Reinforcement Learning for Automated Negotiation

SCML as an Example

Yasser Mohammad [1, 2, 3]

[1] NEC Corporation, Global Innovation Unit

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

[3] Assiut University, Egypt

November 27, 2023

AJCAI 2023 Tutorial Updated on November 27, 2023

Outline

Why Now?

1. Industries are moving online.
2. Automation: Factory floor $\rightarrow$ 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

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

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].

[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

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]

[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

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".

The Automated Negotiation Challenge

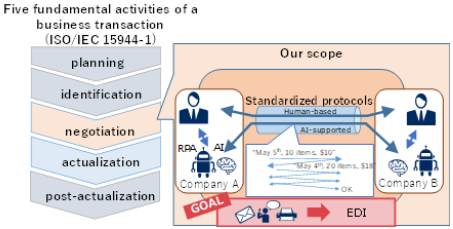
NEC-AIST AI Cooperative Research Laboratory

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-

Tutorial Outline

1. Theoretical Session (55min) Break (5min)
   - Theoretical Foundation
   - The Competition
2. Development Environment (15min) Break (5min)
3. RL for SCML (35min)
4. Concluding Remarks (5min)

Materials

1. Tutorial Website: http://yasserm.com/tutorial-ajcai2023/
2. Github Repository: https://github.com/yasserfarouk/ajcai2023autoneg
3. Handouts: https://github.com/yasserfarouk/ajcai2023autoneg/raw/main/ajcai2023autoneg.pdf
4. Negmas Documentation: https://negmas.readthedocs.io
5. SCML Documentation: https://scml.readthedocs.io
6. SCML Competition: https://scml.cs.brown.edu

Outline

Outline

Motivation

NEC-AIST AI Cooperative Research Laboratory

Goal

define and understand the automated negotiation problem in relation to other AI problems.

Summary

- AN in general is a General Game Playing problem in which you are **not** told the game.
- Depending on how much prior information, AN can vary in difficulty.

Types of Games

NEC-AIST AI Cooperative Research Laboratory

Copmlete Information Game

- All game parameters are **common knowledge** except player's policies.
- All information sets has a single node.

Imperfect Information Game

- Some partner infomration is unknown.
- Some information sets has multiple nodes.

Incomplete Information Game

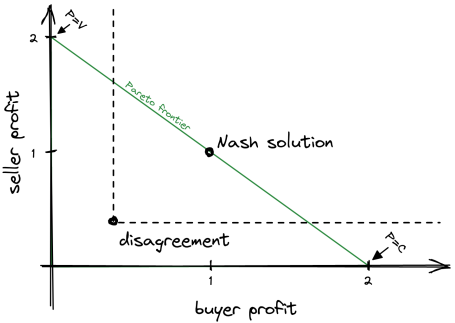
- The game is not fully specified (e.g. unknown opponent ufun)
- Can be convereted to an Imprefect information game by adding **Nature** as a player.

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 than V (more than C)



Sketch by Jackson de Campos

No-go Theorem

NEC-AIST AI Cooperative Research Laboratory

Desiderata:

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

Myerson-Satterthwaite Impossibility Result

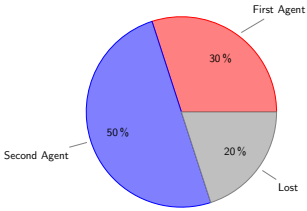
Theorem:[5] No mechanism can achieve all four of these desiderata.

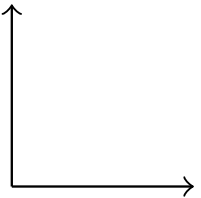
- A buyer values a good at V.
- A seller can create the good at cost C.
- V (C) is private information, known only to the buyer (seller).
- There is no IR, IC, and BB mechanism that results in agreement, for all $V > C$ values.

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

Nash Bargaining Game: Description

A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:



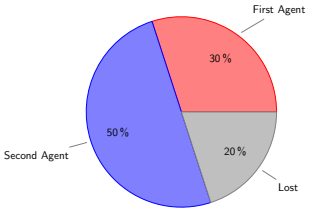


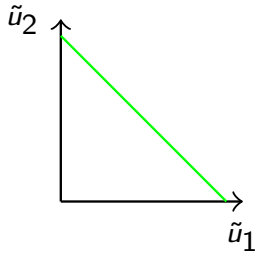
Nash Bargaining Game: Description

A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$



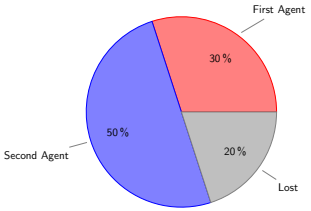


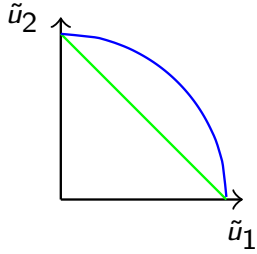
Nash Bargaining Game: Description

A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$



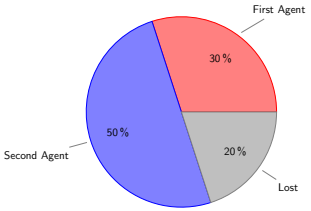


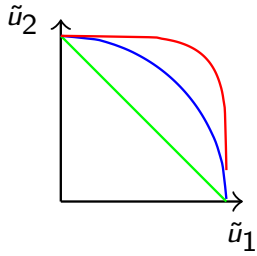
Nash Bargaining Game: Description

A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$





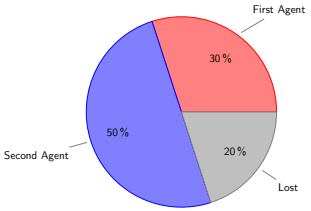
Nash Bargaining Game: Description

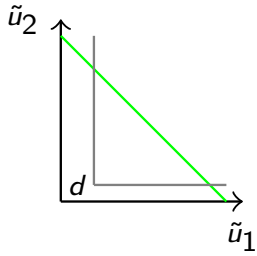
A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$

- A disagreement point $d \equiv \tilde{u}_1(\phi) + \tilde{u}_2(\phi) \in \Re^2$ which is the utility value received by the two players in case of disagreement (reserved values).





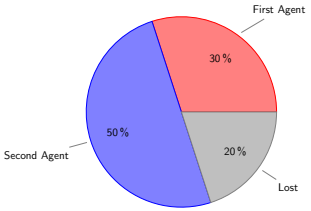
Nash Bargaining Game: Description

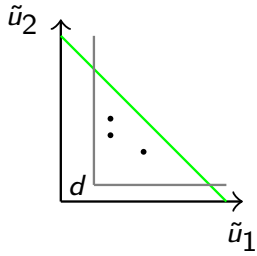
A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$

- A disagreement point $d \equiv \tilde{u}_1(\phi) + \tilde{u}_2(\phi) \in \Re^2$ which is the utility value received by the two players in case of disagreement (reserved values).





Other Bargaining Solutions

NEC-AIST AI Cooperative Research Laboratory

- Nash Bargaining Solution (1950): The point at which the product of surplus utility (above reservation value) of negotiators is maximized

$$\arg\max_{\omega_1,\omega_2} \prod_{i=1}^{2} (u_i(\omega_i) - u_i(\phi))$$

- Kalai-Smorodinsky Bargaining Solution (1975): The Pareto outcome with equal ratios of achieved surplus utility and maximum feasible surplus utility

$$\arg\max_{\omega_1,\omega_2 \in F} (\omega_1 + \omega_2) \text{ s.t. } \left(\frac{u_1(\omega_1) - u_1(\phi)}{u_2(\omega_2) - u_2(\phi)} = \frac{\max_{v \in F} (u_1(v)) - u_1(\phi)}{\max_{v \in F} (u_2(v)) - u_2(\phi)} \right)$$

- Kalai Bargaining Solution (1977): The Pareto outcome maximizing the utility for the unfortunate player. Defining P as the Pareto front

$$\arg\max_{\omega_1,\omega_2 \in P} \min_{i \in \{1,2\}} (u_i(\omega_i) - u_i(\phi))$$

Rubinstein's Bargaining Protocol: Description

The Game

- Two agents sharing a pie.

Rubinstein's Bargaining Protocol: Description

NEC-AIST AI Cooperative Research Laboratory

The Game

- Two agents sharing a pie.
- Each agent is under a different time-pressure: $u_i^{t+\Delta}(\omega) < u_i^t(\omega)$. Examples of time-pressure:

Rubinstein's Bargaining Protocol: Description

NEC-AIST AI Cooperative Research Laboratory

The Game

- Two agents sharing a pie.
- Each agent is under a different time-pressure: $u_i^{t+\Delta}(\omega) < u_i^t(\omega)$. Examples of time-pressure:
  - Exponential $u_i^{t+\Delta}(\omega) = \delta_i^{\Delta} u_i^t(\omega)$.
  - Linear $u_i^{t+\Delta}(\omega) = u_i^t(\omega) - \Delta c_i$

Rubinstein's Bargaining Protocol: Description

The Game

- Two agents sharing a pie.
- Each agent is under a different time-pressure: $u_i^{t+\Delta}(\omega) < u_i^t(\omega)$. Examples of time-pressure:
  - Exponential $u_i^{t+\Delta}(\omega) = \delta_i^{\Delta} u_i^t(\omega)$.
  - Linear $u_i^{t+\Delta}(\omega) = u_i^t(\omega) - \Delta c_i$
- Agent's initial utility is the assigned part of the pie: $u_i^0 = \omega_i$.

Rubinstein's Bargaining Protocol: Description

NEC-AIST AI Cooperative Research Laboratory

The Game

- Two agents sharing a pie.
- Each agent is under a different time-pressure: $u_i^{t+\Delta}(\omega) < u_i^t(\omega)$. Examples of time-pressure:
  - Exponential $u_i^{t+\Delta}(\omega) = \delta_i^{\Delta} u_i^t(\omega)$.
  - Linear $u_i^{t+\Delta}(\omega) = u_i^t(\omega) - \Delta c_i$
- Agent's initial utility is the assigned part of the pie: $u_i^0 = \omega_i$.
- Time pressure and utility information are common knowledge.

Rubinstein's Bargaining Protocol: Description

NEC-AIST AI Cooperative Research Laboratory

The Game

- Two agents sharing a pie.
- Each agent is under a different time-pressure: $u_i^{t+\Delta}(\omega) < u_i^t(\omega)$. Examples of time-pressure:
  - Exponential $u_i^{t+\Delta}(\omega) = \delta_i^{\Delta} u_i^t(\omega)$.
  - Linear $u_i^{t+\Delta}(\omega) = u_i^t(\omega) - \Delta c_i$
- Agent's initial utility is the assigned part of the pie: $u_i^0 = \omega_i$.
- Time pressure and utility information are common knowledge.
- No externally imposed time-limit.

Rubinstein's Bargaining Protocol: Description

NEC-AIST AI Cooperative Research Laboratory

The Game

- Two agents sharing a pie.
- Each agent is under a different time-pressure: $u_i^{t+\Delta}(\omega) < u_i^t(\omega)$. Examples of time-pressure:
  - Exponential $u_i^{t+\Delta}(\omega) = \delta_i^{\Delta} u_i^t(\omega)$.
  - Linear $u_i^{t+\Delta}(\omega) = u_i^t(\omega) - \Delta c_i$
- Agent's initial utility is the assigned part of the pie: $u_i^0 = \omega_i$.
- Time pressure and utility information are common knowledge.
- No externally imposed time-limit.
- Zero reservation value: $u_i^{\tau}(\phi) = 0 \ \forall \tau$.

Rubinstein's Bargaining Protocol: Description

NEC-AIST AI Cooperative Research Laboratory

The Game

- Two agents sharing a pie.
- Each agent is under a different time-pressure: $u_i^{t+\Delta}(\omega) < u_i^t(\omega)$. Examples of time-pressure:
  - Exponential $u_i^{t+\Delta}(\omega) = \delta_i^{\Delta} u_i^t(\omega)$.
  - Linear $u_i^{t+\Delta}(\omega) = u_i^t(\omega) - \Delta c_i$
- Agent's initial utility is the assigned part of the pie: $u_i^0 = \omega_i$.
- Time pressure and utility information are common knowledge.
- No externally imposed time-limit.
- Zero reservation value: $u_i^{\tau}(\phi) = 0 \ \forall \tau$.

Main Result

There is a unique *sub-game perfect equilibrium* that requires a single negotiation step in most cases.

Rubinstein's Bargaining Protocol: Equilibrium

Exponential Discounting

The negotiation ends in **one step** with the first agent proposing and the second agent accepting *for asymmetric cases*:

$$(\omega_1^*, \omega_2^*) = \left(\frac{1-\delta_2}{1-\delta_1\delta_2}, \frac{\delta_2(1-\delta_1)}{1-\delta_1\delta_2} \right)$$

Rubinstein's Bargaining Protocol: Equilibrium

NEC-AIST AI Cooperative Research Laboratory

Exponential Discounting

The negotiation ends in **one step** with the first agent proposing and the second agent accepting *for asymmetric cases*:

$$(\omega_1^*, \omega_2^*) = \left(\frac{1-\delta_2}{1-\delta_1\delta_2}, \frac{\delta_2(1-\delta_1)}{1-\delta_1\delta_2} \right)$$

Linear Discounting

The negotiation ends in **one step** with the first agent proposing and the second agent accepting:

$$(\omega_1^*, \omega_2^*) = \begin{cases} (c_2, 1-c_2) & c_1 > c_2 \\ (x, 1-x)\ \forall x \in [c_1, 1] & c_1 = c_2 \\ (1, 0) & c_1 < c_2 \end{cases}$$

Negotiation with Complete Information

NEC-AIST AI Cooperative Research Laboratory

Hick's Paradox

Why do rational parties negotiate when they have full information?

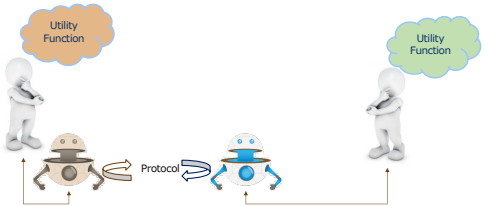
Labor vs. Management

- 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.

Components of Negotiation



Negotiation Scenario Defines who is negotiating about what

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
- ...

Notation

NEC-AIST AI Cooperative Research Laboratory

Negotiation Scenario $\mathcal{S}$

Agent Indices $\mathcal{A}^{\mathcal{S}}$: The set of n_A agents.

Domain $\mathcal{D}^{\mathcal{S}}$: A tuple $(\Omega, \mathcal{F}^{\mathcal{S}})$ defining the situation:

- **Outcome Space Ω:** A set of all possible agreements with $\phi \notin \Omega$ representing disagreement. $\Omega^{+} \equiv \Omega \bigcup \{\phi\}$ is called the Extended Outcome Space.
- **Preferences Tuple $\mathcal{F}^{\mathcal{S}}$:** A tuple of n_A members that defines the preferences of each agent engaged in the negotiation.
- **Information Set Tuple $\mathcal{I}^{\mathcal{S}}$:** A tuple of n_A information sets $\left(\left\{\mathrm{I}_f^{\mathcal{S}}(\mathcal{F}^{\mathcal{S}}) \mid f \in \mathcal{A}^{\mathcal{S}}\right\}\right)$ where $\mathrm{I}_f^{\mathcal{S}}(\mathcal{F}^{\mathcal{S}})$ represents all the information available to agent x about the preferences $\mathcal{F}^{\mathcal{S}}$ of all agents including itself.

Protocol $\mathcal{P}$: The negotiation protocol.

Negotiation Protocol Execution: $\mathcal{P}(\mathcal{S})$

NEC-AIST AI Cooperative Research Laboratory

Protocol

Breaking Rule when the negotiation ends with a timeout.

Action Constraints **what** actions are valid.

Activation Rule **which** each agent is activated.

Evaluation Rule the next step:

- **Continue** ▷ Goto breaking rule.
- **Agreement** Succeed with ω.
- **Disagreement** Fail with ϕ.

Agent (Strategy)

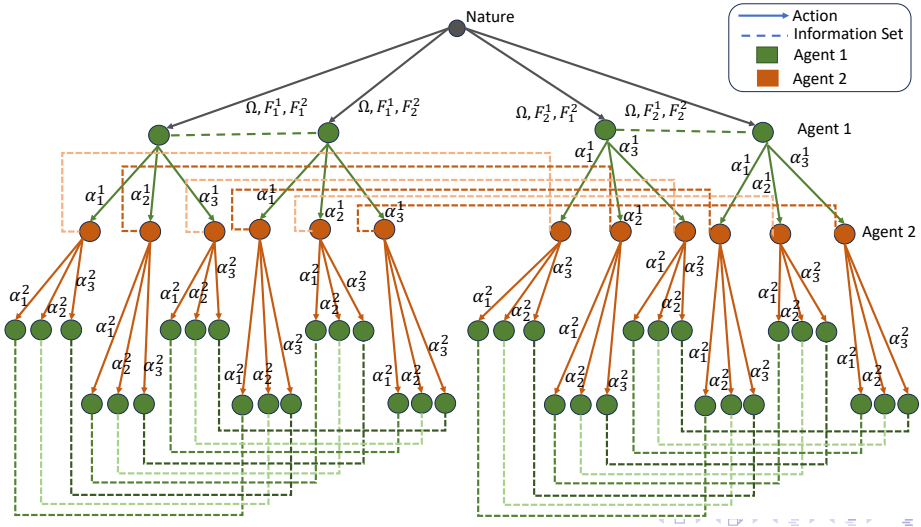
Selects an action from the **valid action set**.

Algorithm 1 Negotiation Mechanism Execution: $\mathcal{P}_\Upsilon(\lambda)$

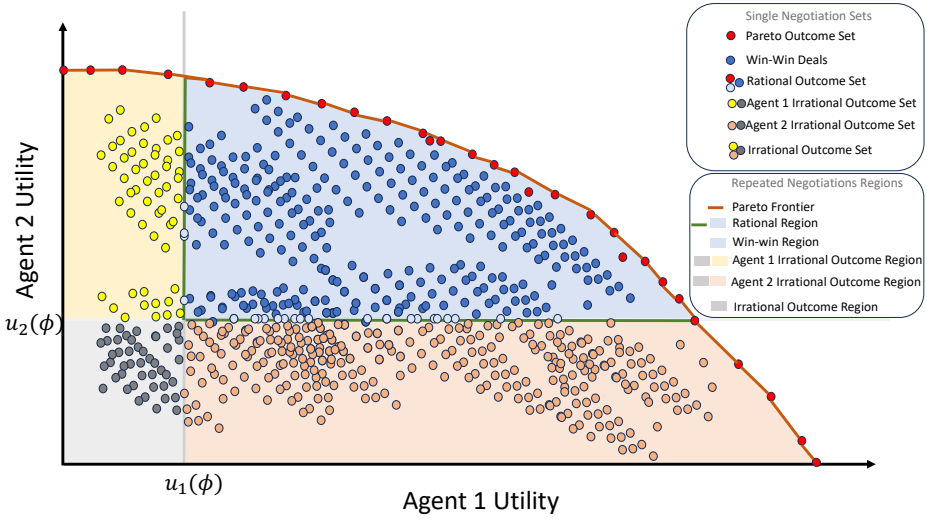
Input: Scenario $\lambda \equiv \left(\mathcal{N}^\lambda, (\Omega, \mathcal{F}^\lambda), \mathcal{I}^\lambda \equiv \left\{\mathrm{I}_x^\lambda : x \in \mathcal{N}^\lambda\right\}\right)$, Mechanism $\mathcal{P}_\Upsilon = (\mathcal{P}, \Upsilon) : \mathcal{P} = (\zeta, \chi, \sigma)$
Output: $\omega_* \in \Omega \cup \{\phi\}$

$x \approx \pi_x \forall x \in \mathcal{N}^\lambda \leftarrow \Upsilon(\lambda)$ ▷ Assign strategies
$i, t, t_0, \mathcal{T} \leftarrow 0, 0, clock(), \langle\rangle$ ▷ Init step, time and trace
while $\zeta(\mathcal{T}, i, t; \lambda) = 0$ **do** ▷ check the breaking rule
  $\mathcal{N}_i^\lambda, \leftarrow \chi(\mathcal{T}, i; \lambda)$ ▷ Next active agents
  for $x \in \mathcal{N}_i^\lambda$ **do** ▷ Activate agents
    $\mathbb{A}_x^v \leftarrow \gamma(\mathcal{T}, x; \lambda)$ ▷ Valid action set
    $\alpha_i^x \leftarrow \pi_x\left(\mathcal{T}, \mathbb{A}_x^v; \mathrm{I}_x^\lambda\right)$ ▷ Receive action
    $\mathcal{T} \leftarrow \mathcal{T} + \langle \alpha_i^x, x \rangle$ ▷ Update the trace
    $z \leftarrow \sigma(\mathcal{T}, i; \lambda)$ ▷ Evaluation rule
    if $z = \phi$ **then return** ϕ ▷ End with disagreement
    else if $z \in \Omega$ **then return** ω_i ▷ agreement
    end if ▷ Otherwise continue to the next step
  end for
  $i, t \leftarrow i + 1, clock() - t_0$ ▷ Update step and time.
end while
return ϕ ▷ Disagreement on breaking rule returning 1

Induced Game with Incomplete Information



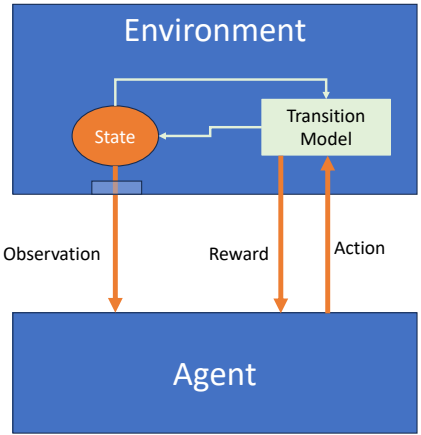
Visualizing a negotiation



Outline

Reinforcement Learning

- One of the most successful ML techniques.
- The agent learns a policy that maximizes its expected reward by interacting with the environment (or a simulation of it).
- Main Components:
  - **State** Full information about the current state of the environment.
  - **Transition Model** Changes the state based on current state and agent' s action.
  - **Observation** The state as *appears* to the agent.
  - **Action** The action generated by the agent's policy
  - **Reward** The feedback given from the environment to the agent.



RL Successes in Games

NEC-AIST AI Cooperative Research Laboratory

- Earlier games solved by RL were: **complete information games**.
  - Chess, Go
- Same method can be used to solve multiple games:
  - Atari
- Some **incomplete information zero-sum games** were also solved.
  - Poker, Stratego
- Next: incomplete information general-sum
  - E.g. Automated Negotiation

Outline

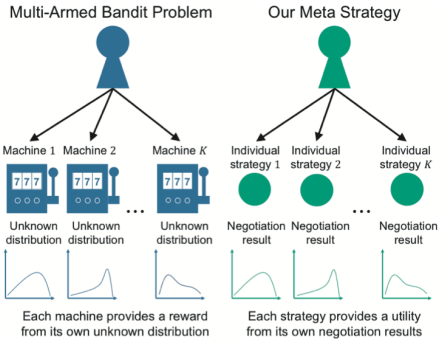
Multiaremed Bandits for Repeated Negotiations[7]

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

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



[7] 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.

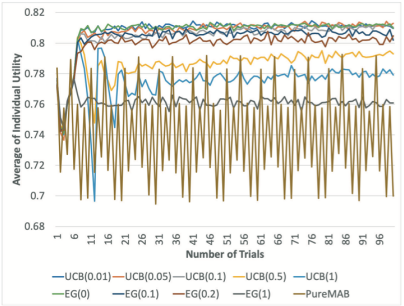
Multiaremed Bandits for Repeated Negotiations[7]

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

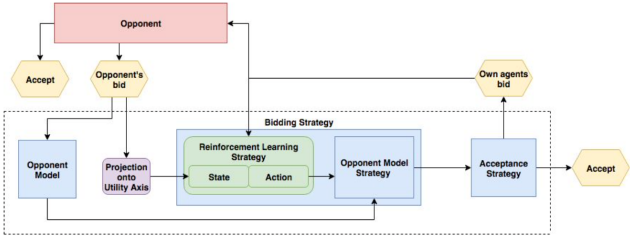
$$UCB(s) = \hat{\mu}_s + c\sqrt{\frac{lnN}{N_s}}$$



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

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

RLBOA: Learning Offering Strategy[8]



Main Points

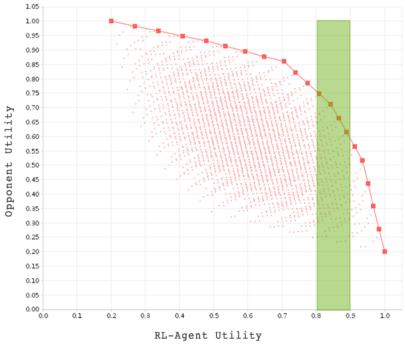
- Extends the BOA architecture.
- Learns only a bidding strategy:
  - The agent learns how to move *in its own utility axis*.

[8] 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: $\{\hat{u}(\omega^s_t), \hat{u}(\omega^s_{t-1}), \hat{u}(\omega^p_t), \hat{u}(\omega^p_{t-1}), t\}$.
  - $\hat{u}(\omega) = \lfloor N \times u(\omega) \rfloor$[9]
- Action Space: $\leftarrow$, $-$, $\rightarrow$.
  - First step $\rightarrow i \in [0, N-1]$
  - Out-of-boundary correction: $-$.
- Training Method: Q-learning
- Acceptance Strategy [Recommended]: $AC_{next}(\alpha = 1, \beta = 0)$[10]

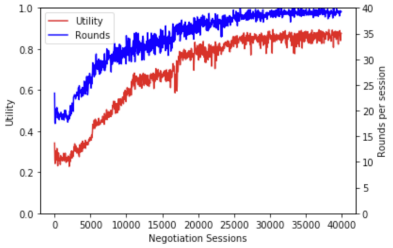
$$a(\omega) = \begin{cases} \text{Accept}, & \text{if } \alpha u(\omega) + \beta \geq u(o(s)) \\ \text{Reject}, & \text{otherwise} \end{cases}$$



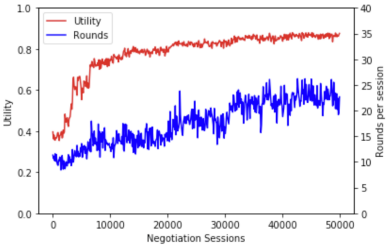
[9] Just ignore the special case at $u == 1$.

[10] 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!

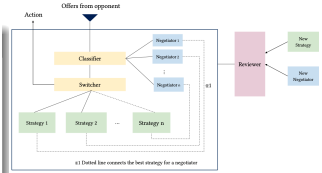
Domain	Outcome space	Low opp.	High opp.
Small	256	0.2615	0.5178
Medium	3.125	0.3111	0.5444
Large	46.656	0.2595	0.5250

A Framework for Learning Offer Strategies

NEC-AIST AI Cooperative Research Laboratory

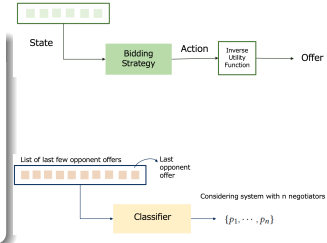
Main Idea[11]

- 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.



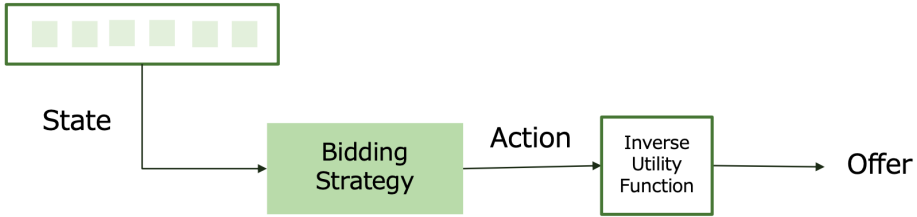
Phases

- 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**.



[11] 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



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)

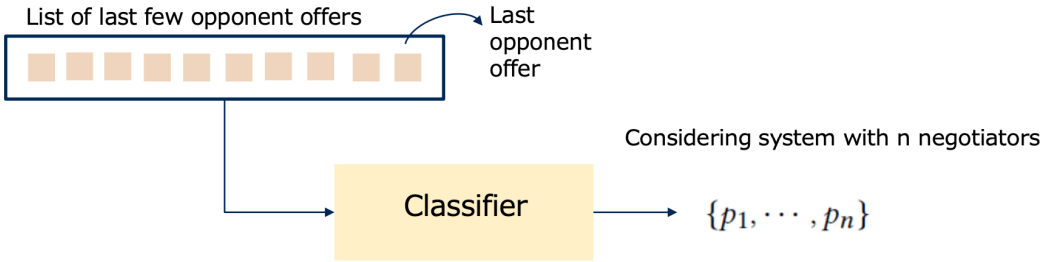
$$s_t = \{t_r, U_s(\omega_s^{t-2}), U_s(\omega_o^{t-2}), U_s(\omega_s^{t-1}), U_s(\omega_o^{t-1}), U_s(\omega_s^t), U_s(\omega_o^t)\}$$

$$a_t = u_s^{t+1} \quad \text{such that} \quad u_r < u_s \leq 1$$

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

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

During: Learning realtime Partner Classification



The SL Components

- **Features** Opponent last K offers.
- **Target** Opponent Type (discrete set)

After: Reviewing New Pairs

NEC-AIST AI Cooperative Research Laboratory

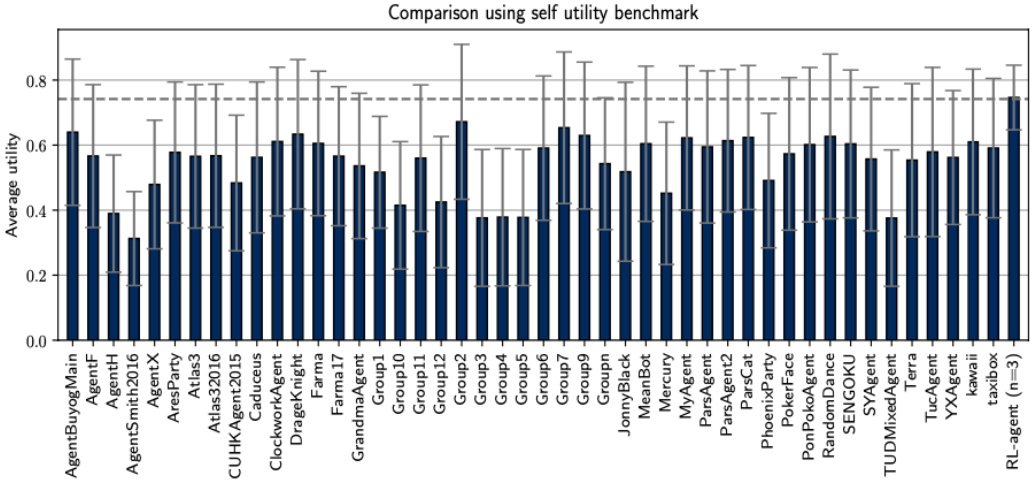
New Partner Type (N_{new}) Ecncountered

- 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 $\downarrow$.

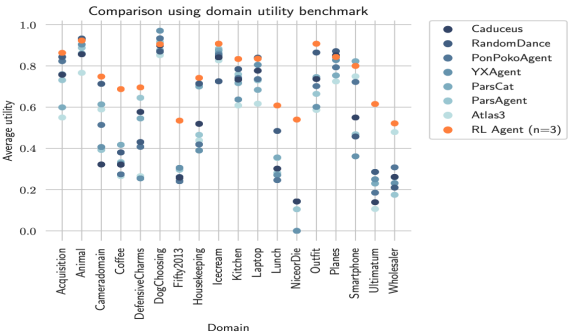
Update Best Responses

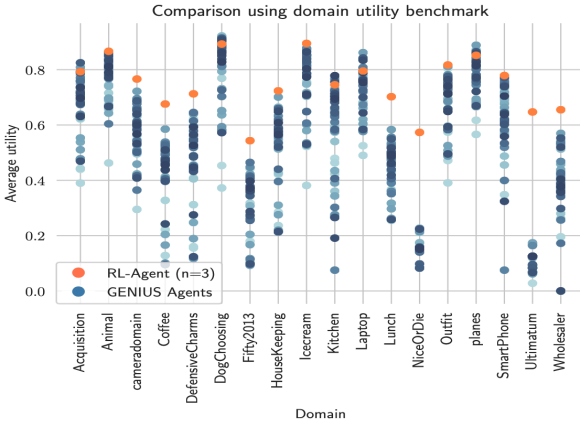
- For every learned ABR, negotiator pair (S, N):
  - Evaluate S_{new} against $N \rightarrow U(S_{new})$
  - Evaluate S against $N \rightarrow U(S)$
  - Replace S with S_{new} iff $\alpha U(S) < U(S_{new})$

Results: Against Different Opponents

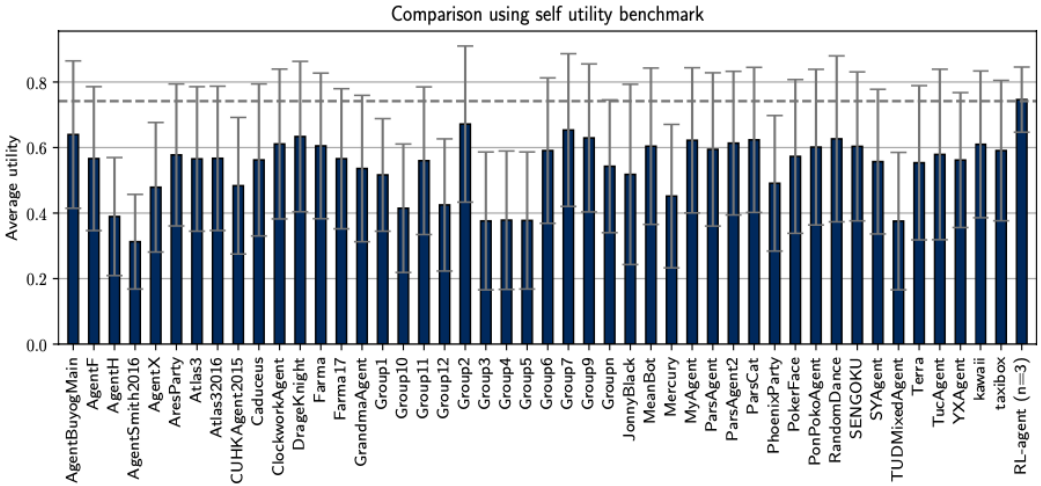


Results: In Different Domains

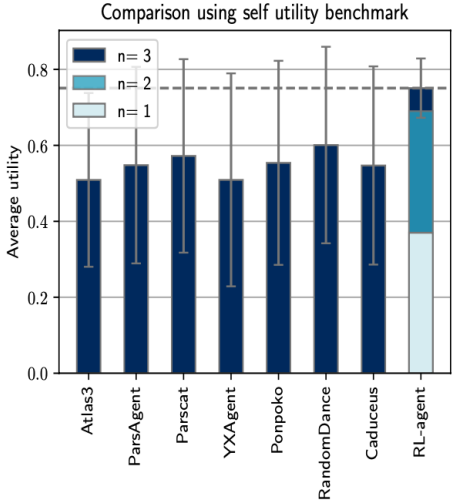




Results: Compared with SOTA Agents



Results: Improvement with new best responses



DQN for learning Acceptance Strategy[12]

NEC-AIST AI Cooperative Research Laboratory

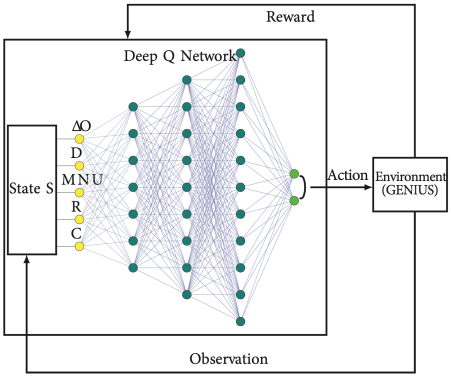
Main Idea

- 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

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



[12] 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

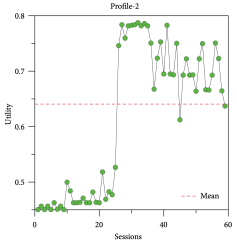
NEC-AIST AI Cooperative Research Laboratory

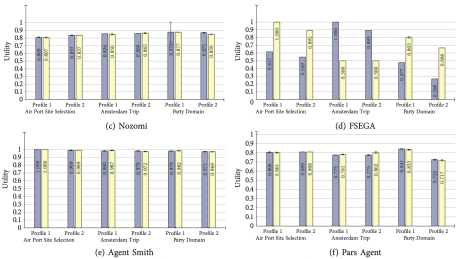
Training

- 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



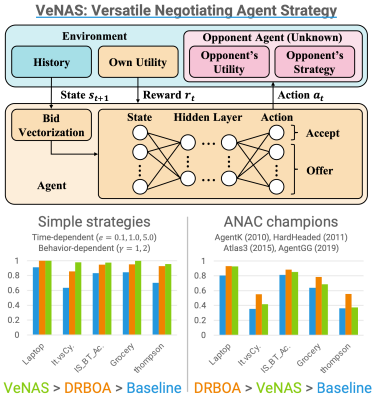


Learning Offer and Acceptance Policies together[13]

NEC-AIST AI Cooperative Research Laboratory

- 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 $= \begin{cases} u(\omega_a), & \text{At the end} \\ 0 & \text{non terminal state} \end{cases}$



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

Reward-based negotiating agent strategies[14]

NEC-AIST AI Cooperative Research Laboratory

State set of state $s_t \in \mathcal{S}$: The agent's offer ω_t, opponent's offer ω'_t and accept signal η'_t, and t/T.

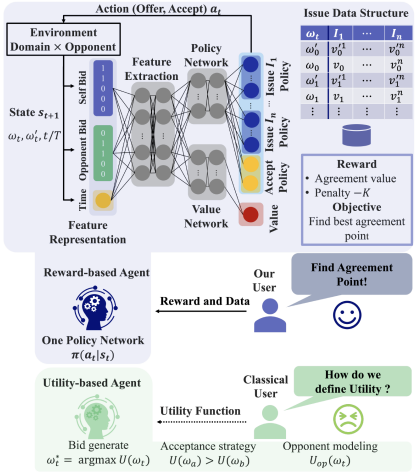
Action set of action $a_t \in \mathcal{A}$: The agent's selected offer ω_t and accept signal η_t. The action space is proportional to the domain size of the utility function.

Reward $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$: When the agent accepts, the agent obtains the final utility value. Penalty $-K$ is given when the negotiation ends without reaching an agreement. Otherwise, the reward is 0.

Policy function $\mathcal{S} \times \mathcal{A} \to [0, 1]$: The policy function is defined as $\pi_\theta(a_t \mid s_t) := \Pr(A_t = a_t \mid S_t = s_t, \theta_t = \theta)$.

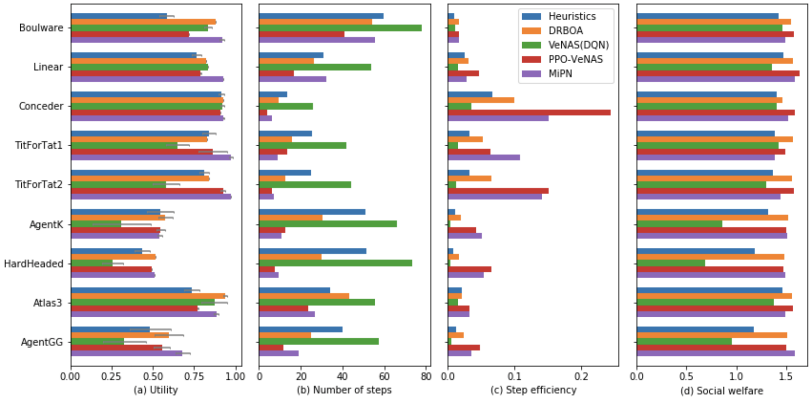
Transition function $\mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$: Transition function is defined as $p(s_{t+1}|s_t, a_t) := \Pr(S_{t+1} = s_{t+1}|S_t = s_t, A_t = a_t)$.

History $\mathcal{D} = (\omega_0, \omega'_0, \cdots, \omega_t, \omega'_t)$: The observable data during a negotiation.



[14] Ryota Higa et al. "Reward-based negotiating agent strategies". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 10. 2023, pp. 11569–11577.

Reward-based negotiating agent strategies[15]



[15] Higa et al., "Reward-based negotiating agent strategies".

Outline

Outline

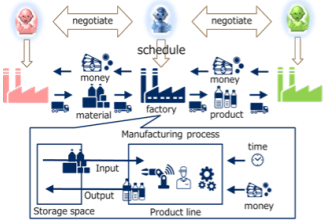
Automated Negotiating Agents Competition (ANAC)

- An international competition dedicated to automated negotiation.
- Started in conjunction with AAMAS in 2010.
- Running every year with AAMAS or IJCAI (14 iterations so far).
- Next ANAC will be in conjunction with AAMAS 2024 (not official yet).

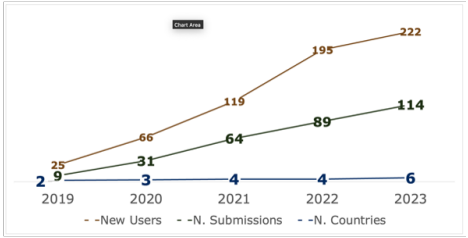
Outline

Supply Chain Management League (SCML)

- Running as part of ANAC since 2019.
- **Scenario** A market in which all trade is done through concurrent closed bilateral negotiations:
  - Represents several applications in the real world.
- Agent preferences are endogenous to the simulation.
- **Goal** Maximize your profit.
- **Main Challenges**:
  - Situated Negotiations.
  - Sequential Negotiation (learning between negotiations).
  - Concurrent Negotiation (outside-options).
  - Trust management.



Rank	Agent	Earned money
2		
3		
4		



SCML World

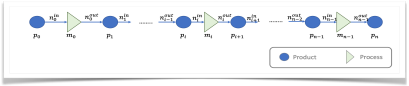
NEC-AIST AI Cooperative Research Laboratory

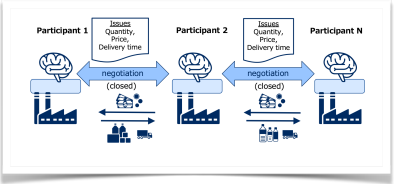
Challenge

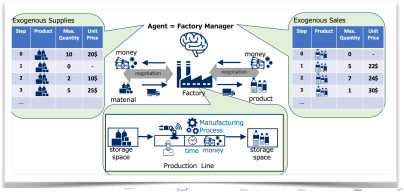
- Negotiation game with imperfect information
- Concurrent negotiations.
- Repeated negotiations → OneShot.
- Sequential negotiations → Standard.

Information

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







SCML Competition

NEC-AIST AI Cooperative Research Laboratory

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

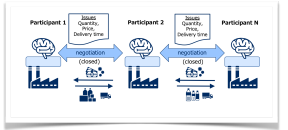
NEC-AIST AI Cooperative Research Laboratory

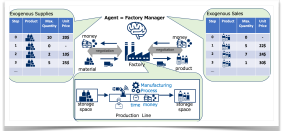
Competition Details

- 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

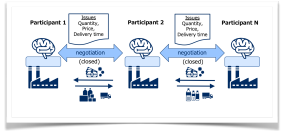
NEC-AIST AI Cooperative Research Laboratory

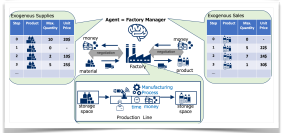
Competition Details

- 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).

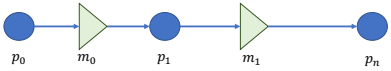
Flavors

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



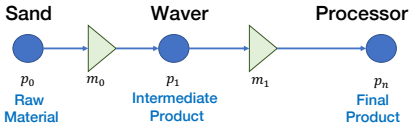


Overview



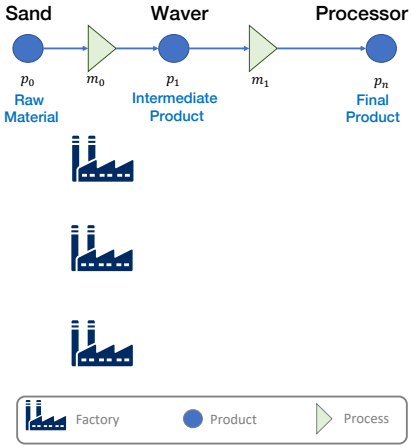
- A **production-graph** defines what can be produced and how.

Overview



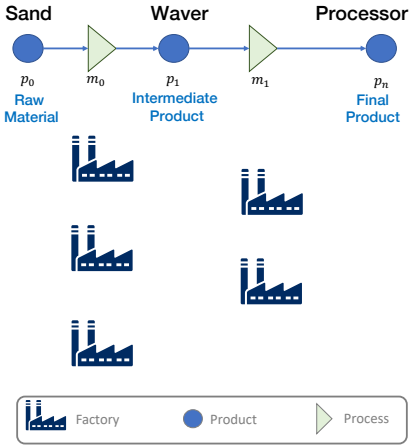
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.

Overview



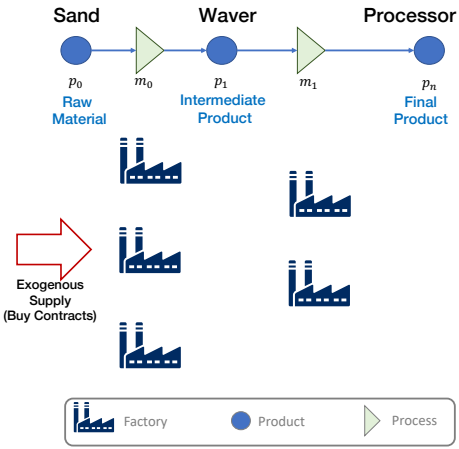
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.
- Factories can run **manufacturing processes** converting input products into output products on their **production lines**.

Overview



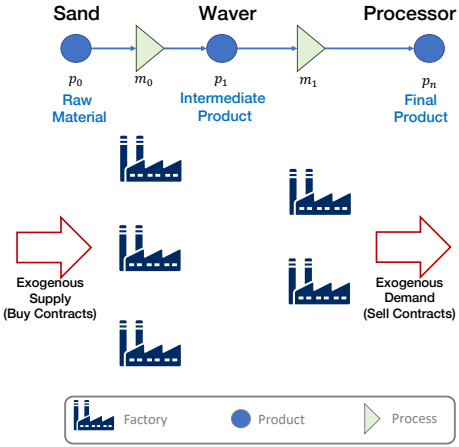
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.
- Factories can run **manufacturing processes** converting input products into output products on their **production lines**.
- We have two layers of factories/agents.

Overview



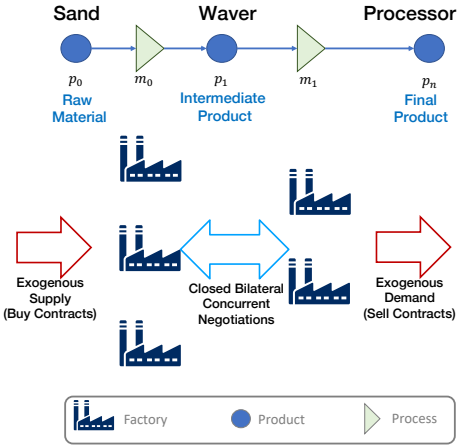
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.
- Factories can run **manufacturing processes** converting input products into output products on their **production lines**.
- We have two layers of factories/agents.
- L_0 factories/agents receive exogenous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

Overview



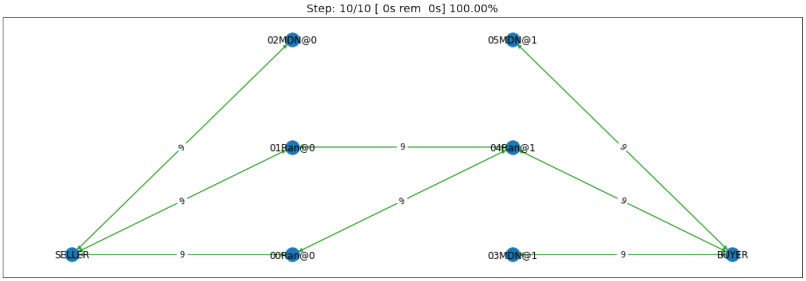
- A **production-graph** defines what can be produced and how.
- We have 3 products, 2 processes.
- Factories can run **manufacturing processes** converting input products into output products on their **production lines**.
- We have two layers of factories/agents.
- L_0 factories/agents receive exogenous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

NEC-AIST AI Cooperative Research Laboratory

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).

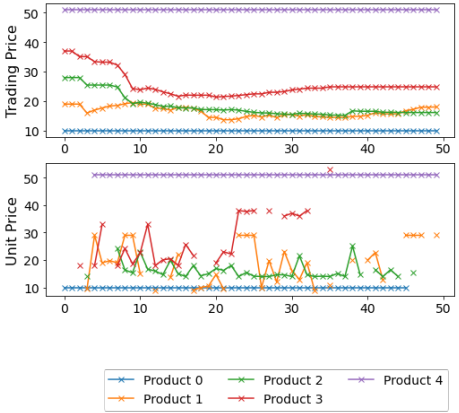
Market Information

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

NEC-AIST AI Cooperative Research Laboratory

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).

Simulation Steps

Once

Initialize all agents

- init()

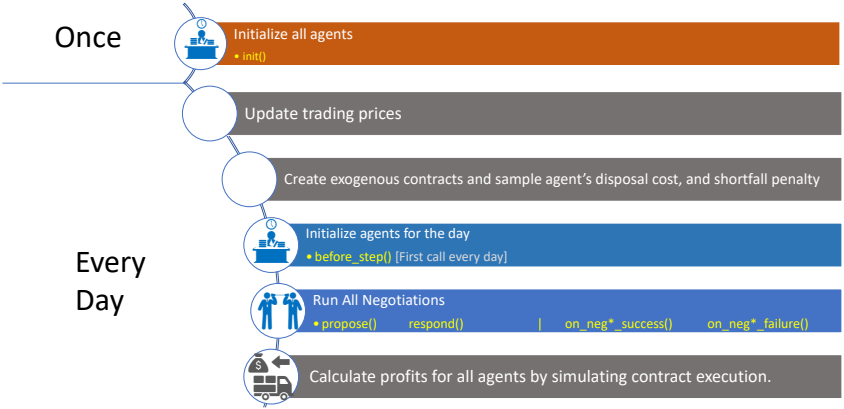
Simulation Steps

Simulation Steps

Simulation Steps

Simulation Steps

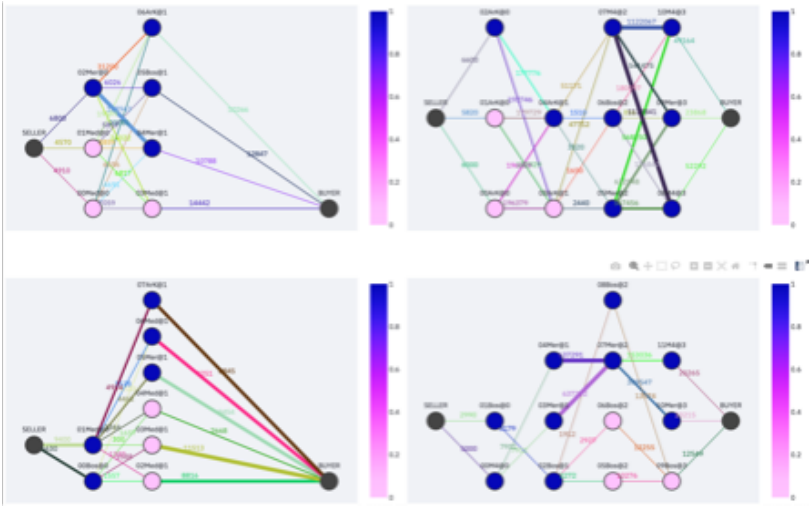
Simulation Steps



Simulation Steps

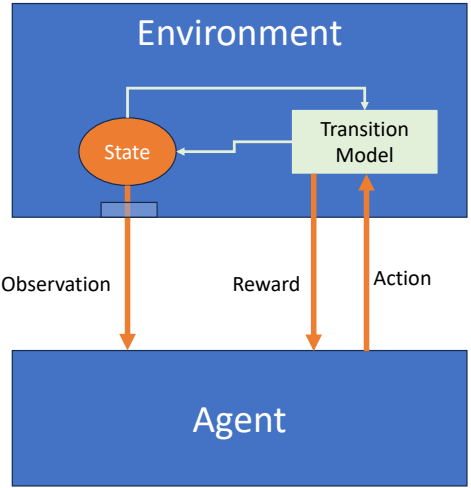
Simulation Steps

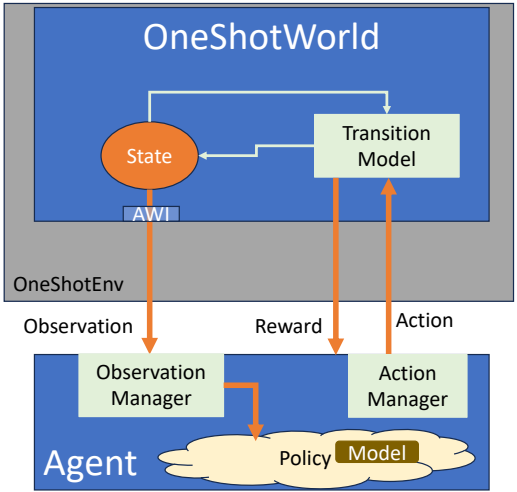
SCML Environment



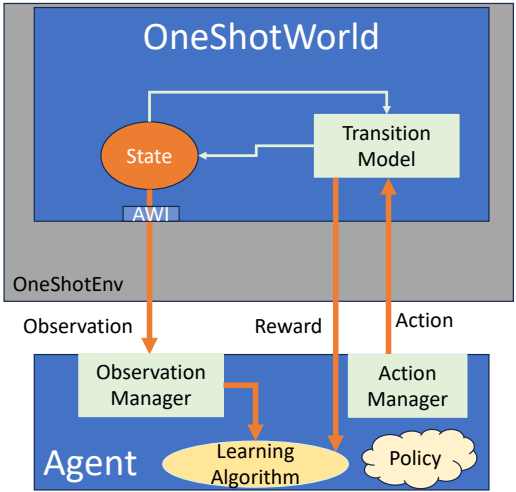
Outline

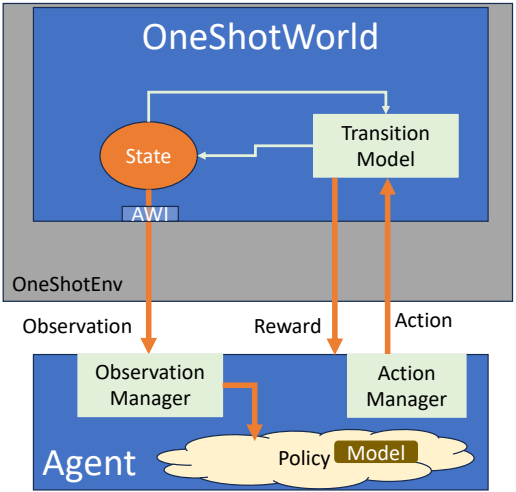
RL for SCML


Environment
State
Transition Model
Observation
Reward
Action
Agent


OneShotWorld
State
Transition Model
AWI
OneShotEnv
Observation
Reward
Action
Observation Manager
Action Manager
Agent
Policy
Model

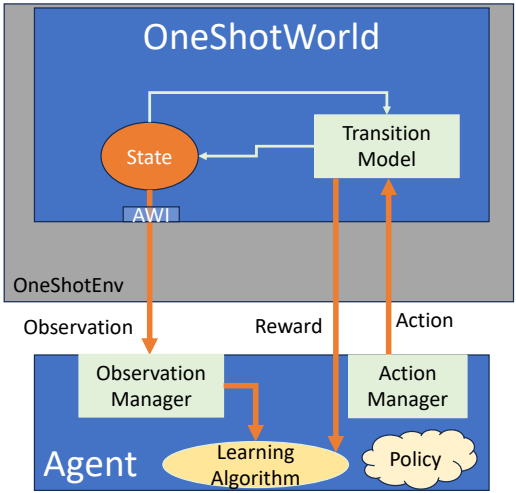
RL for SCML


OneShotWorld
State
Transition Model
AWI
OneShotEnv
Observation
Reward
Action
Observation Manager
Action Manager
Agent
Learning Algorithm
Policy


OneShotWorld
State
Transition Model
AWI
OneShotEnv
Observation
Reward
Action
Observation Manager
Action Manager
Agent
Policy
Model

RL for SCML: Main Challenges

- Huge and hybrid observation space
  - The main challenge is to design a good ObservationManager
- Sparse Reward
  - Does Reward Shaping help?
  - Can we devise a reward signal other than the profit (e.g. KPIs)?
- Every simulation is different (i.e. competitors, suppliers, consumers).
  - How to handle variable number of negotiations (i.e. elastic action space)?
  - How to negotiations ending at different time (in the same day)?
  - How to divide all possible simulations into usable contexts suitable for learning?
- How to learn (learning algorithm)?



Outline

Outline

Outline

Outline