Why Now?

1. Industries are moving online.
2. Automation: Factory floor → The back office.
3. Human-Human Negotiation is cumpersome, and inefficient.
4. Automated Negotiation opens new possiblities:
   - Too fast for people: Repeated smart contracts.
   - Too large for people: complete supply chains

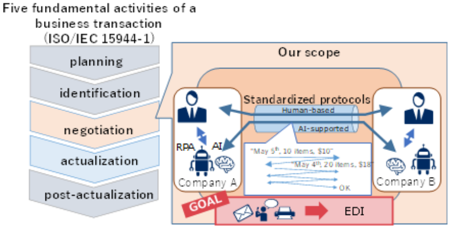
The Automated Negotiation Challenge

Why is it hard?

- Mechanism Design Problem:
  - Better than haggling?
- Negotiator Design Problem:
  - Generality $\times$ Effectiveness

Why is it interesting?

- Easy to state yet hard to solve.
- Multiple levels of abstraction and complexity.
- Several concrete open questions.
- Vibrant yet not saturated research space.



attribution: UNECE eNegotiation Project

Automated Negotiating Agents Competition: 2010-

Outline

1. Introduction and Classic Results (45min) Break (5min)
2. Protocols, Strategies and Platforms (50min)
    1. Hands On Experience

    Break (10min)
3. Learning in Negotiation (40min)
    Break (10min)
4. Supply Chain Management Competition (20min)
    1. Hands On Experience

    Break (5min)
5. Challenges and Open Problems (35min)
6. Concluding Remarks (5min)

Materials

1. Tutorial Website: http://yasserm.com/aaai2022tutorial-automated_negotiation_challenges_and_tools/
2. Github Repository: https://github.com/yasserfarouk/Aaai2022AutomatedNegotiation
3. Handouts: https://github.com/yasserfarouk/Aaai2022AutomatedNegotiation/raw/main/handouts.pdf
4. Negmas Documentation: http://www.yasserm.com/negmas
5. SCML Documentation: http://www.yasserm.com/scml/scmldocs
6. SCML Competition: https://scml.cs.brown.edu

Automated Negotiation: Challenges and Tools

Introduction and Classic Results

Yasser Mohammad [1, 2, 3] Amy Greenwald [4]

[1] NEC Corporation, Global Innovation Unit

[2]National Institute of Advanced Industrial Science and Technology (AIST), Japan

[3]Assiut University, Egypt

[4]Brown University, USA

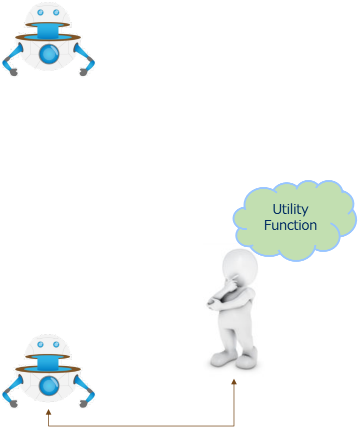
February 23rd, 2022

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory AAAI-22

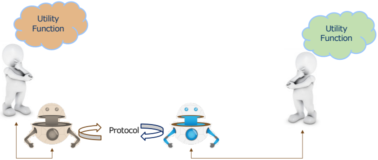
Outline

1. Negotiation
2. Bargaining 101

Outline


Utility
Function

Components of Negotiation



Negotiation Protocol Defines how negotiation is to be conducted

- Alternating Offers Protocol
- Single Text Protocol
- ...

Negotiation Strategy Defines how agents behave during a negotiation

- Time-based strategies: Boulware, conceder, ...
- Tit-for-tat variations
- ...

Dimensions of Automated Negotiation

Negotiator Type

1. Agent-agent
2. Agent-human

Outcome Space Type

1. Single Issue
2. Multiple Issues

Number of Negotiators

1. Bilateral negotiation
2. Multilateral negotiation

Protocol Type

1. Mediated
2. Unmediated

Outline

Outline

A Joint Venture

- A factory F can manufacture a widget at cost C
- A distributor D can sell a widget for revenue R
- $R > C$, so there is surplus $R - C$ to be gained (value creation)
- Bargaining problem: how much should D pay F for the widget? (value division)

Abstract Bargaining Problem

The two-person bargaining problem can be defined abstractly by

- A set $F \subset \mathbb{R}^2$ of feasible utilities (u_1, u_2)
- A disagreement point $\phi \doteq (d_1, d_2) \in \mathbb{R}^2$, also called the status quo. The value d_i is called agent i's reservation value.

Individual rationality assumption: No agent will ever agree to a utility below their disagreement point

An efficient outcome is one on the Pareto frontier, where neither agent can be made strictly better off without making the other worse off

Challenge: We seek a cooperative outcome (i.e., an efficient one) in a non-cooperative game

von Neumann-Morgenstern Utility Theorem (1947)

It is natural to express agent's preferences as comparisons: e.g., "I like apples better than bananas."

Given an agent with partial preferences with at least one strict inequality that satisfy various axioms (completeness, transitivity, continuity, and IIA),

$$\exists u : \Omega \rightarrow \Re \text{ such that } \omega \succ \psi \text{ iff } \mathbb{E}[u(\omega)] < \mathbb{E}[u(\psi)]$$

and u is unique up to scaling.

Why is this relevant?

- Justifies focusing on bargaining assuming utility functions (hereafter, ufuns).
- Justifies modelling the preferences of negotiation partners (hereafter, opponents) via ufuns.

Outline

1. Negotiation
2. Bargaining 101
   - Abstract Problem Definition
   - Bargaining Games and Solutions

Nash's Protocol (1950)

Arguably the simplest possible protocol.

- Both agents announce their demands simultaneously.
- If their demands are feasible, an agreement is reached.
- If not, the outcome is the disagreement point.

In the joint venture game (because there are now rules), we might define the agents' demands to be feasible iff the price announced by the distributor is at least the price announced by the manufacturer.

There are many possible equilibria in bargaining problems.
In the joint venture game, any price at which a trade transpires is an equilibrium.

Nash sought a theory of bargaining which would characterize a unique outcome of a negotiation.

Nash's Demand Game (1950)

The Nash demand game:

- Two agents are offered the chance to split a dollar.
- They both announce a number in $[0, 1]$ simultaneously.
- If the sum of the two numbers does not exceed 1, they each win what they demanded.
- Otherwise, they win nothing.

There are again many possible equilibria in the Nash demand game.

Define a player i's surplus as the difference between the value of an agreement and its reservation value.

Nash's bargaining solution is the point at which the product of each agent's surplus is maximized:

$$\arg\max_{\omega} \left(u_1(\omega) - u_1(\phi)\right)\left(u_2(\omega) - u_2(\phi)\right)$$

This is the unique point that satisfies several natural axioms: efficiency, symmetry, invariance, and IIA.

Automated Negotiation: Challenges and Tools

Protocols and Strategies

Yasser Mohammad [1, 2, 3] Amy Greenwald [4]

[1] NEC Corporation, Global Innovation Unit

[2] National Institute of Advanced Industrial Science and Technology (AIST), Japan

[3] Assiut University, Egypt

[4] Brown University, USA

February 23rd, 2022

BROWN UNIVERSITY

NEC-AIST AI Cooperative Research Laboratory

AAAI-22

Outline

Outline

Some Automated Negotiation Platforms

Genius [1]

a Java-based negotiation platform to develop general negotiating agents and create negotiation scenarios.

GENIUS

General Environment for Negotiation with Intelligent multi-purpose Usage Simulation.

GeniusWeb

A distributed platform for automated negotiation on the internet

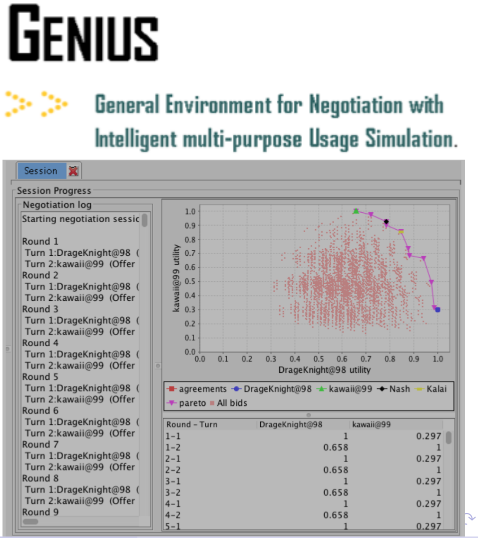
NegMAS [2]

a Python-based negotiation platform for developing autonomous negotiation agents embedded in simulation environments.

GENIUS

Genius[3]

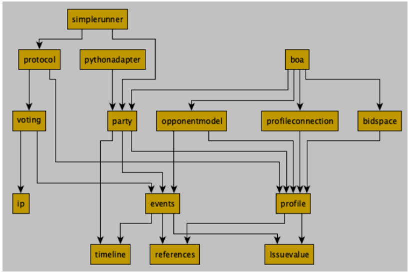
- Developed and maintained by University of Delft.
- Since 2008 and used in all ANAC competitions
- The defacto-standard
- Java-based [4]
- Has a GUI
- Includes SOTA agents from ANAC competitions since 2010.



GeniusWeb

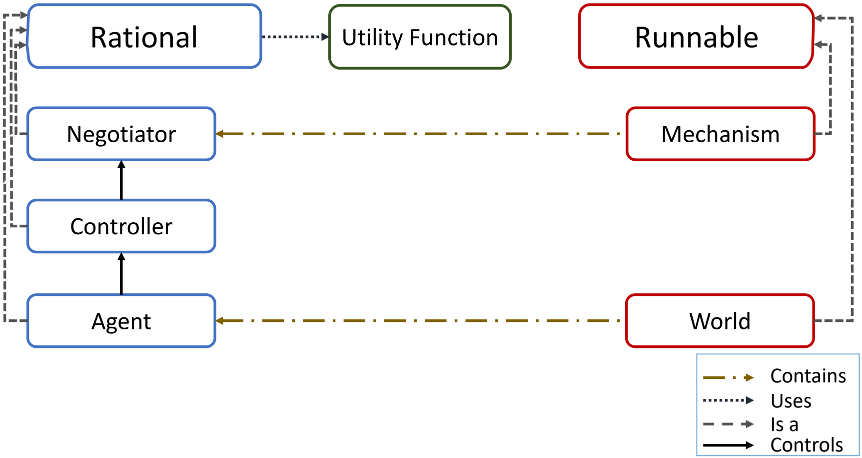
GeniusWeb [5]

- An open architecture for negotiation via the internet.
- Implemented in Java with Python support for developing agents.
- Used since ANAC 2019.
- Negtiators can be distributed over multiple machines.



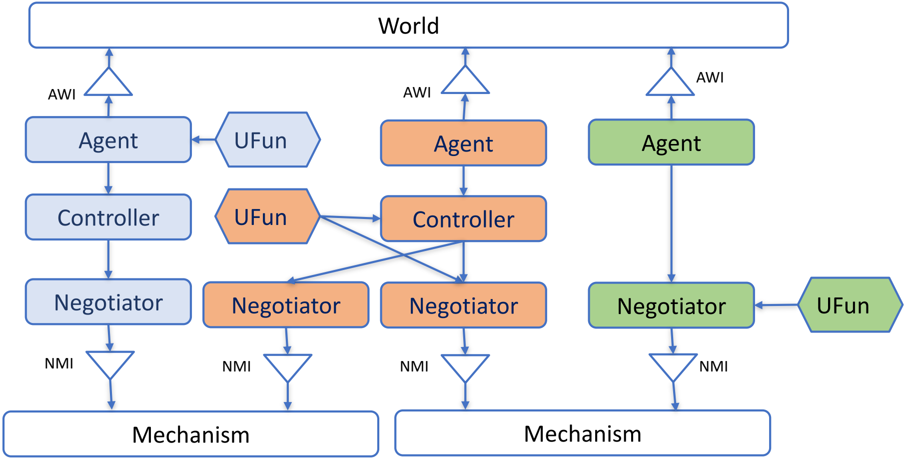
[5] GeniusWeb Team. *GeniusWeb Website*. 2021. URL: https://tracinsy.ewi.tudelft.nl/pubtrac/GeniusWeb.

NegMAS[6] in one slides



[6] https://www.github.com/yasserfarouk/negmas

NegMAS[7] in almost one slide



[7] https://www.github.com/yasserfarouk/negmas

Outline

Outline

Issues and Outcomes

Cartesian Outcome Space

The Cartesian product of a set of issues: $\Omega = I_0 \times I_1 \times \cdots \times I_{N-1}$.

Issue Types

- **Catgorical** Set of values: $\{v_i | v_i \in I\}$
- **Ordinal** with defined order
- **Cardinal** with defined difference
- **Numeric** with defined numeric value (integer/real)

[7] In NegMAS, disagreement ϕ is represented by *None*

Issues and Outcomes

Cartesian Outcome Space

The Cartesian product of a set of issues: $\Omega = I_0 \times I_1 \times \cdots \times I_{N-1}$.

Issue Types

- **Catgorical** Set of values: $\{v_i | v_i \in I\}$
- **Ordinal** with defined order
- **Cardinal** with defined difference
- **Numeric** with defined numeric value (integer/real)

NegMAS

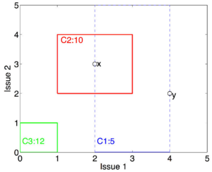
```
make_os([
    make_issue(["to be", "not to be"], "the question"),
    make_issue(10),
    make_issue((0.0, 1.0)),
    make_issue([("happy", "kitten"), ("sad", "dog")])
])
```

[7] In NegMAS, disagreement ϕ is represented by *None*

Preferences and Utility Functions

- Partial Ordering $\omega_i \succeq \omega_j \forall \omega_i, \omega_j \in \Omega$
- Full Ordering $\omega_i \succ \omega_j \forall \omega_i, \omega_j \in \Omega$
- Cardinal $\delta_{ij} = \omega_i - \omega_j \in \Re \forall \omega_i, \omega_j \in \Omega$
- Utility Function $u(\omega) \in \Re \ \forall \omega \in \Omega$
- Normalized Utility Function $u(\omega) \in [0, 1] \forall \omega \in \Omega$

- Linear UFuns $u(\omega) = \sum_{i=0}^{|\omega|} \alpha_i \times \omega_i$
- Linear Additive UFuns $u(\omega) = \sum_{i=0}^{|\omega|} \omega_i \times f_i(\omega_i)$
- Generalized Additive UFuns $u(\omega) = \sum_{i=0}^{K} \omega_k \times f_k(\omega_j \forall j \in G_k)$
- Hyper Rectangle UFuns $u(\omega) = \sum_{k=0}^{K} c_k \times \delta[\omega \in C_k]$
- Genrealized Hyper Rectangle UFuns $u(\omega) = \sum_{k=0}^{K} f_k(\omega) \times \delta[\omega \in C_k]$

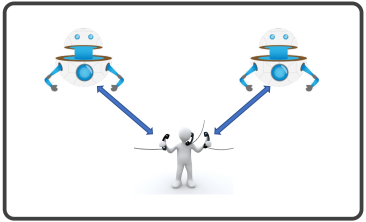


Outline

Mediated Protocols

Main Features

- Has A central *mediator*.
- Agents negotiate by exchanging *messages* with the *mediator*.
- Proposals can come from the mediator or the negotiators.

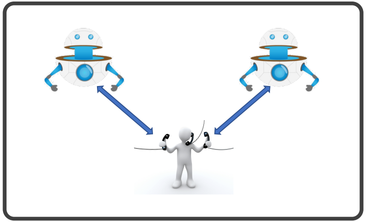


Mediated Protocols

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

Main Features

- Has A central *mediator*.
- Agents negotiate by exchanging *messages* with the *mediator*.
- Proposals can come from the mediator or the negotiators.



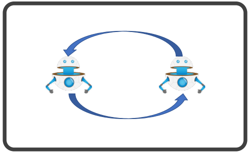
Examples

Single Text Protocol The mediator proposes a single hypothetical agreements, gets feedback about it and modifies it based on this feedback.

Unmediated Protocols

Main Features

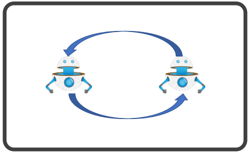
- No central coordinator.
- Agents negotiate by exchanging *messages*.
- All proposals come from negotiators.



Unmediated Protocols

Main Features

- No central coordinator.
- Agents negotiate by exchanging *messages*.
- All proposals come from negotiators.



Examples

Nash Bargaining Game Single iteration, single issue, bilateral protocol with complete information.

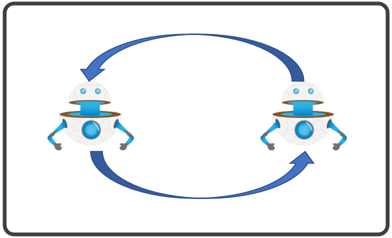
Rubinstein Bargaining Protocol Infinite horizon, single issue, bilateral protocol with complete information[a].

Alternating Offers Protocol Finite horizon, multi-issue, multilateral protocol with partial information[b].

[a] Rubinstein, "Perfect equilibrium in a bargaining model".

Stacked Alternating Offers Protocol

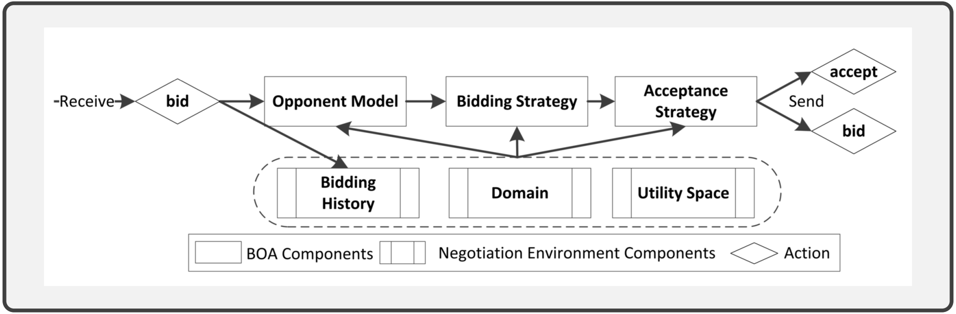
```
n_agreed, current = 0, randint(0, n_agents)
offer = agents[current].offer()
while not timeout()
        current = (current + 1) % n_agents
        response = agents[current].respond(offer)
        if response == 'accept':
                n_agreed += 1
        if n_agreed == n_agents:
                return offer
        elif response == 'end_negotiation':
                return 'failed'
        elif response == 'reject':
                offer = agents[current].offer()
return "timedout"
```



jupyter

Outline

Negotiator Components[8]



BOA Architecture

- **Opponent Model** Learns about the partner/opponent.
- **Offer Policy** Generates new bids, Also called Offer Policy
- **Acceptance Policy** Decides when to accept, Also called Acceptance Policy.

[8] Tim Baarslag et al. "Decoupling Negotiating Agents to Explore the Space of Negotiation Strategies". In: *Novel Insights in Agent-based Complex Automated Negotiation*. Ed. by Ivan Marsa-Maestre et al. Tokyo: Springer Japan, 2014, pp. 61–83. ISBN: 978-4-431-54758-7. DOI: 10.1007/978-4-431-54758-7_4. URL: https://doi.org/10.1007/978-4-431-54758-7_4.

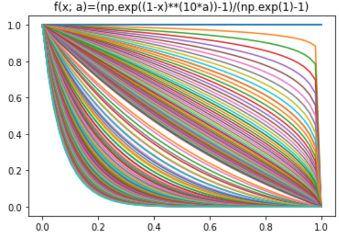
jupyter

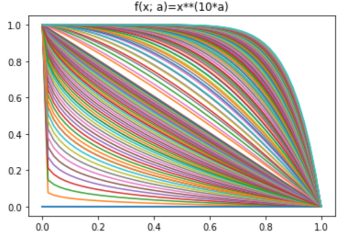
Outline

Time-based Offer Policy

Time-based strategies

- The negotiator's offers and decisions (acceptance, ending) depend **only** on the relative negotiation time.
- Monotonically decreasing utility (usually).
- Usually requires an inverse utility function.





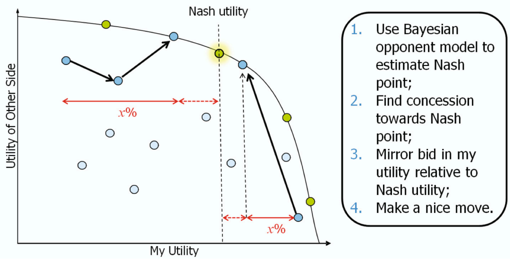
Behavioral Offer Policies

Behavior Based Strategies

- Responds to the opponent offers.
- Usualy Tit-for-Tat.
- Usually requires an opponent model.

(Nice) Tit-for-Tat (bilateral) [9]

Concede as much as the opponent toward the estimated nash-point and do not retaliate.



jupyter

Outline

Acceptance Policy

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

Acceptance Policy

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

Threshold a utility threshold (τ).

Acceptance Policy

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

Threshold a utility threshold (τ).

Constant May be a fraction of maximum utility ($AC_{const}(\gamma)$).

Acceptance Policy

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

- **Threshold** a utility threshold (τ).
  - **Constant** May be a fraction of maximum utility ($AC_{const}(\gamma)$).
  - **Time-based** Monotonically non-increasing with time ($AC_{monotonic}(t)$).

Acceptance Policy

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

- **Threshold** a utility threshold (τ).
  - **Constant** May be a fraction of maximum utility ($AC_{const}(\gamma)$).
  - **Time-based** Monotonically non-increasing with time ($AC_{monotonic}(t)$).
- **Last** my last offer (AC_{last}).

Acceptance Policy

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

- **Threshold** a utility threshold (τ).
  - **Constant** May be a fraction of maximum utility ($AC_{const}(\gamma)$).
  - **Time-based** Monotonically non-increasing with time ($AC_{monotonic}(t)$).
- **Last** my last offer (AC_{last}).
- **Next** what I am about to offer (AC_{next}).

Acceptance Policy

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

- **Threshold** a utility threshold (τ).
  - **Constant** May be a fraction of maximum utility ($AC_{const}(\gamma)$).
  - **Time-based** Monotonically non-increasing with time ($AC_{monotonic}(t)$).
- **Last** my last offer (AC_{last}).
- **Next** what I am about to offer (AC_{next}).
- **Best** the best offer I received in a given window ($AC_{best}(K)$).

Acceptance Policy

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

- **Threshold** a utility threshold (τ).
  - **Constant** May be a fraction of maximum utility ($AC_{const}(\gamma)$).
  - **Time-based** Monotonically non-increasing with time ($AC_{monotonic}(t)$).
- **Last** my last offer (AC_{last}).
- **Next** what I am about to offer (AC_{next}).
- **Best** the best offer I received in a given window ($AC_{best}(K)$).
- **Average** the average utility I received in a given window ($AC_{avg}(K)$).

Acceptance Policy

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

- **Threshold** a utility threshold (τ).
  - **Constant** May be a fraction of maximum utility ($AC_{const}(\gamma)$).
  - **Time-based** Monotonically non-increasing with time ($AC_{monotonic}(t)$).
- **Last** my last offer (AC_{last}).
- **Next** what I am about to offer (AC_{next}).
- **Best** the best offer I received in a given window ($AC_{best}(K)$).
- **Average** the average utility I received in a given window ($AC_{avg}(K)$).
- **Time** the reserved value and $T \in [0, 1]$ fraction of the negotiation have passed ($AC_{time}(T)$)
- **Expected** the best offer I expect to receive (e.g. Gaussian Processes, needs opponent offer and acceptance policies).

Acceptance Policy

Examples

Accept if $\alpha u(\omega) + \beta$ is greater than:

- **Threshold** a utility threshold (τ).
  - **Constant** May be a fraction of maximum utility ($AC_{const}(\gamma)$).
  - **Time-based** Monotonically non-increasing with time ($AC_{monotonic}(t)$).
- **Last** my last offer (AC_{last}).
- **Next** what I am about to offer (AC_{next}).
- **Best** the best offer I received in a given window ($AC_{best}(K)$).
- **Average** the average utility I received in a given window ($AC_{avg}(K)$).
- **Time** the reserved value and $T \in [0, 1]$ fraction of the negotiation have passed ($AC_{time}(T)$)
- **Expected** the best offer I expect to receive (e.g. Gaussian Processes, needs opponent offer and acceptance policies).

Combining Acceptance Policies

Combined Acceptance Strategy[10]

- Combines multiple simple acceptance policies.
- $AC_{combi}(\tau, \gamma) = AC_{next} \vee (AC_{time}(\tau) \wedge AC_{const}(\gamma))$
- $AC^{best}_{combi}(\tau, W) = AC_{next} \vee (AC_{time}(\tau) \wedge AC_{best}(W))$
- $AC^{avg}_{combi}(\tau, W) = AC_{next} \vee (AC_{time}(\tau) \wedge AC_{avg}(W))$
- $AC^{best}_{combi}(\tau) = AC_{next} \vee (AC_{time}(\tau) \wedge AC_{best}(T))$

NegMAS

```
ACCombi = ACNext() or (ACtauime(tau) and ACConst(gamma))
ACBest = ACNext() or (ACtauime(tau) and ACLastKReceived(K))
ACAvg = ACNext() or (ACtauime(tau) and ACLastKReceived(K, op=math.mean))
ACBestAll = ACNext() or (ACtauime(tau) and ACLastKReceived())
```

[10] Tim Baarslag, Koen Hindriks, and Catholijn Jonker. "Effective acceptance conditions in real-time automated negotiation". In: *Decision Support Systems* 60 (2014), pp. 68–77.

Outline

Opponent Modeling

Opponent Components

- Opponent preferences $u^o(\omega) \forall \omega \in \Omega$
- Offer Policy $\pi^o(s)$
- Acceptance Policy $a(\omega, s)$

What is being modeled?

- Any of the 3 components.
- Opponent Type.

Opponent Modeling

Opponent Components

- Opponent preferences $u^o(\omega) \forall \omega \in \Omega$
- Offer Policy $\pi^o(s)$
- Acceptance Policy $a(\omega, s)$

What is being modeled?

- Any of the 3 components.
- Opponent Type.

When is it modeled?

- Before the negotiation: Static Model.
- During the negotiation: Dynamic Model.

Opponent Modeling

Opponent Components

- Opponent preferences $u^o(\omega) \forall \omega \in \Omega$
- Offer Policy $\pi^o(s)$
- Acceptance Policy $a(\omega, s)$

What is being modeled?

- Any of the 3 components.
- Opponent Type.

When is it modeled?

- Before the negotiation: Static Model.
- During the negotiation: Dynamic Model.

Data

- This opponent vs. this opponent group vs. all opponents.
- Only agreements vs. All exchanged offers.

Outline

References I

Aydoğan, Reyhan et al. "Alternating offers protocols for multilateral negotiation". In: *Modern Approaches to Agent-based Complex Automated Negotiation*. Springer, 2017, pp. 153–167.

Baarslag, Tim, Koen Hindriks, and Catholijn Jonker. "A tit for tat negotiation strategy for real-time bilateral negotiations". In: *Complex Automated Negotiations: Theories, Models, and Software Competitions*. Springer, 2013, pp. 229–233.

– ."Effective acceptance conditions in real-time automated negotiation". In: *Decision Support Systems* 60 (2014), pp. 68–77.

Baarslag, Tim et al. "Decoupling Negotiating Agents to Explore the Space of Negotiation Strategies". In: *Novel Insights in Agent-based Complex Automated Negotiation*. Ed. by Ivan Marsa-Maestre et al. Tokyo: Springer Japan, 2014, pp. 61–83. ISBN: 978-4-431-54758-7. DOI: 10.1007/978-4-431-54758-7_4. URL: https://doi.org/10.1007/978-4-431-54758-7_4.

Lin, Raz et al. "Genius: An Integrated Environment for Supporting the Design of Generic Automated Negotiators". In: *Computational Intelligence* 30.1 (2014), pp. 48–70. ISSN: 1467-8640. DOI: 10.1111/j.1467-8640.2012.00463.x. URL: http://dx.doi.org/10.1111/j.1467-8640.2012.00463.x.

References II

Mohammad, Yasser, Amy Greenwald, and Shinji Nakadai. "NegMAS: A platform for situated negotiations". In: *Twelfth International Workshop on Agent-based Complex Automated Negotiations (ACAN2019) in conjunction with IJCAI*. Macau, China, 2019. URL: https://github.com/yasserfarouk/negmas.

Rubinstein, Ariel. "Perfect equilibrium in a bargaining model". In: *Econometrica: Journal of the Econometric Society* (1982), pp. 97–109.

Team, Genius. *Genius Website*. 2021. URL: http://ii.tudelft.nl/genius/.

Team, GeniusWeb. *GeniusWeb Website*. 2021. URL: https://tracinsy.ewi.tudelft.nl/pubtrac/GeniusWeb.

Linear UFuns $u(\omega) = \sum_{i=0}^{|\omega|} \omega_i \times \omega_i$

Linear Additive UFuns $u(\omega) = \sum_{i=0}^{|\omega|} \omega_i \times_i (\omega_i)$

Hyper Rectangle UFuns $u(\omega) = \sum_{k=0}^{K} c_k \times \delta[\omega \in C_k]$

Genrealized Hyper Rectangle UFuns $u(\omega) = \sum_{k=0}^{K} f_k(\omega) \times \delta[\omega \in C_k]$

Automated Negotiation: Challenges and Tools

Learning in Negotiation

Yasser Mohammad [1, 2, 3] Amy Greenwald [4]

[1] NEC Corporation, Global Innovation Unit

[2] National Institute of Advanced Industrial Science and Technology (AIST), Japan

[3] Assiut University, Egypt

[4] Brown University, USA

February 23rd, 2022

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory AAAI-22

Outline

Outline

Learning in Automated Negotiation

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory

What?

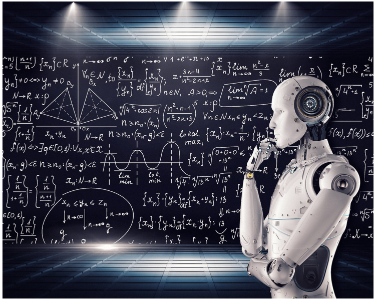
1. Acceptance Policy
2. Offering Policy
3. Opponent/Partner Model

When?

1. Within Negotiation
2. Between Negotiations

How?

1. Supervised Learning
2. Reinforcement Learning
3. Unsupervised Learning



"Artificial Intelligence & AI & Machine Learning" by mikemacmarketing

Outline

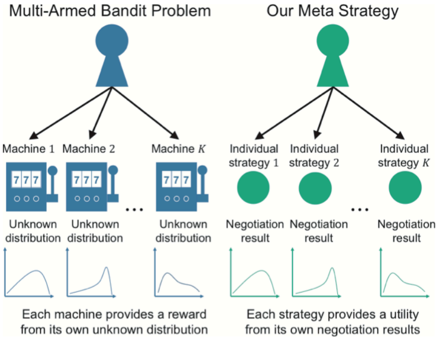
Outline

Multiaremed Bandits for Repeated Negotiations[1]

Treat sub-negotiators as bandits in a standard multi-armed bandits problem.

- Base Strategies: Atlas3, CaduceusDC16, Kawaii, ParsCat, Rubick, YXAgent
- Method:
  - After every negotiation update the corresponding $\hat{\mu}_s$.
  - Use the slot machine (negotiator) that maximizes

$$UCB(s) = \hat{\mu}_s + c\sqrt{\frac{lnN}{N_s}}$$



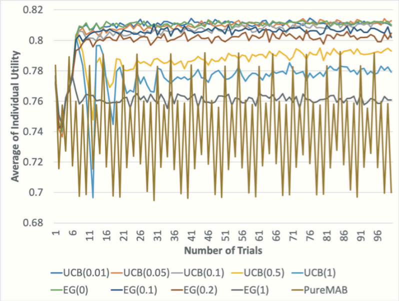
[1] Ryohei Kawata and Katsuhide Fujita. "Meta-Strategy Based on Multi-Armed Bandit Approach for Multi-Time Negotiation". In: *IEICE TRANSACTIONS on Information and Systems* 103.12 (2020), pp. 2540–2548.

Multiaremed Bandits for Repeated Negotiations[1]

Treat sub-negotiators as bandits in a standard multi-armed bandits problem.

- Base Strategies: Atlas3, CaduceusDC16, Kawaii, ParsCat, Rubick, YXAgent
- Method:
  - After every negotiation update the corresponding $\hat{\mu}_s$.
  - Use the slot machine (negotiator) that maximizes

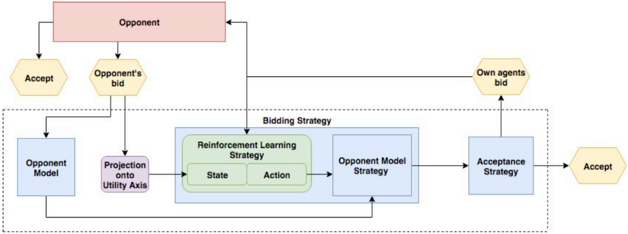
$$UCB(s) = \hat{\mu}_s + c\sqrt{\frac{\ln N}{N_s}}$$



Agent	Individual utility	Social welfare
UCB(0.01)	**0.7734**	1.4575
Agent33	0.6901	**1.4579**
AgentNP2018	0.7082	1.4362
Appaloosa	0.7067	1.3706
Ellen	0.6083	1.2223
TimeTraveler	0.7142	1.4573

[1] Kawata and Fujita, "Meta-Strategy Based on Multi-Armed Bandit Approach for Multi-Time Negotiation".

RLBOA: Learning Offering Strategy[2]



Main Points

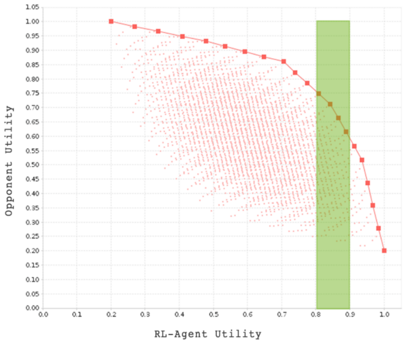
- Extends the BOA architecture.
- Learns only a bidding strategy:
  - The agent learns how to move *in its own utility axis*.

[2] Jasper Bakker et al. "RLBOA: A modular reinforcement learning framework for autonomous negotiating agents". In: *Proceedings of the 18th international conference on autonomous agents and multiagent systems*. 2019, pp. 260–268.

RLBOA: The details

- State Space: $\{\hat{u}(\omega_t^s), \hat{u}(\omega_{t-1}^s), \hat{u}(\omega_t^p), \hat{u}(\omega_{t-1}^p), t\}$.
  - $\hat{u}(\omega) = \lfloor N \times u(\omega) \rfloor^3$
- Action Space: $\leftarrow, -, \rightarrow$.
  - First step $\rightarrow i \in [0, N-1]$
  - Out-of-boundary correction: $-$.
- Training Method: Q-learning
- Acceptance Strategy [Recommended]: $AC_{next}(\alpha = 1, \beta = 0)^4$

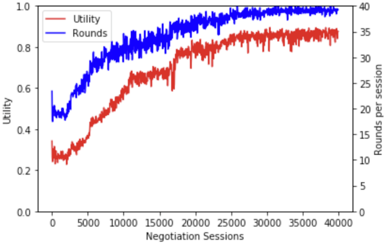
$$a(\omega) = \begin{cases} \text{Accept}, & \text{if } \alpha u(\omega) + \beta \geq u(o(s)) \\ \text{Reject}, & \text{otherwise} \end{cases}$$



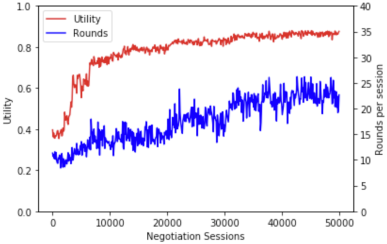
[4] Just ignore the special case at $u == 1$.

[4] Default acceptance strategy in NegMAS

RLBOA: Evaluation and Results



(a) Scenario generality experiment against the Boulware agent.



(b) Opponent generality experiment in the medium sized domain with low opposition.

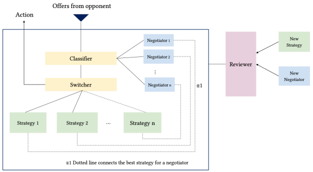
- Partners: TFT, Boulware TB
- Projection into one's utility space is surprisingly effective.
- Faster and better agreements!

Domain	Outcome space	Low opp.	High opp.
Small	256	0.2615	0.5178
Medium	3.125	0.3111	0.5444
Large	46.656	0.2595	0.5250

A Framework for Learning Offer Strategies

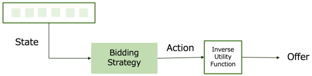
Main Idea[5]

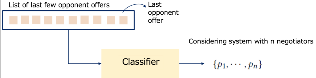
- Uses RL for learning approximate best responses to some agents.
- Uses Supervised Learning to learn a realtime switching strategy between learned best responses.
- Uses a form of Unsupervised Learning for adapting the system to new partner types.



Phases

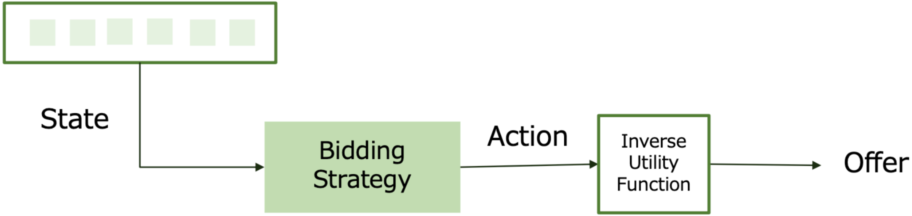
- Before Negotiation Learn approximate best responses to **a few** agents.
- During Negotiation Switch to the most appropriate **learned app. best response**
- After Negotiation Decide whether to add a new **best response**.





[5] Ayan Sengupta, Yasser Mohammad, and Shinji Nakadai. "An Autonomous Negotiating Agent Framework with Reinforcement Learning Based Strategies and Adaptive Strategy Switching

Before: Learning Approximate Best Response



$$\begin{aligned} s_t = \quad & \{t_r, U_s(\omega_s^{t-2}), U_s(\omega_o^{t-2}), U_s(\omega_s^{t-1}), \\ & U_s(\omega_o^{t-1}), U_s(\omega_s^{t}), U_s(\omega_o^{t})\} \\ a_t = \quad & u_s^{t+1} \quad \text{such that} \quad u_r < u_s \leq 1 \end{aligned}$$

The RL Component

State Self utility of last N offers plus relative time.

Action Utility of next offer $\in [0, 1]$.

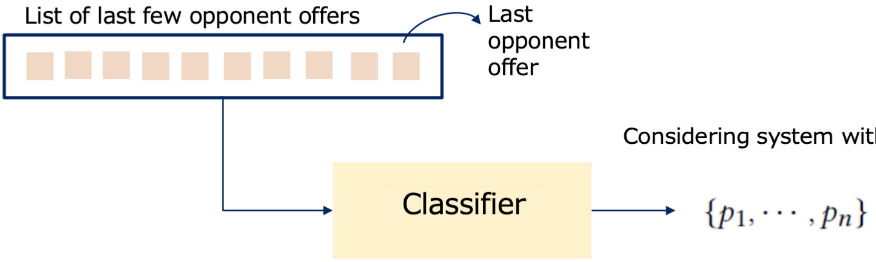
Reward Agreement/disagreement utility.

Trainer Soft Actor Critic (SAC)

$$U_s^{-1}(u_s) = \underset{\omega}{\operatorname{argmin}} f(\omega), \quad \text{where}$$

$$f(\omega) = (U_s(\omega) - u_s)^2 \quad \forall \quad \omega \in \Omega.$$

During: Learning realtime Partner Classification



The SL Components

Features Opponent last K offers.

Target Opponent Type (discrete set)

After: Reviewing New Pairs

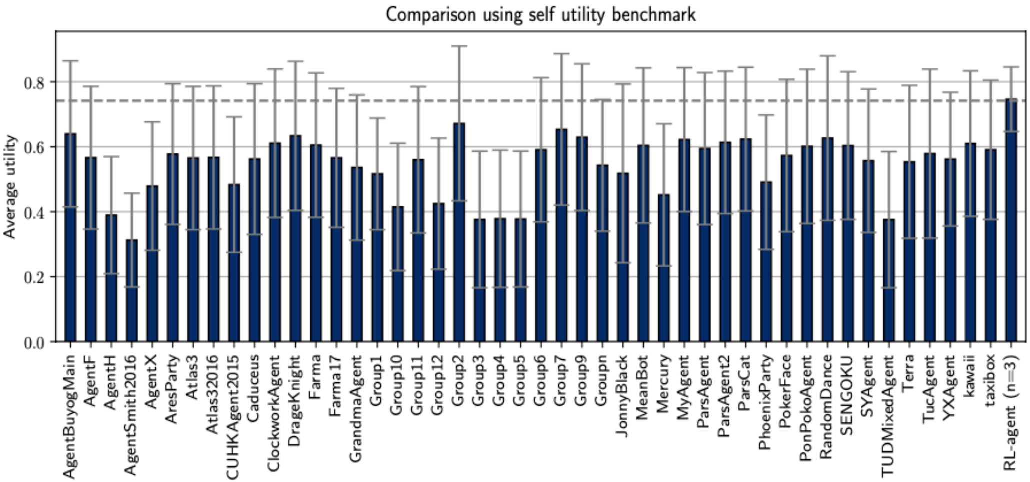
New Partner Type (N_{new}) Ecncountered

- Train a best response (using SAC) $\rightarrow S_{new}$.
- Evaluate S_{new} against $N_{new} \rightarrow U(S_{new})$
- Evaluate *Current* against $N_{new} \rightarrow U(Current)$
- Add (S_{new}, N_{new}) iff $\beta U(Current) < U(S_{new})$
- Update best responses $\downarrow$.

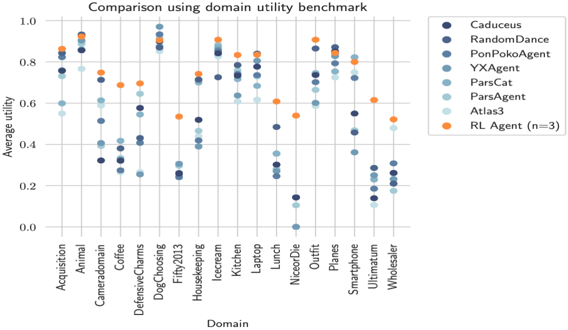
Update Best Responses

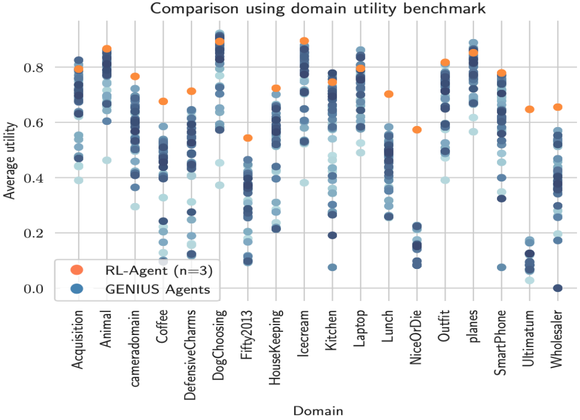
- For every learned ABR, negotiator pair (S, N):
  - Evaluate S_{new} against $N \rightarrow U(S_{new})$
  - Evaluate S against $N \rightarrow U(S)$
  - Replace S with S_{new} iff $\alpha U(S) < U(S_{new})$

Results: Against Different Opponents

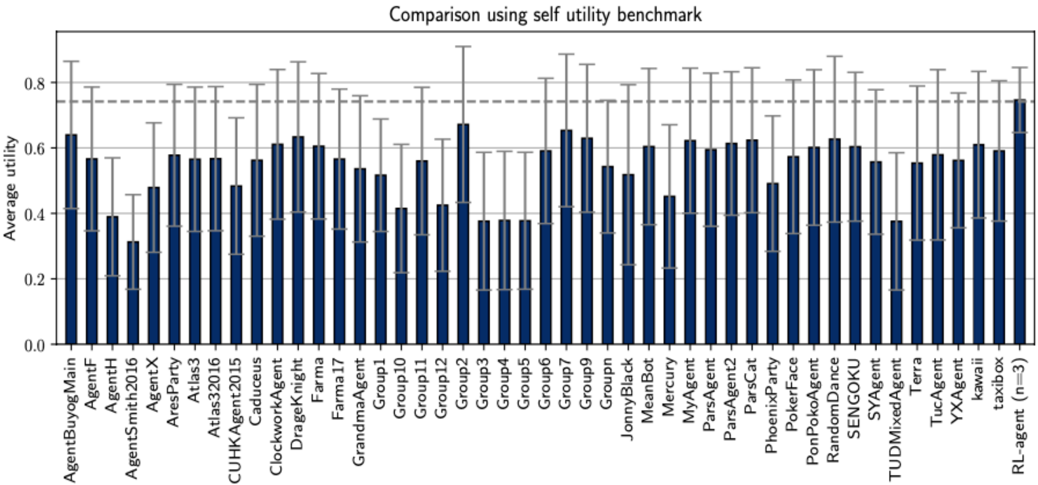


Results: In Different Domains



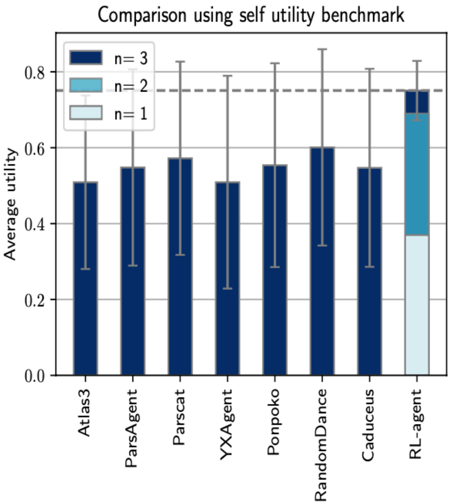


Results: Compared with SOTA Agents



Results: Improvement with new best responses

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory



Outline

DQN for learning Acceptance Strategy[6]

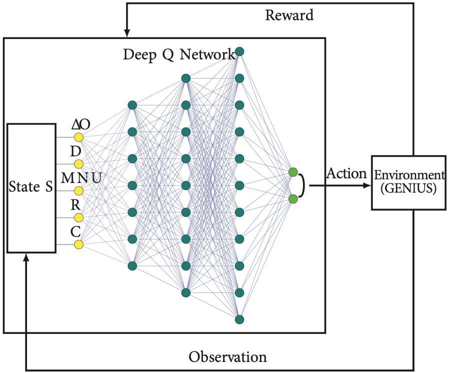
Main Idea

- Reward shaping during negotiation.

Settings

- State Space $u(\omega) - u(\phi), 1 - t, u(o(s)), u_t, u(\omega)$
  - u_t is a relatively large target utility (e.g. 0.8).
- Action Space Accept/Reject
- Reward

$$r = \begin{cases} -2^{|u_t - u_f|}, & \text{if } u_t > u(\omega_a) \\ +2^{|u_t - u_f|}, & \text{if } u_t < u(\omega_a) \\ 0 & \text{if non-terminal} \end{cases}$$



[6] Yousef Razeghi, Celal Ozan Berk Yavuz, and Reyhan Aydoğan. "Deep reinforcement learning for acceptance strategy in bilateral negotiations". In: *Turkish Journal of Electrical Engineering & Computer Sciences* 28.4 (2020), pp. 1824–1840.

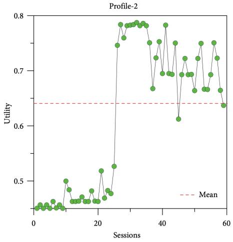
Evaluation

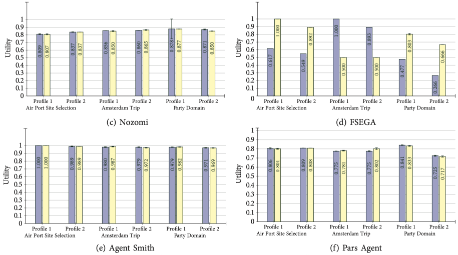
Training

- Domain England-Zimbabwe (576 outcomes)
- Partner Gahboninho
- Offering Strategy AgentK
- Opponent Model AgentLG, Not TFT.

Testing

- Domains Party (3072), Amsterdam (3024), Airport (420)
- Partners Agent Smith, Yushu, FSEGA, IAMHaggler, ParsAgent, Nozomi
- Baseline ACnext





Outline

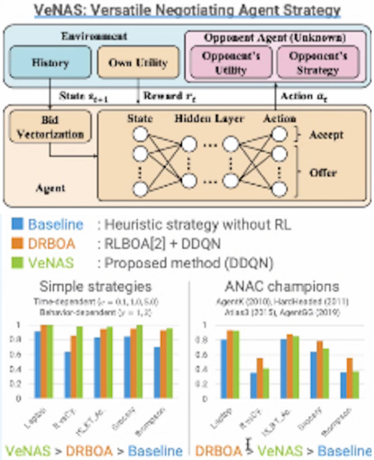
1. Learning in Automated Negotiation
2. Learning how to negotiate
   - Offer Policy
   - Acceptance Strategy
   - Both
3. Learning Preferences
4. References

Learning Offer and Acceptance Policies together[7]

- Fixed domain (i/o using outcomes).
- Discrete Issues: One hot encoding per issue.

- State Space $\omega^s, \omega^o, t, \eta_t$
- Action Space $\Omega \wedge$ Accept
- Reward $= \begin{cases} u(\omega_a), & \text{At the end} \\ 0 & \text{non terminal state} \end{cases}$

- Feb 25, 9:00-10:45am (PST) Blue 5
- Feb 26, 4:45-06:30am (PST) Blue 5



[7] Toki Takahashi et al. "VeNAS: Versatile Negotiating Agent Strategy via Deep Reinforcement Learning". In: *AAAI 2022*. 2022.

Outline

Outline

HardHeaded Opponent Modeling Strategy[8]

Context

- Winner of the ANAC 2011 competition.

$$u(\omega) = \sum_{i=1}^{n} \alpha_i F[i, \omega_i]$$

- Assumes a Discrete Outcome Space, Linear Additive Utility Function and a bilalateral negotiation.
- Learns while negotiaing.

Main Idea

- The opponent is likely to change values for issues that are less important.

[8] Thijs van Krimpen, Daphne Looije, and Siamak Hajizadeh. "HardHeaded". In: *Complex Automated Negotiations: Theories, Models and Software Competitions*. Ed. by Ito Takayuki et al. Springer, 2013, pp. 223–227.

HardHeaded Opponent Model: Pseudo-code

NegMAS-like implementation

```
# M issues and N values per issue

class HardHeaddedOpponentModel(UtilityFunction):
    F = np.zeros((M, N))
    alpha = np.zeros(M)

    epsilon = 0.02
    last_offer = None

    def after_receiving(self, state, offer):
        # update model
        if not self.last_offer:
            self.last_offer = offer
            return
        for i in range(M):
```

Opponent Model: Bayesian

Bayesian Learning

Hypothesis A hypothesis about the opponent's behavior.

Opponent Model: Bayesian

Bayesian Learning

Hypothesis A hypothesis about the opponent's behavior.

Evidence Behavior of the agent (e.g. its counteroffers/rejections).

Opponent Model: Bayesian

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

Bayesian Learning

Hypothesis A hypothesis about the opponent's behavior.

Evidence Behavior of the agent (e.g. its counteroffers/rejections).

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

Opponent Model: Bayesian

Bayesian Learning

Hypothesis A hypothesis about the opponent's behavior.

Evidence Behavior of the agent (e.g. its counteroffers/rejections).

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

Example

Hypothesis space: Utility function as a weighted sum of basis functions

$$u(\omega) = \sum_{i=1}^{n} \alpha_i f_i(\omega_i; \sigma_i)$$

Opponent Model: Bayesian

Bayesian Learning

Hypothesis A hypothesis about the opponent's behavior.

Evidence Behavior of the agent (e.g. its counteroffers/rejections).

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

Example

Hypothesis space: Utility function as a weighted sum of basis functions

$$u(\omega) = \sum_{i=1}^{n} \alpha_i f_i(\omega_i; \sigma_i)$$

Evidence: Rejection and offers (assuming a strategy).

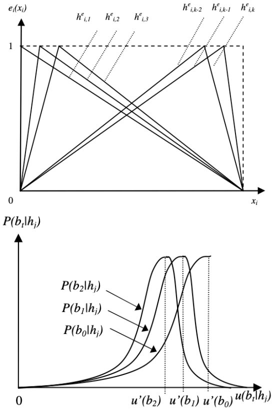
Bayesian Opponent Model Learner[9]

Assumptions

- Opponent has a Linear Additive UFun ($u(\omega) = \sum_{i=1}^{|\omega|} \alpha_i f_i(\omega_i, \sigma_i)$)
- Value functions (f_i) are triangle like (or linear).

Settings

- Hypothesis Space: values of α_i and σ_i
- Evidence: $P(\omega|\alpha_i, \sigma_i) = \frac{1}{\sqrt{2\pi\sigma}} \exp \frac{(u(\omega|\alpha_i,\sigma_i) - \hat{u}(\omega))^2}{\sigma}$ with $\hat{u}(\omega) = 1 - \frac{t}{20}$.
- Estimated opponent utility value: $u^o(\omega) = \sum_{j=1}^{|H|} P(\alpha_j, \sigma_j|\Omega^o) u(\omega|\alpha_j, \sigma_j)$



[9] Koen Hindriks and Dmytro Tykhonov. "Opponent modelling in automated multi-issue negotiation using bayesian learning". In: *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*. 2008, pp. 331–338.

Outline

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

Preference Elicitation

The challenge

How to reduce Uncertainty in user preferences:

- before negotiation (offline preference elicitation).
- while negotiating (online preference elicitation).

Preference Elicitation

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

The challenge

How to reduce Uncertainty in user preferences:

- before negotiation (offline preference elicitation).
- while negotiating (online preference elicitation).

Types of questions

Utility Value what is $\tilde{u}(\omega)$?

Utility Constraint Is $\tilde{u}(\omega) \geq x$? Usually implemented as a standard gamble.

Utility Comparison Is $\omega_1 \succ \omega_2$?

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: .

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: Do not ask about the utility directly..

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: Do not ask about the utility directly..
3. Practical elicitation uses a **series** of comparisons between outcomes to assess utilities.

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: Do not ask about the utility directly..
3. Practical elicitation uses a **series** of comparisons between outcomes to assess utilities.

A Gamble

(ω^*, ω_*, p) : Getting ω^* with probability p otherwise ω_*

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: Do not ask about the utility directly..
3. Practical elicitation uses a **series** of comparisons between outcomes to assess utilities.

A Gamble

(ω^*, ω_*, p) : Getting ω^* with probability p otherwise ω_*

Example query

Do you prefer to get ω for certain over (ω^*, ω_*, p)?

Elicitation Procedures/Strategies

Probability Equivalence

find p so that $\omega = (\omega^*, \omega_*, p)$

Certainty Equivalence

find ω so that $\omega = (\omega^*, \omega_*, p)$

- Both require *normalized* utilities.
- Both require knowledge of $\omega^* \succ \omega \succ \omega_*$.
- Lead to different biases.

Comparison-only Procedures

1. Titration-down: $p_k = 1 - s \times k$
2. Titration-up: $p_k = s \times k$
3. Ping-pong: $p_k = \begin{cases} s \times \lfloor k/2 \rfloor & \text{k is odd} \\ 1 - s \times k/2 & \text{k is even} \end{cases}$

Importance of Elicitation

Negotiation with Elicitation

$m,\ \Omega,\ R,\ \tilde{U}_i\ \forall 1 \le i \le m\ ,\ \hat{U}_i^0\ \forall 1 \le i \le m$

m Number of agents/actors

$\Omega = \{\omega_j\}$ Possible outcomes (assumed countable)

n Number of outcomes $|\Omega|$

$R(i) \equiv r_i$ Reserved value for agent i

$\tilde{U}_i : \Omega \to [0,1]$ Utility of outcomes to **actor** i

$\hat{U}_i^0 : \Omega \to P$ Probability distribution of utility values for **agent** i

$\hat{U}_{ij}^0 \equiv \hat{U}_i^0(\omega_j)$

$P : \{[0,1] \to [0,1]\}$ A probability distribution on the closed interval $[0,1]$

What is Elicitation Doing?

Reduces uncertainty in $\hat{U}$

State of the Art

- Lots of work on preferences/utility elicitation in decision making domain.
- Some work on incremental utility elicitation.
- Few works on incremental utility elicitation during negotiations

State of the Art

- Lots of work on preferences/utility elicitation in decision making domain.
- Some work on incremental utility elicitation.
- Few works on incremental utility elicitation during negotiations

Why Is Negotiation Different

1. The acceptance model changes over time $\rightarrow$ environment dynamics are not static.
2. Exploration is extremely costly.
3. Usually negotiations are not repeated much.
4. Cannot train on a simulator (in most cases).

Value of Information Algorithm

- Based on[10] in decision-support context.
- Adapted to the negotiation context.

Main Idea

- Assume an accurate opponent model (acceptance probability)
- Given a set of queries $Q \rightarrow$ find the one with the maximum difference between the expected expected utility before and after asking it[11][12].

[10] Urszula Chajewska, Daphne Koller, and Ronald Parr. "Making rational decisions using adaptive utility elicitation". In: *AAAI/IAAI*. 2000, pp. 363–369.

[12] Tim Baarslag and Michael Kaisers. "The value of information in automated negotiation: A decision model for eliciting user preferences". In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems. 2017, pp. 391–400.

[12] Yasser Mohammad and Shinji Nakadai. "FastVOI: Efficient Utility Elicitation During Negotiations". In: *International Conference on Principles and Practice of Multi-Agent Systems (PRIMA)*. Springer. 2018, pp. 560–567.

VOI Based Elicitation

Policy

$$\pi^t = \left(\omega^t, \omega^{t+1}, \omega^N\right) \text{ where } \omega^x \in \Omega \; K(\omega|\pi) \equiv \text{index of } \omega \text{ in } \pi, \; \pi(k) = \omega \text{ where } K(\omega|\pi) = k$$

Optimal Policy

$$\pi^{t*} = \arg\max_\pi EEU^t\left(\pi, \left\{\hat{U}^t_\omega\right\}\right)$$

VOI Based Elicitation

Policy

$$\pi^t = \left(\omega^t, \omega^{t+1}, \omega^N\right) \text{ where } \omega^x \in \Omega \;\; K\left(\omega|\pi\right) \equiv \text{index of } \omega \text{ in } \pi,\; \pi\left(k\right) = \omega \text{ where } K\left(\omega|\pi\right) = k$$

Probability of Agreement

$$Pa^t\left(\omega|\pi\right) = \begin{cases} \Lambda^t\left(\omega\right) \prod_{k=1}^{K_\pi(\omega)-1} \left(1 - \Lambda^t\left(\pi\left(k\right)\right)\right) & \omega \in \pi \\ 0 & otherwise \end{cases}$$

Optimal Policy

$$\pi^{t*} = \arg\max_\pi EEU^t\left(\pi, \left\{\hat{U}^t_\omega\right\}\right)$$

VOI Based Elicitation

Policy

$$\pi^t = \left(\omega^t, \omega^{t+1}, \omega^N\right) \text{ where } \omega^x \in \Omega \;\; K\left(\omega|\pi\right) \equiv \text{index of } \omega \text{ in } \pi,\; \pi\left(k\right) = \omega \text{ where } K\left(\omega|\pi\right) = k$$

Probability of Agreement

$$Pa^t\left(\omega|\pi\right) = \begin{cases} \Lambda^t\left(\omega\right)\prod_{k=1}^{K_\pi(\omega)-1}\left(1 - \Lambda^t\left(\pi\left(k\right)\right)\right) & \omega \in \pi \\ 0 & otherwise \end{cases}$$

Expected Expected Utility[13]

$$EEU^t\left(\pi, \left\{\hat{U}^t_\omega\right\}\right) = \sum_{\omega\in\Omega} Pa\left(\omega|\pi\right)\mathbb{E}\left(\hat{U}^t_\omega\right)$$

Optimal Policy

$$\pi^{t*} = \arg\max_\pi EEU^t\left(\pi, \left\{\hat{U}^t_\omega\right\}\right)$$

VOI Based Elicitation II

Questions

$$Q \equiv \{q_l\}$$
$$q_l \equiv \left\{ \left(Ans_s^l, p_s \right) \right\}$$

Answers

$$Ans_s^l \equiv \left\{ \hat{U}_\omega^{t+1} \right\}$$
$$\sum_s p_s = 1$$

VOI Based Elicitation II

Questions

$$Q \equiv \{q_l\}$$
$$q_l \equiv \left\{\left(Ans_s^l, p_s\right)\right\}$$

Answers

$$Ans_s^l \equiv \left\{\hat{U}_\omega^{t+1}\right\}$$
$$\sum_s p_s = 1$$

Expected value of information

$$EVOI\left(q^l, \left\{\hat{U}_\omega^t\right\}\right) = \mathbb{E}_s\left(\max_\pi EEU\left(\pi, Ans_s^l\right)\right) - \max_\pi EEU\left(\pi, \left\{\hat{U}_\omega^t\right\}\right)$$

VOI Based Elicitation II

Questions

$$Q \equiv \{q_l\}$$
$$q_l \equiv \left\{\left(Ans_s^l, p_s\right)\right\}$$

Answers

$$Ans_s^l \equiv \left\{\hat{U}_\omega^{t+1}\right\}$$
$$\sum_s p_s = 1$$

Expected value of information

$$EVOI\left(q^l, \left\{\hat{U}_\omega^t\right\}\right) = \mathbb{E}_s\left(\max_\pi EEU\left(\pi, Ans_s^l\right)\right) - \max_\pi EEU\left(\pi, \left\{\hat{U}_\omega^t\right\}\right)$$

Elicitation

Ask q^* where

$$q^* = \arg\max_q \left(EVOI\left(q^l, \left\{\hat{U}_\omega^t\right\}\right) - c_q\right)$$

c_q Cost of asking question q

VOI main Issues

Accurate Agreement Model Assumption

- Everything depends on the probability of agreement (Pa)
- Pa depends on the **product** of probabilities in the acceptance model (Λ^t)

$$Pa^t\left(\omega|\pi\right) = \begin{cases} \Lambda^t\left(\omega\right)\prod_{k=1}^{K_\pi(\omega)-1}\left(1-\Lambda^t\left(\pi\left(k\right)\right)\right) & \omega\in\pi \\ 0 & otherwise \end{cases}$$

VOI main Issues

Accurate Agreement Model Assumption

- Everything depends on the probability of agreement (Pa)
- Pa depends on the **product** of probabilities in the acceptance model (Λ^t)

$$Pa^t(\omega|\pi) = \begin{cases} \Lambda^t(\omega)\prod_{k=1}^{K_\pi(\omega)-1}(1-\Lambda^t(\pi(k))) & \omega \in \pi \\ 0 & otherwise \end{cases}$$

Speed: Complexity $= O(nN|Q||Ans|)$

- Too many *argmax* and $\mathbb{E}$ operations.
- Every policy extends to the end of the negotiation.

$$q^* = \arg\max_q \left(EVOI\left(q^l, \left\{\hat{U}^t_\omega\right\}\right) - c_q\right)$$

$$EVOI\left(q^l, \left\{\hat{U}^t_\omega\right\}\right) = \mathbb{E}_s\left(\max_\pi EEU\left(\pi, Ans^l_s\right)\right) - \max_\pi EEU\left(\pi, \left\{\hat{U}^t_\omega\right\}\right)$$

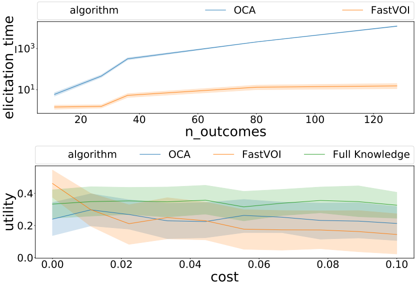
$$\pi^{t*} = \arg\max_\pi EEU^t\left(\pi, \left\{\hat{U}^t_\omega\right\}\right)$$

Extending VOI

BROWN UNIVERSITY | NEC-AIST AI Cooperative Research Laboratory

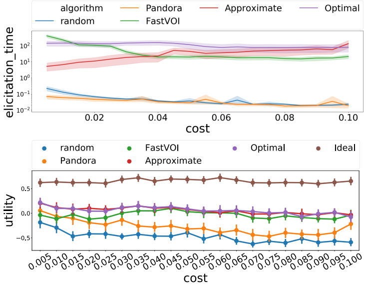
FastVOI[14]

- A faster approximate version of VOI



OptimalVOI[15]

- Extends Applicability to Infinite N. Questions.



[15] Yasser Mohammad and Shinji Nakadai. "Fastvoi: Efficient utility elicitation during negotiations". In: *International Conference on Principles and Practice of Multi-Agent Systems*. Springer. 2018, pp. 560–567.

[15] Yasser Mohammad and Shinji Nakadai. "Optimal value of information based elicitation during negotiation". In: *Proceedings of the 18th international conference on autonomous agents and multiagent systems*. 2019, pp. 242–250.

Outline

References I

Baarslag, Tim and Michael Kaisers. "The value of information in automated negotiation: A decision model for eliciting user preferences". In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems. 2017, pp. 391–400.

Bakker, Jasper et al. "RLBOA: A modular reinforcement learning framework for autonomous negotiating agents". In: *Proceedings of the 18th international conference on autonomous agents and multiagent systems*. 2019, pp. 260–268.

Boutilier, Craig. "On the foundations of expected expected utility". In: *IJCAI*. Vol. 3. Citeseer. 2003, pp. 285–290.

Chajewska, Urszula, Daphne Koller, and Ronald Parr. "Making rational decisions using adaptive utility elicitation". In: *AAAI/IAAI*. 2000, pp. 363–369.

Hindriks, Koen and Dmytro Tykhonov. "Opponent modelling in automated multi-issue negotiation using bayesian learning". In: *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*. 2008, pp. 331–338.

References II

Kawata, Ryohei and Katsuhide Fujita. "Meta-Strategy Based on Multi-Armed Bandit Approach for Multi-Time Negotiation". In: *IEICE TRANSACTIONS on Information and Systems* 103.12 (2020), pp. 2540–2548.

Krimpen, Thijs van, Daphne Looije, and Siamak Hajizadeh. "HardHeaded". In: *Complex Automated Negotiations: Theories, Models and Software Competitions*. Ed. by Ito Takayuki et al. Springer, 2013, pp. 223–227.

Mohammad, Yasser and Shinji Nakadai. "FastVOI: Efficient Utility Elicitation During Negotiations". In: *International Conference on Principles and Practice of Multi-Agent Systems (PRIMA)*. Springer. 2018, pp. 560–567.

– ."Fastvoi: Efficient utility elicitation during negotiations". In: *International Conference on Principles and Practice of Multi-Agent Systems*. Springer. 2018, pp. 560–567.

– ."Optimal value of information based elicitation during negotiation". In: *Proceedings of the 18th international conference on autonomous agents and multiagent systems*. 2019, pp. 242–250.

Razeghi, Yousef, Celal Ozan Berk Yavuz, and Reyhan Aydoğan. "Deep reinforcement learning for acceptance strategy in bilateral negotiations". In: *Turkish Journal of Electrical Engineering & Computer Sciences* 28.4 (2020), pp. 1824–1840.

References III

Sengupta, Ayan, Yasser Mohammad, and Shinji Nakadai. "An Autonomous Negotiating Agent Framework with Reinforcement Learning Based Strategies and Adaptive Strategy Switching Mechanism". In: *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*. AAMAS '21. Virtual Event, United Kingdom: International Foundation for Autonomous Agents and Multiagent Systems, 2021, pp. 1163–1172. ISBN: 9781450383073.

Takahashi, Toki et al. "VeNAS: Versatile Negotiating Agent Strategy via Deep Reinforcement Learning". In: *AAAI 2022*. 2022.

Automated Negotiation: Challenges and Tools

Automagted Negotiation is SCM

Yasser Mohammad [1, 2, 3] Amy Greenwald [4]

[1] NEC Corporation, Global Innovation Unit

[2]National Institute of Advanced Industrial Science and Technology (AIST), Japan

[3]Assiut University, Egypt

[4]Brown University, USA

February 23rd, 2022

Outline

Outline

Negotiation in SCM Business

- Human negotiations lead to an estimated 17-40% *value leakage* in some estimates [1]

[1] KPMG report: https://bit.ly/3kDRy6I
[2] Forrester report: https://bit.ly/3nwXEaY
[3] UN/CEFACT Project website: https://bit.ly/38LOsLX
[4] Y. Mohammad et al. "Supply Chain Management World: A benchmark environment for situated negotiations". In: *Proceedings of the 22nd International Conference on Principles and Practice of Multi-Agent Systems*. 2019.

Negotiation in SCM Business

- Human negotiations lead to an estimated 17-40% *value leakage* in some estimates [1]
- A recent study suggests that at least 15 companies are working in *contracting support systems* [2].

[1] KPMG report: https://bit.ly/3kDRy6I
[2] Forrester report: https://bit.ly/3nwXEaY
[3] UN/CEFACT Project website: https://bit.ly/38LOsLX
[4] Mohammad et al., "Supply Chain Management World: A benchmark environment for situated negotiations".

Negotiation in SCM Business

- Human negotiations lead to an estimated 17-40% *value leakage* in some estimates [1]
- A recent study suggests that at least 15 companies are working in *contracting support systems* [2].
- A recent UNECE UN/CEFACT proposal to standardize negotiation protocols for SCM and other applications [3]

[1] KPMG report: https://bit.ly/3kDRy6I
[2] Forrester report: https://bit.ly/3nwXEaY
[3] UN/CEFACT Project website: https://bit.ly/38LOsLX
[4] Mohammad et al., "Supply Chain Management World: A benchmark environment for situated negotiations".

Negotiation in SCM Business

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory

- Human negotiations lead to an estimated 17-40% *value leakage* in some estimates [1]
- A recent study suggests that at least 15 companies are working in *contracting support systems* [2].
- A recent UNECE UN/CEFACT proposal to standardize negotiation protocols for SCM and other applications [3]
- More to come[4].

CONTRACTROOM

pactum

[1] KPMG report: https://bit.ly/3kDRy6I
[2] Forrester report: https://bit.ly/3nwXEaY
[3] UN/CEFACT Project website: https://bit.ly/38LOsLX
[4] Mohammad et al., "Supply Chain Management World: A benchmark environment for situated negotiations".

Outline

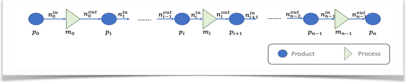
SCML World

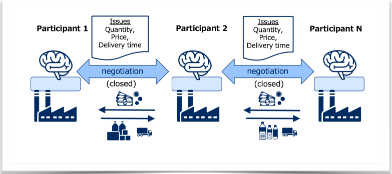
Challenge

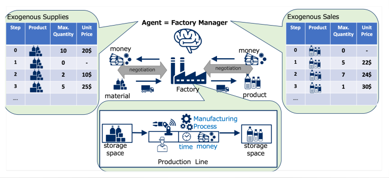
- Negotiation game with imperfect information
- Concurrent negotiations.
- Repeated negotiations → OneShot.
- Sequential negotiations → Standard.

Information

- **Website** https://scml.cs.brown.edu/
- **Code** https://www.github.com/yasserfarouk/scml







SCML Competition

Competition Details

- Runs as part of ANAC IJCAI.
- You control one or more factories.
  - **Oneshot track** → one factory (predefined ufun).
  - **Standard track** → one factory (you define your own ufun).
  - **Collusion track** → multiple factories (3).

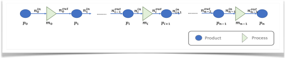
SCML Competition

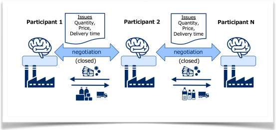
Competition Details

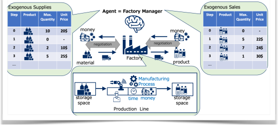
- Runs as part of ANAC IJCAI.
- You control one or more factories.
  - **Oneshot track** → one factory (predefined ufun).
  - **Standard track** → one factory (you define your own ufun).
  - **Collusion track** → multiple factories (3).

Flavors

- Online competition at https://scml.cs.brown.edu
- Official competition as part of ANAC.







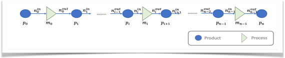
SCML Competition

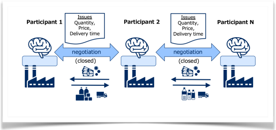
Competition Details

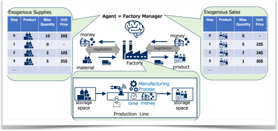
- Runs as part of ANAC IJCAI.
- You control one or more factories.
  - **Oneshot track** → one factory (predefined ufun).
  - **Standard track** → one factory (you define your own ufun).
  - **Collusion track** → multiple factories (3).

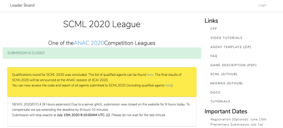
Flavors

- Online competition at https://scml.cs.brown.edu
- Official competition as part of ANAC.



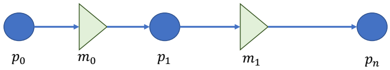






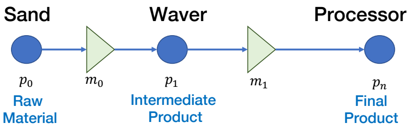
Outline

Overview



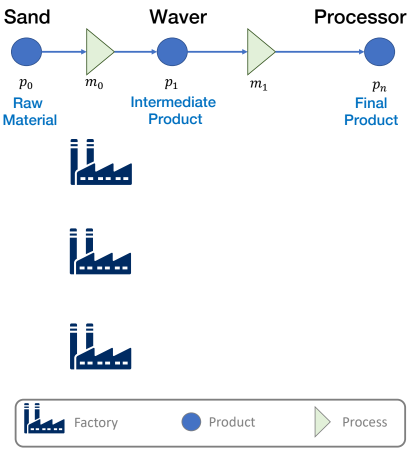
- A **production-graph** defines what can be produced and how.

Overview



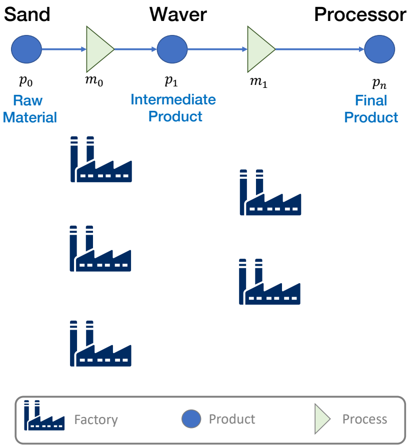
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.

Overview



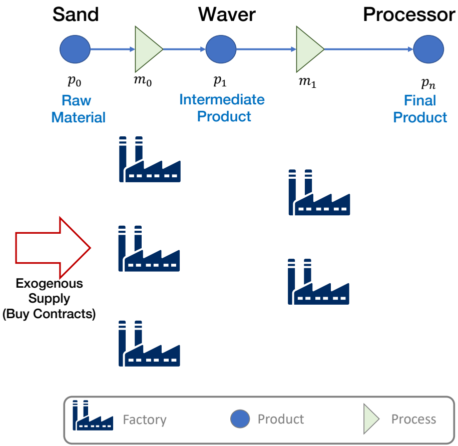
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.
- Factories can run **manufacturing processes** converting input products into output products on their **production lines**.

Overview



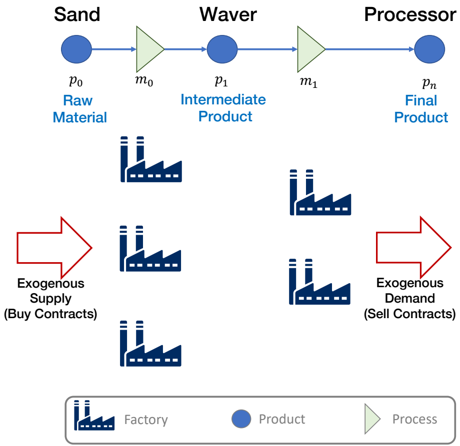
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.
- Factories can run **manufacturing processes** converting input products into output products on their **production lines**.
- We have two layers of factories/agents.

Overview



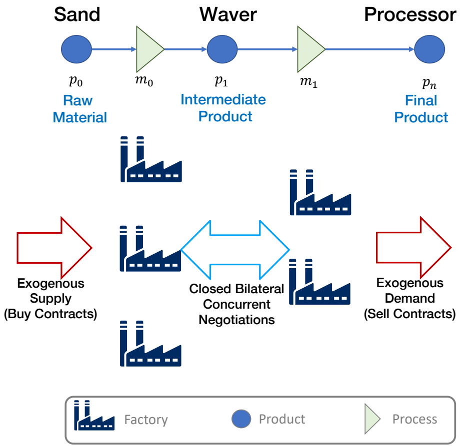
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.
- Factories can run **manufacturing processes** converting input products into output products on their **production lines**.
- We have two layers of factories/agents.
- L_0 factories/agents receive exogenous supplies of raw material.

Overview



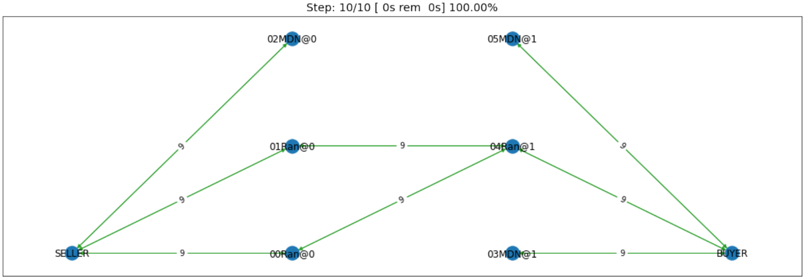
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.
- Factories can run **manufacturing processes** converting input products into output products on their **production lines**.
- We have two layers of factories/agents.
- L_0 factories/agents receive exogenous supplies of raw material.
- L_1 factories/agents receive exogenous sales of final product.

Overview



- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.
- Factories can run **manufacturing processes** converting input products into output products on their **production lines**.
- We have two layers of factories/agents.
- L_0 factories/agents receive exogenous supplies of raw material.
- L_1 factories/agents receive exogenous sales of final product.
- L_0 negotiate with L_1 agents to exchange intermediate product

SCML-OneShot Track



Main Idea

- Agents arranged in two production levels (3 products, 2 processes)
- Every day you get a fresh set of exogenous contracts.
- All products perish in one day (no inventory accomulation).

Utility Function

General Form

Utility = Profit = Sales - Supply cost - Production cost - Disposal cost - Delivery Penalty

Sales unit price $\times$ quantity $\forall$ feasible sales.

Supply cost unit price $\times$ quantity $\forall$ supplies.

Production cost unit production cost $\times$ quantity produced.

Disposal cost unit disposal cost $\times$ quantity bought but not produced.

Shortfall penalty unit shortfall penalty $\times$ infeasible sales.

Information about self

Static Information

- Number of production lines.
- Production cost.
- Mean and variance of disposal cost and shortfall penalty.
- Input/output product, consumers/suppliers, n. input/output negotiations.

Dynamic Information

- current input/output negotiation issues.
- current input/output exogenous contracts (quantity, unit price).
- current disposal cost and shortfall penalty.
- Current balance (money in wallet).

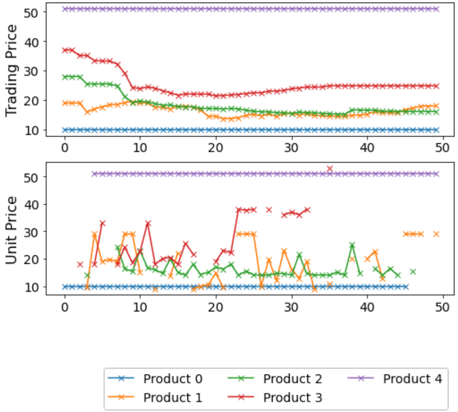
Market Information

Trading prices

- Trading prices represent a weighted running average of different product prices.
- Available to the agent through the AWI in all tracks.

Exogenous Contract Summary

- The total quantity and average prices of all exogenous contracts are now available through the AWI.
  - Exogenous contracts for individual agents are private information.



Other Agents' Information

Financial Reports

For each agent, a financial report is published every *m* days (e.g. 5) with the following information:

- Current balance (money in wallet).
- breach probability (fraction of sale contracts not satisfied).
- breach level (average fraction of sales not satisfied).

Outline

Simulation Steps

Once

Initialize all agents

- init()

Simulation Steps

Once

Initialize all agents

- init()

Update trading prices

Simulation Steps

Once

- Initialize all agents
  - init()
- Update trading prices
- Create exogenous contracts and sample agent's disposal cost, and shortfall penalty

Simulation Steps

Once

- **Initialize all agents**
  - init()

Every

- **Update trading prices**
- **Create exogenous contracts and sample agent's disposal cost, and shortfall penalty**
- **Initialize agents for the day**
  - before_step() [First call every day]

Simulation Steps

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

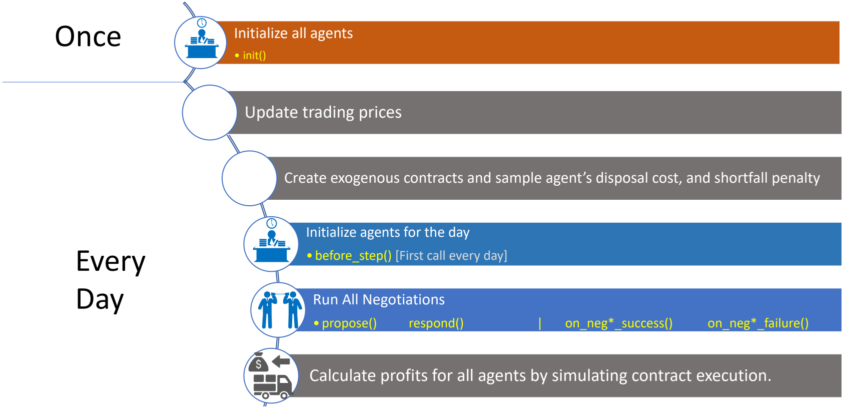
Once

- Initialize all agents
  - init()

Every Day

- Update trading prices
- Create exogenous contracts and sample agent's disposal cost, and shortfall penalty
- Initialize agents for the day
  - before_step() [First call every day]
- Run All Negotiations
  - propose() respond() | on_neg*_success() on_neg*_failure()

Simulation Steps



Simulation Steps

Simulation Steps

Once

- Initialize all agents
  - init()

Every Day

- Update trading prices
- Create exogenous contracts and sample agent's disposal cost, and shortfall penalty
- Initialize agents for the day
  - before_step() [First call every day]
- Run All Negotiations
  - propose() respond() | on_neg*_success() on_neg*_failure()
- Calculate profits for all agents by simulating contract execution.
- Finalize agents for the day
  - step() [Last call every day]
- Publish Financial Reports

jupyter

Outline

1. Automated Negotiation in SCML
2. The SCML Game
3. References

References I

Mohammad, Y. et al. "Supply Chain Management World: A benchmark environment for situated negotiations". In: *Proceedings of the 22nd International Conference on Principles and Practice of Multi-Agent Systems*. 2019.

Automated Negotiation: Challenges and Tools

Future Challenges

Yasser Mohammad [1, 2, 3] Amy Greenwald [4]

[1] NEC Corporation, Global Innovation Unit

[2] National Institute of Advanced Industrial Science and Technology (AIST), Japan

[3] Assiut University, Egypt

[4] Brown University, USA

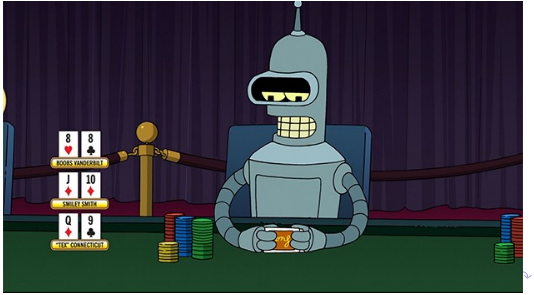
February 23rd, 2022

Outline

Outline

1. AI in Games
2. Finding a Best Response to a Fixed Opponent
3. Finding a Best Response to Multiple Fixed Opponents
4. Conclusion
5. References

AI has learned to play games (well!)



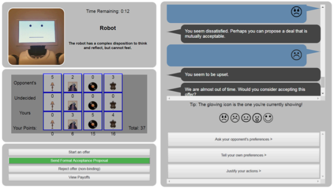
3/ 23

A brief history of ANAC

2022	??, HAN, Werewolf, **SCML**	2016	Energy Grid Theme
2021	ANL (Repeated), HAN, Werewolf, **SCML**	2015	Three-party negotiation
2020	ANL (Elicitation), HAN, Werewolf, **SCML**, HUMAINE	2014	Learning
2019	Uncertainty, Diplomacy, HAN, Werewolf, **SCML**	2012	Reservation Value
2018	Repeated Negotiations, Diplomacy, HAN	2011	Linear Ufuns
2017	Repeated Negotiations, Diplomacy, HAN	2010	Domain Independence

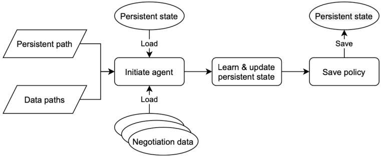
ANAC 2021 Leagues

HUMAINE: Negotiating with a Human

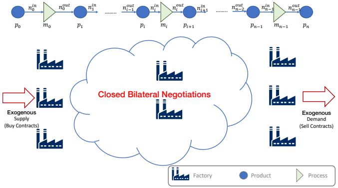


Werewolf: Negotiating with Natural Language

ANL: Learning in Repeated Negotiations (Preference Elicitation)



SCML: Supply Chain Management League



Game Characteristics

Solved Games

- Sequential decision making (e.g., turn-taking)
- Imperfect (and perfect) information
- Two-player & multi-player
- Zero-sum

Game Characteristics

Solved Games

- Sequential decision making (e.g., turn-taking)
- Imperfect (and perfect) information
- Two-player & multi-player
- Zero-sum

Agent Negotiation

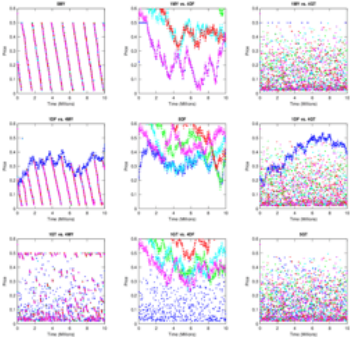
- Sequential decision making (e.g., turn-taking)
- Imperfect (and perfect) information
- Two-player & multi-player
- General-sum

Empirical Game-Theoretic Analysis (EGTA)

EGTA is a MAS tool for analyzing multiagent interactions.[1]

Key Idea: Derive so-called empirical games from data. Solve these games.

Empirical games are often higher-level games, played with higher-level strategies (i.e., heuristics).[2]



[2] Michael P. Wellman et al. "Exploring bidding strategies for market-based scheduling". In: *4th ACM Conference on Electronic Commerce*. San Diego, 2003, pp. 115–124.

Empirical Game-Theoretic Analysis (EGTA)

EGTA is a MAS tool for analyzing multiagent interactions.[1]

Key Idea: Derive so-called empirical games from data. Solve these games.

Empirical games are often higher-level games, played with higher-level strategies (i.e., heuristics).[2]

	4MY	4DF	4GT
1MY	.0337,.0337	.0690,.0225	.0185,.0109
1DF	.0136,.0335	.0387,.0387	.0134,.0159
1GT	.0119,.0169	.0536,.0226	.0129,.0129

[2] Wellman et al., "Exploring bidding strategies for market-based scheduling".

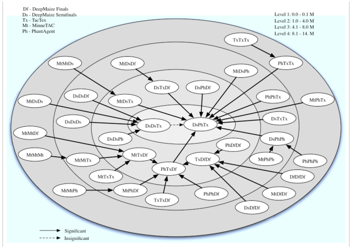
[2] Amy R. Greenwald, Jeffrey O. Kephart, and Gerald J. Tesauro. "Strategic Pricebot Dynamics". In: *1st ACM Conference on Electronic Commerce*. Denver, 1999, pp. 58–67.

Empirical Game-Theoretic Analysis (EGTA)

EGTA is a MAS tool for analyzing multiagent interactions.[1]

Key Idea: Derive so-called empirical games from data. Solve these games.

Empirical games are often higher-level games, played with higher-level strategies (i.e., heuristics).[2]



[2] Wellman et al., "Exploring bidding strategies for market-based scheduling".

[2] Patrick R. Jordan, Christopher Kiekintveld, and Michael P. Wellman. "Empirical game-theoretic analysis of the TAC supply chain game". In: *6th International Joint Conference on Autonomous Agents and Multi-Agent Systems*. Honolulu, 2007, pp. 1188–1195.

Double Oracle Algorithm[3](PSRO[4])

Theorem The double oracle algorithm converges to a Nash equilibrium in zero-sum games.

Algorithm 1 Double Oracle Algorithm

Input: A game with strategy sets Π_i, for $i \in \{1,2\}$
Input: An initial strategy set $\Pi_i^0 \in \Pi_i$, for $i \in \{1,2\}$
Output: Nash equilibrium

repeat $t \in \{0,1,2,\ldots\}$
    Find π^*, a Nash equilibrium in the game Π_i^t, for $i \in \{1,2\}$
    for $i \in \{1,2\}$ **do**
        Find a best response $\beta_i \in \Pi_i$ to π^*_{-i}
        Expand strategy set: $\Pi_i^{t+1} \leftarrow \Pi_i^t \cup \beta_i$
until $\Pi_i^{t+1} = \Pi_i^t$, for $i \in \{1,2\}$
Return: π^*

---

[2] H. Brendan McMahan, Geoffrey J. Gordon, and Avrim Blum. "Planning in the presence of cost functions controlled by an adversary". In: *20th International Conference on Machine Learning*. Washington, DC, 2003, pp. 536–543.

Outline

Finding a Best Response to a Fixed Opponent

Learning a best-response policy, offline

- Fixed acceptance policy: use RL to learn an offer policy
- Fixed offer policy: use deep learning to learn an acceptance policy

Finding a Best Response to a Fixed Opponent

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory

Learning a best-response policy, offline

- Fixed acceptance policy: use RL to learn an offer policy
- Fixed offer policy: use deep learning to learn an acceptance policy

Some issues with this approach

- Why decouple these decisions? Doing so is unlikely to be optimal. Perhaps for tractability?
- Possible way out: Formulate as an MDP and solve
- An even bigger problem: when is a negotiating partner so benevolent that they give you access to a simulator of their strategy that you can run for thousands of iterations to learn a best response?
- Possible way out: EGTA

Negotiation as an MDP

Assume a fixed opponent: one's whose strategy does depend *not* on our actions.
E.g., an aspiration agent.

When the opponent is known, our agent's decision making problem is an MDP.

- States: negotiation round plus two designated states, agreement and disagreement
- Actions: offers and accept/reject
- Transitions: capture opponent's behavior (e.g., if they accept, transition to the agreement state)
- Rewards: zero everywhere except at agreement, where they are given by the agent's utility function

Solving MDPs:

- Planning (e.g., VI), if the opponent model is in closed form
- RL, if we have only a generative model of the opponent model

Opponent Model

An opponent model comprises an acceptance model and an offer policy.

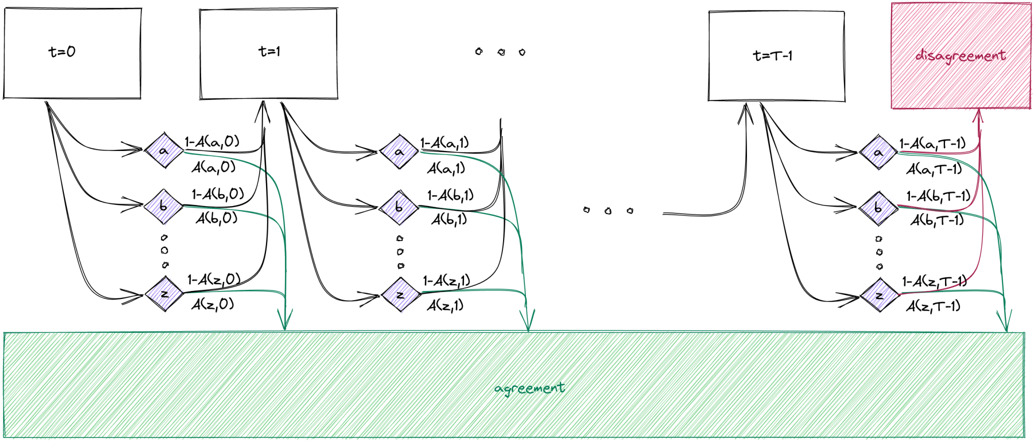
(Opponent ufuns are not modelled, because our decisions are conditionally independent of them, given an acceptance model and an offer policy.)

TDAM: time-dependent acceptance model
A map from a time to an acceptance probability distribution

TDOP: time-dependent offer policy
A map from a time to opponent offers (or probabilities over offers)

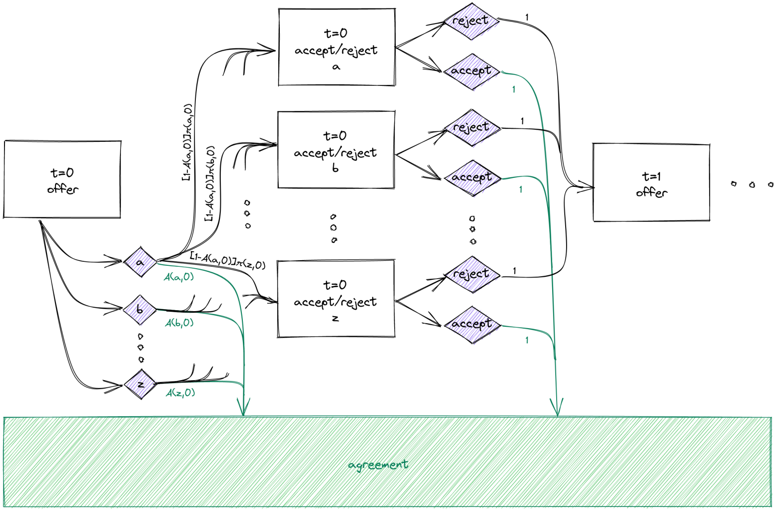
Time = negotiation round.

TDAM MDP



Drawing by Jackson de Campos

TDAM + TDOP MDP



Drawing by Jackson de Campos

EGTA, Revisited

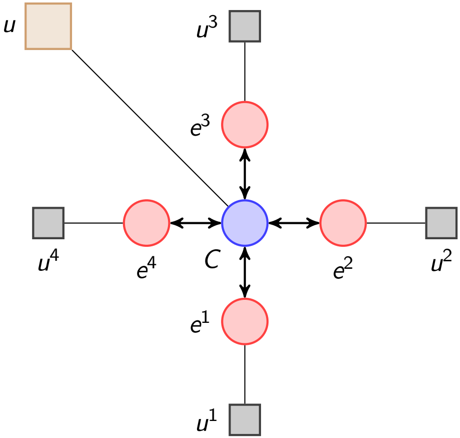
- Goal of EGTA: build agents that are robust, against a population of opponents.
- Each TDAM + TDOP MDP corresponds to an opponent.
  A solution to each MDP is an optimal negotiation strategy for playing against that opponent.
- Start from an initial population of opponents (i.e., MDPs) $\Rightarrow$ negotiation strategies.
- Use EGTA to grow the population of negotiation strategies $\Rightarrow$ opponents (i.e., MDP).
- End result should be a robust agent!

Outline

Concurrent Negotiations

These negotiations are dependent, as evidenced by the global ufun u, which depends on a global outcome: i.e., the outcomes of all negotiations.

The problem of how to negotiate in this setting is reminiscent of how to bid in simultaneous auctions, when bidders' utilities are combinatorial.[5]

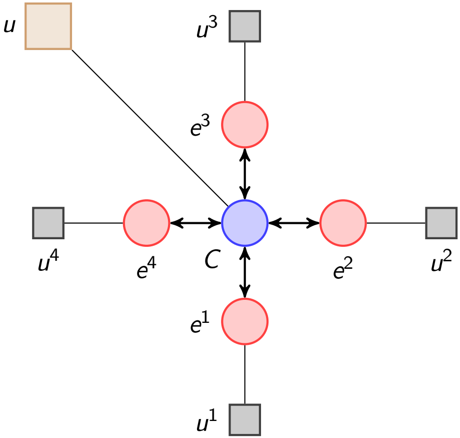


[5] Michael P. Wellman, Eric Sodomka, and Amy Greenwald. "Self-confirming price-prediction strategies for simultaneous one-shot auctions". In: *Games and Economic Behavior* 201 (2017), pp. 339–372.

Concurrent Negotiations

In ANAC SCML, an agent negotiates with multiple agents simultaneously. Moreover, the agent's utility depends on the outcomes of all the negotiations.

An agent represents a factory, whose production depends on the inputs it acquires from all its trading partners, and so, in turn, does its sales.



Expected Marginal Utility

An outcome prediction is a joint distribution over the outcomes of all (say, n) concurrent negotiations: i.e., $P(\omega) = P(\omega_1, \ldots, \omega_n)$, where $\omega_i \in \Omega_i$ and $\Omega = \prod_{i=1}^{n} \Omega_i$.

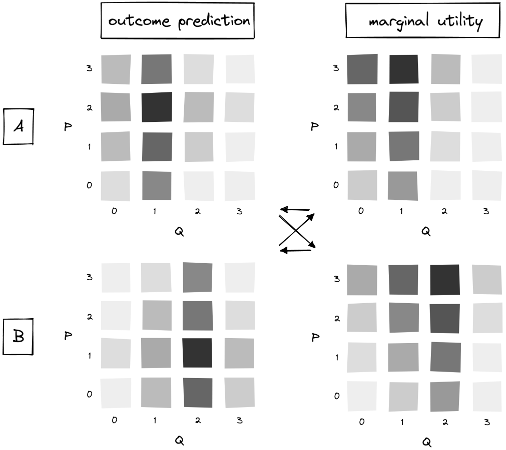
Given an outcome prediction P, the expected marginal utility μ_i of outcome $\omega_i \in \Omega_i$ in negotiation i is:

$$\mu_i(\omega_i; P) = \mathop{\mathbb{E}}_{\Omega_{\neg i} \sim P_{\neg i}} \left[u(\{\omega_i\} \cup \Omega_{\neg i}) - u(\Omega_{\neg i}) \right] \quad ,$$

where $P_{\neg i}$ is the outcome prediction for all negotiations other than i (i.e., P marginalized over i).

SCML OneShot: GodFather Strategy

- We are a seller with three goods to sell
- The arrows indicate dependencies in the expected marginal utility calculations
- Darker squares indicate higher probability predictions and higher marginal utilities
- Since we predict opponent A to most likely want 2 goods and B to most likely want 1, the highest marginal utility is obtained by contracting to sell 1 to opponent A and 2 to opponent B (at the highest possible prices)



Joint work with Jackson de Campos, Ben Fiske, and Chris Mascioli

Outcome Predictions

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory

An outcome prediction is a joint distribution over the outcomes of all concurrent negotiations: i.e., $P(\omega) = P(\omega_1, \ldots, \omega_n)$

- Static model: Depends on factors external to the current negotiation, e.g., the state of the world, past negotiations, etc.

  $P(\omega \mid \text{external factors})$

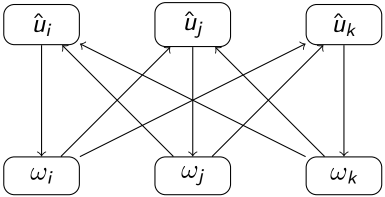
- Dynamic model: Depends on factors both external and internal to the current negotiation, i.e., the current negotiation trace so far

  $P(\omega \mid \text{both external and internal factors})$

- Introspective model: Expected marginal utilities depend on the outcome predictions. Likewise, the outcome predictions depend on the expected marginal utilities (because they affect our behavior).

  $P(\omega \mid \text{both external and internal factors, and expected marginal utilities})$

Introspective Equilibrium



An introspective equilibrium[5] is a situation in which the outcome predictions are consistent with the marginal utilities, meaning $P(\omega \mid \ldots$ and expected marginal utilities $\boldsymbol{\mu}(\cdot; f)) = f$.

Equivalently, for all negotiations x, $\mu_i(\omega_i; P(\omega \mid \ldots$ and expected marginal utilities $\boldsymbol{f})) = f_i(\omega_i)$.

[5] Wellman, Sodomka, and Greenwald, "Self-confirming price-prediction strategies for simultaneous one-shot auctions".

Open Questions

Decouplng negotiations tames the complexity, but is this kosher? How far from optimal is this approach?

How do you to build reliable prediction models: static, dynamic, and introspective?

When does simple iteration converge to an introspective equilibrium?

Outline

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory

Summary

Conclusions

Outline

References I

Greenwald, Amy R., Jeffrey O. Kephart, and Gerald J. Tesauro. "Strategic Pricebot Dynamics". In: *1st ACM Conference on Electronic Commerce*. Denver, 1999, pp. 58–67.

Jordan, Patrick R., Christopher Kiekintveld, and Michael P. Wellman. "Empirical game-theoretic analysis of the TAC supply chain game". In: *6th International Joint Conference on Autonomous Agents and Multi-Agent Systems*. Honolulu, 2007, pp. 1188–1195.

Lanctot, Marc et al. "A unified game-theoretic approach to multiagent reinforcement learning". In: *31st Annual Conference on Neural Information Processing Systems*. Long Beach, CA, 2017, pp. 4190–4203.

McMahan, H. Brendan, Geoffrey J. Gordon, and Avrim Blum. "Planning in the presence of cost functions controlled by an adversary". In: *20th International Conference on Machine Learning*. Washington, DC, 2003, pp. 536–543.

Wellman, Michael P., Eric Sodomka, and Amy Greenwald. "Self-confirming price-prediction strategies for simultaneous one-shot auctions". In: *Games and Economic Behavior* 201 (2017), pp. 339–372.

Wellman, Michael P. et al. "Exploring bidding strategies for market-based scheduling". In: *4th ACM Conference on Electronic Commerce*. San Diego, 2003, pp. 115–124.

references

Automated Negotiation: Challenges and Tools

Future Challenges and Open Problems

Yasser Mohammad [1, 2, 3] Amy Greenwald [4]

[1] NEC Corporation, Global Innovation Unit

[2] National Institute of Advanced Industrial Science and Technology (AIST), Japan

[3] Assiut University, Egypt

[4] Brown University, USA

February 23rd, 2022

AAAI 2022 Tutorial Updated on February 23, 2022

Outline

Outline

1. Negotiation With Incomplete Information
   - Negotiation with Known Acceptance Model
   - Greedy Concession Algorithm
   - Speeding up GCA
   - Extending GCA
2. Preference Elicitation During Negotiation
3. Concurrent Negotiation
4. References

Outline

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

Negotiation with Known Acceptance Model

Scenario

Negotiation with Known Acceptance Model

Scenario

$T \in \mathbb{I}$: The round-limit.

Negotiation with Known Acceptance Model

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

Scenario

$T \in \mathbb{I}$: The round-limit.

$\Omega \equiv \{\omega\}$: Possible outcomes.

- $\phi \in \Omega$ represents disagreement.

Negotiation with Known Acceptance Model

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

Scenario

$T \in \mathbb{I}$: The round-limit.

$\Omega \equiv \{\omega\}$: Possible outcomes.

- $\phi \in \Omega$ represents disagreement.

$u : \Omega \to [0, 1]$ Agent Utility Function.

Negotiation with Known Acceptance Model

Scenario

$T \in \mathbb{I}$: The round-limit.

$\Omega \equiv \{\omega\}$: Possible outcomes.

- $\phi \in \Omega$ represents disagreement.

$u : \Omega \rightarrow [0, 1]$ Agent Utility Function.

$a : \Omega \times [0, T] \rightarrow [0, 1]$ The acceptance model.

Policy (Υ)

A deterministic time-dependent policy $\pi : [1, T] \rightarrow \Omega$ is simply a **sequence** of outcomes

Problem

Find the **optimal** deterministic time-dependent policy π^* such that:

$$\pi^* = \arg\max_{\pi} \mathcal{EU}(\pi|\Upsilon)$$

Acceptance Model Progression

General AM $a(\omega) = f(\omega, t, \pi, x)$

Dynamic AM $a(\omega) = f(\omega, t, \pi)$

Time Dependent AM $a(\omega) = f(\omega, t)$

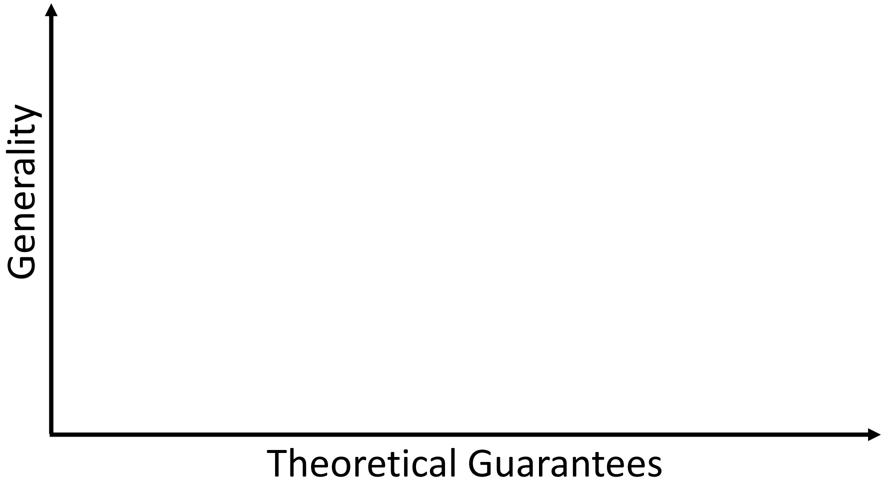
Monotonic AM
$$\big(f(\omega, t+1) - f(\omega, t)\big)\big(f(\omega, t+2) - f(\omega, t+1)\big) \geq 0$$

Homogeneous AM
$$f(\omega_1, t+1) - f(\omega_1, t) = f(\omega_2, t+1) - f(\omega_2, t)$$

Fixed Rate AM
$$f(\omega, t+2) - f(\omega, t+1) = f(\omega, t+1) - f(\omega, t)$$

Static AM $f(\omega, t+1) = f(\omega, t)$

Positioning this work

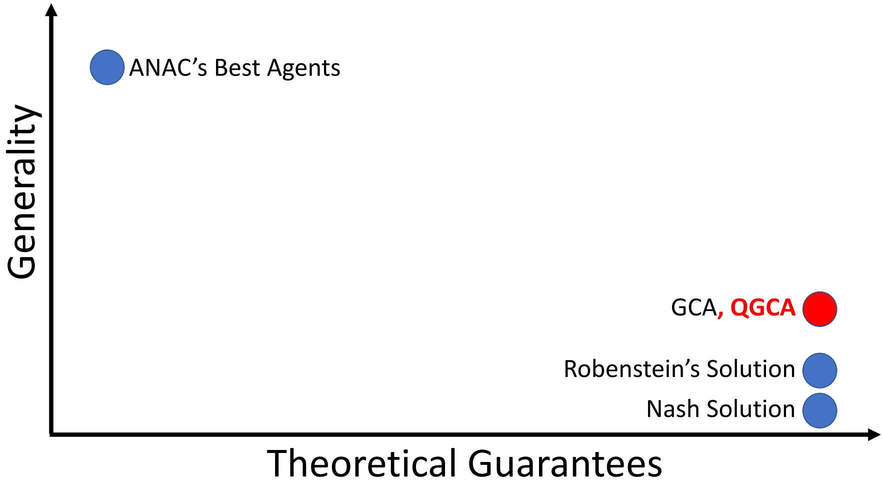


Positioning this work

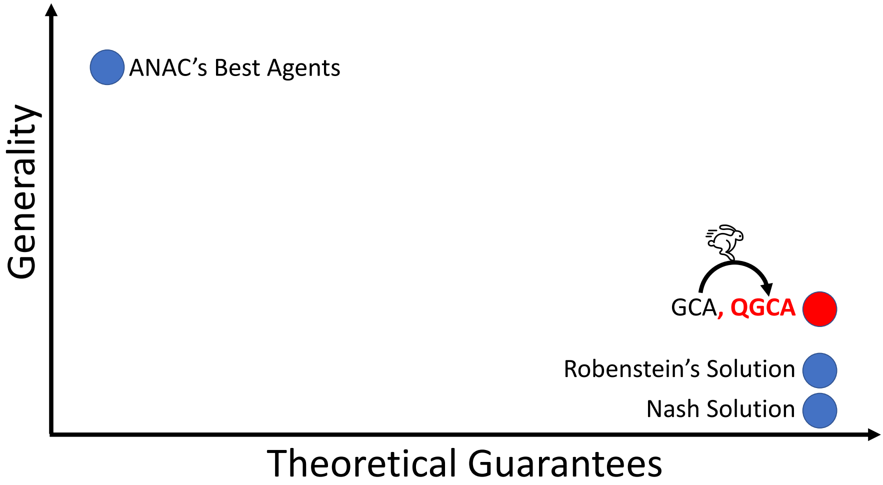
Positioning this work

Positioning this work

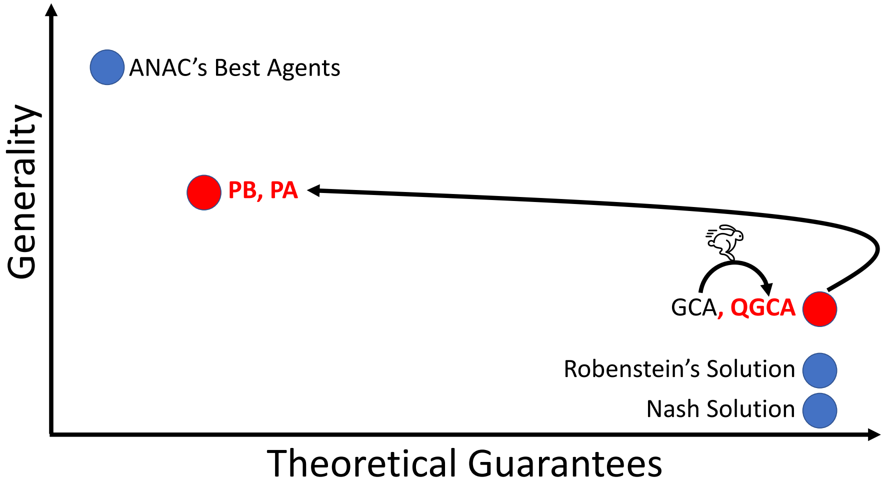
Positioning this work



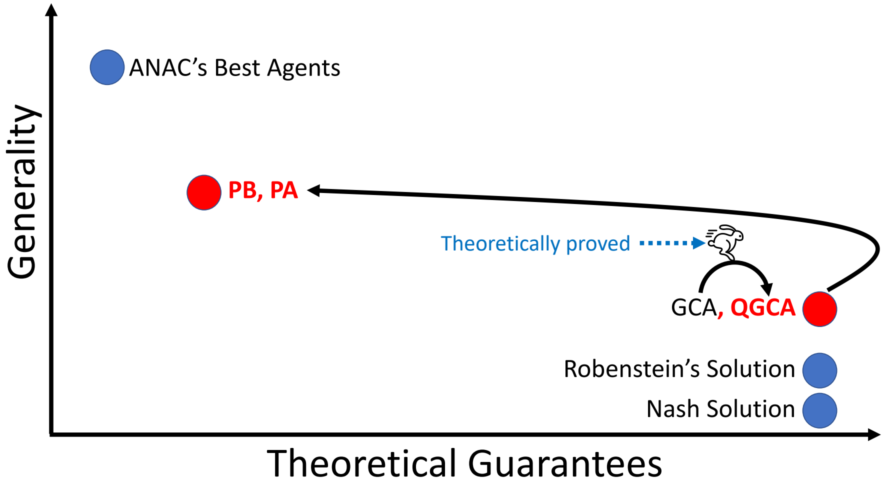
Positioning this work



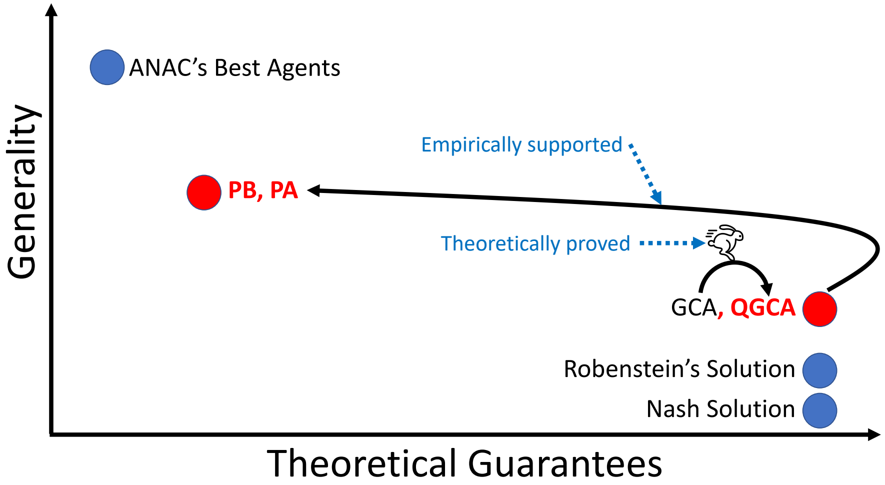
Positioning this work



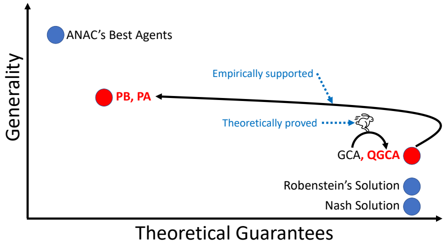
Positioning this work



Positioning this work



Why is this worth doing?



Why are we trying this?

- You cannot argue with .
  - Faster calculation of π^* opens new .
- Advance our Fundamental Understanding.
  - When **exactly** is GCA optimal?
  - Why **exactly** is GCA optimal?
- Provide ideas and inspiration for new strategies to advance the state of the art.

Outline

Greedy Concession Algorithm

Applicability

Designed and proved optimal for negotiations **static** acceptance models, and **no-repetition** of offers[a].

[a] **baarslag2015gca**.

Main Idea

- The optimal policy of length one is known: $\arg\max_{\omega} a(\omega)u(\omega)$
- Optimal policy of length N contains *scrambled* version of the optimal policy of length $N-1 \rightarrow$ Optimality of greediness theorem.
- The optimal policy is a **conceding** policy $\rightarrow$ the concession lemma.

Understanding Optimality of GCA

GCA

$\pi \leftarrow <>$ ▷ Empty policy
for $k \leftarrow 1 : T$ **do**
  $\omega^* \leftarrow \arg\max_{\omega \in \Omega - \pi} \mathcal{EU}(sort_{\mathcal{EU}}(\pi \circ \omega))$
  $\pi \leftarrow sort_{\mathcal{EU}}(\pi \circ \omega^*)$ ▷ Operator o inserts an element in a set

Understanding Optimality of GCA

GCA

1: $\pi \leftarrow <>$ ▷ Empty policy
2: **for** $k \leftarrow 1 : T$ **do**
3: $\quad \omega^* \leftarrow \arg\max_{\omega \in \Omega - \pi} \mathcal{EU}(sort_{\mathcal{EU}}(\pi \circ \omega))$
4: $\quad \pi \leftarrow sort_{\mathcal{EU}}(\pi \circ \omega^*)$ ▷ Operator o inserts an element in a set

Time Complexity

$$O(TK^2)^a$$

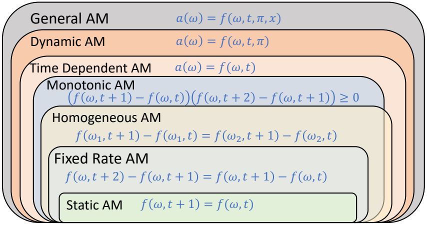
- Linear in negotiation time.
- **Quadratic** in outcome space size.

[a] trees do not help here 😱

Understanding Optimality of GCA

Can it be extended without modifications?

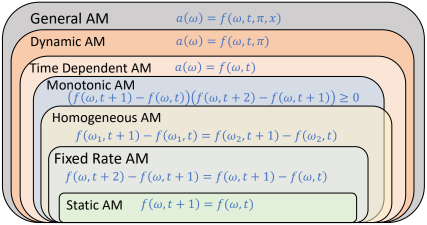
- Allowing repetition $\rightarrow$
- Extension **keeping optimality** to more AMs?
    - General TDAM $\rightarrow$
    - Monotonic AM $\rightarrow$
    - Homogeneous AM $\rightarrow$
    - Fixed Rate AM $\rightarrow$



Understanding Optimality of GCA

Can it be extended without modifications?

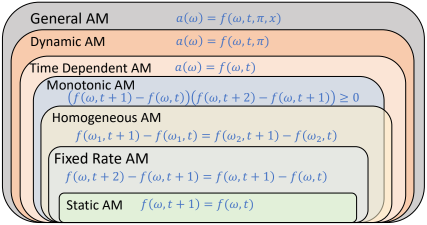
- Allowing repetition $\rightarrow$ **YES.**
- Extension **keeping optimality** to more AMs?
  - General TDAM $\rightarrow$
  - Monotonic AM $\rightarrow$
  - Homogeneous AM $\rightarrow$
  - Fixed Rate AM $\rightarrow$



Understanding Optimality of GCA

Can it be extended without modifications?

- Allowing repetition $\rightarrow$ **YES.**
- Extension **keeping optimality** to more AMs?
  - General TDAM $\rightarrow$ **NO.**
  - Monotonic AM $\rightarrow$ **NO.**
  - Homogeneous AM $\rightarrow$ **NO.**
  - Fixed Rate AM $\rightarrow$ **NO.**



Outline

Quick GCA

Applicability

Static Acceptance Models.

Quick GCA

Applicability

Static Acceptance Models.

Main Idea

90% of CS is caching ... [a]

- Calculating the $\mathcal{EU}$ of a policy **repeats** many calculations of shorter policies.
- Let's **cache** those.
- Allows us to calculate the effects of various operations on the expected utility in **O(1)** time (and space).

[a] this is not a quote. Just invented it.

QGCA Details

Cached Quantities

$P : [0, T] \to [0, 1]$ The aggregate multiplication of probability of rejection used in the evaluation of $\mathcal{EU}$.

$$P_i = \prod_{j=0}^{i-1} 1 - a(\pi_j) \tag{1}$$

$S : [0, T] \to [0, 1]$ The cumulative sum of the expected utility value.

$$S_i = \sum_{j=0}^{i} u(\pi_j)a(\pi_j)P_j = S_{i-1} + u(\pi_i)a(\pi_i)P_i \tag{2}$$

Calculating $\mathcal{EU}$ using ↑

$\mathcal{EU} = S_T$

Operations on Policies

Static AM

Operation	Symbol	Effect on $\mathcal{EU}(\pi)$
Replacing π_i with ω	π_i^ω	$P_i(u(\omega)a(\omega, i) - u(\pi_i)a(\pi_i)) + (S_T - S_i)\frac{a(\omega, i) - a(\pi_i, i)}{1 - a(\pi_i, i)}$
Swapping π_i, π_j	$\pi_{i\leftrightarrow j}$	$u_i\left(a_i^j P_j \frac{1-a_j^i}{1-a_i^i} - a_i^i P_i\right) + u_j\left(a_j^i P_i - a_j^j P_j\right) + \left(S_{j-1} - S_{i+1}\right)\frac{a_i^i - a_j^i}{1 - a_i^i}$
Swapping π_i, π_{i+1}	$\pi_{i\rightarrow 1}$	$u_i\left(a_i^{i+1}P_i(1 - a_{i+1}^i) - a_i^i P_i\right) + u_{i+1}\left(a_{i+1}^i P_i - a_{i+1}^{i+1}P_{i+1}\right)$

General TDAM

Replacing π_i with ω	π_i^ω	$P_i(u(\omega)a(\omega) - u(\pi_i)a(\pi_i)) + (S_T - S_i)\frac{a(\omega) - a(\pi_i)}{1 - a(\pi_i)}$
Swapping π_i, π_j	$\pi_{i\leftrightarrow j}$	$u_i a_i\left(P_j \frac{1-a_j}{1-a_i} - P_i\right) + u_j a_j\left(P_i - P_j\right) + \left(S_{j-1} - S_{i+1}\right)\frac{a_i - a_j}{1 - a_i}$
Swapping π_i, π_{i+1}	$\pi_{i\rightarrow 1}$	$P_k a(\pi_i)a(\pi_{i+1})\left(u(\pi_{i+1}) - u(\pi_i)\right)$
Inserting outcome ω at location t	$\pi_{\omega @ i}$	$u(\omega)a(\omega)P_i - a(\omega)(S_T - S_{i-1})$

Quick GCA

$\pi \leftarrow <>$ ▷ Empty *linked* list
$\mathcal{L}_\omega = 0 \forall \omega \in \Omega$ ▷ Initialize location of all outcomes
$S_{-1}, P_{-1} \leftarrow 0, 1$
for $k \leftarrow 1 : T$ **do**
  $d^*, \omega^* \leftarrow -\infty, \phi$
  for $\omega \in \Omega$ **do**
    $i \leftarrow \mathcal{L}_\omega$ ▷ Lookup location of insertion
    $d_\omega \leftarrow S_{i-1} + (1 - a(\omega))(\mathcal{EU}(\pi) - S_{i-1}) + P_i \mathcal{EU}(\omega) a(\omega)$
    if $d_\omega \geq d^*$ **then**
      $d^*, \omega^* \leftarrow d_\omega, \omega$
  $\pi \leftarrow \pi \circ_{\mathcal{L}_{\omega^*}} \omega^*$ ▷ Insert best outcome in its correct place
  Update S, P, $\mathcal{L}$
  if no-repetition **then**
    $\Omega \leftarrow \Omega - \{\omega^*\}$

Time Complexity

$$O(TK)$$

- Linear in negotiation time.
- **Linear** in outcome space size.

Outline

Extending GCA to General TDAM

Why cannot we use GCA as it is?

- The concession lemma does not hold.
  - The optimal policy may **NOT** be conceding.
- The greedy-is-optimal theorem does not hold.
  - The optimal policy may **NOT** contain shorter optimal policies (even scrambled).

What to do

- Start by running QGCA $\rightarrow$ a conceding policy.
  - Use the mean as the SAM.
- Use fast operations to permute this policy for a better one.
  - Policy Bubbling (PB).
- Starting from PB, search for a better policy (i.e. using simulated annealing)
  - Policy Annealing (PA).

Policy Bubbling

$\pi \leftarrow \pi^c$ ▷ Start from the result of QGCA$^+$

for $r \leftarrow 1 : T$ **do**

    $\pi^- \leftarrow \pi$

    for $k \leftarrow 1 : T - 1$ **do**

        if $\mathcal{EU}(\pi_{k \to 1}) - \mathcal{EU}(\pi) > 0$ **then**

            $\pi \leftarrow \pi_{k \to 1}$

    if $\pi^- = \pi$ **then**

        return $\pi^b = \pi$

return $\pi^b = \pi$

Time Complexity

$$O(T^2)$$

Policy Annealing

$\pi \leftarrow \pi^b$ ▷ Start from the result of PBS
for $r \leftarrow 1 : R$ **do**
    Randomly select a site s and an outcome $\omega \in \Omega - \{\pi_s\}$
    $\delta \leftarrow \mathcal{EU}(\pi_s^\omega) > \mathcal{EU}(\pi)$
    if $\delta > 0 \vee rand() > \exp^{-\delta/\tau(r)}$ **then**
        $\pi \leftarrow \pi_s^\omega$
    for $i \leftarrow 1 : T$ except s **do** ▷ Find the best permutation
        if $\mathcal{EU}(\pi_{i\leftrightarrow s}) > \mathcal{EU}(\pi)$ **then**
            $\pi \leftarrow \pi_{i\leftrightarrow s}$
return $\pi^a = \pi$

Time Complexity

$$O(TR)$$

Outline

Outline

Preference Elicitation

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

The challenge

How to reduce Uncertainty in user preferences:

- before negotiation (offline preference elicitation).
- while negotiating (online preference elicitation).

Preference Elicitation

The challenge

How to reduce Uncertainty in user preferences:

- before negotiation (offline preference elicitation).
- while negotiating (online preference elicitation).

Types of questions

Utility Value what is $\tilde{u}(\omega)$?

Utility Constraint Is $\tilde{u}(\omega) \geq x$? Usually implemented as a standard gamble.

Utility Comparison Is $\omega_1 \succ \omega_2$?

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: .

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: Do not ask about the utility directly..

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: Do not ask about the utility directly..
3. Practical elicitation uses a **series** of comparisons between outcomes to assess utilities.

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: Do not ask about the utility directly..
3. Practical elicitation uses a **series** of comparisons between outcomes to assess utilities.

A Gamble

(ω^*, ω_*, p) : Getting ω^* with probability p otherwise ω_*

Elicitation Procedures

1. Long history in the decision support and economics research community.
2. Take away message: Do not ask about the utility directly..
3. Practical elicitation uses a **series** of comparisons between outcomes to assess utilities.

A Gamble

(ω^*, ω_*, p) : Getting ω^* with probability p otherwise ω_*

Example query

Do you prefer to get ω for certain over (ω^*, ω_*, p)?

Elicitation Procedures/Strategies

Probability Equivalence

find p so that $\omega = (\omega^*, \omega_*, p)$

Certainty Equivalence

find ω so that $\omega = (\omega^*, \omega_*, p)$

- Both require *normalized* utilities.
- Both require knowledge of $\omega^* \succ \omega \succ \omega_*$.
- Lead to different biases.

Comparison-only Procedures

1. Titration-down: $p_k = 1 - s \times k$
2. Titration-up: $p_k = s \times k$
3. Ping-pong: $p_k = \begin{cases} s \times \lfloor k/2 \rfloor & \text{k is odd} \\ 1 - s \times k/2 & \text{k is even} \end{cases}$

Importance of Elicitation

Negotiation with Elicitation

$m,\ \Omega,\ R,\ \tilde{U}_i\ \forall 1 \leq i \leq m\ ,\ \hat{U}_i^0\ \forall 1 \leq i \leq m$

m Number of agents/actors

$\Omega = \{\omega_j\}$ Possible outcomes (assumed countable)

n Number of outcomes $|\Omega|$

$R(i) \equiv r_i$ Reserved value for agent i

$\tilde{U}_i : \Omega \to [0,1]$ Utility of outcomes to **actor** i

$\hat{U}_i^0 : \Omega \to P$ Probability distribution of utility values for **agent** i

$\hat{U}_{ij}^0 \equiv \hat{U}_i^0(\omega_j)$

$P : \{[0,1] \to [0,1]\}$ A probability distribution on the closed interval $[0,1]$

What is Elicitation Doing?

Reduces uncertainty in $\hat{U}$

State of the Art

- Lots of work on preferences/utility elicitation in decision making domain.
- Some work on incremental utility elicitation.
- Few works on incremental utility elicitation during negotiations

State of the Art

- Lots of work on preferences/utility elicitation in decision making domain.
- Some work on incremental utility elicitation.
- Few works on incremental utility elicitation during negotiations

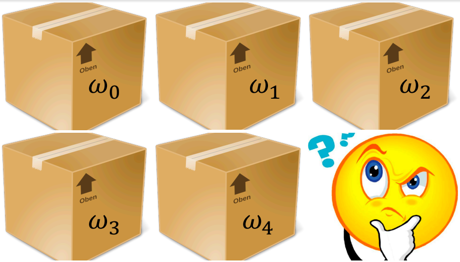
Why Is Negotiation Different

1. The acceptance model changes over time $\rightarrow$ environment dynamics are not static.
2. Exploration is extremely costly.
3. Usually negotiations are not repeated much.
4. Cannot train on a simulator (in most cases).

Outline

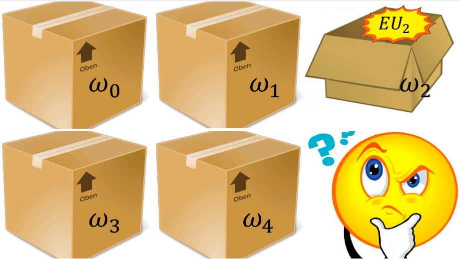
Pandora's Problem [Economics]

1. A set of n boxes ($\{\omega_j\}$).
2. Opening a box j gives a reward between 0 and ∞ according to distribution p_j after t_j time-steps, and costs c_j.
3. Future rewards are discounted with a known factor β.



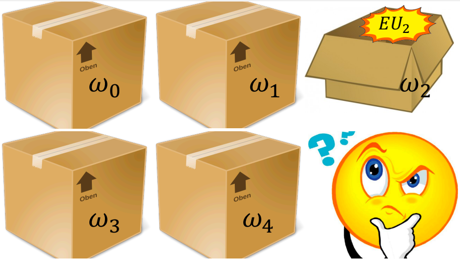
Pandora's Problem [Economics]

1. A set of n boxes ($\{\omega_j\}$).
2. Opening a box j gives a reward between 0 and ∞ according to distribution p_j after t_j time-steps, and costs c_j.
3. Future rewards are discounted with a known factor β.



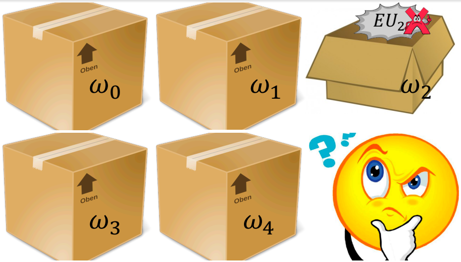
Pandora's Problem [Economics]

1. A set of n boxes ($\{\omega_j\}$).
2. Opening a box j gives a reward between 0 and ∞ according to distribution p_j after t_j time-steps, and costs c_j.
3. Future rewards are discounted with a known factor β.
4. Pandora's Problem:
   1. What is the optimal order to open the boxes?
   2. When should she stop?



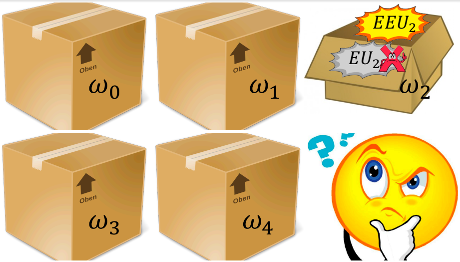
Pandora's Problem [Economics]

1. A set of n boxes ($\{\omega_j\}$).
2. Opening a box j gives a reward between 0 and ∞ according to distribution p_j after t_j time-steps, and costs c_j.
3. Future rewards are discounted with a known factor β.
4. Pandora's Problem:
   1. What is the optimal order to open the boxes?
   2. When should she stop?
5. Similar to elicitation (boxes = outcomes, open = query) but assumes that uncertainty is completely removed.



Pandora's Problem [Economics]

1. A set of n boxes ($\{\omega_j\}$).
2. Opening a box j gives a reward between 0 and ∞ according to distribution p_j after t_j time-steps, and costs c_j.
3. Future rewards are discounted with a known factor β.
4. Pandora's Problem:
   1. What is the optimal order to open the boxes?
   2. When should she stop?
5. Similar to elicitation (boxes = outcomes, open = query) but assumes that uncertainty is completely removed.



Pandora's Problem [Economics]

1. A set of n boxes ($\{\omega_j\}$).
2. Opening a box j gives a reward between 0 and ∞ according to distribution p_j after t_j time-steps, and costs c_j.
3. Future rewards are discounted with a known factor β.
4. Pandora's Problem:
   1. What is the optimal order to open the boxes?
   2. When should she stop?
5. Similar to elicitation (boxes = outcomes, open = query) but assumes that uncertainty is completely removed.

Solution: Pandora's Rule[a]

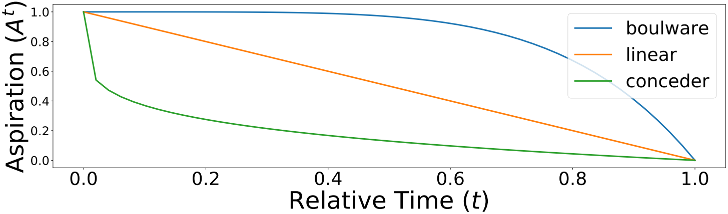
[a] Weitzman, "Optimal search for the best alternative".

For each box j, find z_j which is the solution to:

$$c_j = \beta_j \int_{z_j}^{\infty} (u - z_j)\, p_j(u)\, du - (1 - \beta_j)\, z_j$$

Optimal Elicitation[1]

Adapts Pandora's Rule to the negotiation context:



1. $\beta = 1.0$
2. Define aspiration level as: $A^t \equiv r_i + \left(A^0 - r_i\right) \times \left(1 - \frac{t}{N}^{1/e}\right)$
   $e > 1 \rightarrow$ Boulware, $e = 1 \rightarrow$ Linear, $e < 1 \rightarrow$ Conceder
3. $p_j = \Lambda_i^t(\omega_j) \times \mathbb{E}\left(\hat{U}_{ij}^t\right) + (1 - \Lambda_i^t(\omega_j)) \times A^t(\omega_j)$
4. Assume that there is an open box giving r_i with outcome index 0.
5. End the negotiation once the best box is 0.

[1] Tim Baarslag and Enrico H Gerding. "Optimal Incremental Preference Elicitation during Negotiation". In: IJCAI. 2015, pp. 3–9.

Why is OE sub-optimal?

Main Issue

Assuming that all uncertainty is removed by elicitation.

1. Assuming that $\hat{U}_{ij} \to \delta\left[u = \tilde{U}_i(\omega_j)\right]$

Why is OE sub-optimal?

Main Issue

Assuming that all uncertainty is removed by elicitation.

1. Assuming that $\hat{U}_{ij} \rightarrow \delta\left[u = \tilde{U}_i\left(\omega_j\right)\right]$
2. Consider any practical strategy (e.g. titration-down):
   - After the first question: $\hat{U}^t_{ij} \rightarrow \hat{U}^{t+1}_{ij}$
   - z_j was calculated using $\hat{U}^t_{ij}$ and must be recalculated.

Take-away message

Avoid deep-elicitation.

Extensions to Pandora's algorithm

Closed-form Calculation of z-index[a]

[a] Yasser Mohammad and Shinji Nakadai. "Utility Elicitation During Negotiation with Practical Elicitation Strategies". In: *IEEE SMC*. 2018.

$$z_j = \begin{cases} \frac{a+b}{2}\beta - c_j & z_j \leq a \\ \frac{-\lambda \pm \sqrt{\lambda^2 - 4\zeta}}{2} & a < z_j \leq b \end{cases}$$

λ $-2\left(b + \frac{a-\beta}{\beta}(b-a)\right)$

ζ $b^2 - \frac{2c_j}{\beta}(b-a)$

The balanced expectation operator

$$\mathcal{E}(\hat{U}_{ij}^t) = \frac{t}{N} \times Min\left(\hat{U}_{ij}^t\right) + \left(1 - \frac{t}{N}\right) \times Max\left(\hat{U}_{ij}^t\right)$$

Min/*Max* a *biased* estimator that exaggerate the lower/upper part of its input. For $U(a, b)$, $Min, Max = a, b$.

Outline

Value of Information Algorithm

- Based on[2] in decision-support context.
- Adapted to the negotiation context.

Main Idea

- Assume an accurate opponent model (acceptance probability)
- Given a set of queries $Q \rightarrow$ find the one with the maximum difference between the expected expected utility before and after asking it[a].

[a] Tim Baarslag and Michael Kaisers. "The value of information in automated negotiation: A decision model for eliciting user preferences". In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems. 2017, pp. 391–400; Yasser Mohammad and Shinji Nakadai. "FastVOI: Efficient Utility Elicitation During Negotiations". In: *International Conference on Principles and Practice of Multi-Agent Systems (PRIMA)*. Springer. 2018, pp. 560–567.

[2] Urszula Chajewska, Daphne Koller, and Ronald Parr. "Making rational decisions using adaptive utility elicitation". In: *AAAI/IAAI*. 2000, pp. 363–369.

VOI Based Elicitation

Policy

$$\pi^t = \left(\omega^t, \omega^{t+1}, \omega^N\right) \text{ where } \omega^x \in \Omega$$
$$K(\omega|\pi) \equiv \text{index of } \omega \text{ in } \pi$$
$$\pi(k) = \omega \text{ where } K(\omega|\pi) = k$$

Optimal Policy

$$\pi^{t*} = \arg\max \; EEU^t\left(\pi, \left\{\hat{U}^t\right\}\right)$$

VOI Based Elicitation

Policy

$$\pi^t = \left(\omega^t, \omega^{t+1}, \omega^N\right) \text{ where } \omega^x \in \Omega$$
$$K\left(\omega|\pi\right) \equiv \text{index of } \omega \text{ in } \pi$$
$$\pi\left(k\right) = \omega \text{ where } K\left(\omega|\pi\right) = k$$

Probability of Agreement

$$Pa^t\left(\omega|\pi\right) = \begin{cases} \Lambda^t\left(\omega\right) \prod_{k=1}^{K_\pi(\omega)-1} \left(1 - \Lambda^t\left(\pi\left(k\right)\right)\right) & \omega \in \pi \\ 0 & otherwise \end{cases}$$

Optimal Policy

VOI Based Elicitation

Policy

$$\pi^t = \left(\omega^t, \omega^{t+1}, \omega^N\right) \text{ where } \omega^x \in \Omega$$
$$K(\omega|\pi) \equiv \text{index of } \omega \text{ in } \pi$$
$$\pi(k) = \omega \text{ where } K(\omega|\pi) = k$$

Probability of Agreement

$$Pa^t(\omega|\pi) = \begin{cases} \Lambda^t(\omega) \prod_{k=1}^{K_\pi(\omega)-1} \left(1 - \Lambda^t(\pi(k))\right) & \omega \in \pi \\ 0 & otherwise \end{cases}$$

Expected Expected Utility[a]

[a] Boutilier, "On the foundations of expected expected utility".

$$EEU^t\left(\pi, \left\{\hat{U}^t_\omega\right\}\right) = \sum_{\omega\in\Omega} Pa(\omega|\pi)\, \mathbb{E}\left(\hat{U}^t_\omega\right)$$

Optimal Policy

$$\pi^{t*} = \arg\max\ EEU^t\left(\pi, \left\{\hat{U}^t\right\}\right)$$

VOI Based Elicitation II

Questions and Answers

$$Q \equiv \{q_l\}$$
$$q_l \equiv \left\{ \left(Ans_s^l, p_s \right) \right\}$$
$$Ans_s^l \equiv \left\{ \hat{U}_\omega^{t+1} \right\}$$
$$\sum_s p_s = 1$$

VOI Based Elicitation II

Questions and Answers

$$Q \equiv \{q_l\}$$
$$q_l \equiv \left\{\left(Ans_s^l, p_s\right)\right\}$$
$$Ans_s^l \equiv \left\{\hat{U}_\omega^{t+1}\right\}$$
$$\sum_s p_s = 1$$

Expected value of information

$$EVOI\left(q^l, \left\{\hat{U}_\omega^t\right\}\right) = \mathbb{E}_s\left(\max_\pi EEU\left(\pi, Ans_s^l\right)\right) - \max_\pi EEU\left(\pi, \left\{\hat{U}_\omega^t\right\}\right)$$

VOI Based Elicitation II

Questions and Answers

$$Q \equiv \{q_l\}$$
$$q_l \equiv \left\{ \left(Ans_s^l, p_s\right) \right\}$$
$$Ans_s^l \equiv \left\{ \hat{U}_\omega^{t+1} \right\}$$
$$\sum_s p_s = 1$$

Expected value of information

$$EVOI\left(q^l, \left\{\hat{U}_\omega^t\right\}\right) = \mathbb{E}_s\left(\max_\pi EEU\left(\pi, Ans_s^l\right)\right) - \max_\pi EEU\left(\pi, \left\{\hat{U}_\omega^t\right\}\right)$$

Elicitation

Ask q^* where

$$q^* = \arg\max_q \left(EVOI\left(q^l, \left\{\hat{U}_\omega^t\right\}\right) - c_q\right)$$

c_q Cost of asking question q

VOI main Issues

Accurate Agreement Model Assumption

- Everything depends on the probability of agreement (Pa)
- Pa depends on the **product** of probabilities in the acceptance model (Λ^t)

$$Pa^t(\omega|\pi) = \begin{cases} \Lambda^t(\omega) \prod_{k=1}^{K_\pi(\omega)-1} (1 - \Lambda^t(\pi(k))) & \omega \in \pi \\ 0 & otherwise \end{cases}$$

VOI main Issues

Accurate Agreement Model Assumption

- Everything depends on the probability of agreement (Pa)
- Pa depends on the **product** of probabilities in the acceptance model (Λ^t)

$$Pa^t(\omega|\pi) = \begin{cases} \Lambda^t(\omega) \prod_{k=1}^{K_\pi(\omega)-1} \left(1 - \Lambda^t(\pi(k))\right) & \omega \in \pi \\ 0 & otherwise \end{cases}$$

Speed: Complexity $= O(nN|Q||Ans|)$

- Too many *argmax* and $\mathbb{E}$ operations.
- Every policy extends to the end of the negotiation.

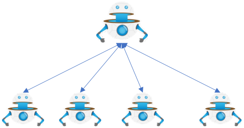
$$q^* = \arg\max_q \left(EVOI\left(q^l, \left\{\hat{U}^t_\omega\right\}\right) - c_q\right)$$

$$EVOI\left(q^l, \left\{\hat{U}^t_\omega\right\}\right) = \mathbb{E}_s\left(\max_\pi EEU\left(\pi, Ans^l_s\right)\right) - \max_\pi EEU\left(\pi, \left\{\hat{U}^t_\omega\right\}\right)$$

$$\pi^{t*} = \arg\max_\pi EEU^t\left(\pi, \left\{\hat{U}^t_\omega\right\}\right)$$

Outline

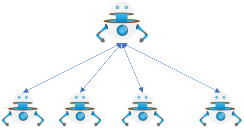
Concurrent Negotiation



Concurrent Negotiation

Generality

- Specific scenario (buyer-seller).
- General domain



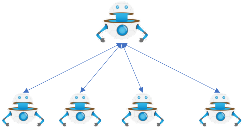
Concurrent Negotiation

Generality

- Specific scenario (buyer-seller).
- General domain

Decommitment

- Symmetric de-commitment.
- Asymmetric de-commitment.
- No de-commitment.



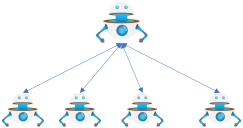
Concurrent Negotiation

Generality

- Specific scenario (buyer-seller).
- General domain

Decommitment

- Symmetric de-commitment.
- Asymmetric de-commitment.
- No de-commitment.



Timing

- Synchronous.
- Any-time.

Outline

Outline

Outline

References I

Baarslag, Tim and Enrico H Gerding. "Optimal Incremental Preference Elicitation during Negotiation.". In: *IJCAI*. 2015, pp. 3–9.

Baarslag, Tim and Michael Kaisers. "The value of information in automated negotiation: A decision model for eliciting user preferences". In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems. 2017, pp. 391–400.

Boutilier, Craig. "On the foundations of expected expected utility". In: *IJCAI*. Vol. 3. Citeseer. 2003, pp. 285–290.

Chajewska, Urszula, Daphne Koller, and Ronald Parr. "Making rational decisions using adaptive utility elicitation". In: *AAAI/IAAI*. 2000, pp. 363–369.

Mohammad, Yasser and Shinji Nakadai. "FastVOI: Efficient Utility Elicitation During Negotiations". In: *International Conference on Principles and Practice of Multi-Agent Systems (PRIMA)*. Springer. 2018, pp. 560–567.

– ."Utility Elicitation During Negotiation with Practical Elicitation Strategies". In: *IEEE SMC*. 2018.

References II

Weitzman, Martin L. "Optimal search for the best alternative". In: *Econometrica: Journal of the Econometric Society* (1979), pp. 641–654.

references

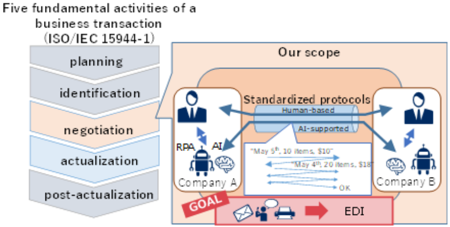
Automated Negotiation is important

Why is it hard?

- Mechanism Design Problem:
  - Better than haggling?
- Negotiator Design Problem:
  - Generality × Effectiveness

Why is it interesting?

- Easy to state yet hard to solve.
- Multiple levels of abstraction and complexity.
- Several concrete open questions.
- Vibrant yet not saturated research space.



attribution: UNECE eNegotiation Project

Automated Negotiating Agents Competition: 2010-

Automated Negotiation Has a Long History

Nash Bargning Game (1950)

How to split a pie when you have one offer? Maximize relative surplus product

Robenstien Bargaining Game (197s)

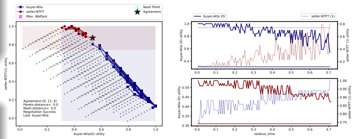
Do we need to negotiate if we know everything? Not really

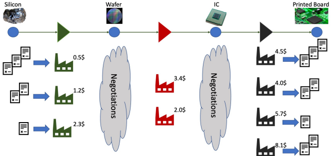
ANAC: 2010 and still running

Can we develop effective agents for automated negotiation in almost any domain? Kind of

SCML 2019 and still running

Can we develop effective agents that orchestrate multiple related negotiations while being embedded in a business like environment? Kind





An example of an SCM world showing four products (circles), three processes (triangles) and few factories. Each process consumes one item of its input and generates one output of its output in one day. Each factory requires a different cost to run its process (shown in its top right). Factories in the first level have exogenous contracts to buy raw material (silicon) and factories at the last level have exogenous contracts to sell the final product (printed boards). These contracts drive the market.

We have Nice Platforms?

Genius [1]

a Java-based negotiation platform to develop general negotiating agents and create negotiation scenarios.

GENIUS

General Environment for Negotiation with Intelligent multi-purpose Usage Simulation.

GeniusWeb

A distributed platform for automated negotiation on the internet

NegMAS [2]

a Python-based negotiation platform for developing autonomous negotiation agents embedded in simulation environments.

You know what it takes to build a negotiator

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory

Offer Policy

What should I offer next?

- Good Heuristics exist
  - Time-based
  - Tit-for-Tat
  - ANAC agents (more than 50)
- Good learners exist but the best ones require unrealistically large amounts of data.

Acceptnce Strategy

When should I accept my partner's offer?

- Several heuristics exist.
- We can learn good acceptance Strategies.
- Still, it may not be a good idea to separate acceptance from offering.

Opponent Model

There is too much still to be done

Strategies with Guarantees

Can we develop strategies that combine the guarantess of classic Game theoretic work with the effectiveness of modern heuristics?

Negotiation under uncertainty

How to behave when I do not know exactly what I want?

Concurrent Negotiation

How coordinate behavior in multiple negotiation threads?

Preference Elicitation

Should I ask? And what exactly should I ask about?