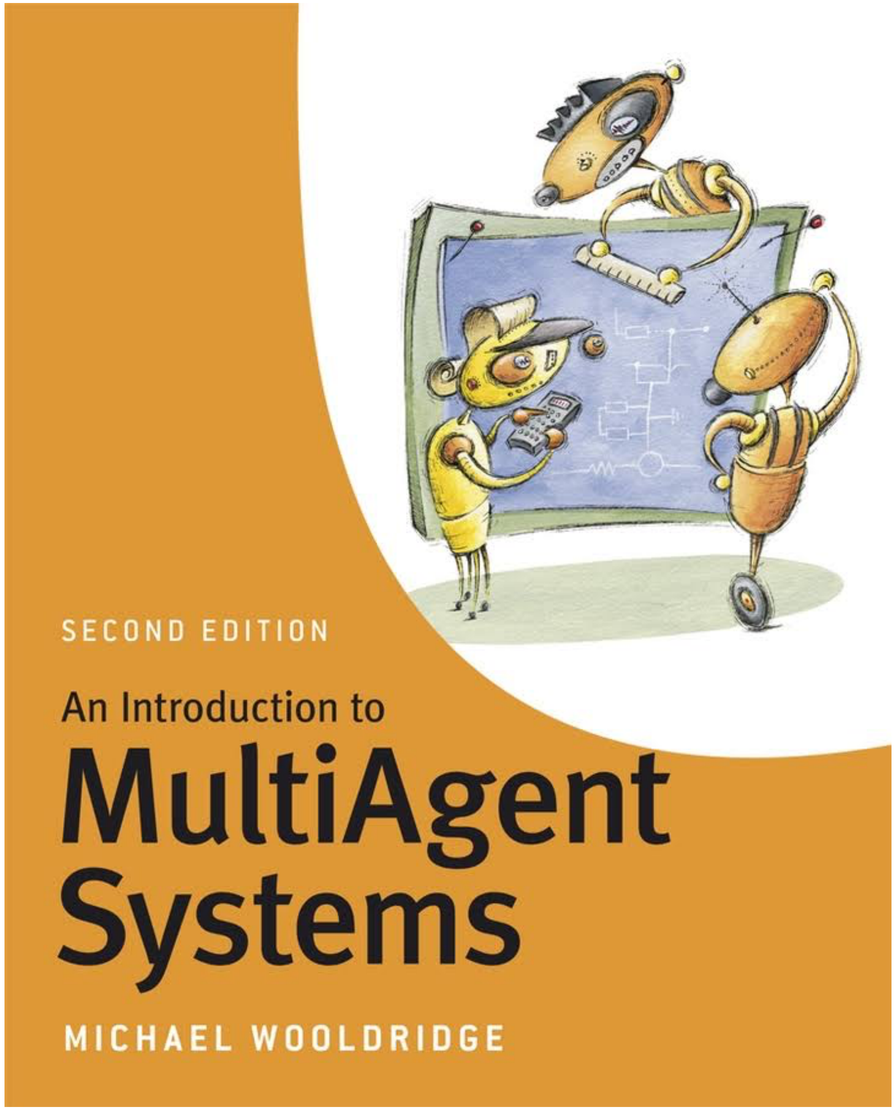
Web-Mining Agents
Rules of Encounter

Prof. Dr. Ralf Möller
Universität zu Lübeck
Institut für Informationssysteme

Tanya Braun (Übungen)

Acknowledgements to…



… and to many other lecturers who have shared their slides on the web!

Mechanisms, Protocols, and Strategies

- The mechanism defines the "rules of encounter" between agents
- *Mechanism design* is designing mechanisms so that they have certain desirable properties
- Given a particular protocol, how can a particular *strategy* be designed that individual agents can use?
- Notion of a dominant strategy
  - Best strategy can be determined w/o considering the (best) strategies of other agents

Example: Prisoner's Dilemma

Two people are arrested for a crime.

- If neither suspect confesses, both are released.
- If both confess then they get sent to jail.
- If one confesses and the other does not, then the confessor gets a light sentence and the other gets a heavy sentence.

Dom. Str. Eq

	A: Confess	A: Don't Confess
B:Confess	B=−5, A=−5	B=−1, A=−10
B:Don't Confess	B=−10, A=−1	B=−2, A=−2

Pareto Optimal Outcome

Dominant strategy exists but is not Pareto efficient

Example: Split or Steal

Does communication help?

Only if agents do not lie

	A: Steal	A: Split
B:Steal	B=0, A=0	B=100, A=-10
B:Split	B=-10, A=100	B=50, A=50

Dom. Str. Eq

Pareto Optimal Outcome

Example: Bach or Stravinsky

A couple likes going to concerts together.

- One loves Bach but not Stravinsky.
- The other loves Stravinsky but not Bach.
- However, they prefer being together than being apart.

	B	S
B	2,1	0,0
S	0,0	1,2

No dom. str. equil.

Nash Equilibrium

- Sometimes an agent's best-response depends on the strategies other agents are playing
  - No dominant strategy equilibria
- A strategy profile is a **Nash equilibrium** if no player has incentive to deviate from his strategy given that others do not deviate

	B	S
B	2,1	0,0
S	0,0	1,2

Mechanism Design

- Protocol such that agent can determine their actions

- Desirable properties of mechanisms:
  - *Convergence/guaranteed success*
  - *Maximizing social welfare*
  - *Pareto efficiency*
  - *Individual rationality*
  - *Stability*
  - *Simplicity*
  - *Distribution*

Auctions

- An auction takes place between an agent known as the *auctioneer* and a collection of agents known as the *bidders*
- The goal of the auction is for the auctioneer to allocate the *good* to one of the bidders
- In most settings the auctioneer desires to maximize the price; bidders desire to minimize price

Auction Parameters

- *Goods can have*
  - *private value*
  - *public/common value*
  - *correlated value*
- *Winner determination may be*
  - *first price*
  - *second price*
- *Bids may be*
  - *open cry*
  - *sealed bid*
- *Bidding may be*
  - *one shot*
  - *ascending*
  - *descending*

English Auctions

- Most commonly known type of auction:
  - *first price*
  - *open cry*
  - *ascending*
- Dominant strategy is for agent to successively bid a small amount more than the current highest bid until it reaches their valuation, then withdraw
- Susceptible to:
  - *winner's curse*
  - *shills*

Dutch Auctions

- Dutch auctions are examples of *open-cry descending* auctions:
  - auctioneer starts by offering good at artificially high value
  - auctioneer lowers offer price until some agent makes a bid equal to the current offer price
  - the good is then allocated to the agent that made the offer

First-Price Sealed-Bid Auctions

- First-price sealed-bid auctions are *one-shot auctions*:
  - there is a single round
  - bidders submit a sealed bid for the good
  - good is allocated to agent that made highest bid
  - winner pays price of highest bid
- Best strategy is to *bid less than true valuation*

Example: 1^{st} price sealed-bid auction

2 agents (1 and 2) with values v_1,v_2 drawn uniformly from [0,1].

Utility of agent i if it bids b_i and wins the item is $u_i=v_i-b_i$.

Assume agent 2's bidding strategy is $b_2(v_2)=v_2/2$

How should 1 bid? (i.e. what is $b_1(v_1)=z$?)

$$\boxed{U_1=\int_{x=0}^{2z}(v_1-x)dx = [v_1x-(1/2)x^2]_0^{2z} = 2zv_1-2z^2}$$

Note: given $b_2(v_2)=v_2/2$, 1 only wins if $v_2<2z$ otherwise U_1 is 0

$$\text{argmax}_z[2zv_1-2z^2] \text{ when } z=b_1(v_1)=v_1/2$$

Similar argument for agent 2, assuming $b_1(v_1)=v_1/2$. We have an equilibrium

Vickrey Auctions

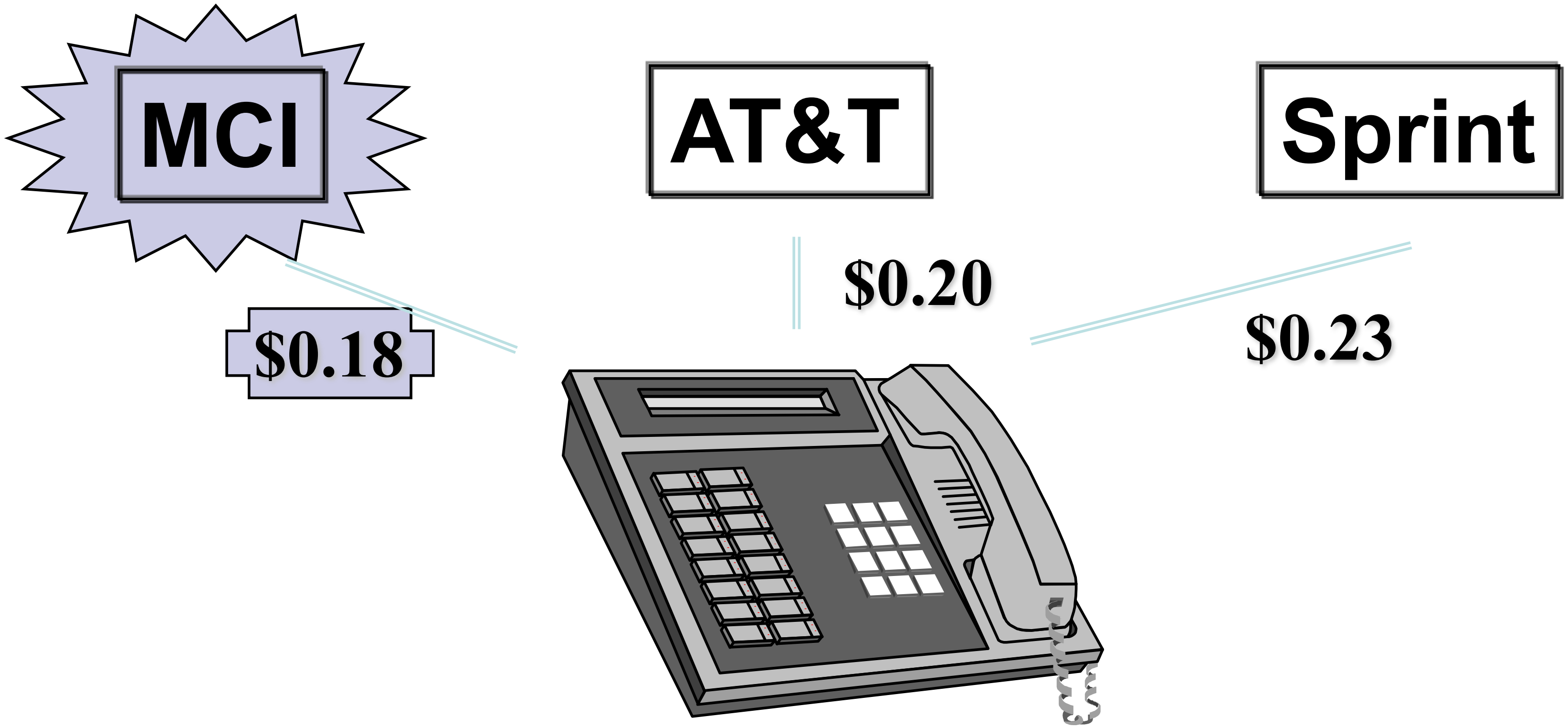
- Vickrey auctions are:
  - *second-price*
  - *sealed-bid*
- Good is awarded to the agent that made the highest bid; at the price of the *second highest* bid
- *Bidding to your true valuation is dominant strategy in Vickrey auctions*
- Vickrey auctions susceptible to *antisocial* behavior

Phone Call Competition Example

- Customer wishes to place long-distance call
- Carriers simultaneously bid, sending proposed prices
- Phone automatically chooses the carrier (dynamically)

Best Bid Wins

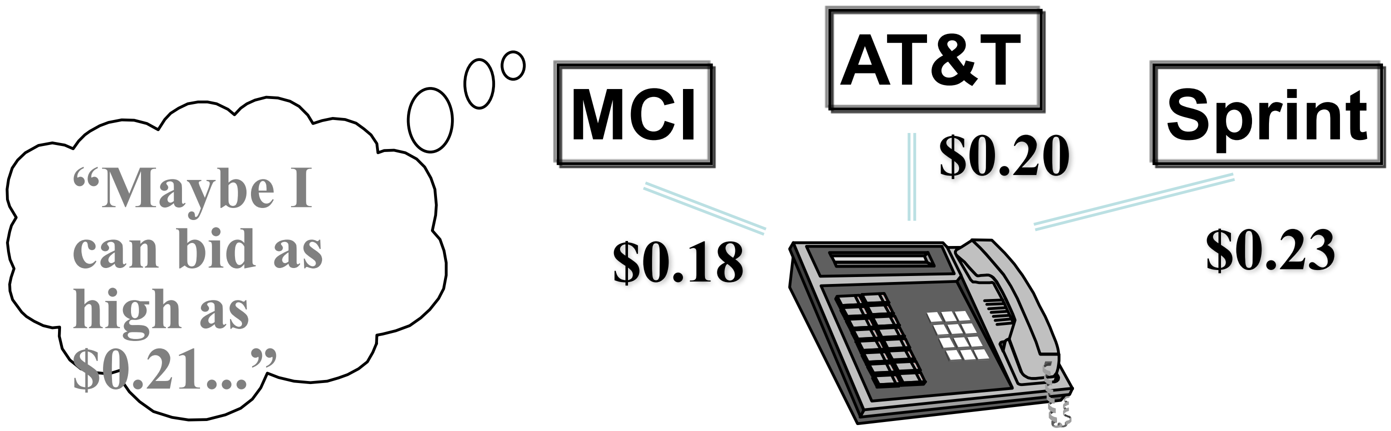
- Phone chooses carrier with lowest bid
- Carrier gets amount that it bid



Attributes of the Mechanism

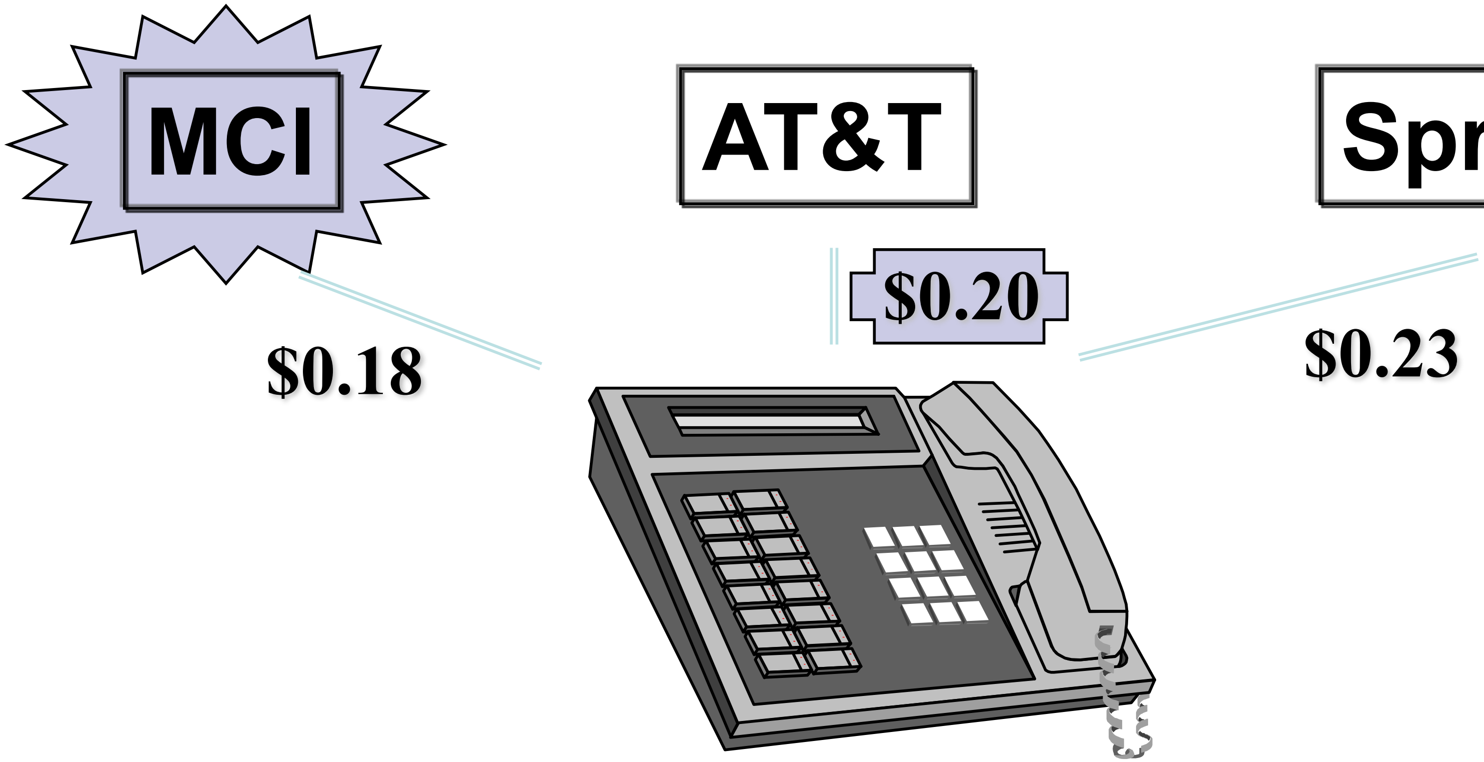
- ✓ *Distributed*
- ✓ *Symmetric*
- ✗ *Stable*
- ✗ *Simple*
- ✗ *Efficient*

Carriers have an incentive to invest effort in strategic behavior



Best Bid Wins, Gets Second Price (Vickrey Auction)

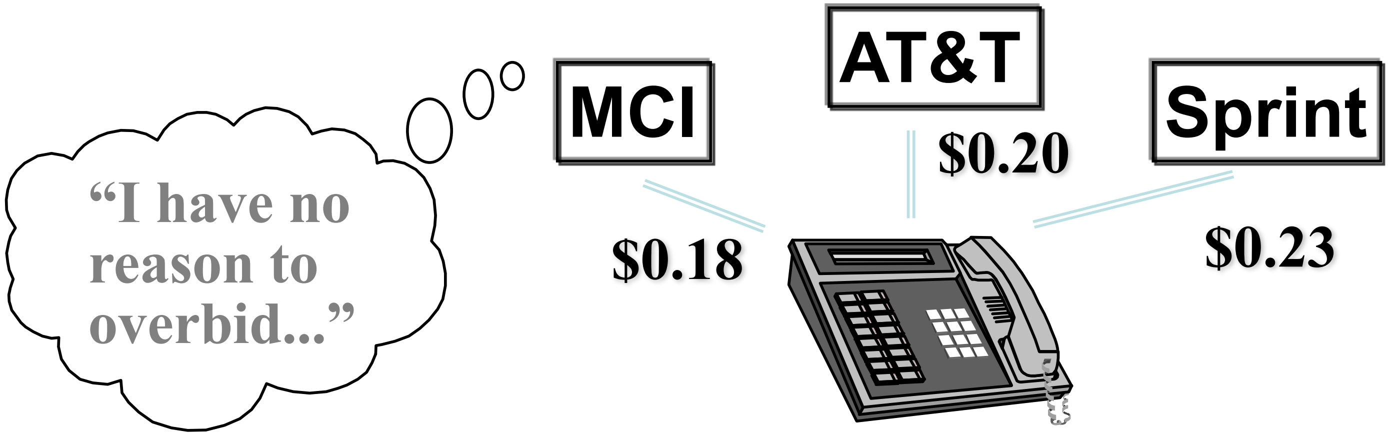
- Phone chooses carrier with lowest bid
- Carrier gets amount of second-best price



Attributes of the Vickrey Mechanism

- ✓ *Distributed*
- ✓ *Symmetric*
- ✓ *Stable*
- ✓ *Simple*
- ✓ *Efficient*

Carriers have *no* incentive to invest effort in strategic behavior



Lies and Collusion

- The various auction protocols are susceptible to lying on the part of the auctioneer, and collusion among bidders, to varying degrees
- All four auctions (English, Dutch, First-Price Sealed Bid, Vickrey) can be manipulated by bidder collusion
- A dishonest auctioneer can exploit the Vickrey auction by lying about the 2nd-highest bid
- *Shills* can be introduced to inflate bidding prices in English auctions

Negotiation

- Auctions are *only* concerned with the allocation of goods: richer techniques for reaching agreements are required
- *Negotiation* is the process of reaching agreements on matters of common interest

Bargaining, Mechanims, Strategies, Deals

- Negotiations can involve
  - Exchange of information
  - Relaxation of initial goals
  - Mutual concession
- Negotiations governed by mechanism (or protocol)
  - Rules of encounter between the agents
    - Public rules by which the agents will come to agreements
    - Stategies that agents should use
  - Deals that can be made
  - Sequence of offers and counter-offers that can be made

Negotiation in Applications

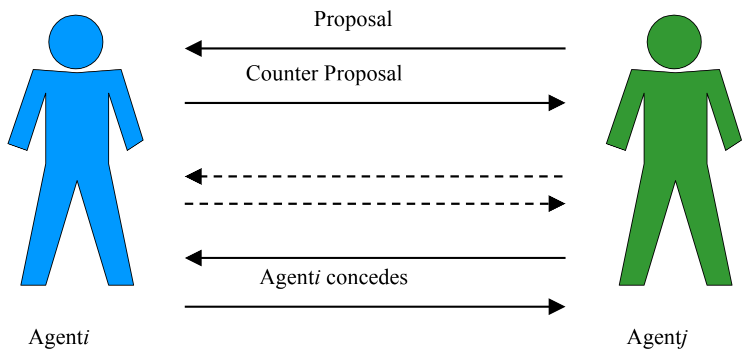
- Task-oriented domains (TOD)
  - Each agent is associated with a set of tasks (e.g., web mining tasks)
  - Goal: redistribute tasks such that costs of completing the tasks is reduced/minimized
- State-oriented domains (SOD $\supseteq$ TOD)
  - Each agent has a set of goal states it would like to achieve
  - Use negotiation to achieve a common goal (actions can have positive or negative side effects)
- Worth-oriented domains (WOD $\supseteq$ SOD)
  - Agents assign worth to state (agent-local utility)
  - Goal: maximize mutual worth / compromise on goals

How many agents?

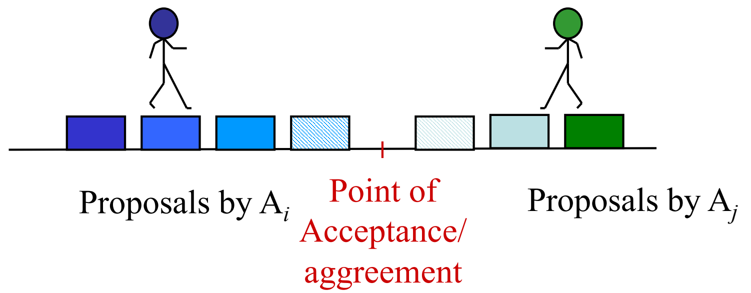
- One to one
- One to many (auction is an example of one seller and many buyers)
- Many to many (could be divided into buyers and sellers, or all could be identical in role – like officemate)
  - n(n-1)/2 number of pairs

Negotiation Process

- Negotiation usually proceeds in a series of rounds, with every agent making a proposal at every round.
- Communication during negotiation:



- Another way of looking at the negotiation process: Who "moves" the farthest



Types of deals

- Conflict deal: keep the same tasks as had originally
- Pure – divide up tasks
- Mixed – we divide up the tasks, but we decide probabilistically who should do what
- All or Nothing (A/N) - Mixed deal, with added requirement that we only have all or nothing deals (one of the tasks sets is empty)

TOD Examples

- Parcel Delivery
  - Several couriers have to deliver sets of parcels to different cities.
  - Target of negotiation is to reallocate deliveries so that the cost of travel for each courier is minimal.
- Database Query Answering / Web Mining
  - Scenario 1:
    - Several agents have access to a common database / web area, and each has to carry out a set of queries
    - Target of negotiation is to arrange queries so as to maximize efficiency of database operations (Selection, Projection, Join, …)
    - E.g., "you are doing a join as part of another operation, so please save the results for me"
  - Scenario 2:
    - Several agents have to access an overlapping set of web areas
    - Agree on reallocation and share results

Negotiation Protocols

- Who begins
- Take turns
- Single or multiple issues
- Build off previous offers
- Give feedback (or not). Tell what utility is (or not)
- Obligations – requirements for later
- Privacy (not share details of offers with others)
- Allowed proposals you can make as a result of negotiation history
- Process terminates (hopefully)

Criteria of a Negotiation Protocols

- Efficiency – do not waste utility. Pareto Optimal
- Stability – no agent have incentive to deviate from dominant strategy
- Simplicity – low computational demands on agents (e.g., no counter-speculation required
  → "dominant strategy" exists)
- Distribution – no central decision maker
- Symmetry (possibly) – may not want agents to play different roles

Task-oriented domain (TOD)

- A task-oriented domain is a triple <T, Ag, c> where
  - T is the (finite) set of all possible tasks
  - $Ag = \{1,\ldots,n\}$ is the set of participating agents
  - $c = \wp(T) \rightarrow \mathbf{R}$ defines the cost of executing each subset of tasks

- Constraints on the cost function c:
  - If $T \subseteq T'$, then $c(T) \leq c\,(T')$ (monotonicity).
  - $c(\emptyset) = 0$

The case of two agents

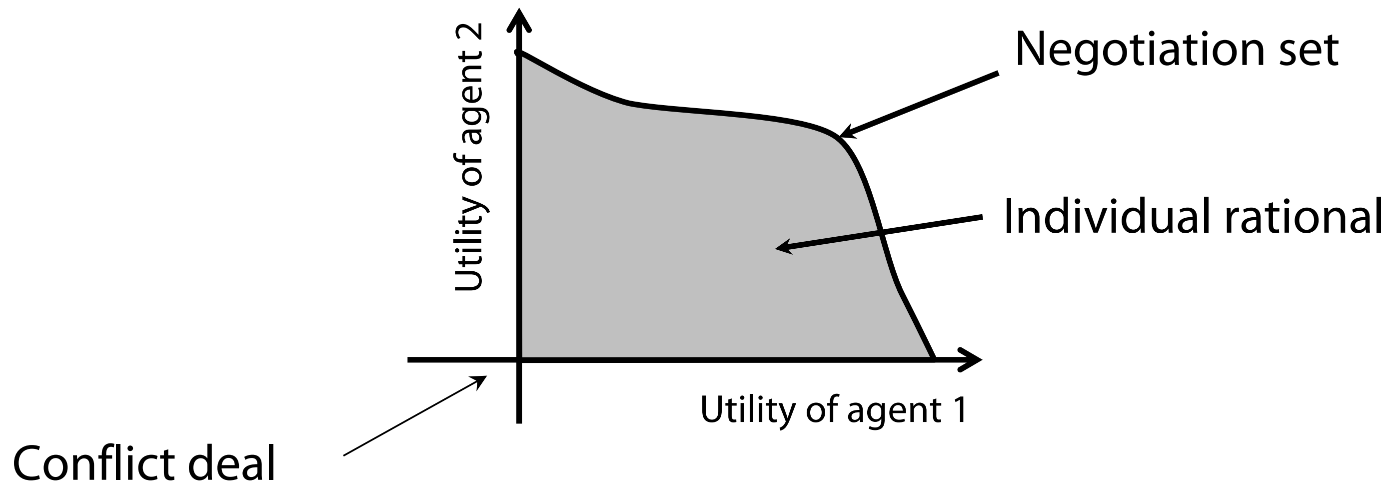
- Let (T_1, T_2) be the original tasks of two agents and let $\delta = (D_1, D_2)$ be a new task allocation (a *deal*), i.e.,
- $$T_1 \cup T_2 = D_1 \cup D_2$$

- An agent *i*'s utility of a deal δ is defined as follows:
- $$utility_i(\delta) = c(T_i) - c(D_i)$$

- δ_1 dominates δ_2 when one agent is better off and none is worse off

The negotiation set

- The **negotiation set** consists of the deals that are Pareto efficient and individual rational.
  - A deal is *Pareto efficient* if it is not dominated by another task allocation
  - A deal is *individual rational* if neither agent is worse off than in the original allocation (the 'conflict deal')

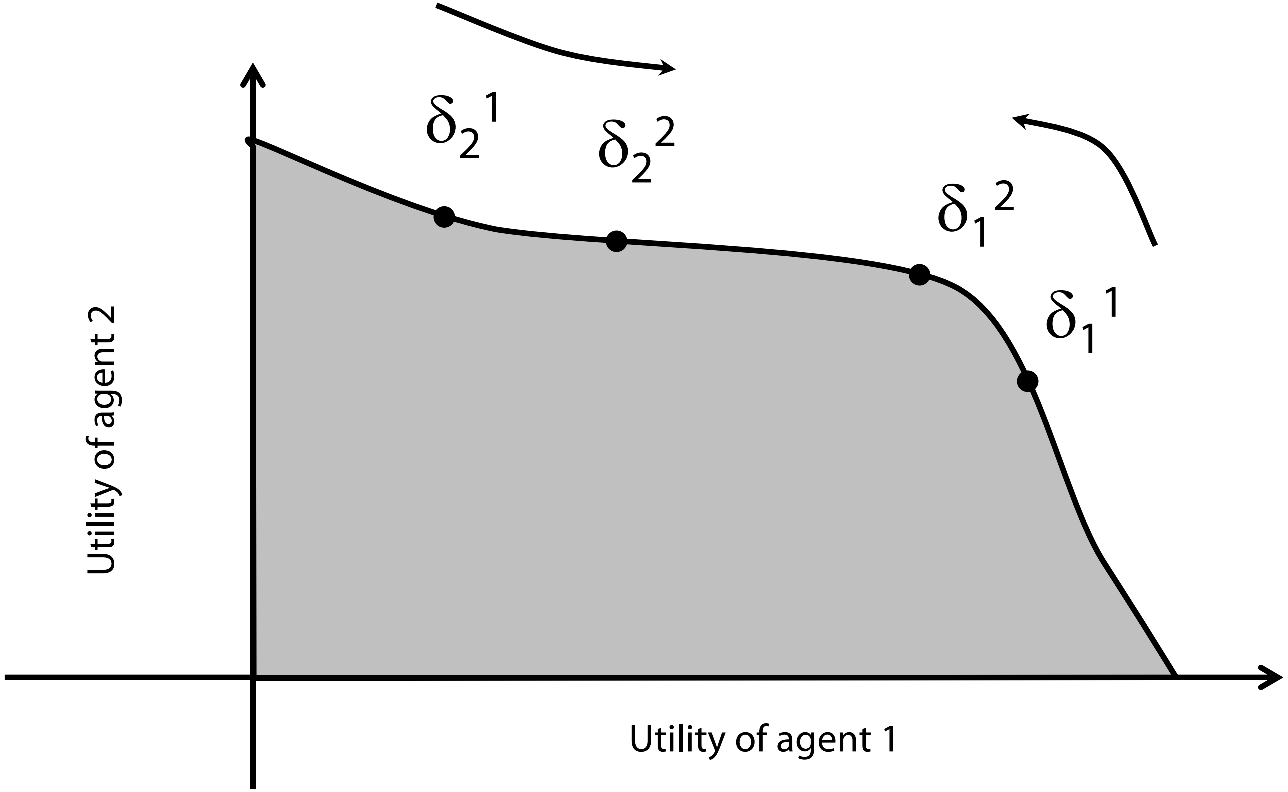


Monotonic Concession Protocol

- Both agents make several small concessions until an agreement is reached.

- Each agent proposes a deal
- If one agent matches or exceeds what the other demands, the negotiation ends
- Else, each agent makes a proposal that is equal or better for the other agent (concede)
- If no agent concedes, the negotiation ends with the conflict deal

Monotonic Concession Protocol



Monotonic Concession Protocol

- Properties
  - Termination: guaranteed if the agreement space is finite
  - Verifiability: easy to check that an opponent really concedes (only one's own utility function matters)

- Criticism
  - You need to know your opponent's utility function to be able to concede (typical assumption in game theory; not always appropriate)

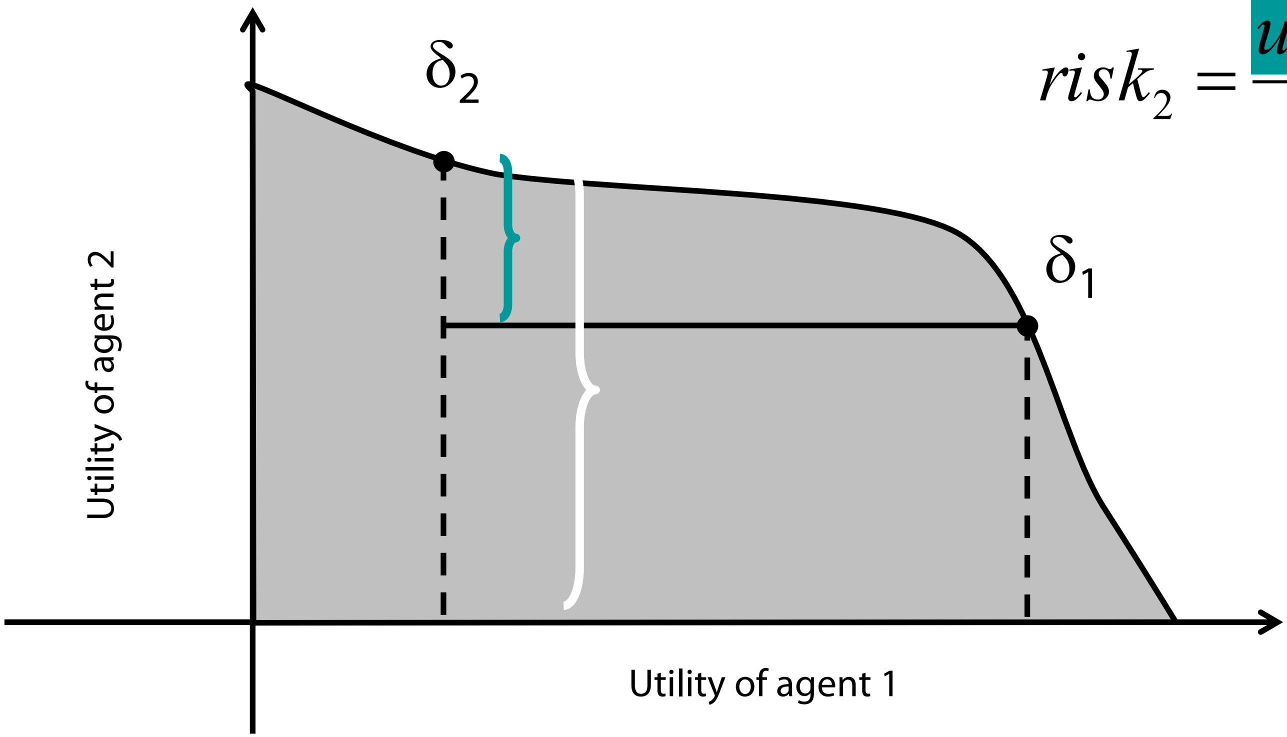
Monotonic Concession Protocol

- What is a good negotiation strategy for the Monotonic Concession Protocol?

- Consider danger of getting it wrong:
  - If you concede too often (or too much), then you risk not getting the best possible deal for yourself.
  - If you do not concede often enough, then you risk conflict (which has utility 0).

Zeuthen strategy

Idea: measure willingness to risk conflict



$$risk_2 = \frac{u_2(\delta_2) - u_2(\delta_1)}{u_2(\delta_2)}$$

Zeuthen strategy

- Start with deal that is best among all deals in the negotiation space
- Calculate willingness to risk conflict of self and opponent
- If willingness to risk conflict is smaller than opponent, offer minimal sufficient concession (a sufficient concession makes opponent's willingness to risk conflict less than yours); else offer original deal

Deception in task-oriented domains

- Deception can benefit agents in two ways:

- Phantom and decoy tasks
  - Pretending that you have been allocated tasks you have not
- Hidden tasks
  - Pretending not to have been allocated tasks that you have been

Evaluation

- The game-theoretic approach to reaching agreement has pros and cons:

- PRO: Desirable properties of protocols provable
- CON: Positions cannot be justified
- CON: Positions cannot be changed

- Alternative: Argumentation

Logic-based Argumentation

- *Database* $\vdash$ (*Sentence*, *Grounds*)

- *Database* is a (possibly inconsistent) set of logical formulae
- *Sentence* is a logical formula known as the conclusion
- *Grounds* is a set of logical formulae such that:
  - *Grounds* $\subseteq$ *Database*; and
  - *Sentence* can be proved from *Grounds*

Argument attack

- Let (C1, G1) and (C2, G2) be arguments from some database D.

- (C1, G1) <u>rebuts</u> (C2, G2) if $C1 \equiv \neg C2$
- (C1, G1) <u>undercuts</u> (C2, G2) if $C1 \equiv \neg S$ for some $S \in G2$

- Rebuttals and undercuts are known as attacks.

Abstract Argumentation

- An <u>abstract argument system</u> is a collection or arguments together with a relation "$\rightarrow$" indicating what attacks what
- Labeling:

  An argument is <u>out</u> (defeated) if (and only if) it has an undefeated attacker, and <u>in</u> (undefeated) if all its attackers are defeated
- Out-in labelings obeying this constraint do not always exist and are not always unique.

Computing labelings

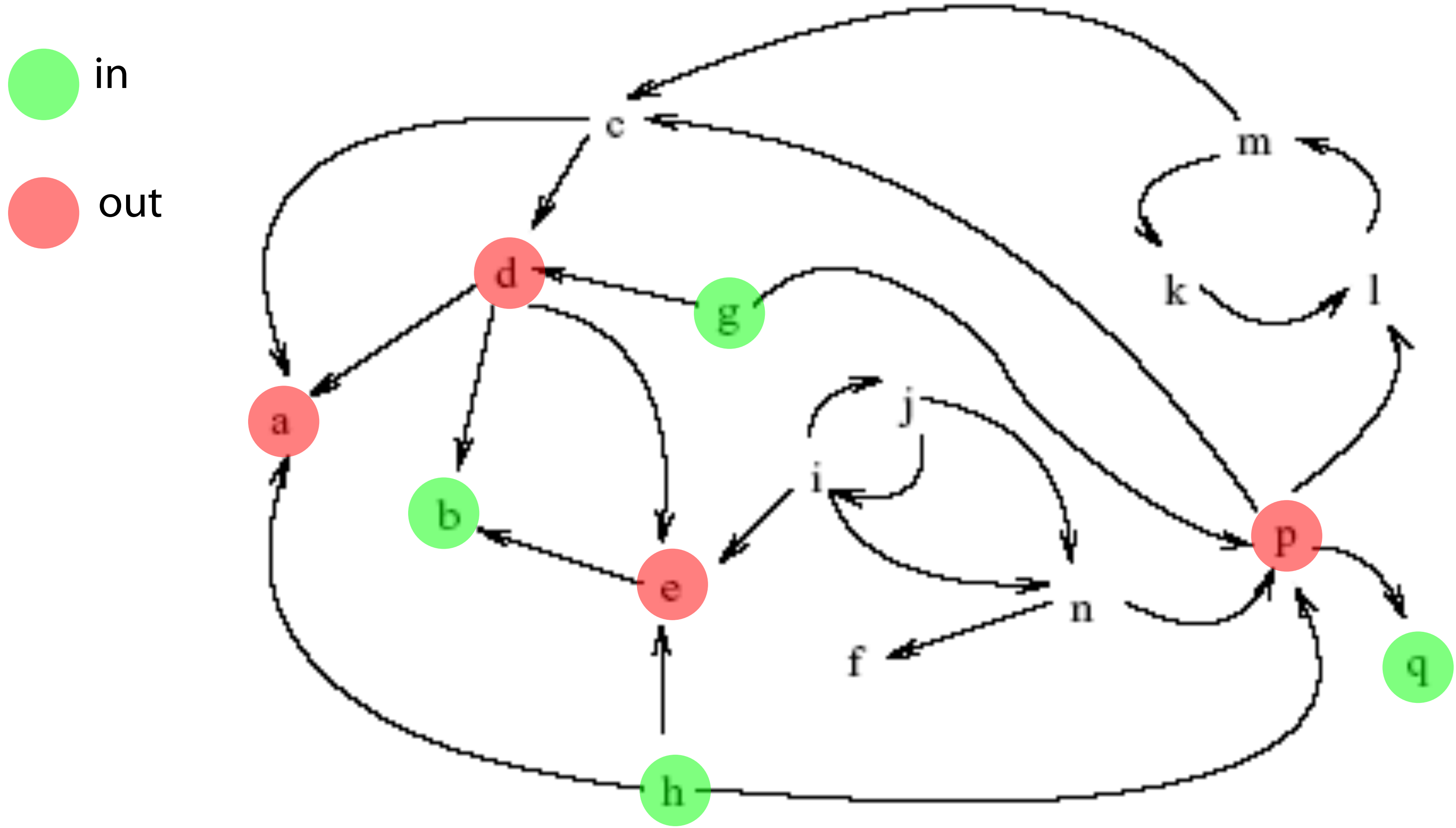
Idea for an algorithm:

1. Label all nodes that can have no <u>in</u> attacker in a complete labeling as <u>in</u>.

   (Having no attackers at all will do.)

2. Label all nodes with an <u>in</u> attacker as <u>out</u>.
3. Go to 1 if changes were made; else stop.

An Example Abstract Argument System



That's it! BTW: In this case there exists no complete labeling. (Why?)