

Group3: A Portfolio-Aware Time-Concession Negotiation Agent for SCML OneShot

Group 3 — Elijah Seun Oluwatayo, Yeabsira Simeka Maru

Department of Computer Science
ANAC/SCML Negotiation Agent Project

CS 551: Introduction to Artificial Intelligence — Spring 2026

Abstract

This report presents **Group3**, a negotiating agent implemented in Python using the SCML/NegMAS framework for the SCML OneShot setting, where negotiation quality depends not only on favourable prices but on whether accepted contracts fit the agent’s daily production and trading portfolio. Group3 uses portfolio-aware quantity control as its main strategic driver: it estimates the quantity still needed for the current day, generates offers that target this remaining gap, and rejects contracts that would create excess or shortfall exposure. Prices are produced by a Boulware-style time-concession rule that starts at utility-preserving values and concedes gradually toward the deadline, and acceptance decisions combine portfolio feasibility with a time-dependent price threshold. Group3 also implements a lightweight opponent model that tracks offer histories, classifies behaviour as silent, selfish, concessionary, or nice, and estimates reciprocal sensitivity, although diagnostic evidence shows that this model functions as a secondary interpretable adaptation layer rather than the main source of performance: most classified moves are silent and sensitivity observations are sparse. In the strongest longer mixed-baseline evaluation, Group3 achieved a mean score of 1.0866 compared with a mixed-baseline mean of 0.7653, a score ratio of $1.42\times$, and a 100% run-level win rate against the mixed-baseline average. Supporting evaluations produced mean scores of 1.0776 and 1.0797, confirming that Group3 consistently outperforms mixed built-in baselines and remains competitive with strong Sync-family agents without consistently dominating them.

Keywords: automated negotiation, SCML OneShot, portfolio management, Boulware concession, opponent modelling, multi-agent systems

1 Introduction

Automated negotiation agents must decide what to offer, when to concede, and when to accept. In many bilateral negotiation domains, the agent’s main objective is to reach a favourable agreement over a fixed set of issues. In the Supply Chain Management League (SCML) [5], the problem is more complex: each bilateral agreement affects the agent’s larger production and trading position. A factory manager may obtain a good unit price but still lose performance if the contract quantity causes overcommitment, excess inventory,

shortfall penalties, or missed production opportunities.

This report presents **Group3**, a negotiation agent for the SCML OneShot setting. Group3 was designed around the observation that SCML OneShot is primarily a *portfolio-balancing* negotiation problem. Price matters, but price cannot be optimized independently of quantity. The agent therefore combines three components:

1. **Portfolio-aware quantity control**, which determines how much quantity the agent should seek or accept.
2. **Boulware-style price concession**, which determines how the agent moves from a strong opening price toward compromise over time.
3. **Lightweight opponent modelling**, which records observed behaviour and makes small adjustments to bidding and acceptance.

The dominant performance driver is the first component: portfolio-aware quantity control. The agent consistently performs well because it avoids accepting or proposing quantities that are harmful to its daily portfolio. The second component, time-based concession, protects utility early and supports agreement later. The third component is implemented and used, but diagnostic evidence (Section 5) shows that it should be interpreted as a secondary and explanatory mechanism rather than the primary reason for the agent’s tournament performance.

The remainder of the report is organised as follows. Section 2 introduces the SCML OneShot setting and reviews the literature motivation. Section 3 states the agent’s strategy formally, covering bidding, acceptance, and opponent modelling. Section 4 describes the development procedure and the key design decisions. Section 5 reports experimental results and defines the evaluation metrics. Section 6 analyses why the agent performs the way it does and discusses limitations, and Section 7 concludes.

2 Background

SCML OneShot [5] is a multi-agent supply-chain negotiation environment built on the NegMAS framework [4]. Agents act as factory managers and negotiate bilaterally over contracts with three discrete issues: quantity, delivery time,

and unit price. In this project we represent an offer as

$$o = (q, t, p), \quad (1)$$

where q is the quantity, t is the delivery time, and p is the unit price. Each day the agent is given an exogenous contract that fixes either its required input quantity (if it is an L0 / first-level agent that must subsequently sell) or its required output quantity (if it is an L1 / last-level agent that must subsequently buy). Profit on the day equals revenue minus costs minus two penalties: a *disposal* penalty when the agent over-acquires inputs it cannot sell, and a *shortfall* penalty when it has committed to sell more than it can produce. Both penalties vanish exactly when bought and sold quantities match, which is why quantity management dominates this setting strategically.

Automated negotiation literature commonly decomposes an agent into three parts: a bidding strategy, an acceptance strategy, and an opponent model [1, 3]. Time-dependent concession functions, including Boulware and Conceder strategies, are a standard way to balance utility preservation with agreement probability [1]. A Boulware strategy concedes slowly at first and more strongly near the deadline, which is useful when the agent wants to avoid unnecessary early concessions while still allowing agreement late.

Opponent modelling is also widely studied in automated negotiation [2, 3], but the SCML setting offers limited observable information: the opponent’s true production costs, penalties, and utility structure are not directly visible. Group3 therefore uses opponent information only in a lightweight way — it does not attempt to learn a full opponent utility function, but instead tracks whether received offers improve or worsen from Group3’s local perspective.

The strategic problem in SCML OneShot is therefore not simply to maximise price advantage. Instead, the agent must approximately solve *maximise profitable portfolio completion under negotiation uncertainty*. This motivates a design in which offer generation and acceptance are dominated by portfolio feasibility, with price and opponent behaviour providing additional guidance.

3 Methodology

Group3 inherits from `OneShotAgent` and uses a strict decision hierarchy: it first asks whether a quantity is useful for the current portfolio, then evaluates whether the price is acceptable under time pressure, and finally applies small adjustments derived from observed opponent behaviour. Figure 1 shows the overall architecture.

The high-level acceptance rule is

$$\text{Accept}(o) \iff F_{\text{portfolio}}(o) = 1 \wedge F_{\text{price}}(o) = 1, \quad (2)$$

where $F_{\text{portfolio}}$ and F_{price} are the two filters defined below.

3.1 Bidding strategy

The bidding engine considers the factors listed in Table 1.

The daily target quantity is estimated from the agent’s exogenous input and output quantities:

$$Q^{\text{target}} = \max(Q^{\text{exo-in}}, Q^{\text{exo-out}}). \quad (3)$$

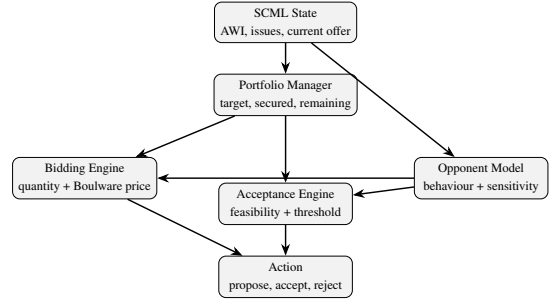


Figure 1: Architecture of Group3. Portfolio management is the central decision driver; price concession and opponent signals modify the final decision.

Table 1: Explicit bidding factors used by Group3.

Factor	Symbol	Purpose
Daily target quantity	Q^{target}	Estimate trading volume
Secured quantity	Q^{secured}	Track agreed quantity
Remaining gap	Q^{rem}	Determine offer quantity
Completion ratio	ρ_c	Reduce risk near completion
Relative time	τ	Control concession
Buyer/seller role	role	Determine price preference
Price range	$[p_{\min}, p_{\max}]$	Keep price valid
Opponent behaviour	b_j	Adjust concession
Opponent sensitivity	s_j	Penalise weak reciprocity

The remaining quantity to secure is

$$Q^{\text{rem}} = \max(0, Q^{\text{target}} - Q^{\text{secured}}). \quad (4)$$

The first offer of a negotiation uses the full remaining gap as its quantity ($q_0 = Q^{\text{rem}}$), but the agent reduces exposure against risky opponents:

$$q = \begin{cases} \text{round}(0.95 q_0), & \text{if } b_j = \text{selfish}, \\ \text{round}(0.97 q_0), & \text{if } s_j < 0.5, \\ q_0, & \text{otherwise.} \end{cases} \quad (5)$$

The opening price is produced by a Boulware-style concession function. Let $\tau \in [0, 1]$ denote relative negotiation time and let $e > 0$ be the concession exponent. Then

$$\alpha(\tau, e) = 1 - \tau^{1/e}. \quad (6)$$

At $\tau = 0$, $\alpha \approx 1$ and the agent proposes its most favourable price; as $\tau \rightarrow 1$, $\alpha \rightarrow 0$ and the agent concedes toward the opponent-favourable boundary. For sellers and buyers respectively,

$$p_{\text{sell}}(\tau) = p_{\min} + \alpha(\tau, e)(p_{\max} - p_{\min}), \quad (7)$$

$$p_{\text{buy}}(\tau) = p_{\max} - \alpha(\tau, e)(p_{\max} - p_{\min}). \quad (8)$$

Counteroffers are produced by the same formulas with updated time, remaining quantity, and opponent classification. Group3 is monotonic in the price-concession dimension under fixed role, price range, and concession exponent: for sellers, $p_{\text{sell}}(\tau)$ weakly decreases in τ ; for buyers, $p_{\text{buy}}(\tau)$ weakly increases. The agent therefore concedes monotonically along the protocol-relevant price axis. The full SCML

Algorithm 1: Group3 bidding

Input: negotiator j , negotiation state s
Output: offer (q, t, p) or no-offer

```

1:  $Q^{\text{rem}} \leftarrow \max(0, Q^{\text{target}} - Q^{\text{secured}})$ 
2: if  $Q^{\text{rem}} \leq 0$  then
3:   return no-offer
4: end if
5:  $q \leftarrow Q^{\text{rem}}$ 
6: if  $b_j$  is selfish then
7:    $q \leftarrow \text{round}(0.95 q)$ 
8: end if
9: if  $s_j < 0.5$  then
10:   $q \leftarrow \text{round}(0.97 q)$ 
11: end if
12:  $\tau \leftarrow$  relative time of state  $s$ 
13:  $e \leftarrow$  opponent-adjusted concession exponent
14:  $\alpha \leftarrow 1 - \tau^{1/e}$ 
15: if Group3 is selling then
16:   $p \leftarrow p_{\min} + \alpha(p_{\max} - p_{\min})$ 
17: else
18:   $p \leftarrow p_{\max} - \alpha(p_{\max} - p_{\min})$ 
19: end if
20: return  $(q, t, p)$ 

```

offer value is not guaranteed to decrease monotonically, however, because quantity is an orthogonal portfolio-control decision: a later offer can reduce quantity exposure or reflect updated portfolio completion, making it safer or more valuable to Group3 even while price concession itself remains monotonic. This is not a violation of the monotonic-concession requirement; it reflects the fact that SCML utility depends on both price and portfolio fit.

3.2 Acceptance strategy

An incoming offer must pass two filters: portfolio feasibility ($F_{\text{portfolio}}$) and a time-dependent price threshold (F_{price}). The completion ratio

$$\rho_c = \frac{Q^{\text{secured}}}{Q^{\text{target}}}, \quad Q^{\text{target}} > 0, \quad (9)$$

controls how strict the feasibility check becomes. Group3 computes a shrinking quantity buffer

$$B(s) = Q^{\text{target}} [\lambda_{\text{early}}(1 - \tau) + \lambda_{\text{late}}\tau] (1 - \rho_c), \quad (10)$$

with $\lambda_{\text{early}} = 0.20$ and $\lambda_{\text{late}} = 0.10$. The safe acceptance limit is

$$L_j(s) = Q^{\text{rem}} + B(s), \quad (11)$$

and when $\rho_c \geq 0.90$ the buffer is removed: $L_j(s) = Q^{\text{rem}}$. Opponent risk further shrinks the limit by a factor of 0.95 if the opponent is selfish or 0.97 if $s_j < 0.5$. The portfolio filter passes when $q \leq L_j(s)$.

The time-dependent price threshold is anchored at the agent's most-favourable boundary at $\tau = 0$ and slides toward the opposite boundary as $\tau \rightarrow 1$. For sellers,

$$\theta_{\text{sell}}(\tau) = p_{\max} - \tau(p_{\max} - p_{\min}) + A_j, \quad (12)$$

Algorithm 2: Group3 acceptance

Input: incoming offer $o = (q, t, p)$ from opponent j
Output: accept or reject

```

1: if  $o$  is empty then
2:   return reject
3: end if
4: Record  $o$  and update opponent model
5: Compute  $Q^{\text{rem}}$ 
6: if  $Q^{\text{rem}} \leq 0$  then
7:   return reject
8: end if
9: Compute safe limit  $L_j(s)$  via Eq. (11)
10: if  $q > L_j(s)$  then
11:   return reject
12: end if
13: Compute price threshold via Eq. (12) or (13)
14: if seller and  $p \geq \theta_{\text{sell}}(\tau)$  then
15:   return accept
16: else if buyer and  $p \leq \theta_{\text{buy}}(\tau)$  then
17:   return accept
18: else
19:   return reject
20: end if

```

and the agent accepts when $p \geq \theta_{\text{sell}}(\tau)$. For buyers,

$$\theta_{\text{buy}}(\tau) = p_{\min} + \tau(p_{\max} - p_{\min}) - A_j, \quad (13)$$

and the agent accepts when $p \leq \theta_{\text{buy}}(\tau)$. The opponent-derived adjustment A_j , with $\Delta p = p_{\max} - p_{\min}$, is

$$A_j = \begin{cases} +0.05 \Delta p, & b_j = \text{selfish}, \\ -0.03 \Delta p, & b_j \in \{\text{concession, nice}\}, \\ +0.03 \Delta p, & s_j < 0.5, \\ -0.02 \Delta p, & s_j \geq 1.0, \\ 0, & \text{otherwise.} \end{cases} \quad (14)$$

3.3 Opponent modelling

For each opponent j the agent records received offers, local values of those offers, sent offers and their proxy values from the opponent's perspective, behaviour counts, a current dominant behaviour, a sensitivity estimate, and an observation counter. The local value of a received offer is

$$V_{\text{us}}(o) = S_p(o) \cdot \min(q, \max(1, Q^{\text{rem}})), \quad (15)$$

where the normalised price score is $S_p(o) = (p - p_{\min}) / (p_{\max} - p_{\min})$ for sellers and $S_p(o) = (p_{\max} - p) / (p_{\max} - p_{\min})$ for buyers.

Behaviour is classified from the round-to-round change $\Delta_j^k = V_{\text{us}}(o_j^k) - V_{\text{us}}(o_j^{k-1})$:

$$b_j^k = \begin{cases} \text{silent}, & |\Delta_j^k| \leq \varepsilon, \\ \text{nice}, & \Delta_j^k / |V_{\text{us}}(o_j^{k-1})| \geq 0.25, \\ \text{concession}, & \Delta_j^k > 0, \\ \text{selfish}, & \Delta_j^k < 0. \end{cases} \quad (16)$$

The current dominant behaviour is the most frequent label observed.

Sensitivity is updated when Group3’s own proxy value to the opponent has improved and the opponent’s response also improves from Group3’s perspective. The smoothed estimate is

$$s_j^{\text{new}} = \eta s_j^{\text{old}} + (1 - \eta) \frac{\max(0, \Delta_j^{\text{opp}})}{\max(\varepsilon, \Delta_j^{\text{us}})}, \quad (17)$$

with $\eta = 0.7$. Together, b_j and s_j influence three decisions: the concession exponent (selfish opponents make Group3 more stubborn; cooperative opponents make it slightly more flexible), the quantity exposure (selfish or low-sensitivity opponents receive reduced quantities), and the acceptance threshold (the same opponents face stricter thresholds via A_j). The model is intentionally lightweight and identity-agnostic: it requires no known opponent class. As Section 5 shows, however, its measured contribution is limited — it should be regarded as an interpretable adaptation layer rather than the primary performance driver.

4 Implementation

The agent is written in Python using NegMAS [4] / SCML [5]. It subclasses `OneShotAgent` and implements the standard callbacks: `before_step()` initialises the daily portfolio state; `propose()` generates first offers and counteroffers via Algorithm 1; `respond()` applies Algorithm 2; `on_negotiation_success()` updates secured quantity; and `step()` records daily diagnostics.

The design was developed through a sequence of controlled improvements. The first objective was to construct a valid SCML agent that imports correctly, runs inside the world, and completes simulations, so that later experiments measured strategy rather than setup errors. The next major change was the addition of portfolio-aware quantity control: the agent estimates a daily target quantity and tracks secured contracts, which converted the strategy from naive offer generation into supply-chain-aware negotiation. Early random-baseline tests showed that this addition alone was the largest single source of improvement.

The Boulware-style price rule was added next, giving Group3 a principled way to start with favourable prices while still conceding over time on a simple, fast, and interpretable curve. Behaviour tracking and sensitivity estimation were then layered on top to make the agent adaptive. These mechanisms were never intended to replace portfolio control; their role is to adjust risk exposure and concession speed when sufficient evidence exists.

Stress-testing against stronger built-in baselines revealed that the agent needed protection against occasional weak runs caused by overcommitment near portfolio completion. A soft stability guard was therefore introduced, parameterised by a completion threshold of 0.90, an early buffer fraction of $\lambda_{\text{early}} = 0.20$, a late buffer fraction of $\lambda_{\text{late}} = 0.10$, a selfish-opponent quantity penalty of 0.95, and a low-sensitivity quantity penalty of 0.97. This configuration produced the best balance of mean score, robustness, and simplicity, and is the version selected for the final evaluation.

Table 2: Headline longer mixed-baseline evaluation.

Metric	Value
Valid runs collected	10
Steps per valid run	10
Mean Group3 score	1.0866
Median Group3 score	1.0783
Best Group3 run average	1.1918
Worst Group3 run average	0.9850
Mean mixed baseline score	0.7653
Median mixed baseline score	0.7570
Mean score improvement	0.3213
Score ratio vs baselines	1.42×
Win rate vs baseline average	100.00%

Additional diagnostics were then used to verify that the opponent model produced interpretable signals; these diagnostics showed that the signals are sparse in practice (see Section 5.5), which led to the interpretation adopted throughout this report: portfolio control and time-based concession are Group3’s core performance drivers, while opponent modelling provides secondary adaptation.

5 Results

5.1 Evaluation methodology

Group3 was evaluated against the built-in SCML OneShot baselines `RandomOneShotAgent`, `SyncRandomOneShotAgent`, `GreedyOneShotAgent`, `GreedySyncAgent`, and `SingleAgreementAspirationAgent`. In the pooled tables, the *Aspiration-family* refers to agents generated from `SingleAgreementAspirationAgent`, and the *Sync-family* refers to sync-style built-in agents (`SyncRandomOneShotAgent` and `GreedySyncAgent`).

The win-rate metric is a *run-level win rate against the mixed-baseline average*: a run counts as a win if Group3’s average score in that generated world is strictly greater than the average score of all non-Group3 baseline agents in the same world. This is not a pairwise win rate against every individual agent. A generated world was marked invalid only when it did not contain a detectable Group3 instance in the score table; the skip criterion is therefore independent of Group3’s performance because no Group3 score exists in such a world. Two table types are reported. Run-average tables average Group3’s mean score per valid simulation. Pooled-family tables aggregate every individual agent of a family across all runs; the pooled count for Group3 can exceed the number of valid runs when a generated world instantiates multiple Group3 agents.

5.2 Headline mixed-baseline evaluation

The strongest 10-step longer mixed-baseline evaluation of the selected Group3 design is reported as the headline result (Table 2), using 10 valid runs at 10 simulation steps per valid run.

Table 3: Supporting 10-step mixed-baseline check.

Metric	Value
Valid runs collected	10
Attempted simulations	12
Skipped invalid runs	2
Steps per valid run	10
Mean Group3 score	1.0776
Median Group3 score	1.0896
Best Group3 run average	1.2187
Worst Group3 run average	0.8073
Mean mixed baseline score	0.8469
Mean score improvement	0.2307
Score ratio vs baselines	1.27×
Win rate vs baseline average	90.00%

Table 4: Pooled family scores in the supporting mixed-baseline check.

Family	n	Mean	Median	Best
Aspiration-family	19	0.7656	0.8570	0.9791
Greedy-family	46	0.8114	0.8471	1.0127
Group3	18	1.0760	1.0885	1.2513
Random-family	20	0.6789	0.7750	0.9446
Sync-family	24	1.0810	1.0885	1.2213

5.3 Supporting check

A later 10-step mixed-baseline check (Table 3) used a slightly different baseline composition that produced harder worlds; the mean score was similar but the win rate was lower, which is consistent with run-to-run stochastic variation.

The two 10-step evaluations agree that Group3 generally scores around 1.07–1.09 and consistently outperforms the mixed-baseline average. The difference in ratio and win rate reflects stochastic world generation and different baseline mixes, not a difference in the agent itself.

5.4 Pooled family comparison

Pooled means rank Sync-family slightly above Group3 (1.0810 vs 1.0760, Table 4). In a separate baseline-family diagnostic run (Section 5.5, Table 5) the ordering reverses: Group3 reaches 1.0640 versus Sync-family 1.0605. Both differences are small and inside the run-to-run variation, so the defensible reading is that Group3 is *competitive* with Sync-family agents without consistently dominating them.

5.5 Baseline-family diagnostic evaluation

The pooled version of this diagnostic run gave Group3 a mean of 1.0640 compared with 1.0605 for Sync-family agents. This again supports the interpretation that Group3 is close to the strongest sync baselines.

5.6 Opponent-model and portfolio diagnostics

Table 6 reports the diagnostic evaluation that probes the internal state of the opponent model and the portfolio matching behaviour.

Table 5: Baseline-family diagnostic evaluation.

Metric	Value
Valid runs collected	10
Attempted simulations	11
Skipped invalid runs	1
Steps per valid run	5
Mean Group3 score	1.0797
Median Group3 score	1.0676
Best Group3 run average	1.2971
Worst Group3 run average	0.9619
Mean mixed baseline score	0.8286
Mean score improvement	0.2511
Score ratio vs baselines	1.30×
Win rate vs baseline average	100.00%

Table 6: Opponent-model and portfolio diagnostic evaluation.

Metric	Value
Valid runs collected	10
Attempted simulations	10
Skipped invalid runs	0
Steps per valid run	10
Mean Group3 score	1.1109
Median Group3 score	1.1082
Worst Group3 run average	1.0177
Mean mixed baseline score	0.8038
Win rate vs baseline average	100.00%
Opponents tracked	96
Sensitivity observations	35
Offer-value samples	433
Zero offer values	340
Non-zero offer values	93
Matched day rate	71.25%
Overshoot day rate	19.38%
Undershoot day rate	9.38%

The diagnostic numbers are important because they prevent the report from overclaiming the role of the opponent model. Although the model is implemented and consulted on every decision, 340 of 433 offer-value samples were zero, and the behaviour counts were dominated by silent classifications (2470 silent, 329 selfish, 330 nice, and 1 concession). This means the opponent model is not the main measured performance driver. The portfolio module, in contrast, shows direct evidence of useful behaviour: the matched-day rate is 71.25%, with 19.38% overshoot and 9.38% undershoot days. Because most decisions are taken when the opponent signal is uninformative, the agent in practice falls through to the unadjusted portfolio-plus-Boulware base case, with the opponent-derived adjustments acting as occasional risk filters rather than as a constant adaptation pressure.

6 Discussion

6.1 Why the agent performs well

Group3 performs well because its strategy matches the structure of the SCML OneShot problem. It does not treat

each negotiation as an isolated price bargain. Instead, it asks whether each contract helps complete the current day’s trading objective. This avoids contracts that look acceptable on price alone but are dangerous on quantity, and it directly attacks the dominant scoring term — the disposal and shortfall penalties — by making quantity matching a first-class decision variable. The empirical signature of this design choice is the 71.25% matched-day rate in Table 6, together with the consistent score range of 1.07–1.11 across every evaluation, including the headline $1.42\times$ ratio and 100% run-level win rate.

6.2 Role of the opponent model

The opponent model is useful for interpretability and for small risk-side adjustments, but the diagnostics show it should not be described as the primary performance source. Many opponent actions appear silent under the local value metric, and sensitivity updates are scarce. This does not make the model useless: it still records behaviour, detects some selfish or nice movement, and adjusts exposure when it does fire. However, its measured effect is small relative to portfolio-aware quantity control, and this distinction matters — a sophisticated component is only valuable if it improves actual tournament outcomes. Group3’s final design therefore keeps the opponent model as a lightweight adaptive layer but deliberately avoids over-relying on it.

6.3 Comparison with Sync-family agents

Sync-family agents are the strongest competitors. In one pooled evaluation, Sync-family slightly exceeded Group3 by mean score (Table 4); in another diagnostic batch, Group3 slightly exceeded Sync-family. These differences are small relative to stochastic variation across generated worlds. The most defensible conclusion is therefore that Group3 is competitive with Sync-family baselines, not that it always dominates them.

6.4 Deficiencies and future work

The main deficiencies of the current implementation are: (i) **limited global synchronisation** — Group3 is built on `OneShotAgent`, so it handles negotiations individually rather than making a single coordinated decision across all simultaneous offers; (ii) **sparse opponent-model updates** — sensitivity updates only occur when the required offer-history pattern exists; (iii) **silent-behaviour dominance** — many offer changes are too small or zero-valued under the local proxy to be classified meaningfully; and (iv) **occasional weaker worlds**, where stochastic generation produces runs in which Sync-family baselines are unusually strong. Natural improvements include adding a synchronised offer-selection layer that evaluates all simultaneous offers together; improving the opponent-value proxy to reduce zero-valued samples; learning concession parameters offline by repeated simulation; running larger tournament samples to obtain statistical significance estimates; and adding confidence-weighted opponent adaptation so that weak signals do not over-influence decisions.

7 Conclusion

We presented Group3, a portfolio-aware negotiation agent for SCML OneShot. The agent implements all required components — bid generation, acceptance decisions, opponent modelling, and strategy adaptation — with portfolio-aware quantity control integrated tightly into Boulware-style price concession and a stability-aware acceptance filter. The opponent model provides interpretability and limited adaptation, but diagnostics show that portfolio management is the dominant performance driver.

Experimental results show that Group3 consistently outperforms mixed built-in baselines and remains competitive with strong Sync-family agents. The strongest longer mixed-baseline evaluation achieved a mean score of 1.0866, a score ratio of $1.42\times$, and a 100% run-level win rate against the mixed-baseline average. Supporting evaluations produced similar means in the 1.0776–1.0797 range. The remaining headroom is structural rather than parametric — a fully synchronised acceptance layer that decides over all concurrent offers at once is the most promising next step.

References

- [1] P. Faratin, C. Sierra, and N. R. Jennings. Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, 24(3–4):159–182, 1998.
- [2] K. Hindriks and D. 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*, 2008.
- [3] T. Baarslag, M. J. C. Hendriks, K. V. Hindriks, and C. M. Jonker. Learning about the opponent in automated bilateral negotiation: a comprehensive survey of opponent modeling techniques. *Autonomous Agents and Multi-Agent Systems*, 30:849–898, 2016.
- [4] Y. Mohammad. NegMAS: A platform for situated negotiations, 2020. Software framework and documentation, <https://negmas.readthedocs.io/>.
- [5] Y. Mohammad and SCML contributors. Supply Chain Management League documentation, 2024. ANAC/SCML competition framework and OneShot environment, <https://scml.readthedocs.io/>.