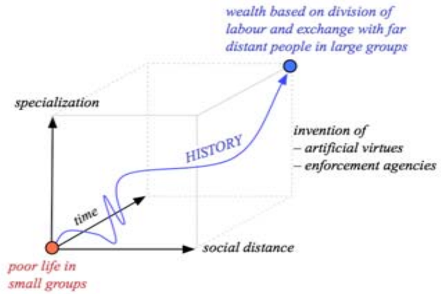
Rainer Hegselmann & Oliver Will (Bayreuth University)

HUME$_{1.0}$ Modelling Hume's moral and political theory

Bielefeld, May 8.–10. 2008

Conference "Norms and Values –The role of social norms as instruments of value realisation"

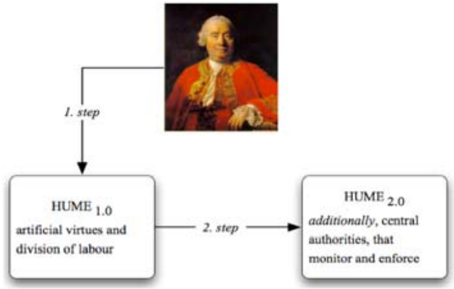
Hume's global view



Hume's view: Other scholars reconstructions

Modelling Hume: Our approach

- Computer model with explicitly introduced parameters
- Agent based
- Systematic exploration of the parameter space



Part of ***EMIL*** (***Em**ergence **i**n the **l**oop*), an EU-project:

SIXTH FRAMEWORK PROGRAMME
PRIORITY 2
Information Society Technologies

Key components of HUME$_{1.0}$

- Specialization and division of labour
- Different exchange regimes
- Two fundamental structural scenarios
- Decision vector
- Type classification
- Matching of agents
- Learning strategies and moral transformation
- The central loop of the model

How to model specialization?

- Period by period at least some of the agents get a problem out of a set of K problems. Problems are characterized by positive integers $\leq K$
- Agents have a time dependent competence vector with K components that sum up to 1.
- By working on a certain problem agents become better in solving the type of problem they are working on. But at the same time their other competences deteriorate.

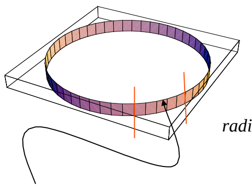
agent$_1$ gets problem k and adresses agent$_2$ for a solution

$$C_2(t) = \langle c_{21}(t), c_{22}(t), \ldots \underbrace{c_{2k}(t)}, \ldots, c_{2K}(t) \rangle, \textit{ where } \sum_{j=1}^{K} c_{2j}(t) = 1$$

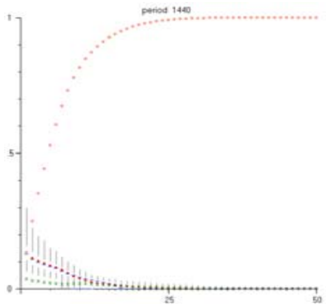
$$c_{2k}(t) + \Delta, \text{if agent}_2 \text{ works on problem } k$$

$$\text{Re-normalization such that: } \sum_{j=1}^{K} c_{2j}(t+1) = 1$$

Dynamics of competences in a 1D-world –NO incentives involved –

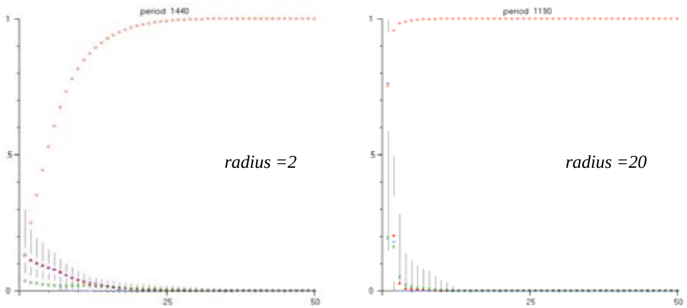


Player$_1$ has a special problem and looks for a player$_2$ with good competence for solving it.



Ordered competence vectors (best component to worst component) over a population of 500 agents with Δ=0.02 (mean, median, range, standard deviation, aggregated values).

Dynamics of competences in a simple problem space: a 1-dimensional example



Ordered competence vectors (best component to worst component) over a population of 500 agents with Δ=0.02 (mean, median, range, standard deviation, aggregated values).

What should competence affect?

VALUE	COSTS	VALUE & COSTS
The higher the competence the higher the value of the solution.	The higher the competence, the lower the costs to produce a solution.	The higher the competence, the higher the value and the lower the costs. VALUE_ADDED = VALUE – COSTS

Which functions $f(c)$ could do the job?

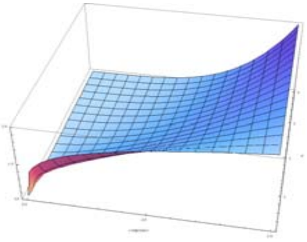
competence, $0 \le c \le 1$

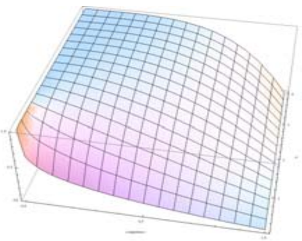
Competence dependent: values and costs

$\text{VALUE} = (1+c)^{\varphi}$

Other constants possible!

$\text{COSTS} = (1-c)^{\sigma}$





Exchange regimes: Two types of agents/roles

P-agent/role

has in period t a problem $k \in K$ *and* a certain own competence c_{Pk} for solving it

The P-agent's *choice*

- Solve k yourself
- Look for a S-agent who solves k – but has to be paid

S-agent/role

has in period t a certain competence c_{Sk} for solving problem $k \in K$

Depending on the competences c_{Pk} and c_{Sk}, the value and costs of the solution of problem k may differ.

Exchange regimes: Who pays/delivers *when*?

Three *risky* possibilities:

The P-agent 'prepays'

and only afterwards the S-agent starts working on the solution which he eventually delivers – *or not*.

The S-agent delivers first

and only afterwards the P-agent does the compensating payment – *or not*.

The P- and S-agent pay/ deliver simultaneously

– *or not* since each one may try to get away without doing his part.

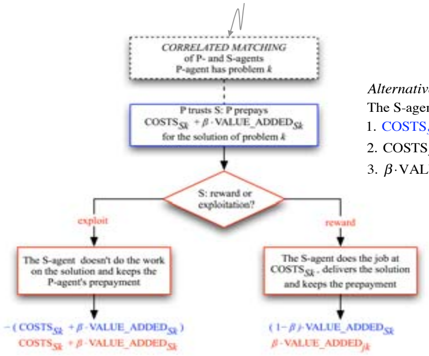
Trust game (TG)

with *either* the P- *or* the S-agent as the first mover.

prisoners' dilemma (PD)

(There exist almost risk free exchange regimes – for instance via mediators. But that implies institutions and an already existing division of labour. In our pre-historical context such institutions are assumed not to exist.)

Exchange regimes: Who pays/delivers *what*? Focus: Trust game structure



excludes matchings that do not make sense:
- *P-agent better off by a solution of her own.*
- *S-agent's costs not covered.*

Alternative payoff regimes
The S-agent has to get for a solution of problem k:
1. $\mathrm{COSTS}_{Sk} + \beta \cdot \mathrm{VALUE\_ADDED}_{Sk}$
2. $\mathrm{COSTS}_{Sk} + \beta \cdot \left(\mathrm{VALUE\_ADDED}_{Sk} - \mathrm{VALUE\_ADDED}_{Pk}\right)$
3. $\beta \cdot \mathrm{VALUE}_{Sk}$

Exchange regimes: *Three* moral dimensions

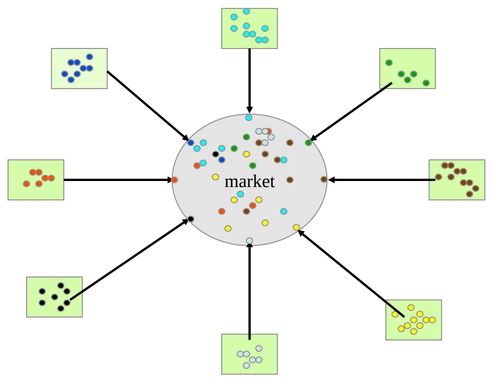
1. *Trustworthiness* (As an S-agent one rewards in the TG structure and cooperates in the PD-structure. As an P-agent one cooperates in the PD-structure.)
2. *Honesty* as to the competencies in the matching procedure
3. *Reliability* as to the effort level

Hume$_{1.0}$ *focuses on trustworthiness only. Honesty as to competencies and reliability as to effort level are assumed.*

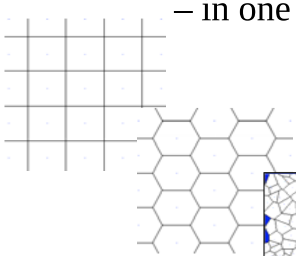
Two social structures

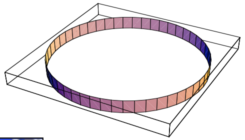
1. Partioning of *non*-overlapping neighborhoods.
   P-agents and S-agents exchange *either* within their neighborhood *or* on a market.
2. Networks of *overlapping* neighborhoods.
   Social distance matters: P-agents look for S-agents only within a certain distance. S-agents reward or exploit depending on distance.

1. Structure: Partition and market

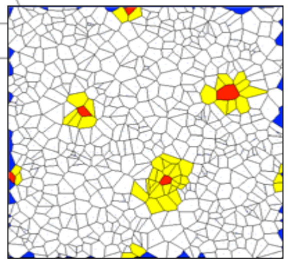


2. Structure: Overlapping neighborhoods – in one or two dimensions –





Torus or not!



Time dependent *decision vector* of agents (Partition/market Szenario)

Propensities of agent *i* at time *t*:

... to enter the *market* as a P-agent

... to enter the *market* as a S-agent

$$D_i^{PM}(t) = \left\langle p_i^{P\_market}(t), p_i^{S\_market}(t), p_i^{reward\_local}(t), p_i^{reward\_market}(t) \right\rangle$$

... to reward *locally*

... to reward in the *market*

Type classification (Partition/market–scenario)

P-agents classify S-agents based on

1. reputation
2. signal reading

For the start we implement signal reading:

$$D_i^{PM}(t) = \left\langle p_i^{P\_market}(t), p_i^{S\_market}(t), \boxed{p_i^{reward\_local}(t), p_i^{reward\_market}(t)} \right\rangle$$

A P-agent classifies an S-agent *i* as a rewarder with a probability that is the higher the higher *i*'s reward probability.

On the market the classification as a rewarder is more cautious than locally.

P-agents do the classification individually. S-agents may there be differently classified by different agents.

We experiment with different functions that satisfy the conditions.

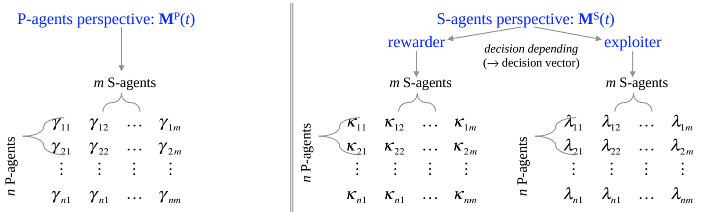
Matching: The central problems

1. Detection of trustworthiness
2. Finding competent problem solvers
3. Finding agents with interesting problems

Modelling strategies:

- Explicitly representing the target process
  [*process representation*]
- Directly generating the effects a process presumably has *plus* an accompanying 'story' that hints to the underlying details
  [*effect generation*] ⇐

Correlated Matching (TG): Basic ideas I



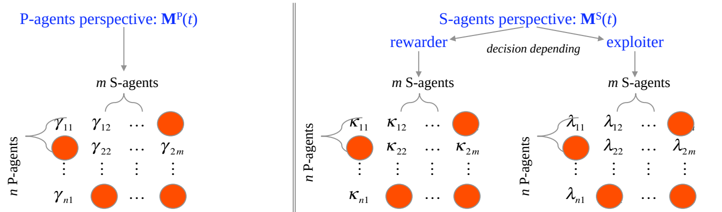
Entries in the $n \times m$ matrix are the *reward* payoffs for a P-agent with problem k if a S-agent solves the problem.

S is *excluded* from a match with P
– if P classifies S as not trustworthy (based on signalling or reputation).
– if S is out of range, i.e. not acting in the same location (market or neighbourhood) as P.
– *Reward* in an exchange with S is worse for P than the *"Solve it at your own"*– solution.
–The prepayment of P does not cover S's costs.

Entries in the $n \times m$ matrix are *either* the reward *or* exploit payoffs for a S-agent if solving problem k of the P-agent.

P-agents are *excluded* from a match with S
– if P classifies j as not trustworthy (based on signalling or reputation).
– if P is out of range, i.e. not acting in the same location (market or neighbourhood) as S.
– *Reward* in an exchange with S is worse for P than the *"Solve it at your own"*– solution.
– The prepayment of P does not cover S's costs.

Correlated Matching (TG): Basic ideas I



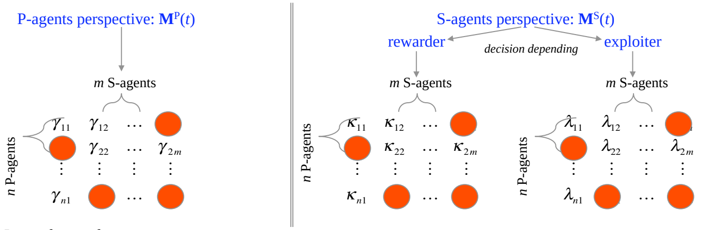
Entries in the $n \times m$ matrix are the *reward* payoffs for P-agent i with problem k if S-agent j solves the problem.

S-agents j are *excluded* from a match with i
– if i classifies j as not trustworthy (based on signalling).
– if j is out of range, i.e. not acting in the same location (market or neighbourhood) as i.
– *Reward* in an exchange with j is worse for i than the *"Solve it at your own"*– solution .
–The prepayment does not cover the S-agent's costs.

Entries in the $n \times m$ matrix are *either* the reward *or* exploit payoffs for S-agent j if solving problem k of P-agent i.

P-agents i are *excluded* from a match with j
– if i classifies j as not trustworthy (based on signalling).
– if i is out of range, i.e. not acting in the same location (market or neighbourhood) as j.
– *Reward* for i is worse than *"Solve it at your own"*.
– The prepayment does not cover the S-agent's costs.

Correlated Matching (TG): Basic ideas II



Iterated procedure:

1. Positive entries in $\mathbf{M}^{\mathrm{P}}(t)$ are normalized such that each *row* sum is 1.
2. Positive entries in $\mathbf{M}^{\mathrm{S}}(t)$ are normalized such that each *column* sum is 1.
3. We select with equal chance $\mathbf{M}^{\mathrm{P}}(t)$ or $\mathbf{M}^{\mathrm{S}}(t)$:
4. Case $\mathbf{M}^{\mathrm{P}}(t)$: With equal chance we select a P-agent i. With a chance corresponding to the P-agents normalized row entries we assign a S-agent j to i and get the match <i,j>. Afterwards row i and column j is eliminated in $\mathbf{M}^{\mathrm{P}}(t)$ and $\mathbf{M}^{\mathrm{S}}(t)$.
5. Case $\mathbf{M}^{\mathrm{S}}(t)$: With equal chance we select a S-agent j. With a chance corresponding to the S-agents normalized column entries we assign a P-agent i to j and get the match <i,j>. Afterwards row i and column j is eliminated in $\mathbf{M}^{\mathrm{P}}(t)$ and $\mathbf{M}^{\mathrm{S}}(t)$.

Correlated Matching: The hopes

We hope the procedure generates and guarantees the following effects:

1. There is no built-in privilege that favours the matching preferences of P- or S-agents.
2. More attractive matches are more often.

Learning & moral transformation of agents in the GD scenario

Learning = Modifications of the propensities in the decision vector.:

$$D_i^{PM}(t)=\left\langle p_i^{P\_market}(t), p_i^{S\_market}(t), p_i^{reward\_local}(t), p_i^{reward\_market}(t)\right\rangle$$

There are many plausible mechanisms:

1. Reinforcement learning (of all sorts).
2. Social comparisons (of all sorts).

A first implemented version:

Agents learn in their local neighbourhood from a *role model*, i.e. the agent with the highest cumulated and discounted payoff according

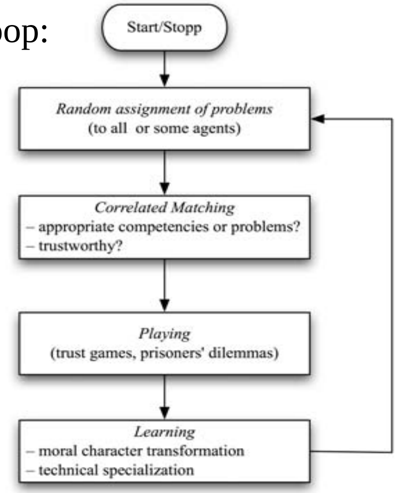
$$\Pi_i(t)=\pi_i(t)+\gamma\cdot\Pi_i(t-1)$$

actual payoff

discounts the past

With a certain probability a learning agent copies the role model's propensity into the own decision vector. There is always some mutation–up and down.

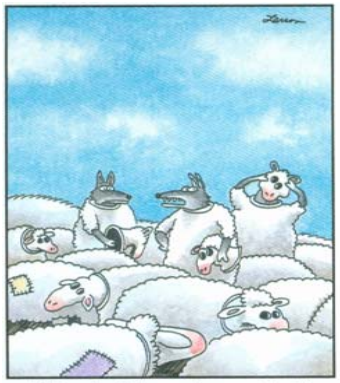
HUME$_{1.0}$ –The central loop:



The general hopes are...

1. ... to understand societal evolution – from living in small groups to living in very large groups – based on a specified set of parameters and explicitly formulated assumptions about relations between them.
2. ... to identify in a high dimensional parameter space those areas that especially further or hinder the evolution of trust and cooperation among strangers.
3. ... to find out when decentralized moral control does not suffice and more central monitoring, enforcing and punishing agencies are necessary.

... and to understand why we did NOT end up here:



"Wait a minute! Isn't anyone here a real sheep?"