Multiagent Systems (SMA-UPC)

4. Multiagent Systems Design

Part 3:
Coordination models (I):
Social Models

Introduction to Coordination.
Trust and Reputation

Javier Vázquez-Salceda
SMA-UPC

Knowledge Engineering and Machine Learning Group
UNIVERSITAT POLITÈCNICA DE CATALUNYA
https://kemlg.upc.edu

Multiagent Systems (SMA-UPC)

Introduction to Coordination Models

- Coordination in MAS
- Types of Coordination
- Coordination Structures
- Social Models for Coordination

Knowledge Engineering and Machine Learning Group
UNIVERSITAT POLITÈCNICA DE CATALUNYA
https://kemlg.upc.edu

Coordination

- Wooldridge and Jennings define an *Agent* as a computer program capable of taking its own decisions with no external control (*autonomy*), based on its perceptions of the environment and the objectives it aims to satisfy. An agent may take actions in response to changes in the environment (*reactivity*) and also it may take initiatives (*proactivity*).
- A further attribute of agents is their ability to communicate with other agents (*social ability*), not only to share information but, more important, to **coordinate actions** in order to achieve goals for which agents do not have plans they can fulfil on their own, solving even more complex problems.

Coordination

- **Coordination** is a desired property in a Multiagent System whose agents should perform complex tasks in a shared environment
- The **degree of coordination** in a Multiagent System depends on:
  - The inability of each individual agent to achieve the whole task(s)
  - The dependency of one agent on others to achieve the tasks
  - The need to reduce/optimize resource usage
  - The need to avoid system halts
  - The need to keep some conditions holding

Coordination
Definitions

- **Coordination** could be defined as the process of managing dependencies between activities. By such process an agent reasons about its local actions and the foreseen actions that other agents may perform, with the aim to make the community to behave in a coherent manner.
- An **activity** is a set of potential operations an **actor** (enacing a role) can perform, with a given goal or set of goals.
- An **actor** can be an agent or an agent group
- A set of **activities** and an ordering among them is a **procedure**.

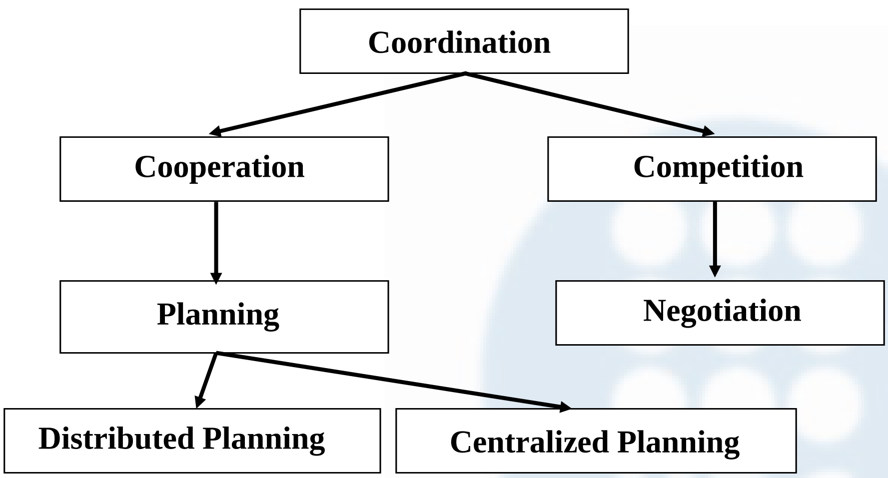
Coordination

- Coordination is a must-have functionality in any Multiagent System implementation
- Coordination becomes critical when agents are **heterogeneous** and **autonomous**
- Coordination consists of a set of mechanisms necessary for the effective operation of a MAS in order to get a well-balanced division of labour (*task allocation techniques*) while reducing logical coupling and resource dependencies of agents.

Coordination
Coordination Theory

- Lots of empirical and theoretical work has been and is currently being done to study coordination, not only for specific domains but in a more generic, domain-independent view.
- Some of this work lead to the creation of coordination theories.
- A **Coordination Theory** can be defined as a set of axioms and the analytical techniques used to create a model of dependency management.
- Examples of coordination theories are
  - *joint-intentions* theory,
  - theories about shared plans
  - domain-independent teamwork models

Coordination
Types of coordination



4. Multiagent Systems Design

Types of Coordination
Cooperation and Planning

- **Cooperation** is a kind of coordination between agents that, in principle, are not antagonist.
- The degree of success in cooperation can be measured by
  - the capability of agents to keep their own goals
  - the capability to allow other agents to reach their goals.
- **Planning** is one of the strongest forms of cooperation
  - There are some shared goals and shared plan
  - Agents allocate tasks among them following the plan

4. Multiagent Systems Design

Types of Coordination
Competition and Negotiation

- **Competition** is kind of coordination between antagonist agents which compete with each other or that are selfish.
- We will be more interested in **Negotiation**, as it is a kind of competition that involves some higher level of intelligence.
- The degree of success in negotiation (for a given agent) can be measured by
  - The capability of this agent to maximize its own benefit
  - The capability of not taking into account the other agents' benefit or even trying to minimize other agents' benefit.

Coordination Structures
Centralised Coordination (I)

- One way to tame the complexity of building a MAS is to create a centralized controller, that is, a specific agent that ensures coordination.
- *Coordinator agents* are agents which have some kind of control on other agents' goals or, at least, on part of the work assigned to an agent, according to the knowledge about the capabilities of each agent that is under the *Coordinator Agent*'s command.
- From the developer's point of view, this approach reduces complexity in MAS building:
  - the ultimate goal of the system is ensured by the goals of the coordinator, which supersedes the goals of the other agents in the system.

Coordination Structures
Centralised Coordination (II)

- Even though these kind of multi-agent architectures are easier to build, the main disadvantages of this approach come from its *centralized control*:
  - the *Coordinator agent* becomes a critical piece of the system, which depends on the reliability of a single agent and the communication lines that connect to it.
  - In the worst case scenario when the *Coordinator Agent* collapses (e.g., it receives more requests and messages than it is able to manage in a given time span), the system may also completely collapse.
  - the other agents have a severe loss of *autonomy*, as the proper behaviour of the systems depends on the agents blindly accepting the commands of the coordinator.

Coordination Structures
Distributed Coordination

- An alternative is to distribute not only the work load but also the control among all the agents in the system (*distributed control*).
- That means to *internalize* control in each agent, which has now to be provided with reasoning and social abilities to make it able to reason about intentions and knowledge of other agents plus the global goal of the society in order to be able to successfully coordinate with others and also resolve conflicts once they arise.
- However, as Moses and Tennenholtz state, in domains where the cost of a conflict is dear, or if conflict resolution is difficult, completely independent behaviour becomes unreasonable.
- Therefore some kind of structure should be defined in order to ease coordination in a *distributed control* scenario.

Coordination
Social Models for Coordination

- One source for inspiration to solve coordination problems are human **societies**
- Sociology is the branch of sciences that studies the interelationships between the individuals and the society
- *Organizational Theory* is a specific area in the middle of Sociology and Economics that studies the way relationships can be structured in human organizations (a specific kind of society)
- There are several social abstractions that have been introduced in Multiagent Systems
  - *Trust* and *Reputation*
  - *Social Structures and Social Roles*
  - *Electronic Organizations*. *Virtual Organizations*
  - *Electronic Institutions*

Trust and Reputation

- **Trust**
- **Trust VS Reputation**
- **Types of Reputation**
- **Examples of Trust/Reputation models**
- **Uses for Trust and Reputation**

Knowledge Engineering and Machine Learning Group
UNIVERSITAT POLITÈCNICA DE CATALUNYA
https://kemlg.upc.edu

What is Trust?

- It depends on the level we apply it:
  - User confidence
    - Can we trust the user behind the agent?
      - Is he/she a trustworthy source of some kind of knowledge? (e.g. an expert in a field)
      - Does he/she acts in the agent system (through his agents in a trustworthy way?
  - Trust of users in agents
    - Issues of autonomy: the more autonomy, less trust
    - How to create trust?
      - Reliability testing for agents
      - Formal methods for open MAS
      - Security and verifiability
  - Trust of agents in agents
    - Reputation mechanisms
    - Contracts
    - Norms and Social Structures

What is Trust?

- We will focus mainly in the Trust of agents in agents
- Def: Gambetta defines *trust* as *a particular level of subjective probability with which an agent a_j will perform a particular action both before [we] can monitor such action … and in a context in which it affects [our] own action.*
- Trust is subjective and contingent on the uncertainty of future outcome (as a result of trusting).

Why Trust? (I)

- In closed environments, cooperation among agents is included as part of the designing process:
  - the multi-agent system is usually built by a single developer or a single team of developers, and the chosen option to reduce complexity is to ensure cooperation among the agents they build including it as an important system requirement.
  - *Benevolence assumption*: an agent a_i requesting information or a certain service from agent a_j can be sure that such agent will answer him if a_j has the capabilities and the resources needed, otherwise a_j will inform a_i that it cannot perform the action requested.
- It can be said that in closed environments **trust is implicit**.

4. Multiagent Systems Design

Why Trust? (II)

- However, in an open environment *trust* is not easy to achieve, as
  - Agents introduced by the system designer can be expected to be nice and trustworthy, but this cannot be ensured for alien agents out of the designer control
  - These *alien* agents may give incomplete or false information to other agents or betray them if such actions allow them to fulfill their individual goals.
- In such scenarios developers use to create *competitive systems* where each agent seeks to maximize its own expected utility at the expense of other agents.
- But, what if solutions can only be constructed by means of *cooperative problem solving*?
  - Agents should try to cooperate, even if there is some uncertainty about the other agent's behaviour
  - That is, to have some explicit representation of trust

4. Multiagent Systems Design

How to compute trust?

- Trust value can be assigned to an agent or to a group of agents
- Trust value is an asymmetrical function between agent `a1` and `a2`
  - `trust_val(a1,a2)` does not need to be equal to `trust_val(a2,a1)`
- Trust can be computed as
  - A binary value (1='I do trust this agent', 0='I don't trust this agent')
  - A set of qualitative values or a discrete set of numerical values (e.g. 'trust always', 'trust conditional to X', 'no trust') (e.g. '2', '1', '0', '-1', '-2')
  - A continuous numerical value (e.g. [-300..300])
  - A probability distribution
  - Degrees over underlying beliefs and intentions (cognitive approach)

4. Multiagent Systems Design

How to compute trust

- Trust values can be externally defined
  - by the system designer: the trust values are pre-defined
  - By the human user: he can introduce his trust values about the humans behind the other agents
- Trust values can be inferred from some existing representation about the interrelations between the agents
  - Communication patterns, cooperation history logs, e-mails, webpage connectivity mapping…
- Trust values can be learnt from current and past experiences
  - Increase trust value for agent a_i if behaves properly with us
  - Decrease trust value for agent a_i if it fails/defects us
- Trust values can be propagated or shared through a MAS
  - Recommender systems, Reputation mechanisms.

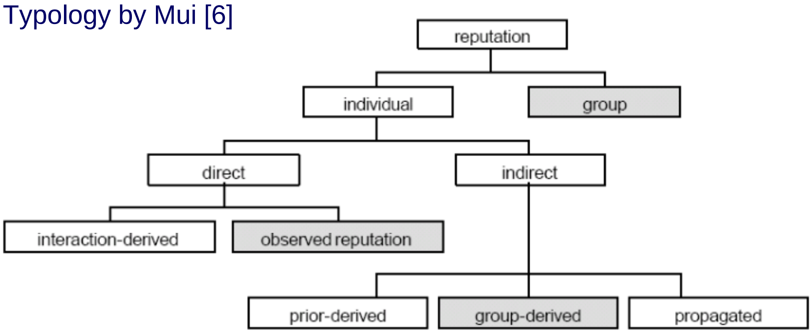
4. Multiagent Systems Design

Trust and Reputation

- Most authors in literature make a mix between trust and reputation
- Some authors make a distinction between them
  - **Trust** is an individual measure of confidence that a given agent has over other agent(s)
  - **Reputation** is a social measure of confidence that a group of agents or a society has over agents or groups
  - (social) Reputation is one mechanism to compute (individual) Trust
    - I will trust more an agent that has good reputation
    - My reputation clearly affects the amount of trust that others have towards me.
    - Reputation can have a ***sanctioning*** role in social groups: a bad reputation can be very costly to one's future transactions.
- Most authors combine (individual) Trust with some form of (social) Reputation in their models

Trust and Reputation

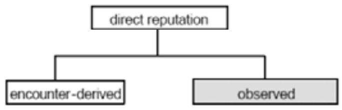
Typology by Mui [6]



- At the topmost level, reputation can be used to describe an individual or a group of individuals
- The most typical in reputation systems is the individual reputation
- Group reputation is the reputation of a set of agents
  - E.g., a team, a firm, a company
- Group reputation can help compute the reputation of an individual.
  - E.g., Mr. Anderson worked for Google Labs in Palo Alto.

Trust and Reputation

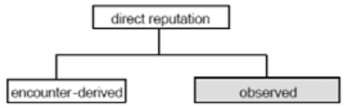
Direct experiences as source (I)



- Direct experiences are the most relevant and reliable information source for individual trust/reputation
- Type 1: Experience based on *direct interaction* with the partner
  - Used by almost all models
  - How to:
    - trust value about that partner increases with good experiences,
    - it decreases with bad ones
  - Problem: how to compute trust if there is no previous interaction?

Trust and Reputation

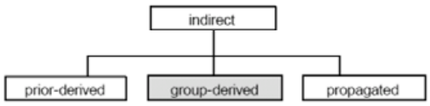
Direct experiences as source (II)



- Type 2: Experience based on *observed interaction* of other members
  - Used only in scenarios prepared for this.
  - How to: depends on what an agent can observe
    a) agents can access to the log of past interactions of other agents
    b) agents can access some feedback from agents about their past interactions (e.g., in eBay)
  - Problem: one has to introduce some noise handling or confidence level on this information

Trust and Reputation

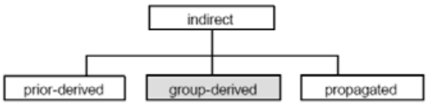
Indirect experiences as source (I)



- *Prior-derived*: agents bring with them prior beliefs about strangers
  - Used by some models to initialize trust/reputation values
  - How-to:
    a) designer or human user assigns prior values
    b) a uniform distribution for reputation priors is set
    c) give new agents the lowest possible reputation value
       - there is no incentive to throw away a cyber identity when an agent's reputation falls below a starting point.
    d) assume neither good nor bad reputation for unknown agents.
       - Avoid lowest reputation for new, valid agents as an obstacle for other agents to realise that they are valid.
  - Problem: prior beliefs are common in human societies (sexual or racial prejudices), but hard to set in software agents

Trust and Reputation

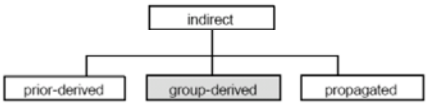
Indirect experiences as source (II)



- *Group-derived*: models for groups can been extended to provide prior reputation estimates for agents in social groups.
  - Used by some models to initialize individual trust/reputation values. See [5] as example.
  - How-to:
    - mapping between the initial individual reputation of a stranger and the group from which he or she comes from.
  - Problem: highly domain-dependant and model-dependant.

Trust and Reputation

Indirect experiences as source (III)



- *Propagated*: agent can attempt to estimate the stranger's reputation based on information garnered from others in the environment. Also called *word-of-mouth*.
  - Used by several models. See [5] as example.
  - How-to: reputation values can be exchanged (recommended) from one agent to another...
    - a) Upon request: one agent request another agent(s) to provide their estimate (a recommendation) of the stranger's reputation, then combines the results coming from these agents depending on the recommenders' reputation
    - b) Propagation mechanism: some mechanism to have a distributed reputation computation.
  - Problem: the combination of the different reputation values tends to be an ad-hoc solution with no social basis.
    - E.g. a weighted sum of a combination of the stranger agent's reputation values and the recommender agents' reputation values

Trust and Reputation
Sociological information as source

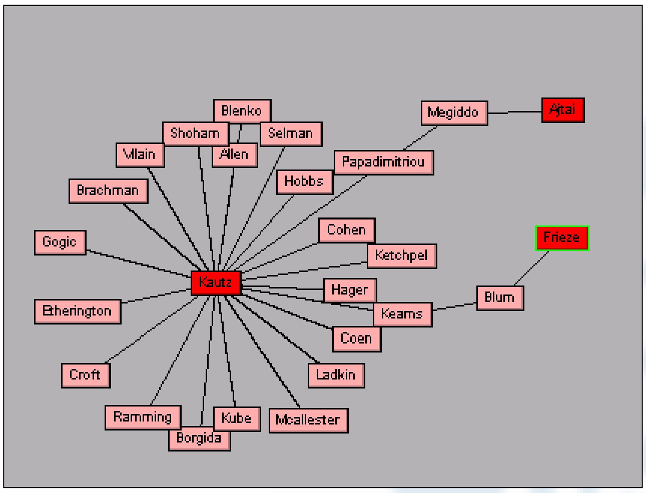
- Sabater [5] and Pujol [4] identify another source for trust/reputation: *Social relations* established between agents.
  - Used only in scenarios where there is a rich interaction between agents. See [4] as an example.
  - How-to: usually by means of *social network analysis*
    - Detect nodes (agents) in the network that are widely used as (trusted) sources of information
      - E.g. Google's page rank analyzes the topology of the network of links. Highly-linked pages get more reputation (nodes with high in-link ratios).
  - Problem: depends on the availability of relational data

Trust and Reputation models
Example 1: Kautz's Referral Web (I)

- Not really for MAS, but can be applied to MAS
- Idea: For serious life / business decisions, you want the opinion of a *trusted expert*
  - If an expert not personally known, then want to find a reference to one via a *chain* of friends and colleagues
- *Referral-chain* provides:
  - Way to judge quality of expert's advice
  - Reason for the expert to respond in a trustworthy manner
- Finding good referral-chains is slow, time-consuming, but vital
  - business gurus on "networking"
- Set of all possible referral-chains = *a social network*

Trust and Reputation models

Example 1: Kautz's Referral Web (II)



Trust and Reputation models

Example 1: Kautz's Referral Web (III)

- Model integrates information from
  - Official organizational charts (online)
  - Personal web pages (+ crawling)
  - External publication databases
  - Internal technical document databases
- Builds a social network based in referral chains
  - Each node is a recommender agent
  - Each node provides reputation values for specific areas
    - E.g. Frieze is good in mathematics
  - Searches in the referral network are made by areas
    - E.g. browsing the network's "mathematics" recommendation chains

Source
Social Network Analysis

Contextual Reputation

Trust and Reputation models

Example 2: A. Abdul-Rahman Distributed Reputation Model (I)

- General, 'common sense' model.
- Distributed: based on recommendations.
- Very useful for multiagent systems (MAS).
- Agents exchange (recommend) reputation information about other agents.
- 'Quality' of information depends on the recommender's reputation.
- 'Loose' areas
  - Trust calculation algorithm too ad hoc.
  - Lacking a concrete definition of trust for distributed systems.

Trust and Reputation models

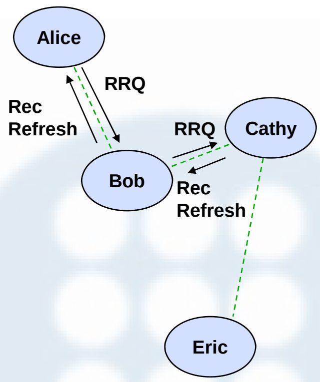
Example 2: A. Abdul-Rahman Distributed Reputation Model (II)

- Trust Model Overview
  - 1-to-1 asymmetric trust relationships.
  - Direct trust and recommender trust.
  - Trust categories and trust values [-1,0,1,2,3,4].
  - Conditional transitivity.
    - *Alice* trusts *Bob* .&. *Bob* trusts *Cathy*
      ⇒ *Alice* trusts *Cathy*
    - *Alice* trusts.rec *Bob* .&. *Bob* says *Bob* trusts *Cathy*
      ⇒ *Alice* may trust *Cathy*
    - *Alice* trusts.rec *Bob* value *X* .&. *Bob* says *Bob* trusts *Cathy* value *Y*
      ⇒ *Alice* may trust *Cathy* value f(*X*,*Y*)

Trust and Reputation models

Example 2: A. Abdul-Rahman Distributed Reputation Model (III)

- Recommendation protocol
  1. Alice → Bob: RRQ(Eric)
  2. Bob → Cathy: RRQ(Eric)
  3. Cathy → Bob: Rec(Eric,3)
  4. Bob → Alice: Rec(Eric,3)



- Refreshing recommendations
  1. Cathy → Bob: Refresh(Eric,0)
  2. Bob → Alice: Refresh(Eric,0)

Trust and Reputation models

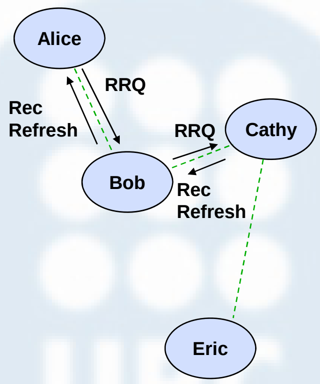
Example 2: A. Abdul-Rahman Distributed Reputation Model (IV)

- Calculating Trust (1 path)
  - $tv_p(T) = tv(R1)/4 \times tv(R2)/4 \times .. \times tv(Rn)/4 \times rtv(T)$

    trust value (for known agents)

    recommended trust value (for stranger agents)

  - ***E.g: tv_p(Eric)***
    = tv(Bob)/4 × tv(Cathy)/4 × rtv(Eric)
    = 3/4 × 2/4 × 3
    = 1.12



Trust and Reputation models

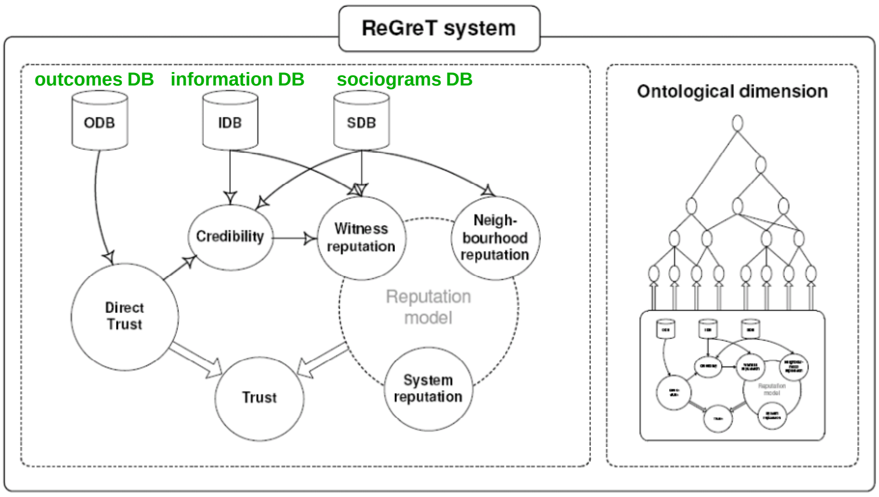
Example 2: A. Abdul-Rahman Distributed Reputation Model (V)

- *Calculating Trust – N Paths*
  - $tv(T) = Average(tv_1(T),..,tv_p(T))$

    trust values computed
    from 1 path
  - ***E.g: tv(Eric)***
    = Average(tv$_1$(T),tv$_2$(T))
    = Average(*1.12,1.75*)
    = 2.375

Trust and Reputation models

Example 3: J. Sabater's ReGreT model (I)



Trust and Reputation models

Example 3: J. Sabater's ReGreT model (II)

- The system maintains three knowledge bases:
  - the *outcomes data base* (*ODB*) to store previous contracts and their result
  - the *information data base* (*IDB*), that is used as a container for the information received from other partners
  - the *sociograms data base* (*SDB*) to store the sociograms that define the agent social view of the world.
- These data bases feed the different modules of the system.
- In the ReGreT system, each trust and reputation value computed by the modules has an associated reliability measure

Trust and Reputation models

Example 3: J. Sabater's ReGreT model (III)

- *Direct Trust:*
  - ReGreT assumes that there is no difference between *direct interaction* and *direct observation* in terms of reliability of the information. It talks about *direct experiences*.
  - The basic element to calculate a *direct trust* is the *outcome*.
  - An *outcome* of a dialog between two agents can be either:
    - An initial contract to take a particular course of action and the actual result of the actions taken, or
    - An initial contract to x the terms and conditions of a transaction and the actual values of the terms of the transaction.

Trust and Reputation models

Example 3: J. Sabater's ReGreT model (IV)

- Reputation Model: *Witness reputation* (I)
  - First step to calculate a witness reputation is to *identify the set of witnesses* that will be taken into account by the agent to perform the calculation.
  - The initial set of potential witnesses might be
    - the set of all agents that have interacted with the target agent in the past.
    - This set, however, can be very big and the information provided by its members probably suffer from the *correlated evidence problem.*
  - Next step is to aggregate these values to obtain a single value for the *witness reputation*.
  - The importance of each piece of information in the final reputation value will be proportional to the *witness credibility.*

Trust and Reputation models

Example 3: J. Sabater's ReGreT model (V)

- Reputation Model: *Witness reputation* (II)
  - Two methods to evaluate *witness credibility*:
    - ReGreT uses *fuzzy rules* to calculate how the structure of social relations influences the credibility on the information. The antecedent of each rule is the type and degree of a social relation (the edges in a sociogram) and the consequent is the credibility of the witness from the point of view of that social relation. E.g.,

$$\text{IF } coop(w_i, b) \text{ is high}$$
$$\text{THEN } socialCr(a, w_i, b) \text{ is very\_low}$$

    - The second method used in the ReGreT system to calculate the credibility of a witness is to *evaluate the accuracy of previous pieces of information* sent by that witness to the agent. The agent is using the *direct trust* value to measure the truthfulness of the information received from witnesses.
      - E.g., an agent A receives information from witness W about agent B saying agent B offers good quality products. Later on, after interacting with agent B realizes that the products that agent B is selling are horrible.

Trust and Reputation models

Example 3: J. Sabater's ReGreT model (VI)

- Reputation Model: *Neighbourhood Reputation*
  - Neighbourhood in a MAS is not related with the physical location of the agents but with the *links* created through interaction.
  - The main idea is that the behaviour of these neighbours and the kind of relation they have with the target agent can give some clues about the behaviour of the target agent.
  - To calculate a *Neighbourhood Reputation* the ReGreT system uses *fuzzy rules*.
    - The antecedents of these rules are one or several *direct trusts* associated to different behavioural aspects and the relation between the target agent and the neighbour.
    - The consequent is the value for a concrete reputation (that can be associated to the same behavioural aspect of the trust values or not).

IF $DT_{a \to n_i}(\mathit{offers\_good\_quality})$ is X AND $\mathit{coop}(b, n_i) \geqslant$ low
THEN $R_{a \xrightarrow{n_i} b}(\mathit{offers\_good\_quality})$ is X
IF $DTRL_{a \to n_i}(\mathit{offers\_good\_quality})$ is X' AND $\mathit{coop}(b, n_i)$ is Y'
THEN $RL_{a \xrightarrow{n_i} b}(\mathit{offers\_good\_quality})$ is T(X', Y')

Trust and Reputation models

Example 3: J. Sabater's ReGreT model (VII)

- Reputation Model: *System Reputation*
  - to use the common knowledge about *social groups* and the role that the agent is playing in the society as a mechanism to assign default reputations to the agents.
  - ReGreT assumes that the members of these groups have one or several *observable* features that unambiguously identify their membership.
  - Each time an agent performs an action we consider that it is playing a single role.
    - E.g. an agent can play the role of buyer and seller but when it is selling a product only the role of seller is relevant.
  - *System reputations* are calculated using a table for each social group where the rows are the roles the agent can play for that group, and the columns the behavioural aspects.

	offers_good_prices	*offers_good_quality*	*delivers_quickly*	*pays_on_time*
seller	-0.6	-0.8	-0.6	-
buyer	-	-	-	-0.6

Trust and Reputation models

Example 3: J. Sabater's ReGreT model (VIII)

- Reputation Model: *Default Reputation*
  - To the previous reputation types we have to add a fourth one, the reputation assigned to a third party agent when there is no information at all: the *default* reputation.
  - Usually this will be a fixed value

Trust and Reputation models

Example 3: J. Sabater's ReGreT model (IX)

- Reputation Model: Combining reputations
  - Each reputation type has different characteristics and there are a lot of *heuristics* that can be used to aggregate the four reputation values to obtain a single and representative reputation value.
  - In ReGreT this heuristic is based on the default and calculated reliability assigned to each type.
  - Assuming we have enough information to calculate all the reputation types, we have the stance that
    - witness reputation is the first type that should be considered, followed by
    - the *neighbourhood reputation*,
    - *system reputation*
    - the *default reputation*.
  - This ranking, however, has to be subordinated to the calculated reliability for each type.

4. Multiagent Systems Design

Trust and Reputation
Uses and Drawbacks

- Most Trust and Reputation models used in MAS are devoted to
  - *Electronic Commerce*
  - *Recommender and Collaborative Systems*
  - *Peer-to-peer file-sharing systems*
- Main criticism to Trust and Reputation research:
  - Proliferation of **ad-hoc models** weakly grounded in social theory
  - **No general, cross-domain model** for reputation
  - **Lack of integration** between models
    - Comparison between models unfeasible
    - Researchers are trying to solve this by, e.g. the ART competition

4. Multiagent Systems Design

References

[1] Wooldridge, M. "Introduction to Multiagent Systems". John Wiley and Sons, 2002.

[2] Haddadi, A. "Communication and Cooperation in Agent Systems: A Pragmatic Theory" Lecture Notes in Artificial Intelligence #1056. Springer-Verlag. 1996. ISBN 3-540-61044-8

[3] J. Vázquez Salceda. "The Role of Norms and Electronic Institutions in Multiagent Systems", Chapter 1. Birkhauser-Verlag, 2004

[4] J. M. Pujol. "Structure in Artificial Societies", Chapter 2. PhD Thesis, UPC, 2006

[5] J. Sabater I Mir. "Trust and reputation for agent societies", Chapter 2 and 4. PhD Thesis, CSIC, 2003.

[6] Mui, L. "Computational Models of Trust and Reputation: Agents, Evolutionary Games, and Social Networks", Chapter 1. PhD Thesis, Massachusets Institute of Technology, 2002.

These slides are based mainly in [3], [4], [5], [6], [2], and some material from U. Cortés