

DecepTor: A Deceptive Negotiating Agent for ANL 2026 Hybrid Bidding, Utility-Equivalent Bid Swaps, and Adaptive Opponent Modeling

Cetin Hosafci¹, Mert Altintas²

¹Department of Computer Science, ²Department of Computer Science

¹cetin.hosafci@ozu.edu.tr, ²mert.altintas@ozu.edu.tr

CS 451: Negotiating Agent Project — Spring 2026

Abstract

This progress report presents the design of **DecepTor**, a bilateral negotiating agent developed for the Automated Negotiation League (ANL) 2026 track of ANAC 2026. The agent is implemented on NegMAS [5] and follows the BOA (Bidding, Opponent modeling, Acceptance) architecture. DecepTor introduces novelty in two components: (1) a deception-aware bid selection mechanism based on utility-equivalent bid swaps that systematically corrupts the opponent's frequency model while maintaining high self-utility, targeting the Kendall τ deception scoring bonus; and (2) an adaptive concession rate that responds to estimated opponent behaviour type using the full DANS classification framework [4]. The acceptance strategy integrates time pressure, a utility threshold, Nash-point proximity, and opponent concession trend detection. A preliminary self-play experiment is reported and a full ablation study against all ANL 2026 baseline agents is planned.

Keywords: automated negotiation, deception, opponent modeling, Boulware strategy, utility-equivalent swaps, DANS framework, ANL 2026, NegMAS, Kendall tau

1 Introduction

Automated bilateral negotiation requires agents to simultaneously obtain favourable agreements and reason about the opponent's preferences and behaviour. The ANL 2026 challenge adds a third dimension: *deception*. Agents are scored not only on agreement utility but also on how well they mislead the opponent's model of their own utility function, measured by the Kendall rank correlation coefficient τ .

Negotiations use the Alternating Offers Protocol (AOP). Time pressure and deadlines play an important role in shaping negotiation dynamics [3]. Utility functions are linear-additive over discrete issues [6]:

$$u(b) = w_1 \cdot u(i_1) + w_2 \cdot u(i_2) + \dots + w_n \cdot u(i_n) \quad (1)$$

The combined ANL 2026 score for agent A is:

$$\text{Score}_A = \text{adv}_A + \frac{\tau^{(A)}}{\tau^{(A)} + \tau^{(B)}} \quad (2)$$

where $\text{adv}_A = (u_A(o) - r_A) / (\max u_A - r_A)$, $\tau^{(A)} = \frac{1}{2}(1 + d(\hat{u}_B, u_B))$, and $\tau^{(B)} = \frac{1}{2}(1 + d(\hat{u}_A, u_A))$. To maximise Score_A we want $\tau^{(A)}$ high and $\tau^{(B)}$ low.

Table 1: Agent decision pipeline and Python methods.

Step	Method	Description
1	<code>on_negotiation_start()</code>	Sort outcomes, build H/D sets, set $e=3$
2	<code>opp_model.update()</code>	Update GSmithFrequencyModel
3	<code>behavior_tracker.update()</code>	Classify move (DANS), update $\hat{\delta}(t)$
4	<code>_adapt_e()</code>	Adjust e from behaviour type
5	<code>_should_accept()</code>	Evaluate Algorithm 1
6	<code>deception_layer.select()</code>	Select from D or H set
7	<code>_call_()</code>	Return SAOResponse

2 Selected League and Agent Overview

We have selected the **Automated Negotiation League (ANL 2026)**. As undergraduate students our eligible leagues are ANL and HAN; ANL was chosen because it provides a clearly defined dual scoring formula, rich baseline agents, and a well-defined research angle through the deception challenge.

DecepTor is built on the BOA architecture with four modules: (1) **Bidding Module** — time-based Boulware with adaptive exponent e ; (2) **Opponent Modeling Module** — preference estimation via GSmithFrequencyModel extended with DANS behaviour classification; (3) **Acceptance Module** — composite four-condition strategy; (4) **Deception Layer** — pre-computed utility-equivalent bid sets with time-varying schedule.

3 Bidding Strategy

3.1 Factors Considered

The bidding decision considers: (i) relative time $t \in [0, 1]$; (ii) own utility — all bids from $F(t) = \{b \mid u_A(b) \geq \alpha(t)\}$; (iii) reservation value r — bids never below $r + 0.05$; (iv) adaptive exponent e ; (v) estimated opponent utility — honest bids maximise Nash-point proximity; (vi) deception schedule $p_{\text{dec}}(t)$.

3.2 Time-Based Aspiration Function

Following Faratin et al. [2]:

$$\alpha(t) = r + 0.05 + (1 - t^{1/e}) \cdot (u_{\max} - r - 0.05) \quad (3)$$

As e increases, $t^{1/e}$ decreases more slowly, keeping $\alpha(t)$ high longer — producing Boulware behaviour. **Boulware mode** ($e = 0.2$): slow decay, used when opponent concedes. **Conceder mode** ($e = 3.0$): fast decay, used when opponent is stubborn.

3.3 Opening Bid and Counteroffers

The first offer is $b^* = \arg \max u_A(b)$, i.e. $\alpha(0) = u_{\max} - 0.05$ — maximum utility, standard Boulware practice [2]. Since $\alpha(t)$ is non-increasing in t for any $e > 0$:

$$u(b_t) \geq \alpha(t) \geq \alpha(t+1) \geq u(b_{t+1}) \quad (4)$$

This formally guarantees **monotonically non-increasing** bid utility, satisfying requirement 2.1(a)(vi). Deceptive bids are constrained to $\geq 0.92 \cdot \alpha(t)$ — a deviation of at most 8% per deceptive round.

3.4 Adaptive Concession Rate

An EMA tracks opponent utility changes:

$$s_t = 0.8 \cdot s_{t-1} + 0.2 \cdot (\hat{u}_t^{\text{opp}} - \hat{u}_{t-1}^{\text{opp}}) \quad (5)$$

If $s_t > 0.05$: increase e (stay Boulware). If $s_t < -0.05$: decrease e toward 1 (concede faster). Otherwise: keep current e . This satisfies requirement 2.1(a)(vii).

4 Opponent Modeling

4.1 Opponent Preference Model

We use GSmithFrequencyModel [5] extended with inverse-change-frequency weighting [7], more stable than variance-based weights in short negotiations:

$$\widehat{\text{val}}(i, v) = \frac{c(i, v)}{\sum_{v'} c(i, v')}, \quad w(i) = \frac{1/(1 + \text{changes}(i))}{\sum_j 1/(1 + \text{changes}(j))} \quad (6)$$

$$\hat{u}_B(b) = \sum_i w(i) \cdot \widehat{\text{val}}(i, b[i]) \quad (7)$$

Used for: (1) Pareto-improving bid selection among $F(t)$ candidates; (2) Nash point approximation for the acceptance strategy.

4.2 Behaviour Classification (DANS)

Following Hindriks et al. [4], each opponent move is classified into one of six categories (Table 2). The dominant type drives e adaptation.

4.3 Concession Rate Estimation

The instantaneous concession rate smoothed by sliding window:

$$\hat{\delta}(t) = \frac{u_A(b_{t-1}^{\text{opp}}) - u_A(b_t^{\text{opp}})}{t - t_{\text{prev}}} \quad (8)$$

Large positive $\hat{\delta}$: opponent conceding. Near zero: stubborn.

Table 2: Opponent move classification — DANS framework [4].

Move	Δu_A	$\Delta \hat{u}_B$	Meaning
Fortunate	↑	↑	Good for both
Selfish	↑	↓	Opponent moving away
Concession	↓	↑	Opponent giving utility
Nice	—	↑	Improving offer
Silent	—	—	No change
Unfortunate	↓	↓	Bad for both

Algorithm 1: Composite Acceptance Decision

Require: offer b^{opp} , time t , reservation value r

```

1: if  $u_A(b^{\text{opp}}) < r$  then
2:   Reject {Never accept below reservation}
3: else if  $u_A(b^{\text{opp}}) \geq \alpha(t)$  then
4:   Accept {Condition A: ACNext guarantee}
5: else if  $t \geq 0.90$  and  $u_A(b^{\text{opp}}) \geq 0.95 \cdot u_A^{\text{Nash}}$  then
6:   Accept {Condition B: Nash-point check}
7: else if  $\hat{\delta}(t) < 0$  and  $t > 0.70$  then
8:   Accept {Condition C: opponent reversing late}
9: else
10:  Reject
11: end if

```

5 Acceptance Strategy

5.1 Factors Considered

(i) Reservation value r — first guard, never accept below; (ii) utility threshold $\alpha(t)$ — ACNext guarantee [1]; (iii) time pressure — $\alpha(t)$ decays toward $r + 0.05$; (iv) Nash proximity in final 10% of rounds; (v) opponent trend $\hat{\delta}(t)$ after $t = 0.70$.

5.2 Composite Acceptance Algorithm

Algorithm 1 gives the full decision. In Condition B, u_A^{Nash} is approximated by maximising $(u_A - r) \cdot \hat{u}_B$ over the top-200 outcomes ranked by u_A .

6 Deception Strategy (Novel Component)

6.1 Motivation

To maximise Score_A we want $\tau^{(B)}$ low, i.e. $d(\hat{u}_A, u_A) \rightarrow -1$. For additive utility functions, bid pairs (b, b') with $|u_A(b) - u_A(b')| < \varepsilon$ but very different issue-value distributions exist. We systematically offer bids from a pre-computed deceptive set to corrupt the opponent's frequency model.

6.2 Utility-Equivalent Bid Swaps

At initialisation, DecepTor pre-computes for each utility level u : (1) $S_u = \{b : |u_A(b) - u| < \varepsilon_u\}$ with $\varepsilon_u = 0.02$; (2) *honest set* H_u (reflecting true preferences); (3) *deceptive set* D_u (high-weight issues take non-optimal values, $\text{rank}_{v_1}(b_d) > \text{rank}_{v_1}(b^*) + \Delta$ with $\Delta = 1$, compensated by optimal values on low-weight issues).

Table 3: Preliminary self-play results.

Metric	Value	Note
Agreement reached	Yes	No deadline failure
Rounds to agreement	38/50	Before deadline
Utility — Agent 1	0.71	$> r = 0.30$
Utility — Agent 2	0.68	$> r = 0.30$
Social welfare $U_A + U_O$	1.39	$\approx 90\%$ of max ~ 1.55
Nash product $U_A \cdot U_O$	0.483	Near-Nash
Agreement rate	100%	Single session
Nash distance	~ 0.11	Near but not Nash
Deception score (τ)	Pending	Requires tournament

6.3 Deceptive Bidding Schedule

The fraction of deceptive bids increases over time:

$$p_{\text{dec}}(t) = \min(0.25, 0.15 + 0.10 \cdot t) \quad (9)$$

At each round, the deception layer samples $p \sim \mathcal{U}(0, 1)$. If $p < p_{\text{dec}}(t)$ and $D_{\alpha(t)} \neq \emptyset$: select from $D_{\alpha(t)}$ the bid maximising Kendall distance to our true ranking. Otherwise: select from $H_{\alpha(t)}$ the bid closest to the Nash point.

6.4 Trade-off and Originality

Deceptive bids ($\geq 0.92 \cdot \alpha(t)$, 15–25% of rounds) cause an expected utility loss below 1.2% while the τ bonus can be substantial. Unlike random noise injection, the structured policy creates a systematic false pattern harder for frequency models to recover. This is our principal novel contribution.

7 Preliminary Experiments

A self-play session (DecepTor vs. DecepTor) was run on a 3-issue domain (Price×Delivery×Warranty, 50 rounds, $r = 0.30$). Table 3 shows results.

Both agents agreed above r . Social welfare 1.39 ($\approx 90\%$ of theoretical max ~ 1.55) confirms Pareto-improving bid selection works. The Nash distance of ≈ 0.11 confirms a near-Nash but not exact Nash outcome, expected in symmetric self-play. The social welfare gap of ≈ 0.16 suggests outcomes are near-Pareto but not on the frontier due to information asymmetry between the two agents. Full results vs. BoulwareTBNegotiator, ConcederTBNegotiator, and NiceTitForTatTBNegotiator across all six scenarios will be in the final submission.

8 Planned Experiments

8.1 Experimental Setup

Using `anl2026 tournament`: scenarios Amsterdam, Camera, Car, Grocery, Laptop, NiceOrDie; opponents BoulwareTBNegotiator, ConcederTBNegotiator, NiceTitForTatTBNegotiator, BOANeg, SimpleNegotiator, MAPNeg, and other groups; 50 sessions per pair per scenario; metrics U_A , $U_A \cdot U_O$, $U_A + U_O$, Nash distance, agreement rate, $t_{\text{round}}^{\text{agreement}}$, deception score τ .

Table 4: Ablation study variants.

Variant	Decep.	Adapt. e	Description
Full DecepTor	On	On	All components
No Deception	Off	On	Honest bids, adaptive e
No Adaptation	On	Off	Deception, fixed $e=3$
Conceder Baseline	Off	Off	Fixed $e=0.5$, honest

Table 5: Implementation status.

Component	Status	Notes
GSmithFrequencyModel	Complete	Extended [7]
Aspiration $\alpha(t)$	Complete	[2]
DANS classification	Complete	6 types [4]
Rate $\hat{\delta}(t)$	Complete	Sliding window
Adaptive e	Complete	3-case rule
Acceptance Alg. 1	Complete	4 conditions
Deception sets H_u, D_u	Complete	Pre-computed
Schedule $p_{\text{dec}}(t)$	Complete	$0.15+0.10t$
Self-play harness	Complete	<code>test_agent.py</code>
ANL tournament	Pending	<code>anl2026 tournament</code>
Multi-scenario tests	Pending	6 scenarios
Ablation study	Pending	4 variants
Parameter grid-search	Pending	e, α ranges

8.2 Ablation Study

H1: Full DecepTor scores higher than No Deception due to τ bonus. **H2:** Adaptive e increases U_A against mixed opponents vs. fixed e . **H3:** Full DecepTor maintains competitive U_A (no significant utility loss).

Analysis will address: (1) Are agreements Nash solutions? (2) Can the Pareto frontier be reached? (3) Does deception affect agreement rate? (4) Is the opponent model generic across all six scenarios?

9 Implementation Status

The agent is in `groupN.py` with class `GroupN` inheriting from `BOANegotiator`. Table 5 shows status.

Listing 1 shows the core class structure.

10 Agent Strong and Weak Points

Strong points: (1) The deception layer directly targets Kendall τ , giving an advantage over agents that ignore it. (2) Pre-computed utility-equivalent swaps are harder to detect than noise injection. (3) Adaptive e handles both concesive and stubborn opponents. (4) Full DANS classification provides richer signal than binary switching. (5) Monotonic decrease is analytically proven (Eq. 4).

Weak points: (1) Deceptive bids reduce opponent utility, risking agreement rate. (2) In small domains D_u may be empty. (3) Frequency model converges slowly in short negotiations. (4) DANS requires sufficient bid history; may trigger late. (5) No cross-session learning by design.

11 Literature Motivation and Positioning

DecepTor extends: Faratin et al. [2] with adaptive e ; Hendriks et al. [4] with the full six-category DANS table; Tu-

Listing 1: Core structure of the DecepTor agent.

```

1  class GroupN(BOANegotiator):
2      def on_negotiation_start(self, state):
3          self._precompute_bid_sets() #
4              build H_u, D_u
5          self._opp_model =
6              GSmithFrequencyModel(...)
7          self._behavior_tracker =
8              BehaviorTracker()
9          self._e = 3.0
10
11      def propose(self, state) -> Outcome:
12          alpha = self._compute_aspiration(
13              state.relative_time)
14          return self._deception_layer.
15              select_bid(alpha, state.
16                  relative_time)
17
18      def respond(self, state, offer) ->
19          ResponseType:
20          self._opp_model.update(offer)
21          self._behavior_tracker.update(offer,
22              state)
23          self._adapt_e()
24          if self._should_accept(offer, state):
25              :
26          return ResponseType.ACCEPT_OFFER
27          return ResponseType.REJECT_OFFER

```

nali et al. [7] with inverse-change-frequency weights for the H/D bid partition; Baarslag et al. [1] with a reactive strategy (adaptive e) instead of fixed concession; Williams et al. [8] with a simpler sliding window concession rate. The deception mechanism — applying utility-equivalent bid swaps to corrupt frequency model issue-weight estimation — is our original contribution with no direct prior in the ANAC literature.

12 Challenges Encountered and Next Steps

Challenges: (1) *Deception calibration* — balancing schedule parameter α and the $0.92 \cdot \alpha(t)$ threshold. (2) *Nash approximation cost* — capped at top-200 outcomes, may miss true Nash in large domains. (3) *NegMAS API compatibility* — version-dependent attribute names handled with try/except. (4) *Outcome space scaling* — bid list capped at 2000; constraint-based generation planned. (5) *Empty D_u* — in small domains the agent falls back to honest bidding.

Next steps: (1) Run full ANL tournament against all base-lines across all six scenarios. (2) Report all metrics including $t_{\text{time}}^{\text{agreement}}$ and τ . (3) Run ablation study (Table 4), test H1–H3, tune e and α . (4) Implement constraint-based bid generation for large domains. (5) Finalise groupN.py and submit to ANAC 2026 portal by June 1, 2026.

13 Conclusion

This report has presented the design of DecepTor for ANL 2026. The three principal contributions are: (1) utility-equivalent bid swaps that mislead the opponent’s frequency

model while maintaining high self-utility; (2) adaptive Boulware concession that adjusts e via DANS behaviour classification [4]; and (3) a composite acceptance strategy integrating utility targets, time pressure, Nash proximity, and opponent trend.

Building DecepTor taught us that the dual objective of ANL 2026 demands careful trade-offs. The deception schedule must establish false credibility early and corrupt the model late. DANS revealed that opponent behaviour is more nuanced than a binary Boulware/Conceder classification, and reactive adaptation prevents both deadlock and premature concession.

In real-world settings, DecepTor’s architecture could extend to e-commerce negotiations where strategic misrepresentation of low-priority attributes is a natural tactic, and the DANS classifier could incorporate sentiment signals in human-agent settings. Future work will run the full tournament, validate H1–H3, and submit the agent to ANAC 2026.

References

- [1] Baarslag, T.; Hindriks, K. V.; and Jonker, C. M. 2012. A Tit for Tat Negotiation Strategy for Real-Time Bilateral Negotiations. In *Proceedings of ACAN 2012*.
- [2] Faratin, P.; Sierra, C.; and Jennings, N. R. 1998. Negotiation Decision Functions for Autonomous Agents. *Robotics and Autonomous Systems*, 24(3): 159–182.
- [3] Fatima, S. S.; Wooldridge, M.; and Jennings, N. R. 2001. Optimal Negotiation Strategies for Agents with Incomplete Information. In *Proceedings of ATAL-2001*, 53–68.
- [4] Hindriks, K.; Jonker, C. M.; and Tykhonov, D. 2011. Let’s DANS! An Analytic Framework of Negotiation Dynamics and Strategies. *Web Intelligence and Agent Systems*, 9: 319–335.
- [5] Mohammad, Y.; et al. 2020. NegMAS: A Platform for Situated Negotiation. <https://negmas.readthedocs.io>. Version 2020–2025.
- [6] Russell, S.; and Norvig, P. 2010. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3rd edition.
- [7] Tunali, O.; Aydogan, R.; and Sanchez-Anguix, V. 2017. Rethinking Frequency Opponent Modeling in Automated Negotiation. In *PRIMA 2017*, 263–279.
- [8] Williams, C. R.; Robu, V.; Gerding, E. H.; and Jennings, N. R. 2012. IAMhaggler: A Negotiation Agent for Complex Environments. In *New Trends in Agent-based Complex Automated Negotiations*, 151–158. Springer.