Combining Agents, ASP and Planning

NICTA, Jul-Aug 2003

- **July and August with the exception of third week in July**.
- **Time:** Thursday, Friday, 14-16, starting on 3rd July 2003.
- Lecture Course is in the **first 3 weeks on theoretical issues in general agent systems and answer set programming**, emphasis on mathematical-logical foundations. **Remaining two weeks devoted to a particular agent system** and some demonstrations.
- `www.cs.man.ac.uk/~jdix/LECTURING/NICTA03.html`.

First and second week (Chapters 1–3)

The first part of this lecture course is mainly based on

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

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

Third week (Chapter 4)

The second part of this lecture course is mainly based on

1. **Knowledge representation, reasoning and declarative problem solving with Answer sets**
   (Chitta Baral), MIT Press, February 2003.
2. **Planning in Answer Set Programming using Ordered Task Decomposition**
   (Jürgen Dix, Ugur Kuter and Dana Nau)
   Theory and Practice of Logic Programming, to appear 2004.
   `<www.cs.umd.edu/users/ukuter/ASP_Planning/>`

We give an introduction to the newly emerged paradigm of **Answer Set Programming** and illustrate it with recent research on how to realise HTN-planning in this paradigm.

Fourth and fifth week (Chapters 5–9)

The third part of this lecture course is mainly based on

> **Heterogenous Agent Systems**
> (Subrahmanian/Bonatti/Dix/Eiter/Kraus/Özcan/Ross)
> MIT Press, August 2000.

We describe the **IMPACT approach** and its **underlying foundations**. We also give two demos and present an approach of monitoring agents through planning (using an ASP engine).

Overview (Agent Systems in general)

1. Introduction
2. Distributed Decision Making (2 Lectures)
3. Contract Nets, Coalition Formation

Overview (Answer Set Programming)

4. ASP: Foundations and an Application to Planning (2 Lectures)

Overview (IMPACT)

5. *IMPACT* Architecture
6. Actions and Agent Programs
7. Implementing Agents: An Application
8. Agent Systems and Planning
9. Extensions of IMPACT

Chapter 1. Introduction

1.1 Motivation

1.2 Intelligent Agents

1.3 Formal Description

1.4 Reactive Agents

1.5 BDI-Architectures

1.6 Layered Architectures

1 Introduction

1.1 Motivation

Three Important Questions

(Q1) What is a (software) **agent**?
$\Rightarrow$ (Franklin and Graesser 1997; Wooldridge and Jennings 1995) and references therein)

(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, where 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**.

(Software) Agent: Programs that are implemented on a platform and have "sensors" and "effectors" to read from and make changes to the environment, respectively.

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.

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 agentise 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.

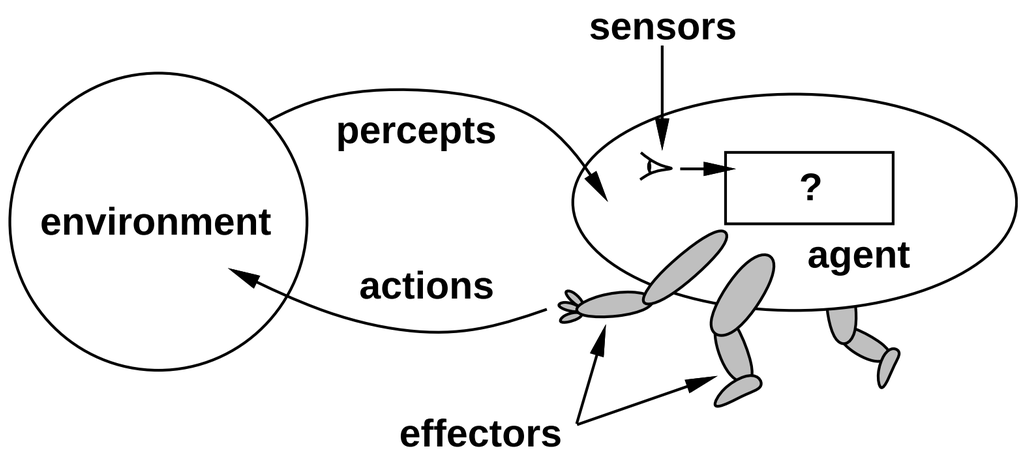
10. The only way of testing the applicability of any theory is to **build a software system based on the theory**, to deploy a set of applications based on the theory, and to report on experiments based on those applications.

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 those, 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**.

Definition 1.4 (Ideal Rational Agent)

An ideal rational agent chooses for each percept sequence exactly the **action** which maximises 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 can be described as

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

How do environment properties influence agent design?

Definition 1.5 (Properties of the Environment)

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

Deterministic/Indeterministc: An inaccessible environment might seem indeterministic, even if it is not.

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

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

Discrete/Continous: Density of observations and actions. Relevant: Level of granularity.

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,software demons are agents (not intelligent).

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

1. **Autonomous**: Operates without direct intervention of others, has some kind of control over its actions and internal state.
2. **Reactive**: Reaction to changes in the environment at certain times to reach its goals.
3. **Pro-active**: Taking the initiative, being goal-directed.
4. **Social**: Interaction with others to reach the goals.

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

Socialisation: Needs coordination, communication, and negotiation skills

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 system as a whole possesses one.

1.3 Formal Description

Definition 1.7 (Actions A, Percepts P, States S)

$\mathbf{A} := \{\mathbf{a_1}, \mathbf{a_2}, \ldots, \mathbf{a_n}, \ldots\}$ is the set of *actions*.
$\mathbf{P} := \{\mathbf{p_1}, \mathbf{p_2}, \ldots, \mathbf{p_n}, \ldots\}$ is the set of *observations*, or *percepts*.
$\mathbf{S} := \{\mathbf{s_1}, \mathbf{s_2}, \ldots, \mathbf{s_n}, \ldots\}$ is the set of *states* of the environment.

What does an agent observe, in a certain state $\mathbf{s}$? We describe this with a function $\boxed{\mathbf{see} : \mathbf{S} \longrightarrow \mathbf{P}.}$

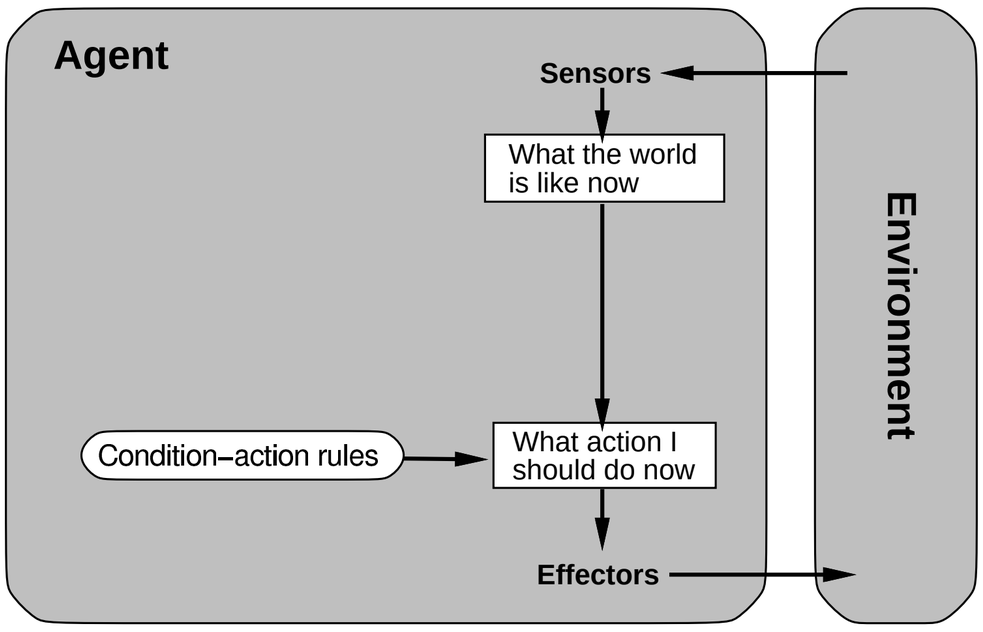
How does the environment develop (the state $\mathbf{s}$) when an action $\mathbf{a}$ is executed? We describe this via a function

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

this includes **non-deterministic** environments.

How do we describe agents?

We could take a function $\boxed{\textbf{action} : \mathbf{P} \longrightarrow \mathbf{A}.}$



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

$$\mathbf{h} : s_0 \rightarrow_{a_0} s_1 \rightarrow_{a_1} \ldots s_n \rightarrow_{a_n} \ldots$$

(or the sequence of observations).

Definition 1.8 (Characteristic Behaviour)
The characteristic behaviour of an (omniscient) agent **action** in an environment **env** is the set

$$\mathbf{Hist}$$

of all histories $\mathbf{h} : \mathbf{s_0} \rightarrow_{a_0} s_1 \rightarrow_{a_1} \ldots s_n \rightarrow_{a_n} \ldots$ with:

1. for all n: $\mathbf{a_n} = \mathbf{action}(\langle \mathbf{s_0}, \ldots, \mathbf{s_n} \rangle)$,
2. for all $n > 0$: $\mathbf{s_n} \in \mathbf{env}(\mathbf{s_{n-1}}, \mathbf{a_{n-1}})$.

For deterministic **env**, the relation "$\in$" can be replaced by "=".

Replace states by percepts:

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

$$\begin{aligned} \mathbf{see} &: \mathbf{S} \longrightarrow \mathbf{P} \\ \text{and} \quad \mathbf{env} &: \mathbf{S} \times \mathbf{A} \longrightarrow \mathbf{2}^{\mathbf{S}}. \end{aligned}$$

Definition 1.10 (Characteristic Behaviour)
The characteristic behaviour of a standard agent **action** in an environment **env** is the set of all sequences

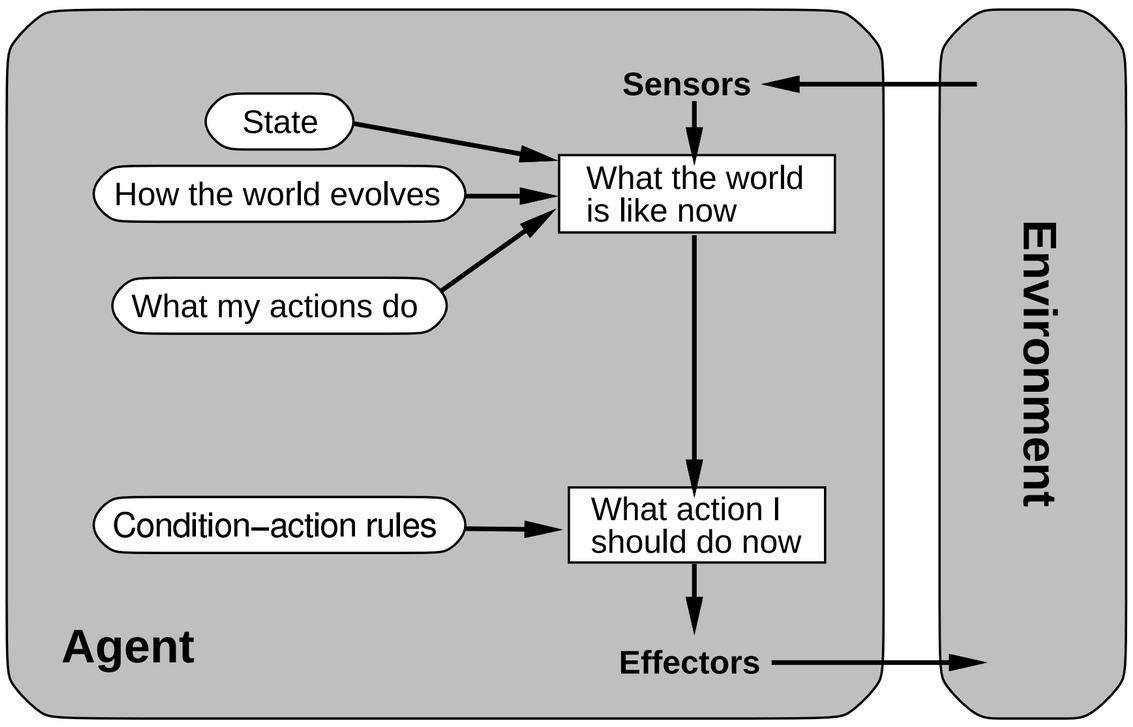
$$\mathbf{p_0} \rightarrow_{a_0} \mathbf{p_1} \rightarrow_{a_1} \ldots \mathbf{p_n} \rightarrow_{a_n} \ldots$$

where

$$\mathbf{p_0} = \mathbf{see}(\mathbf{s_0}),$$
$$\mathbf{a_i} = \mathbf{action}(\langle \mathbf{p_0}, \ldots, \mathbf{p_i} \rangle),$$
$$\mathbf{p_i} = \mathbf{see}(\mathbf{s_i}), \text{ where } \mathbf{s_i} \in \mathbf{env}(\mathbf{s_{i-1}}, \mathbf{a_{i-1}}).$$

Such a sequence (a run), even if deterministic from the agent's view, may cover different histories (environmental behaviours) $\mathbf{s_0} \rightarrow_{a_0} s_1 \rightarrow_{a_1} \ldots s_n \rightarrow_{a_n} \ldots$

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



Definition 1.11 (State-based Agent action)
A state-based agent **action** is given by a function

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

together with

$$\begin{aligned} \mathbf{see} &: \mathbf{S} \longrightarrow \mathbf{P}, \\ \text{and} \quad \mathbf{next} &: \mathbf{I} \times \mathbf{P} \longrightarrow \mathbf{I}. \end{aligned}$$

Here $\mathbf{next}(\mathbf{i}, \mathbf{p})$ is the successor state of $\mathbf{i}$ if $\mathbf{p}$ is observed.

Definition 1.12 (Characteristic Behaviour)
The characteristic behaviour of a state-based agent **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: $\mathbf{a_n} = \mathbf{action}(\mathbf{i_{n+1}})$,
2. for all n: $\mathbf{next}(\mathbf{i_n}, \mathbf{p_n}) = \mathbf{i_{n+1}}$,

Sequence covers the histories $\mathbf{h} : \mathbf{s_0} \rightarrow_{\mathbf{a_0}} \mathbf{s_1} \rightarrow_{\mathbf{a_1}} \ldots$ where

$$\mathbf{a_j} = \mathbf{action}(\mathbf{i_j}),$$
$$\mathbf{s_j} \in \mathbf{env}(\mathbf{s_{j-1}}, a_{j-1}),$$
$$\mathbf{p_j} = \mathbf{see}(\mathbf{s_j})$$

Are state-based agents more expressive than standard agents?
How to measure?

Environmental behaviour of an agent: set of possible histories covered by characteristic behaviour of the agent.

Theorem 1.1 (Equivalence)

Standard agents and state-based agents are equivalent with respect to their environmental behaviour.
More precisely: For each state-based agent action **act** and **next** storage function there exists a standard agent action **act′** which has the same environmental behaviour, and vice versa.

"$\Rightarrow$:" construct act' from act and internal state

"$\Leftarrow$:" internal state simply stores percepts

1.4 Reactive Agents

Intelligent behaviour is Interaction of the agents with their environment. It emerges through splitting in simpler interactions.

Subsumption-Architectures:

- Decision making is realised through **goal-directed behaviours**: each behaviour is an individual action.
  nonsymbolic implementation .
- Many behaviours can be applied **concurrently**. How to select between them?
  Implementation through Subsumption-Hierarchies, Layers .
  Upper layers represent abstract behaviour.

Formal Model

- see: as up to now, but close relation between observation and action: no transformation of the input .
- action: Set of behaviours and inhibition relation.

$$Beh := \{\langle \mathbf{c}, \mathbf{a} \rangle : \mathbf{c} \subseteq \mathbf{P}, \mathbf{a} \in \mathbf{A}\}.$$

$\langle \mathbf{c}, \mathbf{a} \rangle$ "fires" if $\mathrm{see}(\mathbf{s}) \in \mathbf{c}$ ($\mathbf{c}$ stands for "condition").

$$\prec \subseteq Ag_{rules} \times Ag_{rules}$$

is called inhibition-relation, $Ag_{rules} \subseteq Beh$.

$\mathbf{b_1} \prec \mathbf{b_2}$ means: b_1 inhibits b_2, $\mathbf{b_1}$ **has priority over** $\mathbf{b_2}$.

```
function action(p : P) : A
var fired : ℘(R)
var selected : A
begin
    fired := {(c, a) | (c, a) ∈ R and p ∈ c}
    for each (c, a) ∈ fired do
        if ¬(∃(c′, a′) ∈ fired such that (c′, a′) ≺ (c, a)) then
            return a
        end-if
    end-for
    return null
end function action
```

What properties do we need for $\prec$ to make that definition work?

We require $\prec$ to be a total ordering.

42-1

Example 1.1 (Exploring a Planet)
A distant planet (asteroid) is assumed to contain gold. Samples should be brought to a spaceship landed on the planet. It is not known where the gold is. Several autonomous vehicles are available. Due to the topography of the planet there is no connection between the vehicles.

The spaceship sends off radio signals: **gradient field**.

1. Layer (Low Level Behaviour):

(1) **If** detect an obstacle **then** change direction.

2. Layer:

(2) **If** Samples on board **and** at base **then** drop off.

(3) **If** Samples on board **and** not at base **then** follow gradient field.

3. Layer:

(4) **If** Samples found **then** pick them up.

4. Layer:

(5) **If** true **then** take a random walk.

With the following ordering

$$(1) \prec (2) \prec (3) \prec (4) \prec (5).$$

1. Under which assumptions (on the distribution of the gold) does this work perfectly?

2. What if the distribution is more realistic?

- Vehicles can **communicate indirectly** with each other:
  - they put off, and
  - pick up

  radioactive samples that can be sensed.

1. Layer (Low Level Behaviour):

(1) **If** detect an obstacle **then** change direction.

2. Layer:

(2) **If** Samples on board **and** at base **then** drop off.

(3) **If** Samples on board **and** not at base **then** drop off two radioactive crumbs and follow gradient field.

3. Layer:

(4) **If** Samples found **then** pick them up.

(5) **If** radioactive crumbs found **then** take one and follow the gradient field (away from the spaceship).

4. Layer:

(6) **If** true **then** take a random walk.

With the following ordering $(1) \prec (2) \prec (3) \prec (4) \prec (5) \prec (6)$.

Pro: Simple, economic, efficient, robust, elegant.

Contra:

- Without knowledge about the environment agents need to know about the own local environment.
- Decisions only based on local information.
- How about bringing in **learning**?
- Relation between agents, environment and behaviours is not clear.
- Agents with ≤ 10 behaviours are doable. But the more layers the more complicated to understand what is going on.

1.5 BDI-Architecture

Belief, **D**esire, **I**ntention.

From time to time intentions need to be re-examined. But they also should persist: **Pro-active vs. reactive**.

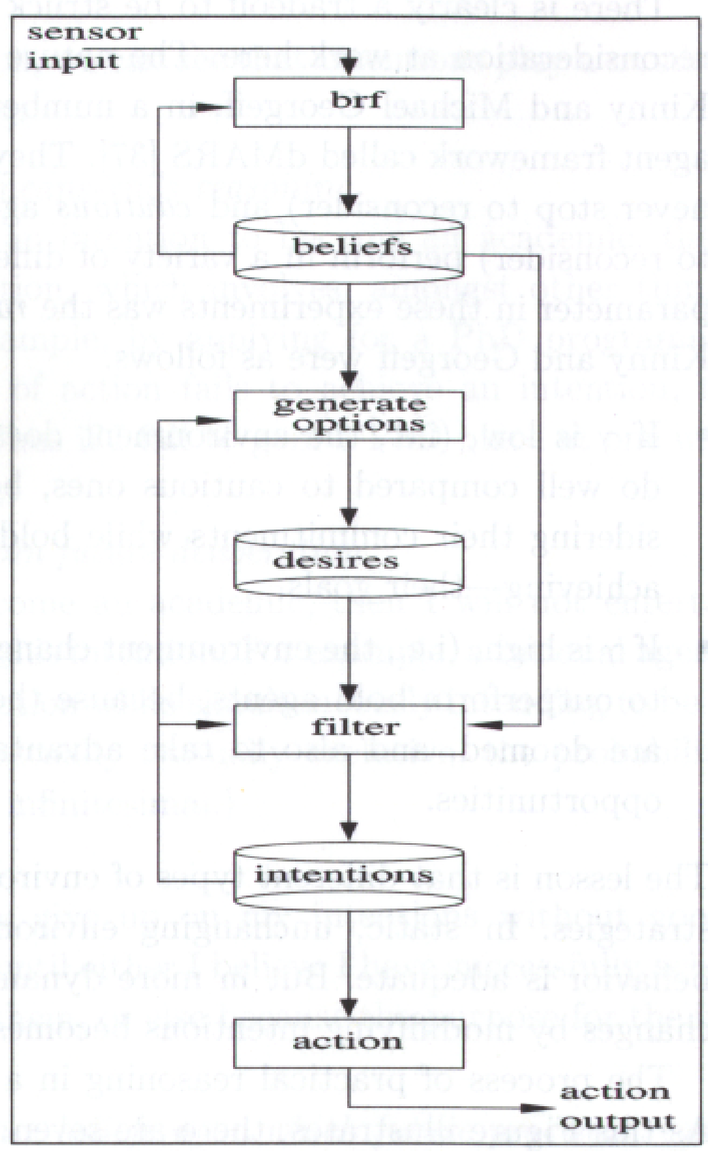
Extreme: *stubborn agents*, *unsure agents*.

What is better? Depends on the environment.
Let γ the **rate of world change**.

1. γ small: stubbornness pays off.
2. γ big: unsureness pays off.

Belief 1:	Going to lectures is worth doing to learn something.
Belief 2:	Dix is a decent lecturer.
Desire 1:	Visit Dix-Lecture, in addition read books.
Intention:	Getting knowledge about Distributed Systems.
New Belief:	Norman does it much better. Therefore revise your Desire.
Desire 2:	Visit Normans's-Lecture, in addition read books.

Of course, Thomas may turn out to be the worst lecturer from all . . .


sensor
input
brf
beliefs
generate
options
desires
filter
intentions
action
action
output

```
function action(p : P) : A
begin
        B := brf(B, p)
        D := options(D, I)
        I := filter(B, D, I)
        return execute(I)
end function action
```

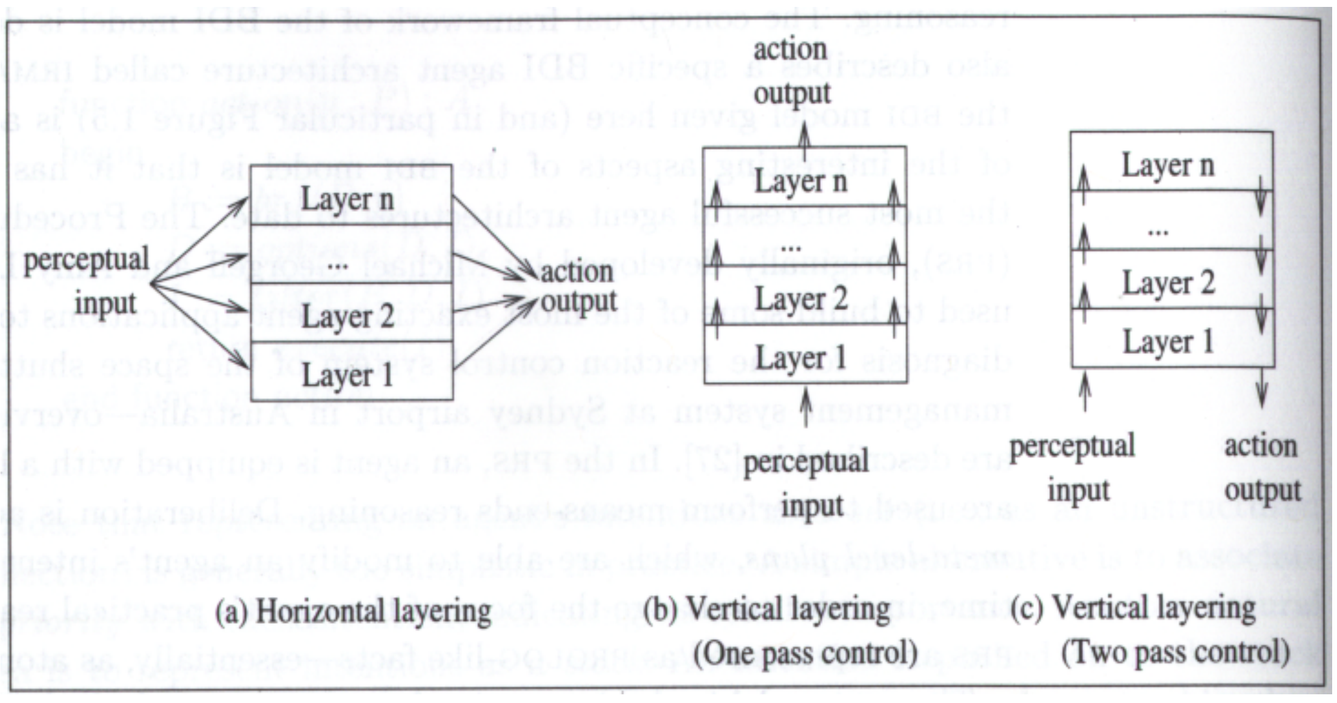
$$(B, D, I) \text{ where } B \subseteq \textbf{Bel}, D \subseteq \textbf{Des}, I \subseteq \textbf{Int}$$

I can be represented as a stack (priorities are available)

- BDI dates back to (Bratman, Israel, and Pollack 1988).
- PRS (*procedural reasoning system*, (Georgeff and Lansky 1987)) uses BDI. Applications: Space Shuttle (Diagnosis), Sydney Airport (air traffic control).
- BDI-Logics: (Rao and Georgeff 1991; Rao and Georgeff 1995; Rao 1995).

1.6 Layered Architectures

At least 2 layers: reactive (event-driven), pro-active (goal directed).



Horizontal:

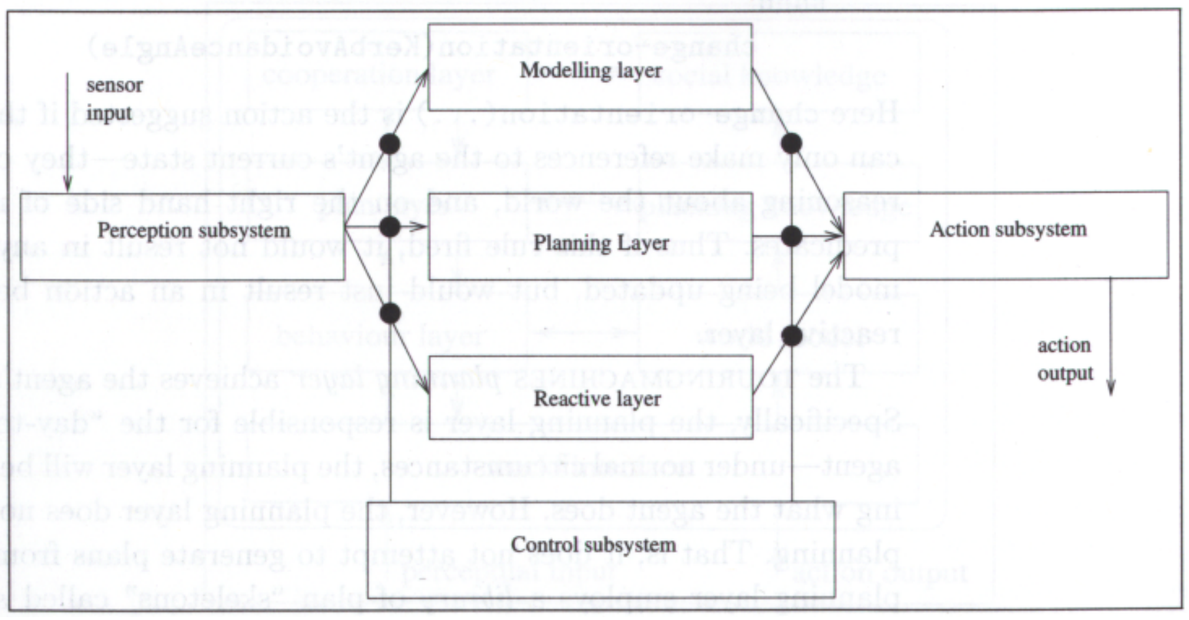
- simple (n behaviours, n layers),
- overall behaviour might be inconsistent,
- Interaction between layers: m^n (m = # actions per layer)
- Control-system is needed.

Vertical:

- Only $m^2(n-1)$ interactions between layers.
- Not fault tolerant: If one layer fails, everything breaks down.

Touring Machine

Autonomous Vehicle.



Rule 1: *Avoid curb*

<u>if</u> $is_in_front(curb, observer)$ and
$speed(observer) > 0$ and
$seperation(curb, observer) < curb_threshold$
<u>then</u> $change_orientation(curb_avoidance_angle)$

Planning-Layer: Pro-active behaviour

Modelling Layer: updating of the world, beliefs, predicts conflicts between agents, changes planning-goals

Control-subsystem: Decides about who is active. Certain observations should never reach certain layers.

Layered architectures do not have a clear semantics and the horizontal interaction is difficult.

References

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

Franklin, S. and A. Graesser (1997). Is it an Agent, or Just a Program? In J. P. Müller, M. Wooldridge, and N. R. Jennings (Eds.), *Intelligent Agents III*, Berlin, Germany. Springer-Verlag. LNAI Volume 1193.

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). *Multi-Agent Systems*. MIT-Press.

Wooldridge, M. J. and N. R. Jennings (1995). Agent Theories, Architectures and Languages: A survey. In M. J. Wooldridge and N. R. Jennings (Eds.), *Intelligent Agents*, Volume 890 of *Lecture Notes in Artificial Intelligence*, pp. 1–39. Springer-Verlag.

Chapter 2. Distributed Decision Making

2.1 Evaluation Criteria

2.2 Voting

2.3 Auctions

2.4 Bargaining

2.5 General Market Criteria

2 Distributed Decision Making

Two and a half lectures: first lecture up to 2.3, second lecture 2.3 – 2.5, half lecture from 2.5 to the end.

Classical DAI: System Designer fixes an **Interaction-Protocol** which is uniform for all agents. The designer also fixes a strategy for each agent.

What is a the outcome, assuming that the **protocol** is followed and the agents follow the strategies?

MAI: **Interaction-Protocol** is given. Each agent determines its own strategy (maximising its own good, via a utility function, without looking at the global task).

What is the outcome, given a **protocol** that guarantees that each agent's desired local strategy is the best one (and is therefore chosen by the agent)?

2.1 General Evaluation Criteria

We need to compare protocols. Each such protocol leads to a solution. So we determine how good these solutions are.

Social Welfare: Sum of all utilities

Pareto Efficiency: A solution x is Pareto-optimal (also called efficient), if

$$\text{there is no solution } x' \text{ with: } (1)\ \exists \text{ agent } \mathbf{ag} : \mathbf{ut}_{\mathbf{ag}}(x') > \mathbf{ut}_{\mathbf{ag}}(x)$$
$$(2)\ \forall \text{ agents } \mathbf{ag}' : \mathbf{ut}_{\mathbf{ag}'}(x') \geq \mathbf{ut}_{\mathbf{ag}'}(x).$$

Individual rational: if the payoff is higher than not participating at all.

Stability:

Case 1: Strategy of an agent depends on the others.

The profile $S^*_{\mathbf{A}} = \langle S^*_1, S^*_2, \ldots, S^*_{|\mathbf{A}|} \rangle$ is called a

Nash-equilibrium , iff

$\forall i$: S^*_i is the best strategy for agent i if all the others choose

$\langle S^*_1, S^*_2, \ldots, S^*_{i-1}, S^*_{i+1}, \ldots, S^*_{|\mathbf{A}|} \rangle$.

Case 2:

Strategy of an agent does not depend on the others.

Such strategies are called dominant.

Prisoner's Dilemma

		Prisoner 2	
		cooperate	defect
Prisoner 1	cooperate	(3,3)	(0,5)
	defect	(5,0)	(1,1)

- **Social Welfare:** Both cooperate,
- **Pareto-Efficiency:** All are Pareto optimal, except when both defect.
- **Dominant Strategy:** Both defect.
- **Nash Equilibrium:** Both defect.

2.2 Voting

Agents give input to a mechanism and the outcome of it is taken as a solution for the agents.

	1	2	3
w_1	A	B	C
w_2	B	C	A
w_3	C	A	B

Figure 2.1: Nonexistence of desired preference ordering.

Comparing A and B: majority for A. Comparing A and C: majority for C. Comparing B and C: majority for B.

Desired Preference ordering: $A > B > C > A$

- Let $\mathbf{A}$ the set of agents, O the set of possible outcomes. (O could be equal to $\mathbf{A}$, or a set of laws).
- The **voting** of agent $\mathbf{i}$ is described by a binary relation

$$\prec_{\mathbf{i}} \subseteq O \times O,$$

which we assume to be asymmetric, strict and transitive. We denote by ***Ord*** the set of all such binary relations.

- Often, not all subsets of O are *votable*, only a subset $V \subseteq 2^O \setminus \{\emptyset\}$.

  Each $v \in V$ represents a possible "set of candidates". The voting model then has to select some of the elements of v.

- Each agent votes independently of the others. But we also allow that only a subset is considered. Let therefore be

$$U \subseteq \prod_{\mathbf{i}=1}^{|\mathbf{A}|} \boldsymbol{Ord}.$$

- A social choice rule wrt. U is a function

$$\mathbf{f^*} : U \rightarrow \mathit{Ord}; (\prec_1, \ldots, \prec_{|\mathbf{A}|}) \mapsto \prec^*$$

For each $V \subseteq 2^O \setminus \{\emptyset\}$ the function $\mathbf{f^*}$ w.r.t. U induces a choice function $\mathbf{C}_{\langle \prec_1, \ldots, \prec_{|\mathbf{A}|} \rangle}$ as follows:

$$\mathbf{C}_{\langle \prec_1, \ldots, \prec_{|\mathbf{A}|} \rangle} =_{def} \begin{cases} V & \longrightarrow & V \\ v & \mapsto & \mathbf{C}_{\langle \prec_1, \ldots, \prec_{|\mathbf{A}|} \rangle}(v) = \max_{\prec^*|_V} v \end{cases}$$

$\max_{\prec^*|_V} v$ is the set of all maximal elements in v according to $\prec^*|_V$.

Each tuple $u = (\prec_1, \ldots, \prec_{|\mathbf{A}|})$ determines the election for all possible $v \in V$.

What are desirable properties for f*?

Pareto-Efficiency: for all $o, o' \in O$: $(\forall \mathbf{i} \in \mathbf{A} : o \prec_{\mathbf{i}} o')$ implies $o \prec^* o'$.

Independence of Irrelevant Alternatives: for all $o, o' \in O$:

$$(\forall \mathbf{i} \in \mathbf{A} : \; o \prec_{\mathbf{i}} o' \;\text{ iff }\; o \prec'_{\mathbf{i}} o') \;\Rightarrow\; (o \prec^* o' \;\text{ iff }\; o \prec'^* o') \,.$$

Note that this implies in particular

$$(\forall \mathbf{i} \in \mathbf{A} : \; \prec_{\mathbf{i}}|_v = \prec'_{\mathbf{i}}|_v)$$
$$\Rightarrow \;\; \forall o, o' \in v, \; \forall v' \in V \text{ s.t. } v \subseteq v' : \; (o \prec^*|_{v'} o' \;\text{ iff }\; o \prec'^*|_{v'} o')$$

The simple **majority vote** protocol does not satisfy the Independence of irrelevant alternatives.

We consider 7 voters ($\mathbf{A} = \{w_1, w_2, \ldots, w_7\}$) and $O = \{a, b, c, d\}$, $V = \{\{a, b, c, d\}, \{a, b, c\}\}$. The columns in the following table represent two different preference orderings of the voters: one is given in black, the second in red.

	$\prec_1$ ($\prec_1$)	$\prec_2$ ($\prec_2$)	$\prec_3$ ($\prec_3$)	$\prec_4$ ($\prec_4$)	$\prec_5$ ($\prec_5$)	$\prec_6$ ($\prec_6$)	$\prec_7$ ($\prec_7$)
a	1 (2)	1 (2)	1 (1)	1 (1)	2 (2)	2 (2)	2 (2)
b	2 (3)	2 (3)	2 (2)	2 (2)	1 (1)	1 (1)	1 (1)
c	3 (4)	3 (4)	3 (3)	3 (3)	3 (3)	3 (3)	3 (3)
d	4 (1)	4 (1)	4 (4)	4 (4)	4 (4)	4 (4)	4 (4)

Let $\prec^*$ be the solution generated by the $\prec_i$ and $\prec^*$ the solution generated by the $\prec_i$. Then we have for $i = 1, \ldots, 7$: $b \prec_i a$ iff $b \prec_i a$, but $b \prec^* a$ and $a \prec^* b$. The latter holds because on the whole set O, for $\prec^*$ a gets selected 4 times

and b only 3 times, while for $\prec^*$ a gets selected only 2 times but b gets still selected 3 times. The former holds because we even have $\prec_i|_{\{a,b,c\}} = \prec_i|_{\{a,b,c\}}$.

The introduction of the irrelevant (concerning the relative ordering of a and b) alternative d changes everything: the original majority of a is split and drops below one of the less preferred alternatives (b).

Theorem 2.1 (Arrows Theorem)
If the choice function $\mathbf{f}^*$ is (1) pareto efficient and (2) independent from irrelevant alternatives, then **there always exists a dictator**: for all $U \subseteq \prod_{\mathbf{i}=1}^{|\mathbf{A}|} \mathit{Ord}$

$$\exists \mathbf{i} \in \mathbf{A} : \ \forall o, o' \in O : o \prec_{\mathbf{i}} o' \ \text{ iff } \ o \prec^* o'.$$

To be more precise: for all $U \subseteq \prod_{\mathbf{i}=1}^{|\mathbf{A}|} \mathit{Ord}$

$$\exists \mathbf{i} \in \mathbf{A} : \forall \langle \prec_1, \ldots, \prec_{|\mathbf{A}|} \rangle \in U :$$
$$\forall o, o' \in O, \ o \prec_{\mathbf{i}} o' \ \text{ iff } \ o \, \mathbf{f}^*(\langle \prec_1, \ldots, \prec_{|\mathbf{A}|} \rangle) \, o'.$$

Ways out:

1. Choice function is not always satisfied.
2. Independence of alternatives is dropped.

The Theorem of Arrow can be even more generalised by weakening the assumption that $\prec^*$ needs to be transitive. In fact, it also holds when using the following definition.

- A social choice rule wrt. U is a function

$$\mathbf{f^*} : U \rightarrow \mathcal{C}(V); (\prec_1, \ldots, \prec_{|\mathbf{A}|}) \mapsto \mathbf{C}_{\langle \prec_1, \ldots, \prec_{|\mathbf{A}|} \rangle},$$

  where $\mathbf{C}_{\langle \prec_1, \ldots, \prec_{|\mathbf{A}|} \rangle}$ **is any function** from V into 2^O satisfying (1) $\mathbf{C}_{\langle \prec_1, \ldots, \prec_{|\mathbf{A}|} \rangle}(v) \neq \emptyset$ and (2) $\mathbf{C}_{\langle \prec_1, \ldots, \prec_{|\mathbf{A}|} \rangle}(v) \subseteq v$.
  Such a function simply selects a subset of v: the elected members of the list v.
  No other assumptions about this function are made.

Binary protocol

Pairwise comparison. Not only introduction of irrelevant alternatives, also ordering may change the outcome.

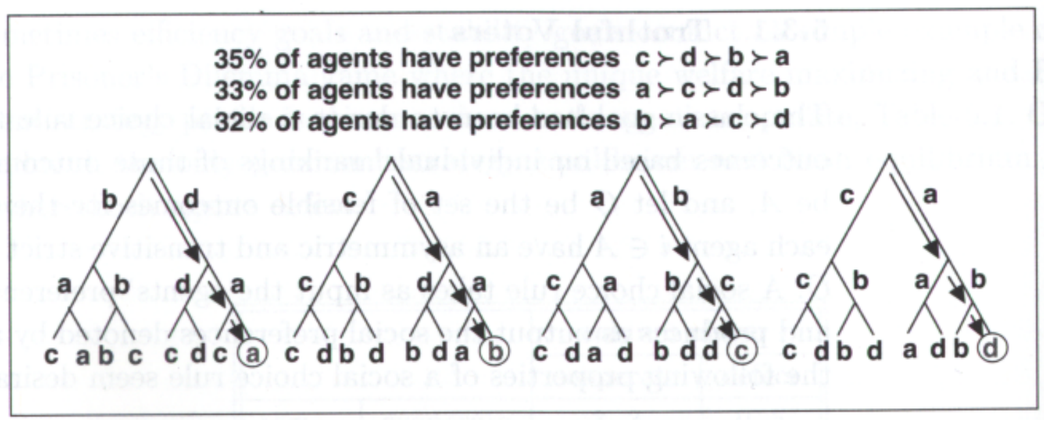


Figure 2.2: Four different orderings and four alternatives.

Last ordering: d wins, but all agents prefer c over d.

Borda protocol

First gets $|O|$ points, second $|O| - 1$, etc. Then sum up, across voters. The highest count wins.

Agent	Preferences
1	$a \succ b \succ c \succ d$
2	$b \succ c \succ d \succ a$
3	$c \succ d \succ a \succ b$
4	$a \succ b \succ c \succ d$
5	$b \succ c \succ d \succ a$
6	$c \succ d \succ a \succ b$
7	$a \succ b \succ c \succ d$
Borda count	c wins with 20, b has 19, a has 18, d loses with 13
Borda count with d removed	a wins with 15, b has 14, c loses with 13

Winner turns loser and loser turns winner

2.3 Auctions

While voting binds all agents, Auctions are always deals between 2.

Types of auctions:

first-price open cry: (English auction), as usual.

first-price sealed bid: one bids without knowing the other bids.

dutch auction: (descending auction) the seller lowers the price until it is taken.

second-price sealed bid: (Vickrey auction) Highest bidder wins, but the price is the second highest bid!

Three different auction settings:

private value: Value depends only on the bidder (cake).

common value: Value depends only on other bidders (treasury bills).

correlated value: Partly on own's values, partly on others.

What is the best strategy in Vickrey auctions?

Theorem 2.2 (Private-value Vickrey auctions)
The **dominant strategy** of a bidder in a Private-value Vickrey auction **is to bid the true valuation**.

Therefore it is equivalent to english auctions.

Vickrey auctions are used to

- allocate computation resources in operating systems,
- allocate bandwidth in computer networks,
- control building heating.

Are first-price auctions better for the auctioneer than second-prize auctions?

Theorem 2.3 (Expected Revenue)
All 4 types of protocols produce the same expected revenue to the auctioneer (assuming (1) private value auctions, (2) values are independently distributed and (3) bidders are risk-neutral).

Why are second price auctions not so popular among humans?

1. Lying auctioneer.
2. When the results are published, subcontractors know the true valuations and what they saved. So they might want to share the profit.

Inefficient Allocation and Lying at Vickrey

Auctioning heterogenous, **interdependent** items.

Example 2.1 (Task Allocation)
Two delivery tasks t_1, t_2. Two agents. $\rightsquigarrow$ **blackboard.**

The global optimal solution is not reached by auctioning independently and truthful bidding.

t_1 goes to agent 2 (for a price of 2) and t_2 goes to agent 1 (for a price of 1.5).

Even if agent 2 considers (when bidding for t_2) that he already got t_1 (so he bids **cost**$(\{t_1, t_2\})$ − **cost**$(\{t_1\}) = 2.5 - 1.5 = 1$) he will get it only with a probability of 0.5.

What about full lookahead ? $\rightsquigarrow$ **blackboard.**

Therefore:

- It pays off for agent 1 to bid more for t_1 (up to 1.5 more than truthful bidding).
- It does not pay off for agent 2, because agent 2 does not make a profit at t_2 anyway.
- Agent 1 bids 0.5 for t_1 (instead of 2), agent 2 bids 1.5. Therefore agent 1 gets it for 1.5. Agent 1 also gets t_2 for 1.5.

Does it make sense to countersperculate at private value Vickrey auctions?

Vickrey auctions were invented to avoid counterspeculation. But what if the private value for a bidder is uncertain? The bidder might be able to determine it, but he needs to invest c.

Example 2.2 (Incentive to counterspeculate)
Suppose bidder 1 does not know the (private-) value v_1 of the item to be auctioned. To determine it, he needs to invest **cost**. We also assume that v_1 is uniformly distributed: satisfies $v_1 \in [0, 1]$.

For bidder 2, the private value v_2 of the item is fixed: $0 \leq v_2 < \frac{1}{2}$. So his dominant strategy is to bid v_2.

Should bidder 1 try to invest **cost** to determine his private value? How does this depend on knowing v_2?

$\rightsquigarrow$ **blackboard.**

Answer: Bidder 1 should invest **cost** if and only if

$$v_2 \geq (2\mathbf{cost})^{\frac{1}{2}}.$$

2.4 Bargaining

Axiomatic Bargaining

We assume two agents $1, 2$, each with a utility function $\mu_i : E \to \mathbb{R}$. If the agents do not agree on a result e the fallback e_{fallback} is taken.

Example 2.3 (Sharing 1 Pound)
How to share 1 Pound?

Agent 1 offers ρ ($0 < \rho < 1$). Agent 2 agrees!

Such deals are individually rational and each one is in Nash-equilibrium!

Therefore we need axioms!

Axioms on the global solution $\mu^* = \langle \mu_1(e^*), \mu_2(e^*) \rangle$.

Invariance: Absolute values of the utility functions do not matter, only relative values.

Symmetry: Changing the agents does not influence the solution.

Irrelevant Alternatives: If E is made smaller but e^* still remains, then e^* remains the solution.

Pareto: The players can not get a higher utility than $\mu^* = \langle \mu_1(e^*), \mu_2(e^*) \rangle$.

Theorem 2.4 (Unique Solution)
The four axioms above uniquely determine a solution. This solution is given by

$$e^* = \text{arg max}_e\{(\mu_1(e) - \mu_1(e_{\text{fallback}})) \times (\mu_2(e) - \mu_2(e_{\text{fallback}}))\}.$$

Strategic Bargaining

No axioms: view it as a game!

Example revisited: Sharing 1 Pound Sterling.

Protocol with finitely many steps: The last offerer just offers ϵ. This should be accepted, so the last offerer gets $1 - \epsilon$.

This is unsatisfiable. Ways out:

1. Add a discountfactor δ: in round n, only the δ^{n-1}th part of the original value is available.
2. Bargaining costs: bargaining is not for free—fees have to be paid.

Finite Games: Suppose $\delta = 0.9$. Then the outcome depends on # rounds.

Round	1's share	2's share	Total value	Offerer
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-3$	0.819	0.181	0.9^{n-4}	2
$n-2$	0.91	0.09	0.9^{n-3}	1
$n-1$	0.9	0.1	0.9^{n-2}	2
n	1	0	0.9^{n-1}	1

Infinite Games: δ_1 factor for agent 1, δ_2 factor for agent 2.

Theorem 2.5 (Unique solution for infinite games)
In a discounted infinite round setting, there exists a unique Nash equilibrium : Agent 1 gets $\frac{1-\delta_2}{1-\delta_1\delta_2}$. Agent 2 gets the rest. Agreement is reached in the first round.

Proof:

Round	1's share	2's share	Offerer
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$t-2$	$1-\delta_2(1-\delta_1\bar{\pi}_1)$		1
$t-1$		$1-\delta_1\bar{\pi}_1$	2
t	$\bar{\pi}_1$		1
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Bargaining Costs

Agent 1 pays c_1, agent 2 pays c_2.

$c_1 = c_2$: Any split is in Nash-equilibrium.

$c_1 < c_2$: Agent 1 gets all.

$c_1 > c_2$: Agent 1 gets c_2, agent 2 gets $1 - c_2$.

2.5 General Equilibrium Mechanisms

A theory for efficiently allocating goods and resources among agents, based on market prices.

Goods: Given $n > 0$ goods g (coffee, mirror sites, parameters of an airplane design). We assume $g \neq g'$ but within g everything is indistinguishable.

Prices: The market has prices $\mathbf{p} = [p_1, p_2, ..., p_n] \in \mathbb{R}^n$: p_i is the price of the good i.

Consumers: Consumer i has $\mu_i(\mathbf{x})$ encoding its preferences over consumption bundles $\mathbf{x_i} = [x_{i1}, ..., x_{in}]^t$, where $x_{ig} \in \mathbb{R}^+$ is consumer i's allocation of good g. Each consumer also has an initial endowment $\mathbf{e_i} = [e_{i1}, ..., e_{in}]^t \in \mathbb{R}$.

Producers: Use some commodities to produce others: $\mathbf{y_j} = [y_{j1}, ..., y_{jn}]^t$, where $y_{jg} \in \mathbb{R}$ is the amount of good g that producer j produces. Y_j is a set of such vectors $\mathbf{y}$.
Profit of producer j: $\mathbf{p} \times \mathbf{y_j}$, where $\mathbf{y_j} \in Y_j$.

Profits: The profits are divided among the consumers (given predetermined proportions Δ_{ij}): Δ_{ij} is the fraction of producer j that consumer i owns (stocks). Profits are divided according to Δ_{ij}.

Definition 2.1 (General Equilibrium)
$(\mathbf{p}^*, \mathbf{x}^*, \mathbf{y}^*)$ is in general equilibrium, if the following holds:

I. The markets are in equilibrium:

$$\sum_i \mathbf{x}_\mathbf{i}^* = \sum_i \mathbf{e}_\mathbf{i} + \sum_j \mathbf{y}_\mathbf{j}^*$$

II. Consumer i maximises preferences according the prices

$$\mathbf{x}_\mathbf{i}^* = \text{arg max}_{\{\mathbf{x}_\mathbf{i} \in \mathbb{R}_+^n \mid \text{cond}_\mathbf{i}\}} \mu_i(\mathbf{x}_\mathbf{i})$$

where $\text{cond}_\mathbf{i}$ stands for $\mathbf{p}^* \times \mathbf{x}_\mathbf{i} \leq \mathbf{p}^* \times \mathbf{e}_\mathbf{i} + \sum_j \Delta_{ij} \mathbf{p}^* \times \mathbf{y}_\mathbf{i}$.

III. Producer j maximises profit wrt. the market

$$\mathbf{y_i^*} = \text{arg max}_{\{\mathbf{y_j} \in Y_j\}} \mathbf{p^*} \times \mathbf{y_j}$$

Theorem 2.6 (Pareto Efficiency)
Each general equilibrium is pareto efficient.

Theorem 2.7 (Coalition Stability)
Each general equilibrium **with no producers** is coalition-stable: no subgroup can increase their utilities by deviating from the equilibrium and building their own market.

Theorem 2.8 (Existence of an Equilibrium)
Let the sets Y_j be closed, convex and bounded above. Let μ_i be continuous, strictly convex and strongly monotone. Assume further that at least one bundle $\mathbf{x_i}$ is producible with only positive entries x_{il}.

Under these assumptions a general equilibrium exists.

2.6 Meaning of the assumptions

Formal definitions: $\rightsquigarrow$ **blackboard.**

Convexity of Y_j: Economies of scale in production do not satisfy it.

Continuity of the μ_i: Not satisfied in bandwidth allocation for video conferences.

Strictly convex: Not satisfied if preference increases when he gets more of this good (drugs, alcohol, dulce de leche).

In general, there exist more than one equilibrium.

Theorem 2.9 (Uniqueness)
If the society-wide demand for each good is non-decreasing in the prices of the other goods, then a unique equilibrium exists.

Positive example: increasing price of meat forces people to eat potatoes (pasta).

Negative example: increasing price of bread implies that the butter consumption decreases.

Chapter 3. Contract Nets, Coalition Formation

3.1 General Contract Nets

3.2 4 Types of Nets

3.3 Abstract Coalition Formation

3.4 Payoff Division

3 Contract Nets, Coalition Formation

3.1 General Contract Nets

How to distribute tasks?

- Global Market Mechanisms. Implementations use a single centralised mediator .
- **Announce, bid, award** -cycle. Distributed Negotiation .

We need the following:

1. **Define a task allocation problem in precise terms.**
2. **Define a formal model for making bidding and awarding decisions.**

Definition 3.1 (Task-Allocation Problem)
A task allocation problem is given by

1. a set of tasks T,
2. a set of agents $\mathbf{A}$,
3. a cost function $\mathbf{cost_i} : 2^T \longrightarrow \mathbb{R} \cup \{\infty\}$ (stating the costs that agent $\mathbf{i}$ incurs by handling some tasks), and
4. the initial allocation of tasks

$$\langle T_{\mathbf{1}}^{init}, \ldots, T_{|\mathbf{A}|}^{init} \rangle,$$

where $T = \bigcup_{\mathbf{i} \in \mathbf{A}} T_{\mathbf{i}}^{init}$, $T_{\mathbf{i}}^{init} \cap T_{\mathbf{j}}^{init} = \emptyset$ for $\mathbf{i} \neq \mathbf{j}$.

Definition 3.2 (Accepting Contracts and Allocating Tasks)
A contractee $\mathbf{q}$ accepts a contract if it gets paid more than the marginal cost of handling the tasks of the contract

$$MC^{add}(T^{contract}|T_{\mathbf{q}}) =_{def} \mathbf{cost}_{\mathbf{q}}(T^{contract} \cup T_{\mathbf{q}}) - \mathbf{cost}_{\mathbf{q}}(T_{\mathbf{q}}).$$

A contractor $\mathbf{r}$ is willing to allocate the tasks $T^{contract}$ from its current task set $T_{\mathbf{r}}$ to a contractee, if it has to pay less than it saves by handling them itself:

$$MC^{remove}(T^{contract}|T_{\mathbf{r}}) =_{def} \mathbf{cost}_{\mathbf{r}}(T_{\mathbf{r}}) - \mathbf{cost}_{\mathbf{r}}(T_{\mathbf{r}} - T^{contract}).$$

Definition 3.3 (The Protocol)
Agents suggest contracts to others and make their decisions according to the above MC^{add} and MC^{remove} sets.

Agents can be both contractors and contractees. Tasks can be recontracted.

- The protocol is domain independent .
- Can only improve at each step: **Hill-climbing in the space of all task allocations**. Maximum is social welfare: $-\sum_{i \in A} \mathbf{cost}_i(T_i)$.
- Anytime algorithm!

3.2 4 Types of Nets

Definition 3.4 (O-, C-, S-, M- Nets)
A contract is called of type

O (Original): if only one task is moved,

C (Cluster): if a set of tasks is moved,

S (Swap): if a pair of agents swaps a pair of tasks,

M (Multi): if more than two agents are involved in an atomic exchange of tasks.

Problem: local maxima.

A contract may be individually rational but the task allocation is not globally optimal.

Theorem 3.1 (Each Type Avoids Local Optima of the Others)
For each of the 4 types there exist task allocations where no IR contract with the remaining 3 types is possible, but an IR contract with the fourth type is.

Theorem 3.2 (O-, C-, S-, M- Nets do not reach Global Optima)
There are instances of the task allocation problem where no IR sequence from the initial task allocation to the optimal one exists using O-, C-, S-, and M- contracts.

Definition 3.5 (OCSM Nets)
A OCSM-contract is a pair $\langle T, \rho \rangle$ of $|\mathbf{A}| \times |\mathbf{A}|$ matrices. An element $T_{\mathbf{i},\mathbf{j}}$ stands for the set of tasks that agent $\mathbf{i}$ gives to agent $\mathbf{j}$. $\rho_{\mathbf{i},\mathbf{j}}$ is the amount that $\mathbf{i}$ pays to $\mathbf{j}$.

Theorem 3.3 (OCSM-Nets Suffice)
Let $|\mathbf{A}|$ and $|T|$ be finite. If a protocol allows OCSM-contracts, any hill-climbing algorithm finds the globally optimal task allocation in a finite number of steps without backtracking.

Theorem 3.4 (OCSM-Nets are Necessary)
If a protocol does not allow a certain OCSM contract, then there are instances of the task allocation problem where no IR-sequence exists from the initial allocation to the optimal one.

3.3 Coalition Formation

Idea:

Consider a protocol (to build coalitions) as a game and consider Nash-equilibrium.

Problem: Nash-Eq is too weak!

Definition 3.6 (Strong Nash Equilibrium)
A profile is in strong Nash-Eq if there is no subgroup that can deviate by changing strategies jointly in a manner that increases the payoff of all its members, given that nonmembers stick to their original choice.

This is often too strong and does not exist.

Definition 3.7 (Characteristic Function Game (CFG))
In a CFG the value of a coalition S is given by a characteristic function v_S.

Thus it is independent of the nonmembers. But:

1. **Positive Externalities:** Caused by overlapping goals. Nonmembers perform actions and move the world closer to the coalition's goal state.
2. **Negative Externalities:** Caused by shared resources. Nonmembers may use the resources so that not enough is left.

Definition 3.8 (Coalition Formation in CFG's)
Coalition Formation in CFG's consists of the following three steps

Forming $\mathcal{CS}$: formation of coalitions such that within each coalition agents coordinate their activities. This partitioning is called coalition structure $\mathcal{CS}$.

Solving Optimisation Problem: For each coalition the tasks and resources of the agents have to be pooled. Maximise monetary value.

Payoff Division: Divide the value of the generated solution among agents.

An interesting property.

Definition 3.9 (Super-additive Games)
A game is called super-additive, if

$$v_{S \cup T} \geq v_S + v_T,$$

where $S, T \subseteq \mathbf{A}$ and $S \cap T = \emptyset$.

Lemma 3.1
Coalition formation for super-additive games is trivial.

Conjecture 3.1
All games are super-additive.

The conjecture is wrong, because the coalition process is not for free:
communication costs, penalties, time limits.

Maximise the social welfare of the agents $\mathcal{A}$ by finding a coalition structure

$$\mathcal{CS}^* = \arg\max_{\mathcal{CS} \in part(\mathcal{A})} \mathrm{Val}(\mathcal{CS}),$$

where

$$\mathrm{Val}(\mathcal{CS}) := \sum_{S \in \mathcal{CS}} v_S.$$

How many coalition structures are there?
Too many: $\Omega(|\mathcal{A}|^{\frac{|\mathcal{A}|}{2}})$. Enumerating is only feasible if $|\mathcal{A}| < 15$.

How can we approximate $\mathrm{Val}(\mathcal{CS})$?

Choose set $\mathcal{N}$ (a subset of all partitions of $\mathbf{A}$) and pick the best coalition seen so far:

$$\mathcal{CS}^*_{\mathcal{N}} = \arg\max_{\mathcal{CS} \in \mathcal{N}} \mathrm{Val}(\mathcal{CS}).$$

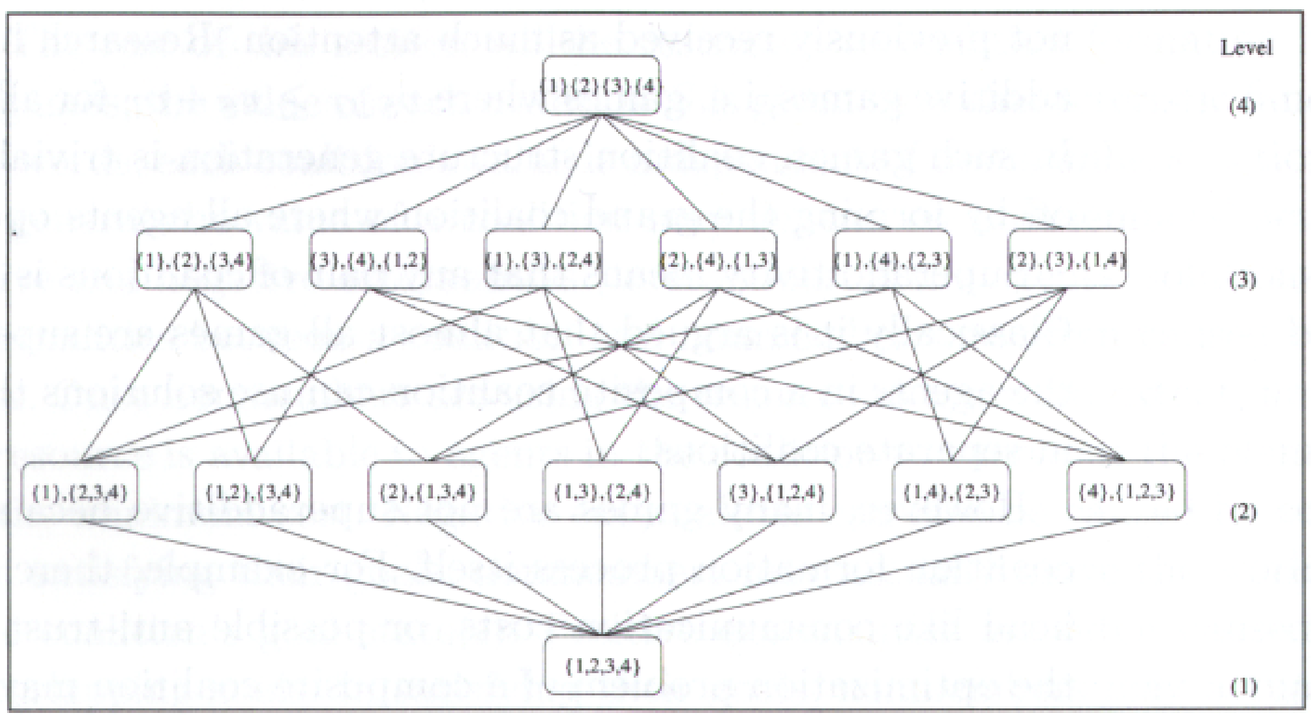


Figure 3.1: Coalition Structure Graph.

We want our approximation as good as possible. That means:

$$\frac{\mathbf{Val}(\mathcal{CS}^*)}{\mathbf{Val}(\mathcal{CS}^*_{\mathcal{N}})} \leq k,$$

where k is as small as possible.

Theorem 3.5 (Minimal Search to get a bound)
To bound k, it suffices to search the lowest two levels of the $\mathcal{CS}$-graph. Using this search, the bound $k = |\mathbf{A}|$ can be taken. This bound is tight and the number of nodes searched is $2^{|\mathbf{A}|-1}$.

No other search algorithm can establish the bound k while searching through less than $2^{|\mathbf{A}|-1}$ nodes.

What exactly means the last theorem? Let n_{min} be the smallest size of $\mathcal{N}$ such that a bound k can be established.

Positive result: $\frac{n_{min}}{\text{partitions of } \mathbf{A}}$ approaches 0 for $|\mathbf{A}| \longrightarrow \infty$.

Negative result: To determine a bound k, one needs to search through exponentially many coalition structures.

Algorithm 3.1 ($\mathcal{CS}$-Search-1)
The algorithm comes in 3 steps:

1. Search the bottom two levels of the $\mathcal{CS}$-graph.
2. Do a breadth-first search from the top of the graph.
3. Return the $\mathcal{CS}$ with the highest value.

This is an **anytime algorithm**.

Theorem 3.6 ($\mathcal{CS}$-Search-1 up to Layer l)
With the algorithm $\mathcal{CS}$-**Search-1** we get the following bound for k after searching through layer l:

$$\begin{cases} \lceil \frac{|\mathbf{A}|}{h} \rceil & \text{if } |\mathbf{A}| \equiv h-1 \mod h \text{ and } |\mathbf{A}| \equiv l \mod 2, \\ \lfloor \frac{|\mathbf{A}|}{h} \rfloor & \text{otherwise.} \end{cases}$$

where $h =_{def} \lfloor \frac{|\mathbf{A}|-l}{2} \rfloor + 2$.

Thus, for $l = |\mathbf{A}|$ (check the top node), k switches from $|\mathbf{A}|$ to $\frac{|\mathbf{A}|}{2}$.

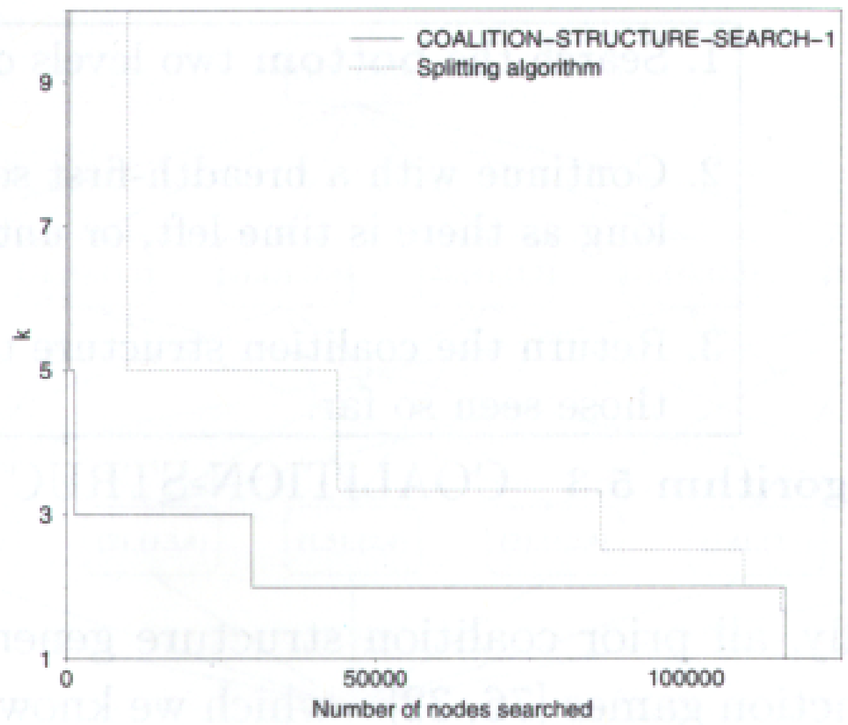


Figure 3.2: Comparing $\mathcal{CS}$-Search-1 with another algorithm.

1. Is $\mathcal{CS}$**-Search-1** the best anytime algorithm?
2. The search for best k for $n' > n$ is perhaps not the same search to get best k for n.
3. $\mathcal{CS}$**-Search-1** does not use any information while searching. Perhaps k can be made smaller by not only considering Val($\mathcal{CS}$) but also v_S in the searched $\mathcal{CS}'$.

3.4 Payoff Division

The payoff division should be fair between the agents, otherwise they leave the coalition.

Definition 3.10 (Dummies, Interchangeable)
Agent i is called a **dummy**, if

for all coalitions S with $i \notin S$: $v_{S \cup \{i\}} - v_S = v_{\{i\}}$.

Agents i and j are called interchangeable, if

for all coalitions S with $i \in S$ and $j \notin S$: $v_{S \setminus \{i\} \cup \{j\}} = v_S$

Three axioms:

Symmetry: If $\mathbf{i}$ and $\mathbf{j}$ are interchangeable, then $x_{\mathbf{i}} = x_{\mathbf{j}}$.

Dummies: For all dummies $\mathbf{i}$: $x_{\mathbf{i}} = \boldsymbol{v}_{\{\mathbf{i}\}}$.

Additivity: For any two games $\boldsymbol{v}, \boldsymbol{w}$:

$$x_{\mathbf{i}}^{\boldsymbol{v} \oplus \boldsymbol{w}} = x_{\mathbf{i}}^{\boldsymbol{v}} + x_{\mathbf{i}}^{\boldsymbol{w}},$$

where $\boldsymbol{v} \oplus \boldsymbol{w}$ denotes the game defined by $(\boldsymbol{v} \oplus \boldsymbol{w})_S = \boldsymbol{v}_S + \boldsymbol{w}_S$.

Theorem 3.7 (Shapley-Value)
There is only one payoff division satisfying the above 3 axioms. It is called the Shapley value of agent i and is defined by

$$x_i = \sum_{S \subseteq A} \frac{(|A| - |S|)!(|S| - 1)!}{|A|!}(v_S - v_{S \setminus \{i\}}).$$

- $(|A| - S)!$ is the number of all possible joining orders of the agents (to form a coalition).
- The Shapley value sums up the marginal contributions of agent i averaged over all joining orders.
- An **expected gain** can be computed by taking a random joining order and computing the Shapley value.

References

Arens, Y., C. Y. Chee, C.-N. Hsu, and C. Knoblock (1993). Retrieving and Integrating Data From Multiple Information Sources. *International Journal of Intelligent Cooperative Information Systems 2*(2), 127–158.

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

Bayardo, R., et al. (1997). Infosleuth: Agent-based Semantic Integration of Information in Open and Dynamic Environments. In J. Peckham (Ed.), *Proceedings of ACM SIGMOD Conference on Management of Data*, Tucson, Arizona, pp. 195–206.

Brass, S. and J. Dix (1994). A disjunctive semantics based on unfolding and bottom-up evaluation. In B. Wolfinger (Ed.), *Innovationen bei Rechen- und Kommunikationssystemen,* (IFIP '94-Congress, Workshop FG2: Disjunctive Logic Programming and Disjunctive

Databases), Berlin, pp. 83–91. Springer.

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

Brink, A., S. Marcus, and V. Subrahmanian (1995). Heterogeneous Multimedia Reasoning. *IEEE Computer 28*(9), 33–39.

Chawathe, S., et al. (1994, October). The TSIMMIS Project: Integration of Heterogeneous Information Sources. In *Proceedings of the 10th Meeting of the Information Processing Society of Japan*, Tokyo, Japan. Also available via anonymous FTP from host db.stanford.edu, file /pub/chawathe/1994/tsimmis-overview.ps.

Currie, K. and A. Tate (1991). O-plan: the open planning architecture. *Artificial Intelligence 52(1)*.

Dix, J., T. Eiter, M. Fink, A. Polleres, and Y. Zhang (2003). Monitoring Agents using Declarative Planning. In R. Kruse (Ed.), *Proceedings of the 27th German Annual Conference on Artificial Intelligence (KI*

'03), Hamburg, Germany, LNAI ???, Berlin. Springer.

Dix, J., T. Eiter, M. Fink, A. Polleres, and Y. Zhang (2004). Monitoring Agents using Declarative Planning. *Fundamenta Informaticae*, to appear.

Dix, J., S. Kraus, and V. Subrahmanian (2001). Temporal agent reasoning. *Artificial Intelligence 127*(1), 87–135.

Dix, J., S. Kraus, and V. Subrahmanian (2002, July). Agents dealing with time and uncertainty. In C. Castelfranchi and W. L. Johnson (Eds.), *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems*. New York: ACM Press.

Dix, J., U. Kuter, and D. Nau (2003). Planning in answer set programming using ordered task decomposition. In R. Kruse (Ed.), *Proceedings of the 27th German Annual Conference on Artificial Intelligence (KI '03), Hamburg, Germany*, LNAI ???, Berlin. Springer.

Dix, J. and M. Müller (1994a). Partial Evaluation and Relevance for Approximations of the Stable Semantics. In Z. Ras and M. Zemankova (Eds.), *Proceedings of the 8th Int. Symp. on Methodologies for Intelligent Systems, Charlotte, NC, 1994*, LNAI 869, Berlin, pp. 511–520. Springer.

Dix, J. and M. Müller (1994b). The Stable Semantics and its Variants: A Comparison of Recent Approaches. In L. Dreschler-Fischer and B. Nebel (Eds.), *Proceedings of the 18th German Annual Conference on Artificial Intelligence (KI '94), Saarbrücken, Germany*, LNAI 861, Berlin, pp. 82–93. Springer.

Dix, J., H. Munoz-Avila, and D. N. an Lingling Zhang (2002). Theoretical and Empirical Aspects of a Planner in a Multi-Agent Environment. In G. Ianni and S. Flesca (Eds.), *Proceedings of Journees Europeens de la Logique en Intelligence artificielle (JELIA '02)*, LNCS 2424, pp. 173–185. Springer.

Dix, J., H. Munoz-Avila, D. Nau, and L. Zhang (2002, July). Planning in

a multi-agent environment: Theory and practice. In C. Castelfranchi and W. L. Johnson (Eds.), *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems*. New York: ACM Press.

Dix, J., H. Munoz-Avila, D. Nau, and L. Zhang (2003). IMPACTing SHOP: Putting an AI planner into a Multi-Agent Environment. *Annals of Mathematics and AI 37(4)*, 381–407.

Dix, J., M. Nanni, and V. S. Subrahmanian (2000). Probabilistic agent reasoning. *ACM Transactions of Computational Logic 1*(2), 201–245.

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

Eiter, T., W. Faber, N. Leone, G. Pfeifer, and A. Polleres (2002). A Logic Programming Approach to Knowledge-State Planning, II: The DLV$^{\mathcal{K}}$ System. *Artificial Intelligence 144*(1-2), 157–211.

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

257–307.

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. Rogers (2000). Heterogeneous Active Agents, III: Polynomially Implementable Agents. *Artificial Intelligence 117*(1), 107–167.

Franklin, S. and A. Graesser (1997). Is it an Agent, or Just a Program? In J. P. Müller, M. Wooldridge, and N. R. Jennings (Eds.), *Intelligent Agents III*, Berlin, Germany. Springer-Verlag. LNAI Volume 1193.

Genesereth, M. R. and S. P. Ketchpel (1994). Software Agents. *Communications of the ACM 37*(7), 49–53.

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.

Munoz-Avila, H., D. Aha, D. Nau, R. Weber, L. Breslow, and F. Yaman (2001). Sin: Integrating case-based reasoning with task

decomposition. In *Proceedings of IJCAI-01*.

Nau, D., Y. Cao, A. Lotem, and H. Muñoz-Avila (1999). Shop: Simple hierarchical ordered planner. In *Proceedings of IJCAI-99*.

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.

Sacerdoti, E. (1977). *A Structure for Plans and Behavior*. American Elsevier Publishing.

Sakama, C. and H. Seki (1994). Partial Deduction of Disjunctive Logic Programs: A Declarative Approach. In *Logic Program Synthesis and Transformation – Meta Programming in Logic*, LNCS 883, Berlin, pp. 170–182. Springer.

Son, T., C. Baral, and S. McIlraith. (2001, September). Planning with domain-dependent knowledge of different kinds – an answer set programming approach. In T. Eiter, M. Truszczyński, and W. Faber (Eds.), *Logic Programming and Non-Monotonic Reasoning, Proceedings of the Sixth International Conference*, LNCS 2173, Berlin, pp. 226–239. Springer.

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

Tate, A. (1977). Generating Project Networks. In *Proc. IJCAI-77*, pp.

888–893.

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

Wiederhold, G. (1993). Intelligent Integration of Information. In *Proceedings of ACM SIGMOD Conference on Management of Data*, Washington, DC, pp. 434–437.

Wilder, F. (1993). *A Guide to the TCP/IP Protocol Suite*. Artech House.

Wilkins, D. (1988). *Practical planning - extending the classical AI planning paradigm*. Morgan Kaufmann.

Wooldridge, M. J. and N. R. Jennings (1995). Agent Theories, Architectures and Languages: A survey. In M. J. Wooldridge and N. R. Jennings (Eds.), *Intelligent Agents*, Volume 890 of *Lecture Notes in Artificial Intelligence*, pp. 1–39. Springer-Verlag.