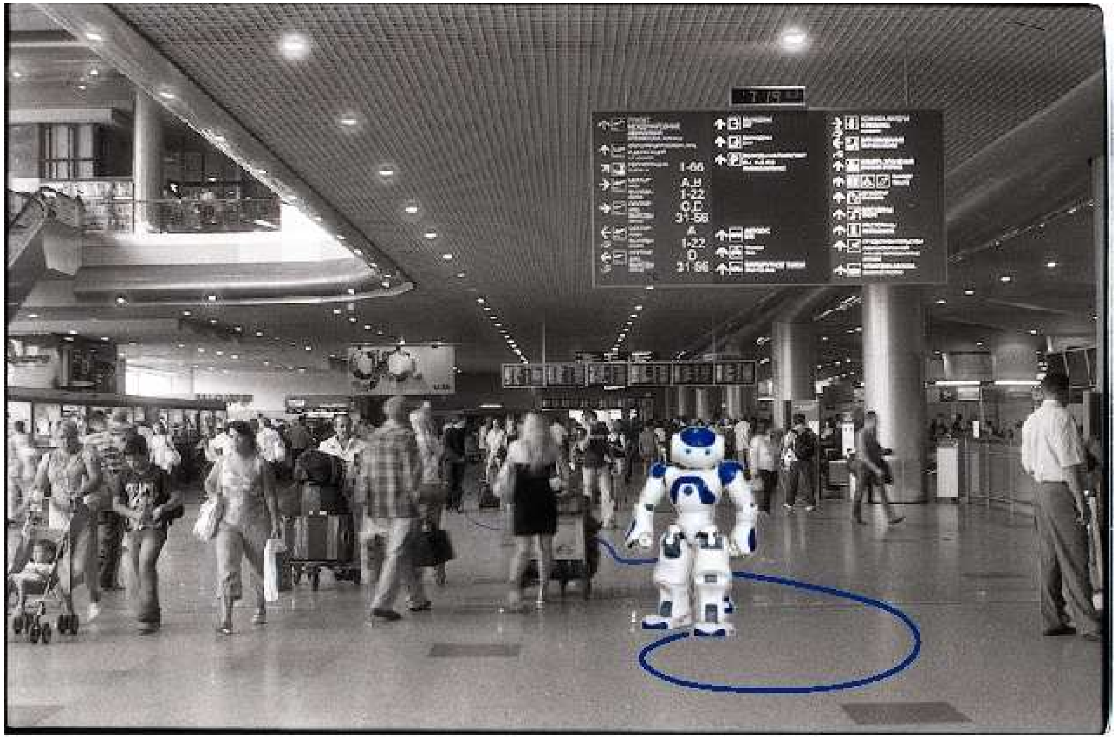
- **autonomy:** the agent acts without its user's intervention, *agent is not an object* (in OOP sense)
- **proactiveness:** the agent acts *purposefully*, i.e., towards reaching goals
- **reactivity:** the agent is responsice to changes of the environment
- **social ability:** it is capable to communicate, coordinate and cooperate with its users and peers

communication $\rightsquigarrow$ symbolic information processing!

Definition 3 (cognitive/knowledge intensive agent)**.** employ cognitive processes, such as knowledge representation and reasoning as the basis for decision making and action selection. I.e., they construct and maintaining a *mental state*.

Definition 4 (mental state)**.** agent's representation of the environment, itself, its peers, etc.

Example: Airport E-Assistant (Ape)



The robot guides and advices passengers at the airport.

- navigates through crowds, maintains energy level
- solves and responds to requests along the way
- communicates with the airport infrastructure, etc...

How to program intelligent agents?

Agent program: action selection (+ state change)

$$next\_action = \mathcal{P}(\ perceptions \bigoplus \sigma\ )$$

$$(\sigma' = \sigma \oslash perceptions)$$

reactive systems: $\sigma \rightarrow \emptyset$

− no/little past, difficult to implement *proactiveness*
+ responsive, robust w.r.t. unexpected = *graceful degradation*

deliberative systems: $perceptions \rightarrow \emptyset$

− weak responsiveness to the environment
+ keeps context (future) = *proactiveness*, planning

environment dynamics, graceful degradation $\rightsquigarrow$ **reactivity**

goals, social abilities $\rightsquigarrow$ **deliberation/planning**

How to marry reactivity & deliberative features?

environment dynamics, graceful degradation ⇝ **reactivity**

goals, social abilities ⇝ **deliberation/planning**

How to marry reactivity & deliberative features?

Hybrid architectures!

procedure AGENT−CYCLE
SENSE
↓ ⟵ PLAN
ACT

Architectures: – *incomplete* –

1987: PRS	(Georgeff and Lansky)
1988: IRMA	(Bratman, Israel and Pollack)
1991: Abstract BDI architecture	(Rao and Georgeff)
1994: INTERRAP	(Müller and Pischel)

Languages: – *incomplete* –

1990: AGENT-0	(Shoham)
1996: AgentSpeak(L)	(Rao)
1996: Golog	(Reiter, Levesque, Lesperance)
1997: 3APL	(Hindriks et al.)
1998: ConGolog	(Giacomo, Levesque, Lesperance)
2000: JACK	(Busetta et al.)
2000: GOAL	(Hindriks et al.)
2002: Jason	(Bordini, Hubner)
2003: Jadex	(Braubach, Pokahr et al.)
2008: Behavioural State Machines/Jazzyk	(Novák)
2008: 2APL	(Dastani)

Theoretical foundations

- **(B)eliefs:** reflect agent's *static* beliefs about its environment, itself, its peers, etc. (*now*)
- **(D)esires:** descriptions of situations the agent wants to bring about (*future*)

Dynamics:

- **automated planning:** provides the connection from *now* to the *future*

Agent's *intentions* are considered to be reducible to (B) and (D).

Cohen & Levesque: *Intention is a choice with commitment*. 1990

- **(B)eliefs:** reflect agent's *static* beliefs about its environment, itself, its peers, etc. (*now*)
- **(D)esires:** descriptions of situations the agent wants to bring about (*future*)
- **(I)ntentions:** *partial plans of action that the agent is committed to execute in order to fulfill the goals*

Belief-Desire-Intention architecture

- **(B)eliefs:** reflect agent's *static* beliefs about its environment, itself, its peers, etc. (*now*)
- **(D)esires:** descriptions of situations the agent wants to bring about (*future*)
- **(I)ntentions:** *partial plans of action that the agent is committed to execute in order to fulfill the goals*

Explicit plans

(*recipes* for how to proceed from now to the future)

$\downarrow$

means-end reasoning $\rightsquigarrow$ **reactive planning**

- planning is not a must!
- partial plans can be encoded in the design time

Bratman: *Intentions, Plans, and Practical Reason.* 1987

Rao & Georgeff: *Modeling Rational Agents within a BDI-Architecture.* 1991

(generalized) computation tree logic:

- branching time
- temporal modalities
- free mixing of temporal operators and quantifiers

$$\theta \quad ::= \quad \mathsf{p} \mid \neg\theta \mid \theta \wedge \theta$$

$$\varphi \quad ::= \quad \theta \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bigcirc\varphi \mid \Diamond\varphi \mid \Box\varphi \mid \varphi\mathcal{U}\varphi \mid \forall\varphi \mid \exists\varphi$$

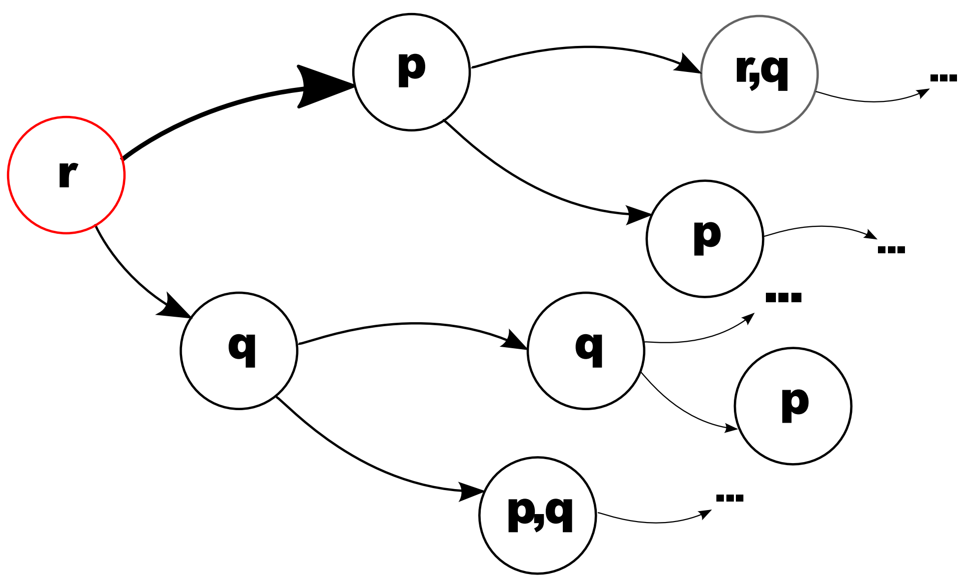
CTL*: the language of the discourse

(generalized) computation tree logic:

- branching time
- temporal modalities
- free mixing of temporal operators and quantifiers

$$\theta ::= \mathsf{p} \mid \neg\theta \mid \theta \wedge \theta$$

$$\varphi ::= \theta \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bigcirc\varphi \mid \Diamond\varphi \mid \Box\varphi \mid \varphi\mathcal{U}\varphi \mid \forall\varphi \mid \exists\varphi$$



r

$\exists \bigcirc p$

$\forall \Diamond p$

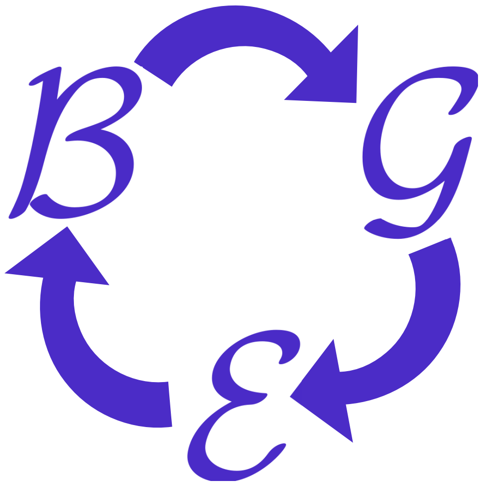
$\exists \Box p$

$\neg \bigcirc \exists \Box p$

$\neg \bigcirc \forall \Diamond r$

I-System (cont.)

AI1:	$\text{GOAL}(\alpha) \Rightarrow \text{BEL}(\alpha)$	(B-G compatibility)
AI2:	$\text{INTEND}(\alpha) \Rightarrow \text{GOAL}(\alpha)$	(G-I compatibility)
AI3:	$\text{INTEND}(does(\alpha)) \Rightarrow does(\alpha)$	(intentions lead to actions)
AI4:	$\text{INTEND}(\phi) \Rightarrow \text{BEL}(\text{INTEND}(\phi))$	(believing in intentions)
AI5:	$\text{GOAL}(\phi) \Rightarrow \text{BEL}(\text{GOAL}(\phi))$	(believing in goals)
AI6:	$\text{INTEND}(\phi) \Rightarrow \text{GOAL}(\text{INTEND}(\phi))$	(desiring intentions)
AI7:	$done(e) \Rightarrow \text{BEL}(done(e))$	(awareness of actions)
AI8:	$\text{INTEND}(\phi) \Rightarrow \forall\Diamond(\neg\text{INTEND}(\phi))$	(no infinite deferral)



blind commitment: the agent maintains intentions until it actually believes that they are achieved

$$\text{INTEND}(\forall\Diamond\phi) \Rightarrow \forall\,(\text{INTEND}(\forall\Diamond\phi)\,\mathcal{U}\,\text{BEL}(\phi))$$

$\leadsto$ I want ϕ! (no care possibility to reach ϕ)

single-minded commitment: the agent maintains intentions as long as as it believes they are still achievable

$$\text{INTEND}(\forall\Diamond\phi) \Rightarrow \forall\,(\text{INTEND}(\forall\Diamond\phi)\,\mathcal{U}\,(\text{BEL}(\phi) \vee \neg\text{BEL}(\exists\Diamond\phi)))$$

$\leadsto$ I want ϕ if it is possible! (regardless of its desires/needs)

open-minded commitment: the agent maintains intentions as long as they are still its goals

$$\text{INTEND}(\forall\Diamond\phi) \Rightarrow \forall\,(\text{INTEND}(\forall\Diamond\phi)\,\mathcal{U}\,(\text{BEL}(\phi)\vee\neg\text{GOAL}(\exists\Diamond\phi)))$$

$\leadsto$ I want ϕ if it makes sense at all.

structural decomposition (knowledge bases):

belief base

- KRR technology $\rightsquigarrow$ First-Order Logic

goal base

- mutual compatibility of goals?
- declarative, procedural, maintenance
- KRR technology $\rightsquigarrow$ set of ground FOL terms

intentions

1. *plan library:* storage of partial plan prescriptions
2. *intention stack:* partial plan instances

agent system dynamics:

1. library plan instantiation
2. gradual plan decomposition & execution

Abstract BDI reasoning cycle


```
initialize−state
loop
    options := generate−options(event−queue)
    selected−options := deliberate(options)
    update−intentions(selected−options)
    execute()
    get−new−external−event()
    drop−successful−attitudes()
    drop−impossible−attitudes()
end loop
```

Rao & Georgeff. BDI Agents: From Theory to Practice, 1995

Agent oriented programming languages

What is an APL?

- operationalization of a BDI architecture
- provides an underlying architecture & *tools for encoding the system dynamics*
- syntax & model of execution (semantics)

Why APLs?

$\leadsto$ software engineering $\leadsto$ systems modeling

$\leadsto$ clear properties (expressivity, verification, etc.)

The landscape

BDI programming systems

Theoretically oriented

- − declarative languages built from scratch ⇝ new syntax
- + clear theoretical properties ⇝ verification
- + declarative KR techniques
- − no integration with external/legacy systems

AgentSpeak(L), 3APL, 2APL, GOAL, CAN, etc.

Engineering approaches

- + layer of specialised language constructs over a robust mainstream programming language (Java) ⇝ code re-usability
- − host language semantics
- − KR in an imperative language
- + easy integration with external systems and environments

JACK, Jadex

- programming language for BDI agents
- notation based on logic programming
- AgentSpeak(L) $\rightsquigarrow$ an abstract programming language
- Jason $\rightsquigarrow$ operational semantics for AgentSpeak
- various pragmatic extensions like external actions (Java)
- (platform for developing multi-agent systems)

- beliefs are FOL terms
- agent program $\rightsquigarrow$ set of rules

  triggering_event : context←body.
- **events**: $+b$, $-b$, $+!g$, $-!g$, $+?g$, $-?g$ $\rightsquigarrow$ implicit goals!
- **context**: logical formula $(\wedge, \vee, \neg)$
- **body**: sequence of actions and subgoals to achieve

developed by Jomi F. Hübner and Rafael H. Bordini

Reasoning cycle

1. Perceiving the Environment
2. Updating the Belief Base
3. Receiving Communication from Other Agents
4. Selecting 'Socially Acceptable' Messages
5. Selecting an Event
6. Retrieving all Relevant Plans
7. Determining the Applicable Plans
8. Selecting one Applicable Plan
9. Selecting an Intention for Further Execution
10. Executing one step of an Intention

```
+green_patch(Rock)
    : not battery_charge(low)
    ← ?location(Rock,Coordinates);
      !at(Coordinates);
      !examine(Rock).

+!at(Coords)
  : not at(Coords) & safe_path(Coords)
  ← move_towards(Coords); !at(Coords).

+!at(Coords) ...
```

Jadex

- programming language for BDI agents
- Java + XML notation $\rightsquigarrow$ towards Agent-Oriented Software Engineering (AOSE)
- mixes object-orientation and BDI concepts
- explicit goals $\rightsquigarrow$ reasoning about (manipulation of) goals
- independent on middleware (adapters for integration with JADE and DIET)

- **beliefs**: Java objects (sets)
- **goals**: explicit (XML) $\rightsquigarrow$ perform, achieve, query maintain
- **events** $\rightsquigarrow$ the central element
- **plans:** represent procedural knowledge $\rightsquigarrow$ head, context (both XML), body (Java)
- capabilities - modularity

Reasoning cycle

(central point = event list)

1. Communication
   (a) select message ⇝ update the event list
2. Dispatcher
   (b) select event (possibly an incomming message)
   (c) find applicable candidate plans/capabilities
3. Scheduler
   (d) select intention
   (e) execute one intention step ⇝ event list update!

Jadex (example)


```
<agent name="SampleAgent" package="jadex.examples.SampleAgent" ... >
<!-- ... imports ... -->
<beliefs>
    <belief name="rockExperiment" class="Experiment">
        <fact>new Experiment("rock")</fact>
    </belief>
</beliefs>
<goals>
    <achievegoal name="examineRock">
        <parameter name="position" class="GPS"/>
        <targetcondition>$beliefbase.rockExperiment.isFinished()</targetcond
    </achievegoal>
</goals>
<plans>
    <plan name="moveToTheRock">
        <trigger><goal ref="examineRock"/></trigger>
        <body>new MotionPlan($event.goal.position)</body>
    </plan>
</plans>
</agent>
```

Research outlook

State of the art & on-going research

- active developments in the APL arena
  - $\rightsquigarrow$ 2APL, GOAL, Jadex, JACK, Jazzyk, ...
- **goals** $\rightsquigarrow$ semantics & dynamics (Dastani, van Riemsdijk, Sardina)
- **planning** $\rightsquigarrow$ integration/relationships
  - lookahead (Hindriks, Padgham)
  - AgentSpeak + planning (Menneguzi)
  - relationship to HTN (de Silva, Padgham, Sardina)
- **model checking** and **verification** (Bordini, Fisher)
- **software engineering** (...)
  - modularity, debugging

Conclusion

Summary: Agent-oriented programming languages

1. intelligent agents $\rightsquigarrow$ *social* abilities $\rightsquigarrow$ symbolic KRR
2. computational model: reactivity vs. deliberation
3. theoretical foundations = BDI + KRR + dynamics
4. I-system = model of rationality
5. BDI operationalization & software engineering $\rightsquigarrow$ BDI APL
6. Jason
7. Jadex
8. State of the art + on-going threads

Thank you for your attention.