Multi-Agent Systems

Sept. 2000, Bahia Blanca

University Nacional del Sur

- **Last two weeks in September**.
- **Tentative Dates:** Tuesday, Sept. 19th, Thursday, Sept. 21st, Friday, Sept. 22nd, Tuesday, Sept. 26th, Thursday, Sept. 28th, Friday, Sept. 29th.
- **Time:** From 4–6 pm, unless otherwise indicated.
- Lecture Course is on theoretical issues, emphasis on mathematical-logical foundations.

Overview

1. Introduction, Terminology
2. Three Basic Architectures
3. Logic Based Architectures
4. Distributed Decision Making
5. Contract Nets, Coalition Formation

Chapter 1. Introduction, Terminology

1.1 General

1.2 Intelligent Agents

1.3 Mathematical Description

1 Introduction, Terminology

1.1 General

This lecture course is mainly based on

> **Multi-Agent Systems** (Gerhard Weiss), MIT Press, June 1999.

We describe **general methods** and **techniques**.

Three Important Questions

(Q1) What is an **Agent**?

(Q2) If some program *P* is not an agent, how can it be **transformed into an agent**?

(Q3) If (Q1) is clear, what kind of **Software Infrastructure** is needed for the interaction of agents? What services are necessary?

Definition 1.1 (Distributed Artificial Intelligence (DAI))

The area investigating systems, in which several autonomous acting entities work together to reach a given goal.

*The entities are called **Agents**, the area **Multiagent Systems**.*

Example: Robocup (simulation league, middle league)

Why do we need them?

Information systems are **distributed**, **open**, **heterogenous**.
We therefore need **intelligent, interactive agents**, that **act autonomously**.

Agent: Programs that are implemented on a platform and have sensors to react to the environment.

Intelligent: Performance measures, to reach goals. **Rational** vs. **omniscient**, **decision making**

Interactive: with other agents (or humans) by observing the environment. **Coordination:** Cooperation vs. Competition

MAS versus Classical DAI

(MAS) **Several Agents coordinate their knowledge and actions (semantics describes this).**

(DAI) **Particular problem is divided into smaller problems (nodes). These nodes have common knowledge. The solution method is given.**

Today DAI is often used synonymous with MAS: (1) as well as (2).

AI	**DAI**
Agent	**Multiple** Agents
Intelligence: Property of a **single** Agent	Intelligence: Property of **several** Agents
Cognitive Processes of a **single** Agent	**Social** Processes of **several** Agents

10 Desiderata

1. **Agents are for everyone!** We need a method to agentize given programs.
2. Take into account that **Data is stored in a wide variety of data structures, and data is manipulated by an existing corpus of algorithms.**
3. A theory of agents must *not* depend upon the set of actions that the agent performs. Rather, **the set of actions that the agent performs must be a *parameter* that is taken into account in the semantics.**

4. **Every agent should execute actions based on some *clearly articulated* decision policy.** A **declarative** framework for articulating decision policies of agents is imperative.

5. Any agent construction framework must allow agents to perform the following types of reasoning:
   - **Reasoning about its beliefs** about other agents.
   - **Reasoning about uncertainty** in its beliefs about the world and about its beliefs about other agents.
   - **Reasoning about time**.

   These capabilities should be viewed as *extensions* to a core agent action language.

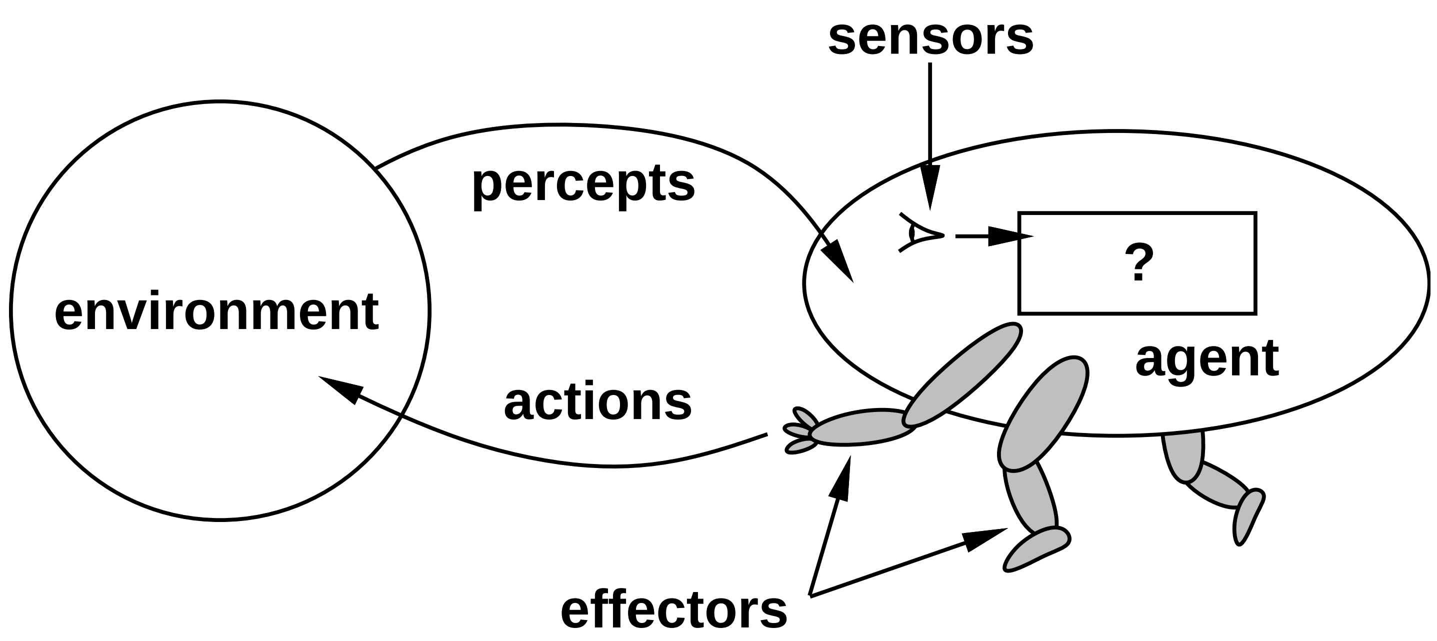
6. **Any infrastructure to support multiagent interactions *must* provide security.**
7. While the efficiency of the code underlying a software agent cannot be guaranteed (as it will vary from one application to another), **guarantees are needed that provide information on the performance of an agent relative to an oracle that supports calls to underlying software code.**

8. **We must identify efficiently computable *fragments* of the general hierarchy of languages alluded to above**, and our implementations must take advantage of the specific structure of such language fragments.
9. **A critical point is *reliability***—there is no point in a highly efficient implementation, if all agents deployed in the implementation come to a grinding halt when the agent “infrastructure” crashes.
10. The only way of testing the applicability of any theory is to **build a software system based on the theory**, to deploy a set of applications based on the theory, and to report on experiments based on those applications.

1.2 Intelligent Agents

Definition 1.2 (Agent)
An agent is a computer system that acts in its environment and executes autonomous actions to reach certain goals.

Learning, **Intelligence**. Environment is non-deterministic.



Definition 1.3 (Rational, Omniscient Agent)

Rational *Agents are those, that always do* **the right thing***.*
(A performance measure is needed).)

Omniscient *agents are agents, that know the results of their actions in advance.*

Rational agents are in general not omniscient!

Aphorism of Karl Kraus: In case of doubt, just choose the right thing.

How is the **right thing** defined and from what does it depend?

1. **Performance measure** (as objective as possible),
2. **Percept sequence**: what has been observed,
3. **Knowledge of the agent** about the environment,
4. **how** the agent **can act**.

An ideal rational agent chooses for each percept sequence exactly the **action**, *that maximizes its* **performance measure** *(given knowledge about the environment).*

Agents can be described mathematically by a function

$$\boxed{\text{Set of percept sequences} \mapsto \text{Set of Actions.}}$$

The internal structure of an agent is

Agent = Architecture + Program

Agents and their PAGE description:

Agent Type	Percepts	Actions	Goals	Environment
Medical diagnosis system	Symptoms, findings, patient's answers	Questions, tests, treatments	Healthy patient, minimize costs	Patient, hospital
Satellite image analysis system	Pixels of varying intensity, color	Print a categorization of scene	Correct categorization	Images from orbiting satellite
Part-picking robot	Pixels of varying intensity	Pick up parts and sort into bins	Place parts in correct bins	Conveyor belt with parts
Refinery controller	Temperature, pressure readings	Open, close valves; adjust temperature	Maximize purity, yield, safety	Refinery
Interactive English tutor	Typed words	Print exercises, suggestions, corrections	Maximize student's score on test	Set of students

Question:

How do properties of the environment influence the design of an agent?

Definition 1.5 (Properties of the Environment)

Accessible/Inaccessible: *If not completely accessible, one needs internal states.*

Determinist./Indeterm.: *If inaccessible the environment might seem indeterministic, even if it is not.*

Episodic/Nonepisodic: *Percept-Action-Sequences are independent from each other. Closed episodes.*

Static/Dynamic: *Dynamic: while the agent is thinking, the world is changing. Semi-dynamic: The world does not change, but the performance measure.*

Discrete/Continous: *concerning the set of observations and actions.*

Example for semi-dynamic: playing chess with a clock.

Environment	Accessible	Deterministic	Episodic	Static	Discrete
Chess with a clock	Yes	Yes	No	Semi	Yes
Chess without a clock	Yes	Yes	No	Yes	Yes
Poker	No	No	No	Yes	Yes
Backgammon	Yes	No	No	Yes	Yes
Taxi driving	No	No	No	No	No
Medical diagnosis system	No	No	No	No	No
Image-analysis system	Yes	Yes	Yes	Semi	No
Part-picking robot	No	No	Yes	No	No
Refinery controller	No	No	No	No	No
Interactive English tutor	No	No	No	No	Yes

xbiff and **software demons** are agents. But certainly not intelligent.

Definition 1.6 (Intelligent Agents)
An intelligent agent is an agent with the following properties:

1. **Reactive**: *Reaction to changes in the environment at certain times to reach its goals.*
2. **Pro-active**: *Taking the initiative, goal-directed behaviour.*
3. **Social**: *Interaction with others to reach the goals.*

Pro-active alone is not sufficient (C-Programs): the environment can change during execution.

Difficulty: Right balance between pro-active and reactive!

Agents vs. Object Orientation

Objects have a

1. **state** (encapsulated): control over internal state,
2. message passing capabilities.

Java: private and public methods.

- Objects have control over their state, but **not over their behaviour**.
- An object can **not prevent others to use** its public methods.

Agents: They call other agents and request them to execute actions.

- Objects do it for free, agents do it for money.
- No analoga to **reactive**, **pro-active**, **social** in OO.
- **MAS are multi-threaded**: each agent has a control thread.
  In OO only the sytem as a whole posesses one.

1.3 Mathematical Description

Definition 1.7 (Actions A, Percepts P, States S)
Let $A := \{a_1, a_2, \ldots, a_n, \ldots\}$, *the set of* actions, *and* $P := \{p_1, p_2, \ldots, p_n, \ldots\}$ *the set of* observations, *or* percepts *of an agent. Let* $S := \{s_1, s_2, \ldots, s_n, \ldots\}$ *the set of* states, *with which the environment is described.*

What does an agent observe, in a certain state s? We describe this with a function

$$\boxed{see : S \longrightarrow P.}$$

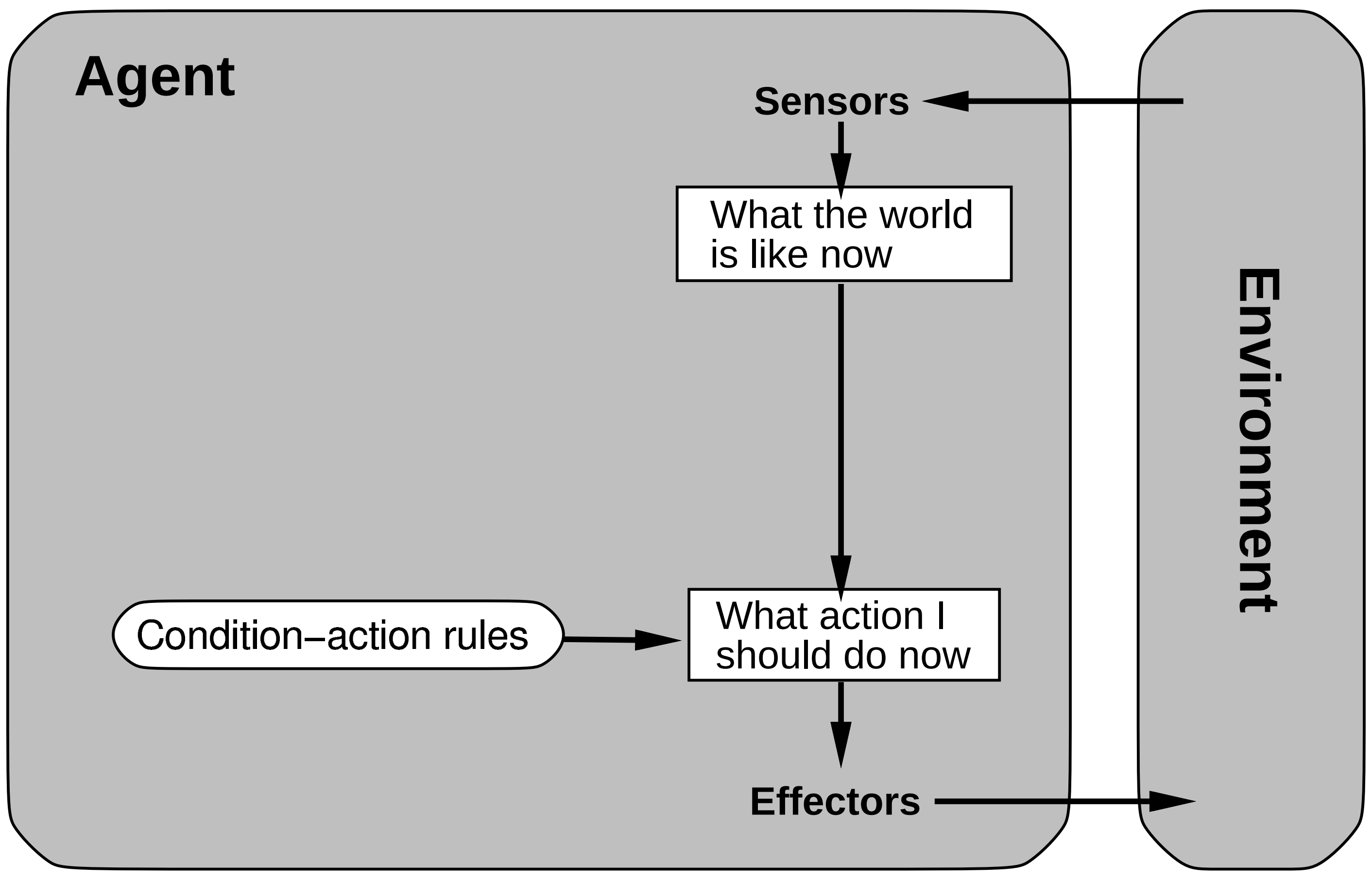
How does the environment develop (the state s) when an action a is executed? We describe this via a function

$$\boxed{env : S \times A \longrightarrow 2^S,}$$

this includes **indeterministic** environments.

How do we describe agents. We could take a function

$$\boxed{\mathit{action} : \mathbf{P} \longrightarrow \mathbf{A}.}$$



This is too weak! Better take the whole history into account

$$\boxed{h : s_0 \longrightarrow_{a_0} s_1 \longrightarrow_{a_1} \ldots s_n \longrightarrow_{a_n} \ldots}$$

(or the sequence of observations).

Definition 1.8 (Characteristic Behaviour)
The characteristic behaviour of an agent $\mathfrak{action}$ *in an environment* $\mathbf{env}$ *is the set* $\mathbf{Hist}$ *of all histories* $\mathbf{h} : s_0 \rightarrow_{a_0} s_1 \rightarrow_{a_1} \ldots s_n \rightarrow_{a_n} \ldots$ *with:*

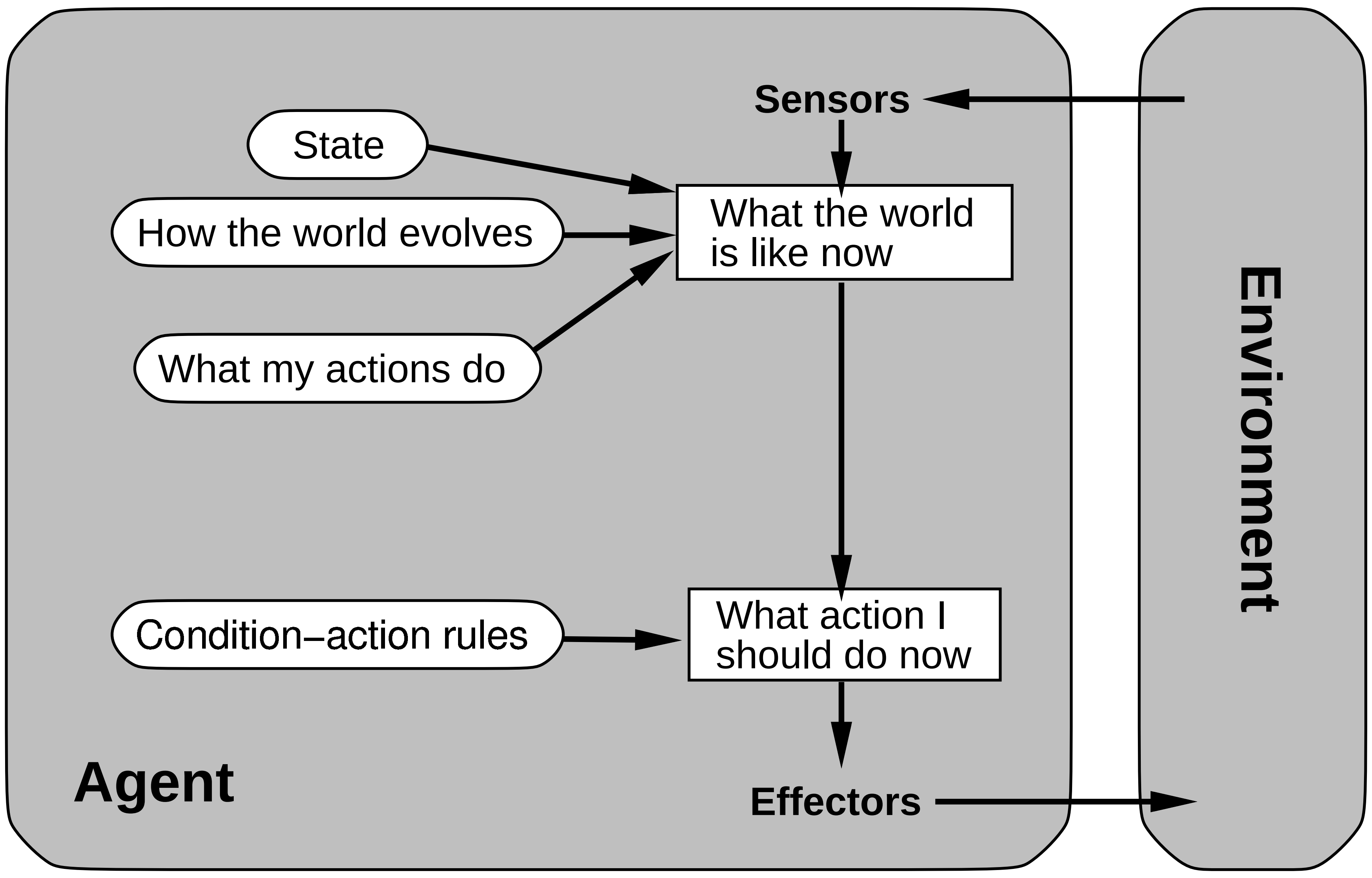
1. *for all n:* $a_n = \mathfrak{action}(\langle s_1, \ldots, s_n \rangle)$,
2. *for all n:* $s_n = \mathbf{env}(s_{n-1}, a_{n-1})$.

Definition 1.9 (Standard Agent action)
A standard agent action is given by a function

$$\boxed{\mathbf{action} : \mathbf{P}^* \longrightarrow \mathbf{A}}$$

together with $\mathbf{see} : \mathbf{S} \longrightarrow \mathbf{P}$ *and* $\mathbf{env} : \mathbf{S} \times \mathbf{A} \longrightarrow 2^{\mathbf{S}}$.

Instead of using the whole history, resp. $\mathbf{P}^*$, one can also use **internal states** $\mathbf{I} := \{\mathbf{i_1}, \mathbf{i_2}, \ldots \mathbf{i_n}, \ldots\}$.


Sensors
State
How the world evolves
What the world is like now
What my actions do
Environment
Condition–action rules
What action I should do now
Agent
Effectors

Definition 1.10 (State-based Agent $\mathfrak{action}$)
A state-based agent $\mathfrak{action}$ *is given by a function*

$$\boxed{\mathfrak{action} : \mathbf{I} \longrightarrow \mathbf{A}}$$

together with $\mathbf{see} : \mathbf{S} \longrightarrow \mathbf{P}$ *und* $\mathbf{next} : \mathbf{I} \times \mathbf{P} \longrightarrow \mathbf{I}$. *Here* $\mathbf{next}(\mathbf{i}, \mathbf{p})$ *is the succesor state of* $\mathbf{i}$ *if* $\mathbf{p}$ *is observed.*

Definition 1.11 (Characteristic Behaviour)
The characteristic behaviour of a state-based agent $\mathfrak{action}$ *in an environment* **env** *is the set of all sequences*

$$(\mathbf{i}_0, \mathbf{p_0}) \rightarrow_{a_0} (\mathbf{i}_1, \mathbf{p_1}) \rightarrow_{a_1} \ldots \rightarrow_{a_n} (\mathbf{i}_n, \mathbf{p_n}), \ldots$$

with

1. *for all n:* $\mathbf{a_n} = \mathfrak{action}(\mathbf{i_n})$,
2. *for alle n:* $\mathbf{next}(\mathbf{i_n}, \mathbf{p_n}) = \mathbf{i_{n+1}}$,

Lemma 1.1 (Equivalence)
Standard and state-based agents are equivalent wrt. their characteristic behaviour.

References

Arisha, K., F. Ozcan, R. Ross, V. S. Subrahmanian, T. Eiter, and S. Kraus (1999, March/April). IMPACT: A Platform for Collaborating Agents. *IEEE Intelligent Systems 14*, 64–72.

Bratman, M., D. Israel, and M. Pollack (1988). Plans and Resource-Bounded Practical Reasoning. *Computational Intelligence 4*(4), 349–355.

Dix, J., S. Kraus, and V. Subrahmanian (2001). Temporal agent reasoning. *Artificial Intelligence to appear*.

Dix, J., M. Nanni, and V. S. Subrahmanian (2000). Probabilistic agent reasoning. *Transactions of Computational Logic 1*(2).

Dix, J., V. S. Subrahmanian, and G. Pick (2000). Meta Agent Programs. *Journal of Logic Programming 46*(1-2), 1–60.

Eiter, T., V. Subrahmanian, and G. Pick (1999). Heterogeneous Active Agents, I: Semantics. *Artificial Intelligence 108*(1-2), 179–255.

Eiter, T., V. Subrahmanian, and T. J. Rogers (2000). Heterogeneous Active Agents, III: Polynomially Implementable Agents. *Artificial Intelligence 117*(1), 107–167.

Eiter, T. and V. S. Subrahmanian (1999). Heterogeneous Active Agents, II: Algorithms and Complexity. *Artificial Intelligence 108*(1-2), 257–307.

Georgeff, M. and A. Lansky (1987). Reactive Reasoning and Planning. In *Proceedings of the Conference of the American Association of Artificial Intelligence*, Seattle, WA, pp. 677–682.

Rao, A. S. (1995). Decision Procedures for Propositional Linear-Time Belief-Desire-Intention Logics. In M. Wooldridge, J. Müller, and M. Tambe (Eds.), *Intelligent Agents II – Proceedings of the 1995 Workshop on Agent Theories, Architectures and Languages (ATAL-95)*, Volume 890 of *LNAI*, pp. 1–39. Berlin, Germany: Springer-Verlag.

Rao, A. S. and M. Georgeff (1991). Modeling Rational Agents within a BDI-Architecture. In J. F. Allen, R. Fikes, and E. Sandewall (Eds.), *Proceedings of the International Conference on Knowledge Representation and Reasoning*, Cambridge, MA, pp. 473–484. Morgan Kaufmann.

Rao, A. S. and M. Georgeff (1995, June). Formal models and decision procedures for multi-agent systems. Technical Report 61, Australian Artificial Intelligence Institute, Melbourne.

Subrahmanian, V., P. Bonatti, J. Dix, T. Eiter, S. Kraus, F. Özcan, and R. Ross (2000). *Heterogenous Active Agents*. MIT-Press.

Weiss, G. (Ed.) (1999). *Multiagent Systems*. MIT-Press.