

IscasAgent: A Hybrid Preference-Concealing Negotiation Agent for ANL 2026

ANL 2026 Submission

Abstract

This report describes **IscasAgent**, a bilateral negotiation agent designed for the ANL 2026 league. The league rewards both agreement utility and preference concealment, so the agent must avoid a simple utility-maximizing trajectory that reveals its private preferences. Our design follows a three-module architecture: an analytical “brain” controls aspiration and reservation safety, a statistical “eyes” module estimates the opponent’s preferences from observed offers, and a probabilistic “hands” module samples offers from a utility-equivalent pool. The result is a white-box agent that can produce an explicit opponent utility estimate while making its own offer path noisy, adaptive, and less directly recoverable.

1 Introduction

ANL 2026 changes the usual negotiation objective by scoring an agent not only on the utility of the reached agreement, but also on how well it models its opponent and hides its own preferences. A strong agent therefore needs to satisfy three constraints at the same time. First, it should protect its own advantage by maintaining a disciplined concession curve. Second, it should expose an opponent model through `private_info["opponent_ufun"]`. Third, it should not bid in a rigid pattern, because deterministic high-utility bidding makes the agent easy to reverse engineer.

IscasAgent implements this trade-off in `iscas.py`. It inherits from the provided BOA-style negotiator when the example package is available, while falling back to `SAOCallNegotiator` in a submission-only environment. The implementation keeps all strategic logic local to the submitted file and uses only lightweight statistics, avoiding cross-session learning or heavy inference.

2 Agent Architecture

The agent is organized as a hybrid of analytical control, statistical opponent modeling, and probabilistic offer generation.

Module	Main functions	Role
Brain	<code>_aspiration_floor</code> , <code>acceptance_strategy</code>	Protect utility and deadline safety
Eyes	<code>update_opponent_model</code> , <code>_estimate_opponent_utility</code>	Infer opponent utility
Hands	<code>_utility_equivalence_pool</code> , <code>_bid_score</code> , <code>_softmax_pick</code>	Generate concealed offers

During `on_preferences_changed`, the agent enumerates the outcome space, caches its own utility for every outcome, records all rational outcomes above the reservation value, and estimates the importance of each issue for its own preferences. This precomputation allows the agent to make fast online decisions in finite domains. It also initializes the opponent model as a `LambdaMultiFun`, so that the competition scorer can compare the predicted utility function against the real opponent utility.

Let O be the finite outcome space, $u(o)$ our utility, and r the reservation value. The agent caches

$$R = \{o \in O \mid u(o) > r\}, \quad U_m(o) = \frac{u(o) - u_{\min}}{u_{\max} - u_{\min} + \epsilon},$$

where R is the feasible rational set used by all bidding and acceptance decisions.

3 Concealing Bidding Strategy

The bidding strategy is designed to avoid the two common failure modes of simple agents: always selecting one of the best outcomes, or randomly selecting from all rational outcomes. The first reveals preferences too quickly; the second sacrifices too much utility. Instead, **IscasAgent** uses utility-equivalence sampling.

At each turn, `_utility_equivalence_pool` builds a candidate pool around the current aspiration target. The pool is centered on the analytical floor, with a time-dependent band that widens near the deadline and in deadlock situations. In sparse domains, the pool is repaired by selecting the nearest utility peers without falling far below the current target. This means that stochasticity is applied among outcomes with comparable value to the agent rather than across the whole rational set.

The equivalence pool is formally

$$P_t = \{o \in R \mid L_t \leq u(o) \leq H_t\}, \quad L_t = \max(r + \epsilon, A_t - b_t), \quad H_t = \min(u_{\max}, A_t + b_t),$$

where A_t is the current aspiration floor and

$$b_t = (u_{\max} - r)(0.035 + 0.055t) \cdot d_t.$$

The multiplier d_t is enlarged under detected deadlock or when the opponent appears to stop conceding. This is the main concealment safeguard: randomization is restricted to a utility-equivalent neighborhood.

Each candidate is scored by `_bid_score` using a weighted sum:

$$S(o) = w_m U_m(o) + w_o \hat{U}_o(o) + w_c C(o) + w_n N(o) + w_p U_m(o) \hat{U}_o(o) - w_g G(o).$$

Here U_m is the agent’s normalized utility, $\hat{U}_o$ is the estimated opponent utility, C rewards hiding the agent’s best-issue signature, N rewards novelty relative to recent offers, the product term biases toward Nash-like outcomes in the later phase, and G penalizes offers estimated to be below the opponent’s likely acceptance level. The own-utility weight is deliberately high early in the negotiation, while opponent and Nash terms gain influence later or during deadlock.

The main time-varying weights are

$$w_m = 2.26 - 0.30t, \quad w_o = 0.10 + 0.36t, \quad w_c = 0.22(1 - 0.30t), \quad w_n = 0.10(1 - 0.35t),$$

$$w_p = 0.03 + 0.28 \max\left(0, \frac{t - 0.55}{0.45}\right).$$

Thus, own utility dominates early, while opponent utility and the Nash product become more important only when agreement pressure increases.

Finally, `_softmax_pick` samples from the top-scored shortlist using

$$\Pr(o \mid P_t) = \frac{\exp(S(o)/\tau_t)}{\sum_{o' \in P_t} \exp(S(o')/\tau_t)}, \quad \tau_t = 0.14(1 - t) + 0.028.$$

The temperature decreases over time, making early offers more diverse and late offers more convergent. This stochastic layer makes the trajectory harder to fit while still staying near the current utility target.

4 Aspiration and Adaptation

The analytical brain controls concession with a dynamic Boulware-like floor. Its exponent is not fixed; it reacts to estimated conflict, domain size, deadlock, and opponent concession. In simplified form,

$$e_t = 2 + 2.7\Omega_t + \Delta_{\text{anchor}} + \Delta_{\text{domain}} + \Delta_{\text{slope}} - \Delta_{\text{deadlock}},$$

where Ω_t is the estimated opposition level. The aspiration is

$$A_t = \max(r + B_t, \min(u_{\max}, u_{\max} - (u_{\max} - r)(\lambda t^{e_t} + 0.14\ell_t) + J_t)).$$

Here B_t is a dynamic reservation buffer, $\lambda \in \{0.66, 0.72\}$ depends on the opponent anchor utility, ℓ_t is a late-release term, and J_t is a small sinusoidal perturbation. This formula covers four design choices: adaptive exponent, reservation-value safety, late deadline release, and non-deterministic trajectory masking.

5 Acceptance Strategy

The acceptance policy combines reservation safety, aspiration control, and ACNext-style reasoning.

- Offers below the reservation value are rejected immediately.

- Very high utility offers are accepted immediately.
- The agent computes an expected next-offer utility using `_expected_next_utility`. If the current offer is at least as good as what the agent can reasonably propose next, it is accepted.
- Otherwise the offer must beat the current analytical floor plus a time-dependent slack.
- Near the deadline, the policy allows agreement above a buffered reservation level to reduce timeout risk.

The ACNext part is intentionally based on the best utility among several high-scored planned bids, not on a single random bid. This prevents the concealment noise from making the agent accept weak offers simply because one stochastic sample happened to be low.

The main acceptance rule can be summarized as

$$\text{accept}(o_t) = \begin{cases} \text{true,} & \text{if } u(o_t) > r \text{ and } [u(o_t) \geq u_{\max} \text{ or } u(o_t) \geq E_t^{\text{next}} - m_t \\ & \text{or } u(o_t) \geq A_t + s_t \text{ or } D_t(o_t)], \\ \text{false,} & \text{otherwise.} \end{cases}$$

where E_t^{next} is the expected utility of the next planned counteroffer, $m_t = (u_{\max} - r)(0.003 + 0.010t)$, s_t is the aspiration slack, and D_t is the deadline safety condition. The opponent acceptance gap used in bidding is

$$G(o) = \max(0, \theta_t - \hat{U}_o(o)), \quad \theta_t = \text{clip}(a(1 - (t + 0.045)^k) - 0.035, 0.08, 0.98),$$

where a is the opponent anchor utility and k is larger in sparse domains. This discourages offers that are good for us but estimated to be unacceptable to the opponent.

6 Opponent Model

The opponent model is explicit and frequency-based, with several robustness corrections. Every opponent offer is treated as evidence about issue values. Early offers receive stronger weight because they are usually closer to the opponent's ideal point, while `_blended_value_counts` mixes this cumulative evidence with a short recent window so that the model can still adapt to later concessions.

For issue i and value v , the count update is

$$c_i(v) \leftarrow \begin{cases} c_i(v) + \alpha(t), & \text{if } o_t^i = v, \\ c_i(v), & \text{otherwise,} \end{cases} \quad \alpha(t) = 0.45 + 2.80(1 - t)^2.$$

The blended count also adds a recency window:

$$\tilde{c}_i(v) = c_i(v) + \sum_{a=0}^{K-1} \gamma_a(v), \quad \gamma_a(v) = \begin{cases} 1.10(0.72)^a, & \text{if } o_{t-a}^i = v, \\ 0, & \text{otherwise.} \end{cases}$$

Issue weights are inferred from two signals:

- **Value concentration:** if an opponent repeatedly uses a small subset of values on an issue, that issue is considered more important.
- **Issue stickiness:** if an issue changes rarely across consecutive opponent offers, it receives additional weight.

Value utilities are estimated with Laplace smoothing, normalized per issue, and combined with the inferred issue weights. A very small opposite-preference prior is blended in for highly competitive domains, but the frequency evidence remains dominant. The model is exposed through `private_info["opponent_ufun"]`.

More explicitly, value concentration is

$$\rho_i = 1 - \frac{\sum_v p_i(v) \log(p_i(v) + \epsilon)}{\log |V_i| + \epsilon}, \quad p_i(v) = \frac{\tilde{c}_i(v)}{\sum_{v'} \tilde{c}_i(v')}.$$

The raw issue weight is

$$\bar{w}_i = 0.35 + 0.50\rho_i + 0.25\sigma_i, \quad w_i = \frac{\bar{w}_i}{\sum_j \bar{w}_j},$$

where σ_i is issue stickiness. Value scores are smoothed and normalized as

$$q_i(v) = \frac{\tilde{c}_i(v) + 0.7}{\sum_{v'} \tilde{c}_i(v') + 0.7|V_i|}, \quad s_i(v) = 0.15 + 0.85 \frac{q_i(v) - q_i^{\min}}{q_i^{\max} - q_i^{\min} + \epsilon}.$$

The opponent utility estimate is

$$\hat{U}_o(o) = (1 - \beta) \sum_i w_i s_i(o_i) + \beta(1 - U_m(o)), \quad \beta = \min(0.08, 0.02 + 0.06\Omega_t).$$

The model also produces strategic signals for the bidding and acceptance modules. `_detect_deadlock` detects repeated or near-repeated offers. `_estimate_concession_slope` fits a lightweight trend over recent estimated opponent utilities. `_estimate_opposition_level` compares the overlap between the agent’s top outcomes and the estimated opponent top outcomes, blending this with issue-weight disagreement. These signals determine how hard the aspiration curve should remain and how much the bidding pool should widen.

The opposition level is

$$\Omega_t = 0.70 \left(1 - \frac{|T_m \cap T_o|}{K} \right) + 0.30 \left(1 - \frac{\mathbf{w}_m \cdot \mathbf{w}_o}{\|\mathbf{w}_m\| \|\mathbf{w}_o\| + \epsilon} \right),$$

where T_m and T_o are the top- K outcome sets under our utility and the estimated opponent utility. This is the signal used by the adaptive exponent in the aspiration formula.

7 Evaluation

The implementation was checked with the local test suite and local command-line negotiations. The tests verify instantiation, BOA compatibility, negotiation completion, valid offers, valid agreements, multiple issue counts, and compatibility against different example opponents. The full project tests passed locally after allowing the tournament tests to write their output directory.

The qualitative behavior is summarized below.

Opponent type	Observed behavior	Strategic response
Conceder/time-based	Gives ground reliably	Maintain high floor and accept strong offers
Boulware	Concedes late	Preserve advantage, widen near deadline
Frequency/BOA/MAP	Models bid statistics well	Use equivalent-pool stochasticity and novelty
Simple/opposite-like	May accept high self-utility deals	Keep own utility dominant early

The design does not specialize against a single opponent. Against strong BOA/MAP-style agents, individual runs can still be lost because both sides use frequency reasoning. However, the current strategy is intended to generalize across scenario sizes and opponent types: it protects advantage through an analytical floor, avoids deterministic offer paths, and maintains an explicit model for the scoring component.

8 Lessons and Future Work

The main lesson is that concealment should be applied inside a high-utility neighborhood. Randomness alone is harmful if it moves the agent too far from its own good outcomes. A second lesson is that opponent modeling should be simple enough to be stable online; a smoothed frequency model with recency correction was more reliable than a heavier model with many assumptions.

Future work would tune the weights in `_bid_score` using larger tournaments, add a more principled estimate of opponent acceptance thresholds, and test whether the final opponent utility ranking can be improved by a Kendall-distance-oriented post-processing step. These additions should remain lightweight and should not rely on cross-negotiation memory, preserving compatibility with the competition setting.

Conclusion

IscasAgent is a transparent hybrid negotiation agent for ANL 2026. It combines an adaptive aspiration curve, an explicit frequency-based opponent model, and SoftMax sampling from utility-equivalent offers. This gives the agent a practical balance between agreement utility, opponent modeling, and preference concealment.