Trust in Multiagent Systems

Pınar Yolum
Email: p.yolum@uu.nl

Department of Information and Computing Sciences
Utrecht University

Working together

- Finding the right service provider from a set of providers
- Yellow-pages
  - Lookup based on service criteria
  - May not always exist
  - May return many results
- Example?

Economic service selection

- Market-oriented programming
  - Design an environment with mechanisms for buying and selling
  - Little interaction between agents; mostly for exchanging goods at different prices
  - Preferences or abilities of agents are not explicitly considered
- Consumer and producer agents
  - Self-interested
  - Maximize their utility

Prices

- The computational state is described completely by current prices for the various goods
- Communications are between each participant and the market, and only in terms of prices
- Participants reason about others and choose strategies entirely in terms of prices being bid

Functions of a Market

- Provides this information to participants
- Takes requests (buy, sell bids) from participants, enforcing rules such as bid increments and time limits
- Decides outcome based on messages from participants, considering rules such as reserve prices

Auctions

- Market where prices are determined dynamically
- Online auctions where agents participate
  - Must be fair and secure
  - Preserve privacy
- Auction types vary
  - Ascending (English) vs. Descending (Dutch)
  - Silent (auctioneer names a price; bids are silent) vs. outcry (bids name prices; auctioneer listens)
  - Hidden identity or not.
  - Combinatorial: involve bundles or sets of goods

Problems

- More applicable for services that differ only on price
- Services differ on quality?
- Negotiation of service descriptions
- Suggestions of service providers
- Semantic service selection
  - Requires deciding on which service provider will do the job best for the user
  - Take into account provider's reputation or customer's trust in the provider

Reputation

- Someone's reputation is a general opinion about that party
- By definition centralized
- Sometimes partially probed by asking others
- Computed by reputation agency
  - Authenticates users
  - Records, aggregates, and reveals ratings (e.g., amazon.com)
  - Provides the conceptual schema for
    - How to capture ratings (typically a number and text)
    - How to aggregate them
    - How to decay them over time

Problems with explicit aggregation

- Context and understanding: The contexts of usage may not be in agreement.
- Privacy: The parties providing their ratings are stating publicly (or to the reputation agency) what they may only wish to reveal in private.
- Trust: The parties using the ratings don't necessarily know where the ratings come from.

Confidence vs. trust

Luhmann's distinction:

- Hope: Wish it will come true (no basis).
- Confidence: Think it will come true (based on evidence).
- Trust: Commit to action with partly uncertain consequences.

Computational trust

1. Send request to SPx
2. Receive a service
3. Evaluate the service
4. Update the model of SPx based on evidence

Local trust

- Based on personal evidence
- Using prior interactions
- Try all others on your own
- Too costly if:
  - There are too many service providers
  - Service providers enter and leave

Institutional trust

- Organizations monitor members' actions
- Ensure a quality of service
- Centralized reputation systems
- Challenges:
  - Privacy: Raters may not want to reveal true ratings in public
  - Trust: Users of ratings don't necessarily know where the ratings come from

Social trust

- Based on evidence from others
- Information sources should be trustworthy
- Ask those you trust yourself
- Challenges:
  - Context: The contexts of usage may be unspecified
  - Satisfaction criteria: The expectations of the raters may be significantly different

Beta-Reputation System (Jøsang and Ismail, 2002)

- Collect ratings from others
- Agent counts the positive and negative ratings
- Uses a beta distribution to predict the reputation
- Assume most ratings are fair

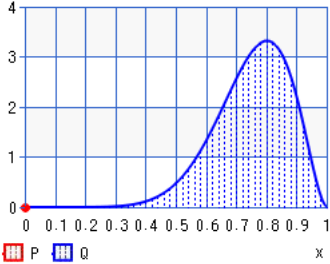
Beta-Reputation System

Estimate the probability of events

- The beta-family of probability density functions is a continuous family, indexed by α and β
- Beta Density Function is used for binary events
- Count the past occurrences and decide based on frequency
- Let r be observed occurrences of x (successful transactions) and s be the observed occurrences of $\bar{x}$ (unsuccessful transactions)
- Set $\alpha = r + 1$ and $\beta = s + 1$

Beta-Reputation System

- The probability expectation value of the beta distribution is:
  $E(p) = \alpha/(\alpha + \beta)$
- Example:
  - 8 positive transactions, 2 negative
  - α = 9; β= 3
  - E(p) = 9/12 =0.75 (Most likely frequency of the outcome)



Beta-Reputation System

- Transactions are not always binary
- Consider the extent of satisfaction (r,s), where r is how satisfied and s is how dissatisfied
- Calculate T's reputation function by agent X (p_T^X)
- Collective amount of positive and negative feedback (r_T^X, s_T^X)
  $E(p_T^X) = (r_T^X + 1)/(r_T^X + s_T^X + 2)$
- Reputation function yields a result between [0,1], where 0.5 is neutral
- Scale to give a result between [-1,1]

$$Rep(r_T^X, s_T^X) = (E(p_T^X) - 0.5) * 2$$

Combining feedback

- Combine equally from everyone
  - $r_T^{X,Y} = r_T^X + r_T^Y$
  - $s_T^{X,Y} = s_T^X + s_T^Y$
- Discount based on how much you trust; e.g., both Y and Z have opinion about T, if X trusts Y more than Z, then Y's opinion should count more.
- Using belief theory: $w_Y^X = (b_Y^X, d_Y^X, u_Y^X)$: X's opinion about Y for belief, disbelief, uncertainty.
- Given X's opinion of Y ($w_Y^X = (b_Y^X, d_Y^X, u_Y^X)$) and Y's opinion of T ($w_T^Y = (b_T^Y, d_T^Y, u_T^Y)$), how much X should trust T? ($w_T^{X:Y} = (b_T^{X:Y}, d_T^{X:Y}, u_T^{X:Y})$)

Calculating beliefs

- Combining beliefs
  - Belief: $b_T^{X:Y} = b_Y^X * b_T^Y$
  - Disbelief: $d_T^{X:Y} = b_Y^X * d_T^Y$
  - Uncertainty: $u_T^{X:Y} = d_Y^X + u_Y^X + b_Y^X * u_T^Y$
- Individual belief values
  - $b = r/(r+s+2)$
  - $d = s/(r+s+2)$
  - $u = 2/(r+s+2)$
- Mapping from belief to reputation
- Discounted reputation (r, s) = (2*b/u, 2*d/u)

Example

- Agent A has collected feedback about agent B over time and represents this as $(10, 8)$, where 10 is the positive feedback and 8 is the negative feedback. Calculate B's reputation rating by A.
- Agent A needs to find the reputation of agent C but it has not interacted with agent C before. However, agent B has interacted with agent C. B's opinion of C is $\omega_C^B = (0.7, 0.2, 0.1)$. Assuming A does not have any other information about C, calculate C's reputation rating by A. Make sure you first calculate A's opinion of B and take that into account.

Example

- Agent *A* has collected feedback about agent *B* over time and represents this as $(10, 8)$, where 10 is the positive feedback and 8 is the negative feedback. Calculate *B*'s reputation rating by *A*.
- The expected value for *B* to provide satisfactory service is: $11/20 = 0.55$. The reputation rating is (0.55-0.5)*2=0.1.

Example

- Agent A needs to find the reputation of agent C but it has not interacted with agent C before. However, agent B has interacted with agent C. B's opinion of C is $\omega_C^B = (0.7, 0.2, 0.1)$. Assuming A does not have any other information about C, calculate C's reputation rating by A. Make sure you first calculate A's opinion of B and take that into account.
- A's opinion of B, based on $(10, 8)$ is $(10/20, 8/20, 2/20)$, yielding $(0.5, 0.4, 0.1)$. Then, A's opinion of C though B is discounted as: $(0.5 * 0.7, 0.5 * 0.2, 0.4 + 0.1 + 0.5 * 0.1)$, yielding $(0.35, 0.1, 0.55)$. Converting this to $(r, s) = (2 * b/u, 2 * d/u)$ yields: $(1.27, 0.36)$. The expected value is 2.27/3.63=0.625. The reputation rating is: $(0.625 - 0.5)$*2=0.25.

Forgetting

- The recent interactions should count for more.
  - Introduce an age and multiply evidence to age the older ones
  - Need to keep track of the time-stamp
- $r^Q_T = \sum_{i=1}^{n} r^Q_{T,i}$ and $s^Q_T = \sum_{i=1}^{n} s^Q_{T,i}$
- Add forgetting factor based on i
- $r^Q_{T,\lambda} = \sum_{i=1}^{n} r^Q_{T,i} * \lambda^{(n-i)}$ and $s^Q_{T,\lambda} = \sum_{i=1}^{n} s^Q_{T,i} * \lambda^{(n-i)}$, where $0 \leq \lambda \leq 1$

Challenges

- What if there are untruthful agents?
- Which agents do you ask?
- Could evaluation of a service differ for a person?
- How could the context be captured?