

AssariAsari: An Agent Submitted to the ANAC 2026 SCML OneShot League

Rinon Asanuma
Tokyo University of Agriculture and Technology
asanuma@katfujii.lab.tuat.ac.jp

June 11, 2026

Abstract

This report describes ASSARIASARI, a synchronous negotiation agent for the ANAC 2026 SCML OneShot league. The agent follows a compact need-driven strategy: it distributes current supply and sales needs across active partners, uses time-dependent price concession, accepts compatible subsets of concurrent offers, and applies simple risk controls to avoid excessive over-commitment.

1 Introduction

The SCML OneShot track requires an agent to negotiate simultaneously with suppliers and consumers under short-horizon production constraints. In this setting, a good agreement is not only a good bilateral deal: it must also fit the agent’s aggregate quantity needs. Accepting too little causes shortage, while accepting too much can create costly surplus.

ASSARIASARI is designed as a conservative and interpretable OneShot agent. Instead of optimizing each negotiation independently, it treats negotiations on the same market side as a group. It proposes quantities by splitting the current need across active partners and later accepts groups of offers whose combined quantity best matches that need.

2 The Design of AssariAsari

2.1 Negotiation Choices

ASSARIASARI extends `OneShotSyncAgent`. Its `first_proposals()` method creates one proposal for each active partner. The proposed quantity is based on the current values of `awi.needed_supplies` and `awi.needed_sales`. These needs are split across active suppliers and consumers using mild capacity-based weights, so larger-capacity partners receive slightly larger quantities without making the strategy depend on a single partner.

Prices are generated using a concession function

$$c(t) = t^\alpha,$$

where t is the relative negotiation time and $\alpha = 0.50$ by default. When selling, the agent starts near the maximum unit price and concedes downward. When buying, it starts near the minimum unit price and concedes upward. Incoming prices are normalized so that high prices are preferred for sales and low prices are preferred for purchases.

2.2 Concurrent Negotiation

The main decision logic is implemented in `counter_all()`. For each market side, the agent evaluates currently received offers as a set rather than one by one. If there are at most ten partners, it enumerates all non-empty subsets of active offers. For larger cases, it uses a bounded heuristic based on quantity closeness and price quality.

For a candidate subset S , the agent compares the total offered quantity q_S with the current need q_{need} . The mismatch cost is

$$\max(0, q_{\text{need}} - q_S) + 1.30 \max(0, q_S - q_{\text{need}}).$$

Thus, surplus is penalized more strongly than shortage. The agent chooses the subset with the best quantity fit, using price quality and partner reliability as tie-breakers. If the selected subset is within the current quantity slack, all offers in that subset are accepted together; otherwise, the agent rejects and sends updated counter-offers.

2.3 Risk Management

The strategy contains three simple risk-management mechanisms. First, negotiations are ended when there is no remaining need on that side of the market. Second, surplus quantity is explicitly penalized through the overfill term above. Third, the agent maintains a lightweight reliability estimate for each partner:

$$r_p = \frac{s_p + 1}{s_p + f_p + 2},$$

where s_p and f_p are the observed successful and failed negotiations with partner p . This smoothed estimate helps the agent prefer partners that have historically reached agreements.

The agent also becomes more tolerant late in negotiation. Its quantity slack is proportional to $t^{3.5}$, making it strict early and more willing to accept small quantity mismatches near the deadline.

3 Evaluation

The implementation is intended for evaluation with the official ANAC 2026 SCML OneShot simulator. Since the agent does not require offline training, the main evaluation criteria are quantity coordination, agreement rate, and price discipline.

Mechanism	Implementation idea	Expected effect
Need splitting	Split current needs across active partners.	Avoid dependence on a single negotiation.
Time concession	Use $t^{0.50}$ for price concession.	Increase agreement chance over time.
Subset acceptance	Accept compatible groups of offers.	Coordinate simultaneous contracts.
Overfill penalty	Penalize surplus quantity by factor 1.30.	Reduce over-contracting risk.
Reliability estimate	Use a smoothed success ratio.	Prefer partners with successful history.
Late slack	Let quantity tolerance grow as $t^{3.5}$.	Avoid losing late acceptable deals.

Table 1: Main design components of ASSARIASARI.

Qualitatively, ASSARIASARI should be strongest when aggregate quantity matching is crucial. Its main limitation is that the opponent model is shallow: it records success and failure, but does not estimate opponent reservation values or detailed concession patterns.

4 Lessons and Suggestions

The main lesson is that OneShot negotiation should prioritize coordinated quantity decisions. A good price can still be harmful if it creates an infeasible set of contracts. The subset-acceptance rule directly addresses this by accepting offers only when they fit the current aggregate need.

Future work could add a richer opponent model, tune parameters such as the concession exponent and overfill penalty, or make the initial quantity distribution depend on learned partner behavior.

Conclusions

ASSARIASARI is a compact synchronous agent for the ANAC 2026 SCML OneShot league. It combines need-driven proposals, time-based price concession, concurrent subset acceptance, and conservative risk management. The design is simple, interpretable, and suitable as a robust baseline for future extensions.