Automated Negotiation: Challenges and Tools

AAAI 2022 Tutorial

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 24, 2022

AAAI 2022 Tutorial Updated on February 24, 2022

Outline

1. Why [Business]?
2. Why [Academia]?
3. What?

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

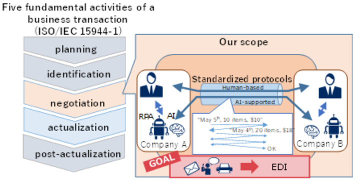
The Automated Negotiation Challenge

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/r
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

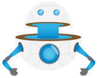
[4]Brown University, USA

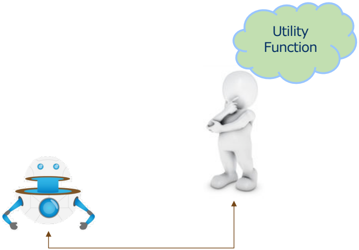
February 24, 2022

AAAI 2022 Tutorial Updated on February 24, 2022

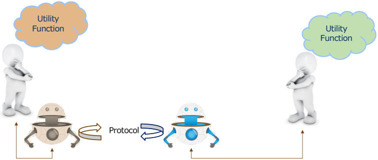
Outline

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

BROWN UNIVERSITY

NEC-AIST AI Cooperative Research Laboratory

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

1. Introduction Negotiation

2. Bargaining 101
   - Abstract Problem Definition
   - Bargaining Games and Solutions
   - Summary

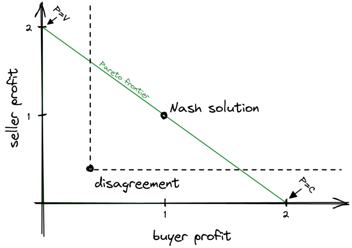
3. References

Outline

A Simple Trading Problem

- A buyer values a good at V
- A seller can create the good at cost C
- If $V > C$, then there is surplus $V - C$ to be gained (value creation)
- Bargaining problem: how much should the buyer pay the seller for the good? (value division)

- We might also assume there is an outside option (e.g., eBay), if the negotiation breaks down (i.e., they do not reach an agreement):
  - The buyer (seller) can buy (sell) the good elsewhere for slightly less



Sketch by Jackson de Campos

Abstract Bargaining Problem

The two-agent bargaining problem can be defined abstractly by

- A set $F \subset \Omega$ of feasible outcomes
- Two agents with utility functions $u_1, u_2 : \Omega \to \mathbb{R}$
- A disagreement point $\phi \in \Omega$, also called the status quo.
  The value $u_i(\phi)$ is called agent i's reservation value.

Individual rationality assumption: No agent will ever agree to a utility below their reservation value.

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

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

We can also compare lotteries (i.e., randomized outcomes): e.g., when I am very hungry,
"I prefer a banana with probability 90% to an apple with probability 50%."

Theorem[1] Given an agent with preferences over randomized outcomes that satisfy various axioms,
there exists a unique utility function $u : \Omega \to \mathbb{R}$ s.t. $\sigma \succ \tau$ iff $\mathbb{E}[u(\sigma)] > \mathbb{E}[u(\tau)]$), up to scaling.
The axioms are completeness, transitivity, continuity, and IIA.

Why is this relevant?

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

Outline

1. Introduction Negotiation

2. Bargaining 101
   - Abstract Problem Definition
   - Bargaining Games and Solutions
   - Summary

3. References

Nash's Bargaining Protocol

Arguably the simplest possible protocol.

- Both agents announce their offers simultaneously.
- If their offers are feasible, an agreement is reached.
- If not, the outcome is the status quo.

In the simple trading game,[2] we might define the agents' offers to be feasible iff
the price announced by the buyer is at least the price announced by the seller.

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

Nash[3] sought a theory of bargaining which would characterize a unique outcome of a negotiation.

[2] because there are now rules

Nash's Demand Game and his Solution

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 efficient 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[4] is the point at which the product of each agent's surplus is maximized:

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

This is the unique point that satisfies several natural axioms: efficiency,

Rubinstein's Bargaining Protocol

- Two agents are dividing a pie (the value, say 1, that can be created).
- Alternating offers game: players take turns making offers and counteroffers (proposed divisions).
  If ever one player accepts the other's offer, the pie is divided as specified.
  Otherwise, the game continues (potentially forever).
- Each agent is under some time pressure: $u_i^{t+\Delta}(\omega) < u_i^t(\omega)$, for all rounds t and $\Delta > 0$.
  E.g., exponential discounting: $u_i^{t+\Delta}(\omega) = \delta_i u_i^t(\omega)$, for some $\delta_i \in [0, 1)$.

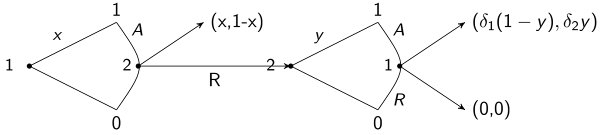


Image from Jack Fanning's actual Bargaining 101 class notes.

Two-Period Game

WLOG, assume P1 with discount factor δ_1 moves first, and P2 with discount factor δ_2 moves second.
Time pressures are common knowledge.

In period 2, the players play a subgame which is called an ultimatum game.

There is a unique equilibrium in this ultimatum game in which P2 offers P1 $\epsilon > 0$, and P1 accepts.
As ϵ can be arbitrarily small, the payoffs in this subgame are $(0, \delta_2)$.

Working backwards (backward induction), P1 knows that P2's payoffs in this game will be at least δ_2
(if P2 rejects). So P1 must offer P2 at least δ_2.

In the subgame perfect equilibrium of this game, P1 offers P2 δ_2 in the first round, and P2 accepts.
The equilibrium payoffs are thus $(1 - \delta_2, \delta_2)$.

Conclusions

Infinite-Horizon Game

P1 places offers on odd rounds. P2 places offers on even rounds.

Because the game has an infinite horizon, all rounds of negotiation are strategically identical to round 1!
Backwards induction won't work! Rubinstein's solution is ingenious!

Theorem[5]
The unique subgame perfect equilibrium of this game is for P1 to offer P2 as follows in the first round, and for P2 to accept:

$$\left(\frac{1-\delta_2}{1-\delta_1\delta_2}, \frac{\delta_2\left(1-\delta_1\right)}{1-\delta_1\delta_2}\right)$$

Conclusions

- Equilibrium is unique.
- Agreement is immediate.
- The higher P2's discount factor (the more patient they are), the higher P2's payoffs

Negotiation with Complete Information

Hick's Paradox

Why do rational parties negotiate when they have full information?

Labor vs. Management

- A union negotiating (in rounds) with management about a wage raise.
- Both parties are perfectly rational and fully informed.
- Threat: the union *can* strike.

Theorem[6] Subgame perfect equilibria exist in which there is a finite strike followed by agreement.

[6] Raquel Fernandez and Jacob Glazer. *Striking for a bargain between two completely informed agents*. Tech. rep. National Bureau of Economic Research, 1989.

Negotiation with Incomplete Information

Desiderata:

- Efficient outcome.
- Individual rationality (IR).
- Incentive compatible (IC).
- Budget balance (BB).

Myerson-Satterthwaite Impossibility Result

Theorem:[7] No mechanism can achieve all four of these desiderata.

- A buyer values a good at V.
- A seller can create the good at cost C.
- V (C) is private information, known only to the buyer (seller).
- There is no IR, IC, and BB mechanism that results in agreement, for all $V > C$ values.

[7] Roger B Myerson and Mark A Satterthwaite. "Efficient mechanisms for bilateral trading". In: *Journal of economic theory* 29.2 (1983), pp. 265–281.

Outline

1. Introduction Negotiation
2. Bargaining 101
   - Abstract Problem Definition
   - Bargaining Games and Solutions
   - Summary
3. References

Summary

- Human negotiation is important. Automated negotiation is posed to become as important!
- Bargaining theory dates back at least to Nash (1950), with many beautiful theorems
  (e.g., Rubinstein's) for simplified negotiation scenarios.
- The rest of this tutorial mostly concerns automated negotiation, assuming an alternating offers protocol, without full information (about your opponent, or perhaps even yourself!).

Outline

1. Introduction Negotiation
2. Bargaining 101
3. References

References I

Fernandez, Raquel and Jacob Glazer. *Striking for a bargain between two completely informed agents*. Tech. rep. National Bureau of Economic Research, 1989.

Myerson, Roger B and Mark A Satterthwaite. "Efficient mechanisms for bilateral trading". In: *Journal of economic theory* 29.2 (1983), pp. 265–281.

Nash, John. "The Bargaining Problem". In: *Econometrica* 18.2 (Apr. 1950), pp. 155–162.

Neumann, J. von and O. Morgenstern. *Theory of games and economic behavior*. Princeton University Press, 1947.

Rubinstein, Ariel. "Perfect Equilibrium in A Bargaining Model". In: *Econometrica* 50 (Feb. 1982), pp. 97–109.

references

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 24, 2022

AAAI 2022 Tutorial Updated on February 24, 2022

Outline

Outline

Some Automated Negotiation Platforms

Genius [1]

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

GeniusWeb

A distributed platform for automated negotiation on the internet

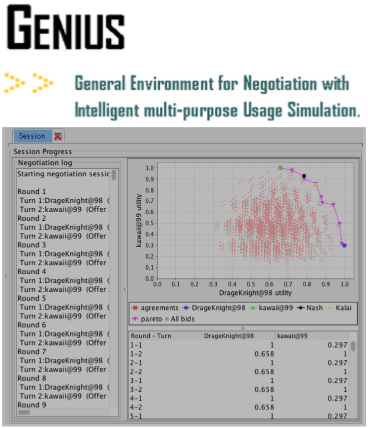
NegMAS [2]

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

GENIUS

Genius[3]

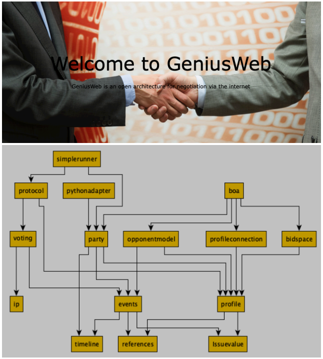
- Originally Developed by University of Delft and University of Bar Ilan. Currently maintained by a consortium of researchers.
- Since 2008 and used in all ANAC competitions
- The defacto-standard.
- Java-based [4].
- Has a GUI for running negotiations and tournaments.
- Includes SOTA agents from ANAC competitions since 2010.



GeniusWeb

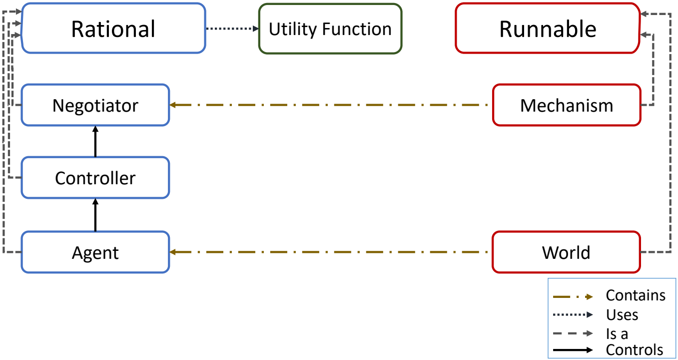
GeniusWeb [5]

- 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.



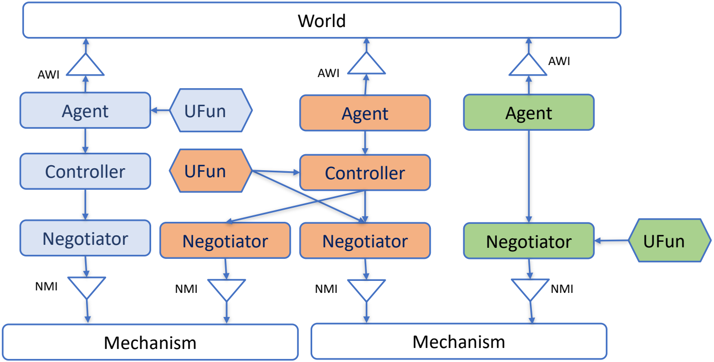
[5] GeniusWeb Team. *GeniusWeb Website*. 2021. URL: https://tracinsy.ewi.tudelft.nl/pubtrac/GeniusWeb.

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: $\{v_i \mid v_i \in I\}$
Ordinal	with defined order
Cardinal	with defined difference
Numeric	with defined numeric value (integer/real)

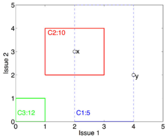
NegMAS

```
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")])
])
```

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

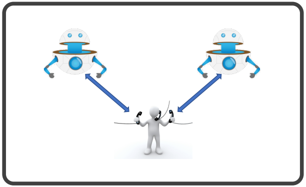


Outline

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.



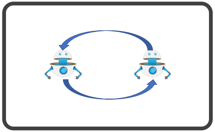
Examples

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.



Examples

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

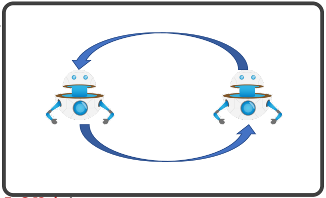
Rubinstein Bargaining Protocol Infinite horizon, single issue, bilateral protocol with complete information[a].

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

[a] Ariel Rubinstein. "Perfect equilibrium in a bargaining model". In: Econometrica: Journal of the Econometric

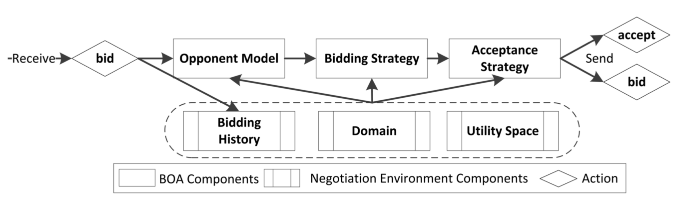
Stacked Alternating Offers Protocol

```
n_agreed, current = 0, randint(0, n_agents)
offer = agents[current].offer()
while not timeout()
        current = (current + 1) % n_agents
        response = agents[current].res
        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"
```



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.

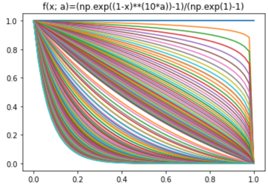
jupyter

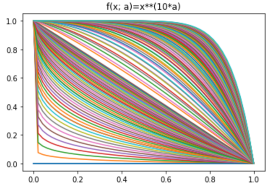
Outline

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.





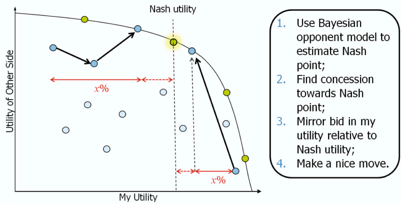
Behavioral Offer Policies

Behvior 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.



[9] Tim Baarslag, 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.

jupyter

Outline

Acceptance Policy

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=m
ACBestAll = ACNext() or (ACtauime(tau) and ACLastKReceived())
```

[10] Tim Baarslag, Koen Hindriks, and Catholijn Jonker. "Effective acceptance conditions in real-time automated negotiation".

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.

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

BROWN UNIVERSITY

NEC-AIST AI Cooperative Research Laboratory

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.

References II

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.

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/.

References III

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

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 24, 2022

AAAI 2022 Tutorial Updated on February 24, 2022

Outline

Outline

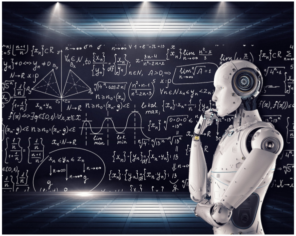
Learning in Automated Negotiation

What?

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

When?

1. Within Negotiation
2. Between Negotiations



"Artificial Intelligence & AI & Machine Learning" by mikemacmarketing

How?

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

Outline

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

Outline

Multiaremed Bandits for Repeated Negotiations

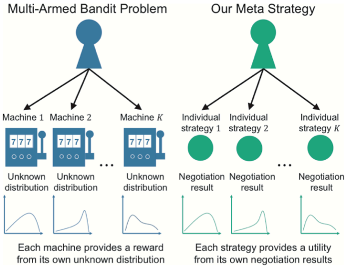
BROWN UNIVERSITY

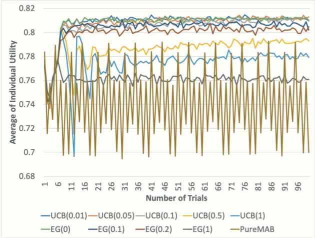
NEC-AIST AI Cooperative Research Laboratory

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

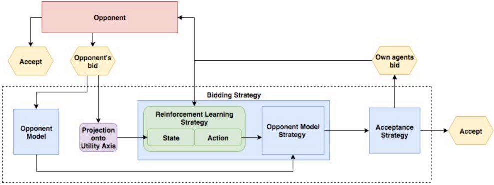
$$UCB(s) = \hat{\mu}_s + c\sqrt{\frac{lnN}{N_s}}$$





Agent	Individual utility	Social welfare
UCB(0.01)	**0.7734**	1.4575

RLBOA: Learning Offering Strategy[2]



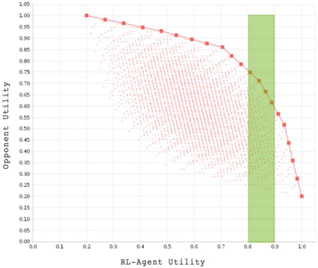
Main Points

- 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.

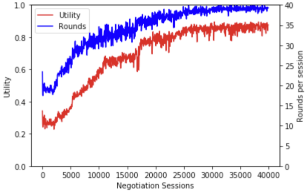
RLBOA: The details

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

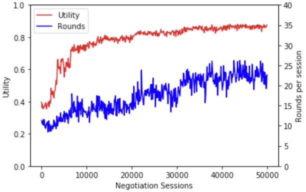


$$a(\omega) = \begin{cases} \text{Accept}, & \text{if } \alpha u(\omega) + \beta \geq u(o(s)) \end{cases}$$

RLBOA: Evaluation and Results



(a) Scenario generality experiment against the Boulware agent.



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

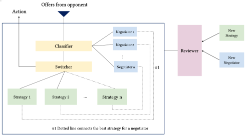
- Partners: TFT, Boulware TB
- Projection into one's utility space is surprisingly effective.
- Faster and better agreements!

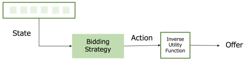
Domain	**Outcome space**	**Low opp.**	**High opp.**
Small	256	0.2615	0.5178
Medium	3.125	0.3111	0.5444
Large	46.656	0.2595	0.5250

A Framework for Learning Offer Strategies

Main Idea[5]

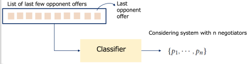
- Uses RL for learning approximate best responses to some agents.
- Uses Supervised Learning to learn a realtime switching strategy between learned best responses.
- Uses a form of Unsupervised Learning for adapting the system to new partner types.



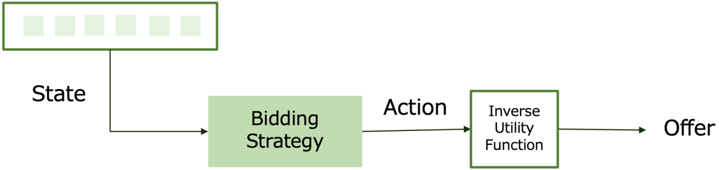


Phases

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



Before: Learning Approximate Best Response



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.

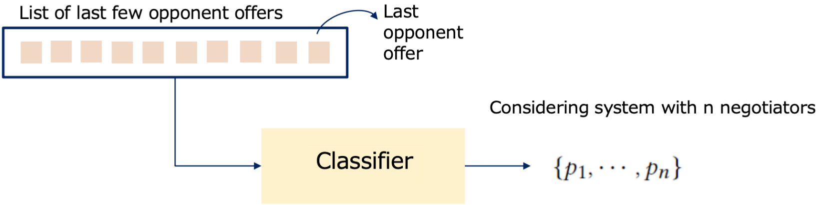
Trainer Soft Actor Critic (SAC)

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

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

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

During: Learning realtime Partner Classification



The SL Components

Features Opponent last K offers.

Target Opponent Type (discrete set)

After: Reviewing New Pairs

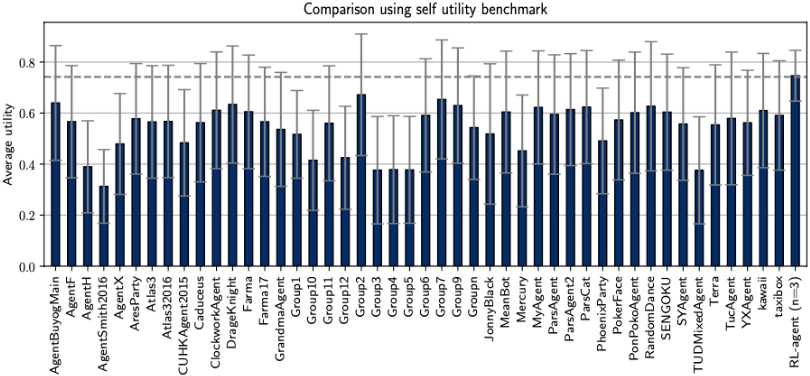
New Partner Type (N_{new}) Ecncountered

- 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 $\downarrow$.

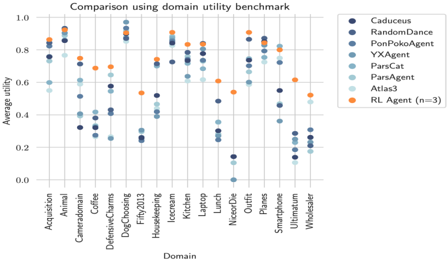
Update Best Responses

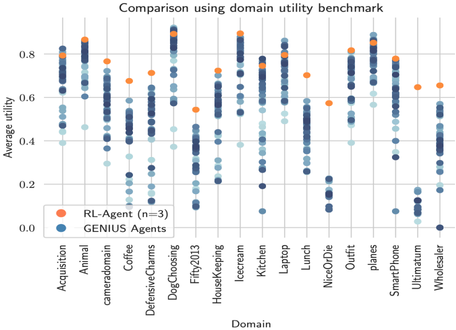
- For every learned ABR, negotiator pair (S, N):
  - Evaluate S_{new} against $N \rightarrow U(S_{new})$
  - Evaluate S against $N \rightarrow U(S)$
  - Replace S with S_{new} iff $\alpha U(S) < U(S_{new})$

Results: Against Different Opponents



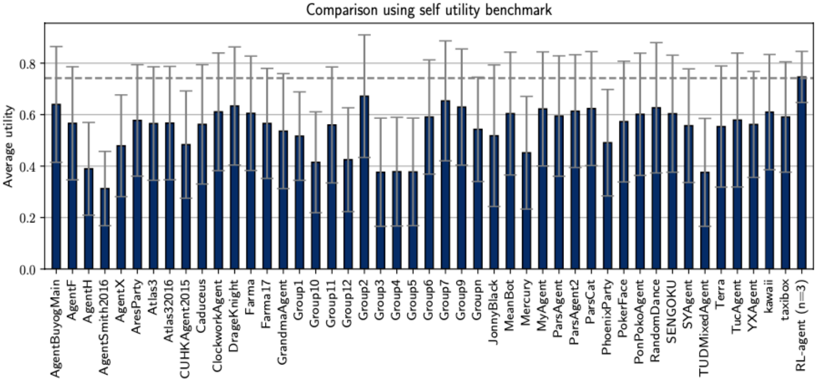
Results: In Different Domains



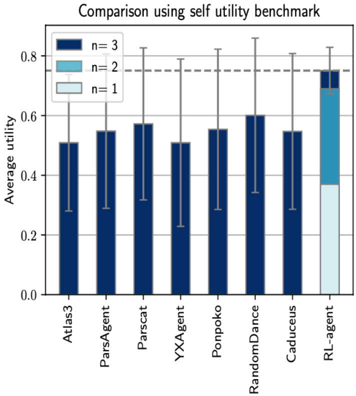


Results: Compared with SOTA Agents

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory



Results: Improvement with new best responses



Outline

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

DQN for learning Acceptance Strategy[6]

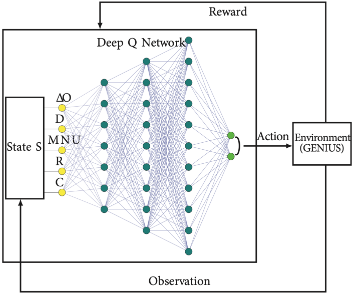
Main Idea

- Learning the acceptance strategy for a fixed offering strategy.

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

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



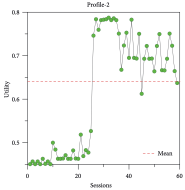
Evaluation

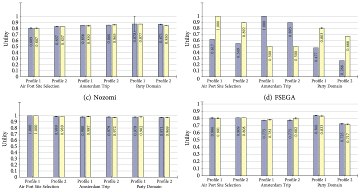
Training

- **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,





(c) Nozomi
(d) FSEGA
(e) Agent Smith
(f) Pars Agent

Outline

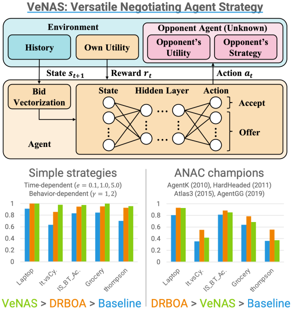
Learning Offer and Acceptance Policies together

- 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



Outline

1. Learning in Automated Negotiation

2. Learning how to negotiate

3. Learning Preferences
   - Partner Preferences: Opponent Modeling
   - Own Preferences: Elicitation

4. References

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

```
# M issues and N values per issue
F, alpha = np.zeros((M, N)), np.zeros(M)
epsilon, last_offer = 0.02, None
def after_receiving(self, state, offer): # update model
    if not self.last_offer:
        self.last_offer = offer
        return
    for i in range(M):
        if offer[i] != self.last_offer[i]:
            continue
        self.F[i, offer[i]] += 1
        self.alpha[i] += self.epsilon
    self.alpha /= self.alpha.sum()

def eval(self, offer): # estimate partner's utility value
```

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).

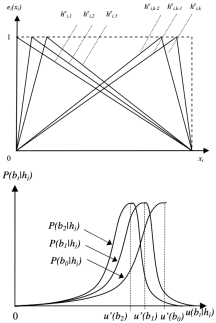
Bayesian Opponent Model Learner[9]

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".

Outline

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: 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

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

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

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

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

$P : \{[0,1] \to [0,1]\}$ 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

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

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

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

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

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)

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

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

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

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

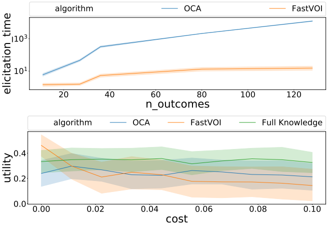
$$EVOI\left(q^l, \left\{\hat{U}^t_\omega\right\}\right) = \mathbb{E}_s\left(\max_\pi EEU\left(\pi, Ans^l_s\right)\right) - \max_\pi EEU\left(\pi, \left\{\hat{U}^t_\omega\right\}\right)$$

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

Extending VOI

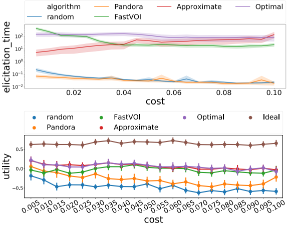
FastVOI[14]

- 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

1. Learning in Automated Negotiation
2. Learning how to negotiate
3. Learning Preferences
4. References

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.

References II

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.

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.

References III

Mohammad, Yasser 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.

– .“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 IV

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 24, 2022

AAAI 2022 Tutorial Updated on February 24, 2022

Outline

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

Outline

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]
- More to come[4].

[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.

Outline

1. Automated Negotiation in SCML

2. The SCML Game
   - SCML-OneShot
   - Simulation Steps

3. References

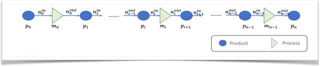
SCML World

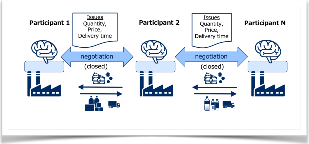
Challenge

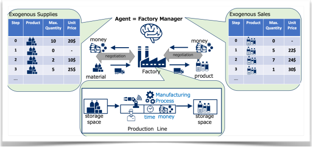
- Negotiation game with imperfect information
- Concurrent negotiations.
- Repeated negotiations → OneShot.
- Sequential negotiations → Standard.

Information

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







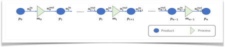
SCML Competition

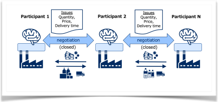
Competition Details

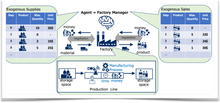
- Runs as part of ANAC IJCAI.
- You control one or more factories.
  - **Oneshot track** → one factory (predefined ufun).
  - **Standard track** → one factory (you define your own ufun).
  - **Collusion track** → multiple factories (3).

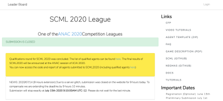
Flavors

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



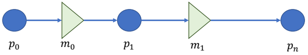


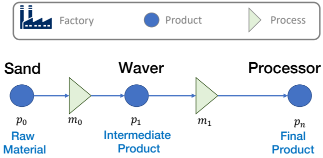




Outline

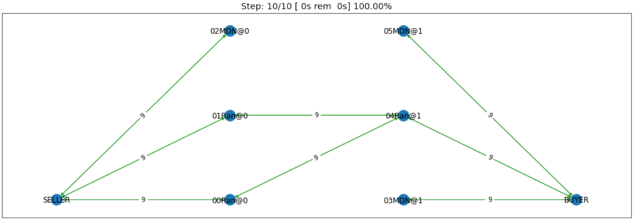
Overview





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

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory

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.

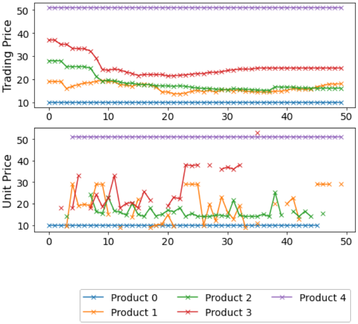
Dynamic Information

- 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

1. Automated Negotiation in SCML
2. The SCML Game
   - SCML-OneShot
   - Simulation Steps
3. References

Simulation Steps

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory

Once

Initialize all agents
- init()

Once

Initialize all agents
- init()

Update trading prices

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 24, 2022

AAAI 2022 Tutorial Updated on February 24, 2022

1/ 26

Outline

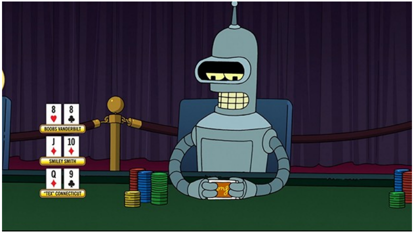
Outline

BROWN UNIVERSITY

NEC-AIST AI Cooperative Research Laboratory

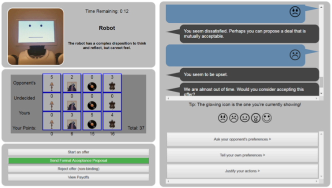
AI has learned to play games (well!)

BROWN UNIVERSITY
NEC-AIST AI Cooperative Research Laboratory



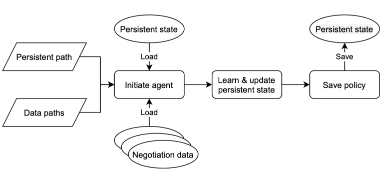
ANAC 2021 Leagues

HUMAINE: Negotiating with a Human

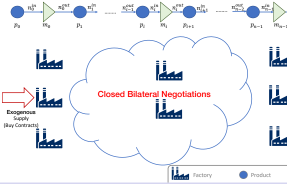


Werewolf: Negotiating with Natu

ANL: Learning in Repeated Negotiations (Preference Elicitation)



SCML: Supply Chain Managem



Game Characteristics

Solved Games

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

Agent Negotiation

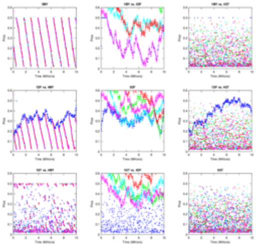
- Sequential decision making (e.g., turn-taking)
- Imperfect (and perfect) information
- Two-player & multi-player
- General-sum

Empirical Game-Theoretic Analysis (EGTA)

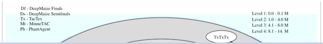
EGTA is a MAS tool for analyzing multiagent interactions.[1]

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).[2,3]



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



Double Oracle Algorithm[4](PSRO[5])

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^*

Outline

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 whose strategy does depend *not* on our actions.
E.g., an aspiration agent.

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

- States: negotiation rounds plus 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.

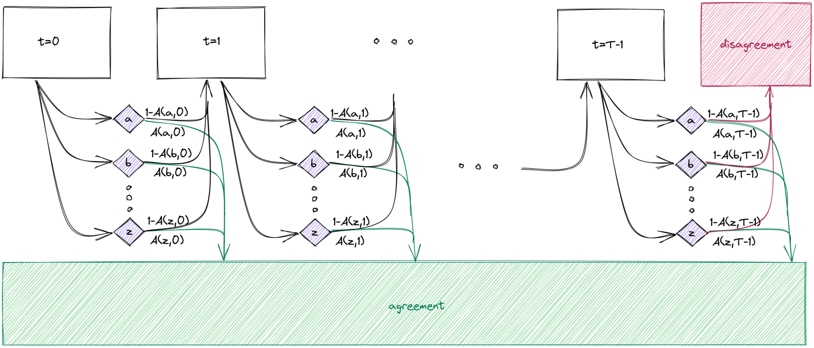
(Opponent ufuns are not modelled, because our decisions are conditionally independent of them, given an acceptance model and an offer policy.)

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)

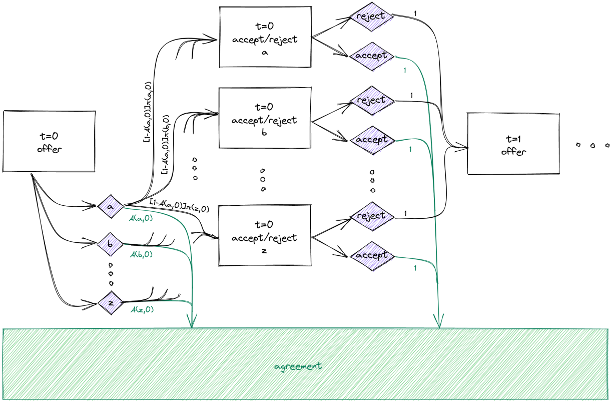
Time = negotiation round.

TDAM MDP



Sketch by Jackson de Campos

TDAM + TDOP MDP



Sketch by Jackson de Campos

EGTA, Revisited

BROWN UNIVERSITY NEC-AIST AI Cooperative Research Laboratory

- Goal: 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: i.e., a best response.
- 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

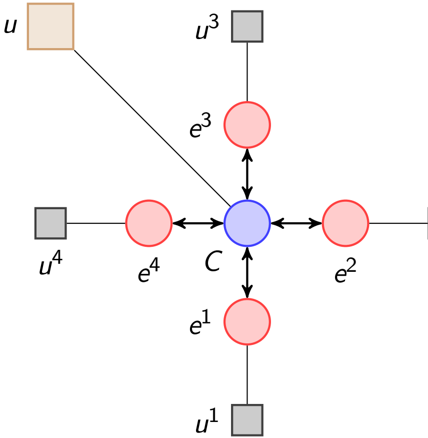
1. AI in Games
2. Finding a Best Response to a Fixed Opponent
3. Finding a Best Response to Multiple Fixed Opponents
4. Conclusion
5. References

Concurrent Negotiations

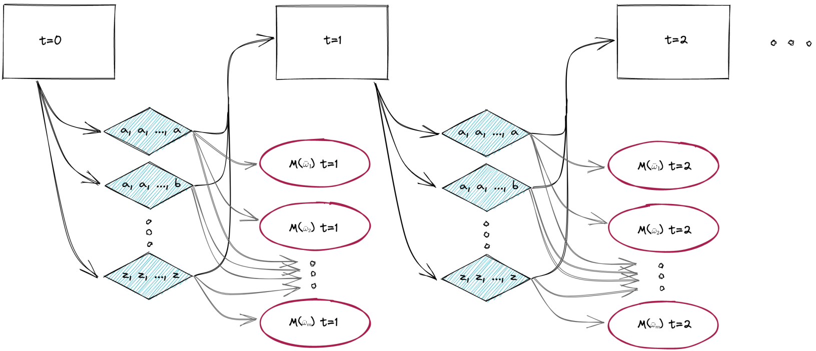
These negotiations are dependent, as evidenced by the global ufun u, which depends on a global outcome: i.e., the outcomes of all negotiations.

The problem of how to negotiate in this setting is reminiscent of how to bid in simultaneous auctions, when bidders' utilities are combinatorial.[6] 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



(A very rough) Concurrent TDAM MDP



Sketch by Jackson de Campos

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 $\Omega = \prod_{i=1}^{n} \Omega_i$.

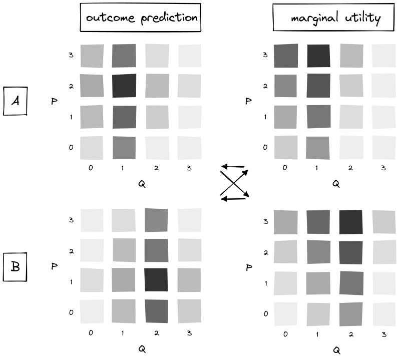
Given an outcome prediction P, the expected marginal utility μ_i of outcome $\omega_i \in \Omega_i$ in negotiation i is:

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

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 3 goods to sell
- 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 A and 2 to B (at the highest possible prices)
- The arrows indicate dependencies in the expected marginal utility calculations



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

Outcome Predictions

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.

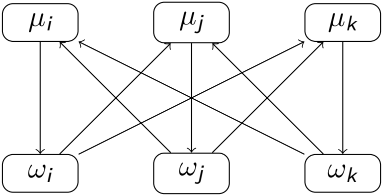
  $P(\omega \mid \text{external factors})$

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

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

- 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).

Introspective Equilibrium



An introspective equilibrium[7] 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 $\boldsymbol{f})) = f_i(\omega_i; \cdot)$.

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

Open Questions

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

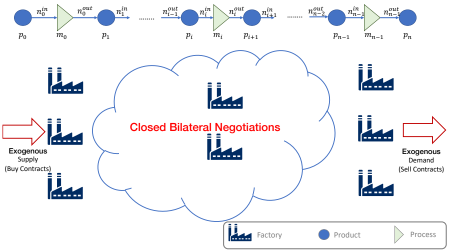
Can we build reliable static and dynamic prediction models? What about introspective models?

When does simple iteration (update strategy; update model; update marginal utilities; repeat)
converge to an introspective equilibrium?

Outline

Summary

- Negotiation is important! Humans negotiate all the time, so negotiation is yet another area ripe for automation.
- Bargaining theory dates back at least to Nash (1950), with many beautiful theorems for simple negotiation games.
- ANAC: much more complicated negotiation games.



Summary

What does AI bring to the table?

- Reinforcement and deep learning to learn negotiation strategies: i.e., offer and acceptance policies
- Preference elicitation: agents to learn about the users they represent (analogous problem in recommender systems)
- Frequentist and Bayesian learning of opponents' ufuns
- Empirical game-theoretic analysis

Open Questions

Almost everything is open!

- Can we develop an agent that best-responds to a class of opponents: to e.g., aspiration agents?
- Can we develop an agent for concurrent negotiations that is near optimal in that the outcomes it generates well approximate the optimal outcomes in the corresponding global negotiation problem?
- What more can we say about preference elicitation, beyond what is presently known for very fragile cases (e.g., query independence)?
- The mechanism design problem: Are there other negotiation protocols worthy of study beyond alternating offers?

Call to Arms!

- Negotiation is a core problem in economics, and is becoming an ever more important problem in AI.
- The AI negotiation problems are complex, but they are also relatively unexplored. There is still plenty of low hanging fruit.
- ANAC has been running for over a decade. It is a vibrant research community, with a low barrier to entry. Join us!

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

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.

References II

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

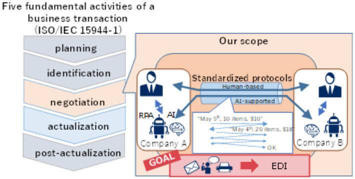
Automated Negotiation is important

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.



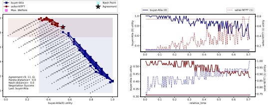
attribution: UNECE eNegotiation Project

Automated Negotiating Agents Competition: 2010-

Automated Negotiation Has a Long History

Nash Bargning Game (1950)

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

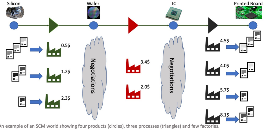


Robenstien Bargaining Game (197s)

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



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.

SCML 2019 and still running

Can we develop effective agents that orchestrate

We have Nice Platforms?

Genius [1]

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

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.

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?

Concurrent Negotiation

How coordinate behavior in multiple negotiation threads?

Negotiation under uncertainty

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

Preference Elicitation

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