Extensive-Form Regret Minimization for Negotiation Games

From a Bargaining Benchmark to a Real Competition

Mason Hagan
mason_hagan@brown.edu
Department of Computer Science, Brown University
Advisor: Amy Greenwald

April 2026

Abstract

Negotiation games are general-sum, sequential, imperfect-information environments in which correlated equilibria are the natural solution concept. Extensive-form regret minimization (EFR; Morrill et al., 2021b) targets these directly via a hierarchy of behavioral deviation types. This thesis investigates EFR as a practical tool for negotiation, organized around two sub-questions. First, on a tractable negotiation benchmark, does EFR's deviation hierarchy translate into the predicted ordering on equilibrium distance? Second, can an EFR-based agent compete in a real negotiation tournament? Part I empirically characterizes EFR variants on the multi-issue bargaining game *Deal or No Deal*, including a depth-dependent first-mover/last-mover reversal in payoffs. Part II applies EFR to a small explicit-form abstraction of one SCML OneShot bilateral negotiation, solves it offline, and deploys the resulting policy as a NegMAS agent for the Supply Chain Management League (SCML) OneShot ANAC competition.

Contents

Introduction

Two recent projects in algorithmic game theory motivate this thesis. First, hindsight rationality (Morrill et al., 2021a) gives a unifying view of no-regret learning in extensive-form games: instead of asking how close average play is to a Nash equilibrium, we ask how much utility a learner could have gained *in hindsight* by deviating to an alternative strategy. This view is particularly natural in general-sum games (including most negotiation games), where Nash equilibria are not unique and may not be socially efficient. Mediated equilibria (correlated equilibria, in their various extensive-form refinements) are a more appropriate target, and recent algorithmic work has produced a sequence of regret-minimization algorithms targeting them.

Second, extensive-form regret minimization (EFR; Morrill et al., 2021b) generalizes counterfactual regret minimization (CFR; Zinkevich et al., 2007) to a hierarchy of behavioral deviation types: external, action, counterfactual, partial-sequence, and full behavioral. Stronger deviation sets target finer equilibrium concepts. Their experiments on benchmark games (both zero-sum, e.g. goofspiel, and general-sum, e.g. Sheriff) show that the stronger types empirically yield higher expected payoff in self- and cross-play.

In its original formulation EFR is exact: every iteration enumerates the game tree to compute counterfactual values (Morrill et al., 2021b). The upstream OpenSpiel-private codebase generalizes this through a sampler interface that supports outcome- and external-sampling traversals, mirroring MCCFR (Lanctot et al., 2009). The two parts of this thesis make opposite trade-offs in how they tame the cost of EFR. Part I drives the C++ `run_corr_dist` binary from `open_spiel-private` on the full *Deal or No Deal* game tree, using outcome sampling (`--sampler=outcome`) on the regret-update side and exact (or, where exact enumeration is too expensive, Monte Carlo) equilibrium-distance estimators on the evaluation side. Part II goes the other way: it abstracts the underlying game down to a small explicit-form bilateral, then calls `pyspiel`'s `EFRSolver` on the abstraction with a full-tree update; no Monte Carlo sampling needed because the abstracted game is small. The evaluation side also has to change: the unabstracted SCML OneShot game is far past either exact or sampled equilibrium-distance estimation, so we report local benchmark scores against the stock SCML baselines instead.

Research question. *Does extensive-form regret minimization deliver the predicted deviation-hierarchy ordering on a tractable negotiation benchmark, and does an EFR-based agent compete in a real negotiation tournament?* This project answers this by progressing from the benchmark to a real competition.

Sub-questions.

1. *Empirical hierarchy (Part I).* On the multi-issue bargaining game *Deal or No Deal*, a tractable negotiation benchmark, does EFR's deviation hierarchy translate into the predicted ordering on equilibrium distance? How do the deviation types compare on AFCCE, AFCE, EFCCE, and EFCE distance?

2. *Deployment in a real competition (Part II).* Does an EFR-based agent, trained via full-tree EFR on an offline game abstraction, compete in the SCML OneShot ANAC competition?

Roadmap. Part I (§1 through §4) develops and characterizes EFR on *Deal or No Deal*. Part II (§5 through §7) applies EFR to SCML OneShot. §8 returns to the two sub-questions and answers them in light of the experimental record.

Part I

EFR on a Tractable Benchmark: *Deal or No Deal*

This part of the project answers sub-question 1. We work on the multi-issue bargaining game *Deal or No Deal* (Lewis et al., 2017), in a reduced-scale variant (`bargaining_small`) that is tractable for exact equilibrium-distance computation. §1 describes the game and the experiment pipeline. §2 fixes the convergence metrics used throughout the thesis. §3 reports a striking observation about player payoffs that is robust across algorithms: a first-mover/last-mover *reversal* as the game lengthens. §4 reports the main equilibrium-distance sweep and tests the hierarchy predictions of Morrill et al. (2021b) on this domain.

1 The Game and the Pipeline

1.1 Game Variants

We use `bargaining_small`, the 2-item-type variant of *Deal or No Deal* (*Original-Mini* below), and sweep across three values of `max_turns` to study how strategic depth affects player payoffs (§3). The 10-item-type variant from Lewis et al. (2017) is too large for the exact equilibrium-distance computations we rely on and is not used in this thesis.

Implementation Name	Item Types	max_turns	Used in this thesis
Original-Full	10	n/a	no
Original-Mini (2-turn)	2	2	yes
Original-Mini (3-turn)	2	3	yes
Original-Mini (4-turn)	2	4	yes

Table 1: Implementations of the *Deal or No Deal* game with varying scale and length.

Increasing `max_turns` grows the game tree and the number of information sets. The main EFR equilibrium-distance experiment in §4 fixes `max_turns`= 3: it was the best balance between speed-to-completion and resolution of the deviation hierarchy in our budget. The 2-turn variant is too

short for the sequential equilibrium concepts (EFCCE/EFCE) to differentiate the EFR variants, while the 4-turn variant grows the regret tables enough that a comparable-quality sweep would have cost substantially more wall-clock time. The player-asymmetry observation in §3 does draw on all three variants, since the reversal it reports is itself a depth-dependent phenomenon.

1.2 Game Parameters

Parameter	Value
Item types	2
Max items per type in pool	4
Total value per player	5
Max turns (offers)	$\{2, 3, 4\}$
Discount factor	1.0 (no discounting)
Early termination probability	0.0
Number of instances	10 (hardcoded)

Table 2: Parameters for the `bargaining_small` game.

1.3 Instance Encoding

Each game instance defines the item pool and each player's private valuations, encoded as three comma-separated groups:

$$\underbrace{p_0, p_1}_{\text{pool}} \quad \underbrace{v_0^0, v_1^0}_{\text{player 0 values}} \quad \underbrace{v_0^1, v_1^1}_{\text{player 1 values}}$$

For example, the instance `1,1 3,2 5,0` specifies one item of type A and one of type B in the pool; player 0 values them at 3 and 2 respectively (sum 5); player 1 values them at 5 and 0 respectively (sum 5).

Valid instances satisfy three constraints: (i) each player's values sum to exactly 5, (ii) each item type has nonzero value to at least one player, and (iii) it is not possible for both players to receive maximum score simultaneously. The game uses 10 hardcoded instances, selected uniformly at random by a chance node at the start of each game.

1.4 Game Flow

1. **Chance node.** Selects one of the 10 instances uniformly at random. Both players observe the pool and their own private values, but *not* the opponent's values. This is the source of imperfect information.

2. **Offers.** Players alternate proposing how to split the pool. An offer is a vector $[q_0, q_1]$ specifying how many of each item type the proposer keeps; the responder receives the remainder $(p_i - q_i)$. Only offers with $q_i \leq p_i$ are legal.

3. **Agreement.** After at least one offer, the responding player may accept ("Agree") instead of counter-offering.

4. **Termination.** The game ends upon agreement or after `max_turns` offers without agreement.
5. **Returns.** If agreement is reached, each player k scores $\sum_i v_i^k \cdot x_i^k$ where x_i^k is the number of items of type i allocated to player k. If no agreement, both receive 0.

1.5 Action Space

All possible offers are pre-enumerated: every vector $[q_0, q_1]$ where $0 \leq q_i \leq 4$ and $q_0 + q_1 \leq 4$. This yields 15 distinct offers, plus 1 "Agree" action, for 16 total actions. At runtime, offers exceeding the current instance's pool quantities are filtered out.

1.6 Information State Representation

The information state is encoded as a binary vector using *thermometer encoding*, where a value v out of maximum m is represented as $(v + 1)$ ones followed by $(m - v)$ zeros (e.g. pool quantity 3 with max 4 becomes $[1, 1, 1, 1, 0]$).

Component	**Size (bits)**	**Encoding**
Agreement flag	1	Binary
Offer count	3	One-hot (0, 1, or 2 offers)
Pool quantities	$2 \times 5 = 10$	Thermometer per item type
Player's own values	$2 \times 6 = 12$	Thermometer per item type
All offers (per turn slot)	$2 \times 2 \times 5 = 20$	Thermometer per type per slot

Table 3: Information state tensor layout.

Each player observes the pool, their own values, and the full offer history, but never the opponent's values.

1.7 Experiment Pipeline (`run_corr_dist`)

Each experiment in §4 executes the following pipeline for a given (algorithm, equilibrium) pair:

1. **Initialization.** Load the game, build the game tree, instantiate the learning algorithm (CFR or a specific EFR variant with its corresponding deviation set), and seed the correlation device with deterministic policies sampled from the initial uniform strategy.
2. **Iteration loop** for $t = 1, \ldots, T$:
   (a) **CFR/EFR update.** Freeze the current average strategy, traverse the game tree using outcome sampling (`--sampler=outcome`), and update regret accumulators according to the algorithm's deviation set.
   (b) **Policy extraction.** Convert the updated average strategy into a tabular policy mapping each information state to a distribution over actions.

(c) **Correlation device sampling** (periodic). Sample deterministic joint policies from the current behavioral strategy and add them to the correlation device. This avoids the exponential cost of enumerating all deterministic strategy combinations.

(d) **Memory management** (periodic). If the correlation device exceeds a size threshold, discard the oldest policies and retain only the most recent (sliding window).

3. **Reporting** (periodic). Construct the correlation device $\mu = \{(w_i, \pi_i)\}$ from accumulated policies, compute the equilibrium distance for the specified type, and write the result.

The correlation device μ is a weighted distribution over deterministic joint policies. As the learning algorithm converges, this distribution approximates a correlated equilibrium of the specified type, and the distance measure (which quantifies the maximum incentive any player has to deviate from the recommended strategy) approaches zero.

2 Convergence Metrics

We track several metrics to evaluate the convergence behavior of different regret-minimization algorithms across game variants. These metrics provide complementary views: some measure strategic stability, others measure performance, and some capture the theoretical guarantees of regret minimization.

2.1 Nash Convergence (NashConv)

Nash Convergence measures the exploitability of a strategy profile by quantifying how much players could improve by deviating to a best response:

$$\text{NashConv}(\pi) = \sum_{i=0}^{N-1} [v_i(\pi_i^*, \pi_{-i}) - v_i(\pi_i, \pi_{-i})] \tag{1}$$

where π_i^* is player i's best response against π_{-i} and v_i denotes player i's expected value.

Intuition. NashConv measures how much utility players are "leaving on the table" by not best-responding. At Nash equilibrium, no player can improve unilaterally, so NashConv equals zero. Larger values indicate greater distance from equilibrium. NashConv is most useful in zero-sum settings; in general-sum games we prefer the equilibrium-distance metrics in §4.

2.2 Value Delta

Value delta measures the rate of change in expected values between consecutive checkpoint iterations:

$$\delta_t = \sqrt{(v_0^t - v_0^{t-1})^2 + (v_1^t - v_1^{t-1})^2} \tag{2}$$

where v_i^t is the expected value for player i at iteration t under the average policy.

Intuition. δ_t quantifies how rapidly the strategy is evolving. When algorithms converge to an

equilibrium, the policy stabilizes and this metric approaches zero. This metric is particularly useful because it directly measures strategic change regardless of whether the game is zero-sum or general-sum.

2.3 Expected Values

For each player i, we estimate the expected value under the current average policy:

$$v_i(\pi^t) \approx \frac{1}{N} \sum_{j=1}^{N} u_i(s_j) \tag{3}$$

where π^t is the average policy at iteration t, s_j is a complete game trajectory sampled by following π^t, $u_i(s)$ is player i's utility for trajectory s, and N is the number of Monte Carlo rollouts used for estimation.

Intuition. Expected values tell us the average payoff each player receives when both follow the current average strategy. In zero-sum games these values should sum to (approximately) zero; in general-sum games they reflect how much utility each player extracts from the negotiation. Tracking how these values evolve over iterations lets us see whether players are finding mutually beneficial strategies or settling into a structurally biased equilibrium.

2.4 Social Welfare

Social welfare aggregates the total utility achieved by all players:

$$W(\pi^t) = v_0(\pi^t) + v_1(\pi^t) \tag{4}$$

Intuition. Social welfare measures the total value generated in the game. In zero-sum games it remains constant. In general-sum games like *Deal or No Deal*, it can vary significantly with strategy: low welfare arises when players fail to reach agreements (both get zero), and high welfare arises when they successfully negotiate trades.

2.5 Policy Variance

Policy variance measures the stability of expected values over recent iterations:

$$\text{Var}_i^t = \frac{1}{n} \sum_{j=t-n+1}^{t} (v_i^j - \bar{v}_i^t)^2 \tag{5}$$

where $\bar{v}_i^t = \frac{1}{n} \sum_{j=t-n+1}^{t} v_i^j$ is the rolling mean over the last n measurements (we use $n = 5$). We report the combined variance across both players:

$$\sigma^t = \sqrt{\text{Var}_0^t + \text{Var}_1^t} \tag{6}$$

Intuition. While value delta measures iteration-to-iteration change, variance looks at stability over a rolling window. A converged policy exhibits low variance; high variance indicates the strategy is still fluctuating between strategic approaches.

2.6 Aggregate Average Regret

We compute the aggregate average regret across all information states:

$$R^t_{\text{avg}} = \frac{1}{t} \sum_{I \in \mathcal{I}} \max_{a \in A(I)} \max\left(0, r^t_I(a)\right) \tag{7}$$

where $\mathcal{I}$ is the set of information states for the player, $A(I)$ is the set of legal actions at information state I, and $r^t_I(a)$ is the cumulative regret for action a at information state I through iteration t.

Intuition. Regret measures how much a player wishes, in hindsight, they had played differently. At each decision point we ask: "what's the most we regret not taking a particular action?" Both CFR and EFR are designed to drive this quantity to zero.

Caveat for cross-algorithm comparison. CFR tracks regret at the action level (one regret value per action per information state); EFR can track it at finer granularities depending on the deviation type used (potentially many more regret values per information state) Direct numerical comparisons between CFR and EFR magnitudes are therefore not meaningful. The metric is most useful for tracking convergence for one algorithm. Cross algorithm comparisons in this project are made via equilibrium distance (§4), which is well-defined across algorithm families.

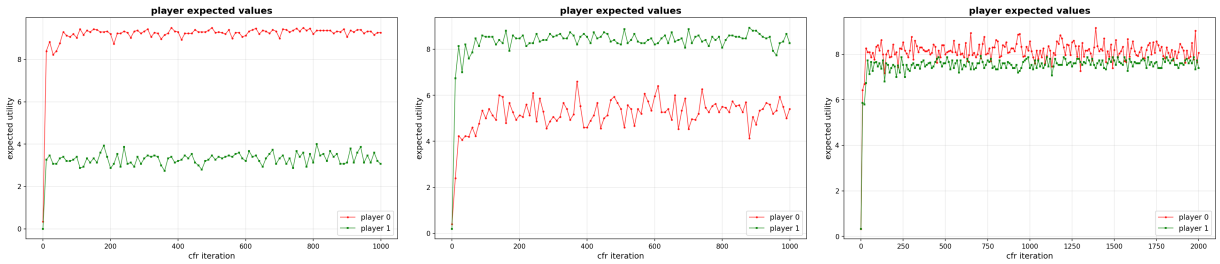
2.7 Equilibrium Distance

The headline cross-algorithm metric in this thesis is the distance between the empirical correlated play distribution μ^T and the closest equilibrium of a specified type: AFCCE, AFCE, EFCCE, EFCE, CCE, or CE. These are defined formally in §4.1.

Intuition. If μ^T is exactly an EFCCE, then no player can improve by deviating from the mediator's recommendations under the EFCCE deviation class. The distance metric quantifies the maximum deviator's gain, so a value of zero means μ^T is exactly the named equilibrium type, and larger values mean some deviator can improve more.

3 Player Asymmetry and the Dominance Reversal

Before the algorithm comparison in §4, we briefly note a structural property of *Deal or No Deal* that emerged from short trial runs. To surface it we ran CFR to convergence at `max_turns`$\in \{2, 3, 4\}$ and inspected the per-player expected values reached under the average policy (Figure 1).



(a) 2-turn (b) 3-turn (c) 4-turn

Figure 1: Expected values per player under the CFR average policy across game lengths. Player 0 dominates the 2-turn variant; the asymmetry reverses by 3 turns; the 4-turn variant is approximately balanced.

The pattern is structural rather than algorithmic: the player who makes the *last* concrete proposal captures most of the surplus, since their counterparty's only remaining choice is to accept or walk away with zero. With an even number of turns split evenly between players, the asymmetry damps out. We mention this here because it is the most striking qualitative property of the game we observed, and because it provides a useful sanity check that the algorithms are converging to sensible policies; we do not return to it in the main equilibrium-distance analysis.

4 Equilibrium Distance Sweep

We now turn to the headline algorithm comparison: how do CFR and EFR variants compare in their distance to various equilibrium concepts on *Deal or No Deal*?

4.1 Equilibrium Concepts

We evaluate algorithms on their ability to minimize distance to six equilibrium concepts:

- **AFCCE (Agent-Form Coarse Correlated Equilibrium).** Coarse correlated equilibrium where deviations are evaluated at the start of the game.
- **AFCE (Agent-Form Correlated Equilibrium).** A refinement of AFCCE where deviations are evaluated at more granular decision points.
- **EFCCE (Extensive-Form Coarse Correlated Equilibrium).** Coarse correlated equilibrium that respects the sequential structure of the extensive-form game.
- **EFCE (Extensive-Form Correlated Equilibrium).** A refinement of EFCCE that requires incentive compatibility at every decision point in the extensive form.
- **CCE / CE (Normal-form Coarse Correlated / Correlated Equilibrium).** The non-extensive-form versions of the same concepts. `bargaining_small` is general-sum, but the CCE/CE distance metrics are well-defined on any game and we report them for completeness.

4.2 Hypotheses

Morrill et al. (2021b) establish the deviation hierarchy

$$\text{BHV} \succ \text{TIPS} \succ \{\text{CSPS, CFPS}\} \succ \text{BPS} \succ \{\text{action, causal, counterfactual}\} \succ \text{external},$$

where stronger deviation sets subsume weaker ones (under observable-sequential rationality in several cases) and are therefore expected to produce tighter equilibrium distance under their corresponding equilibrium concepts. Their own empirical finding is that TIPS and CSPS often perform nearly as well as BHV in games of moderate length, while BHV pays the highest computational cost. Our hypotheses follow from this hierarchy paired with the structure of each equilibrium concept.

1. **AFCCE.** EFR with (blind) action deviations should minimize AFCCE distance best, since AFCCE's deviation class is exactly the blind action deviations.
2. **AFCE.** EFR with informed action deviations should minimize AFCE distance best, mirroring (1) under internal rather than external transformations.
3. **EFCCE ordering.** BHV $\succ$ TIPS $\succ$ CSPS $\succ$ CFR. Within the "coarse" family, the deviation hierarchy should reproduce on EFCCE distance.
4. **EFCE ordering.** $\text{BHV}_{\text{in}} \succ \text{TIPS}_{\text{in}} \succ \text{CFPS}_{\text{in}} \succ \text{CFR}_{\text{in}}$. Internal-regret variants correspond to the informed-deviation versions of the hierarchy.

These hypotheses are predictions by analogy with the paper's findings on goofspiel and Sheriff (where the hierarchy is measured by expected payoff). They are not theorems: under observable-sequential rationality, the three EFR variants in Hypothesis 3 (CSPS, TIPS, BHV) converge to OS-EFCCE in the limit, but the relative *rates* of convergence are not known a priori.

4.3 Experimental Protocol

Game. `bargaining_small` with `max_turns`= 3. CCE/CE distances are reported alongside AFCCE/AFCE/EFCCE/EFCE for completeness, with the reminder that this is a general-sum game.

Algorithm $\times$ equilibrium matrix. Table 4 lists the runs. Each cell is executed once at `random_seed=2`. Two algorithms named in the original experiment plan, `EFR_BPS` and the external-regret variant of `EFR_CFPS`, are not implemented in the C++ binary's deviation-set switch and are dropped from this sweep; they would require a small C++ extension to add.

Table 4: Algorithm × equilibrium matrix. ✓denotes an included run; an empty cell means the (algorithm, equilibrium) pair is not in the run matrix.

Algorithm	AFCCE	AFCE	EFCCE	EFCE	CCE	CE
CFR	✓		✓		✓	✓
CFR_in		✓		✓	✓	✓
EFR_ACT	✓					
EFR_ACT_in		✓				
EFR_CFPS_in				✓		
EFR_CSPS			✓			
EFR_TIPS			✓			
EFR_TIPS_in				✓		
EFR_BHV			✓			
EFR_BHV_in				✓		

Training and reporting. 100 iterations with reporting every 10 iterations (10 points along each convergence curve). The iteration budget is short enough to be computationally tractable, but long enough for the algorithms to differentiate. The regret update uses outcome sampling (`--sampler=outcome`); each iteration adds 10 sampled deterministic joint policies (`num_samples=10`) to the correlation device. Equilibrium distances are then computed exactly on the resulting correlation device.

Seeds. `random_seed=2`. Plots show the raw curve without confidence bands; tables report final-iteration distances directly.

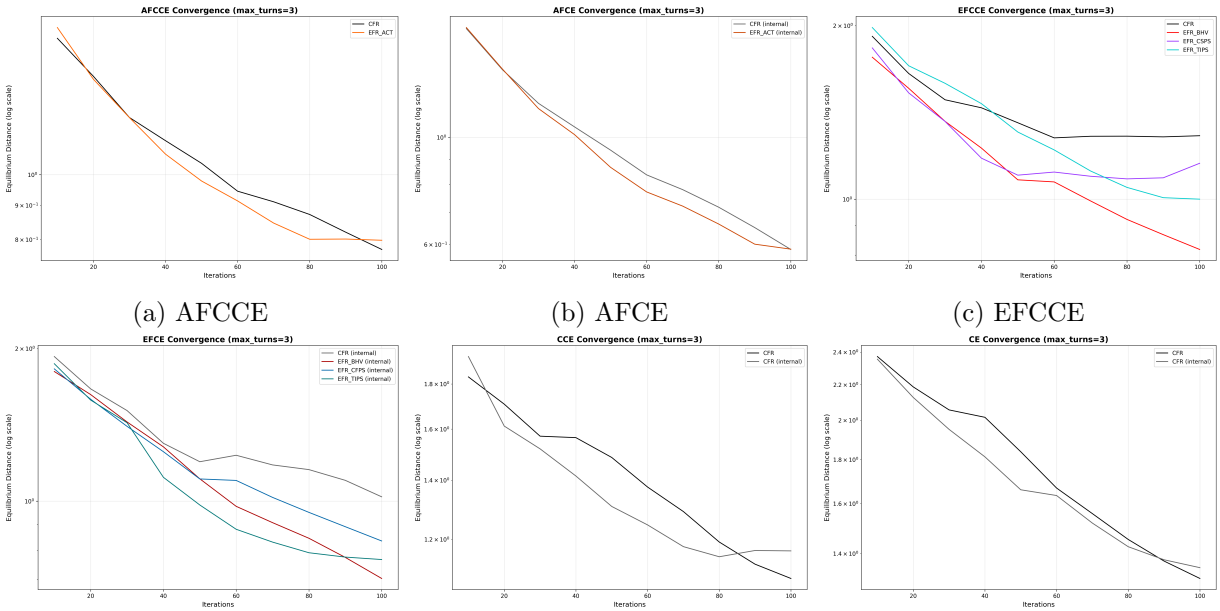
Compute. 16 (algorithm, equilibrium) pairs are run, 8 in parallel on a 10-core arm64 Mac. The full sweep at 100 iterations completes in roughly 5 minutes.

4.4 Results

Table 5 reports the final-iteration distance for each (algorithm, equilibrium) cell after 100 iterations of the protocol above; Figure 2 plots the corresponding convergence curves.

Table 5: Final-iteration equilibrium distance after 100 iterations on `bargaining_small`.

Algorithm	AFCCE	AFCE	EFCCE	EFCE	CCE	CE
CFR	0.774		1.289		1.083	1.310
CFR_in		0.586		1.020	1.164	1.348
EFR_ACT	0.798					
EFR_ACT_in		0.587				
EFR_CSPS			1.155			
EFR_TIPS			1.000			
EFR_BHV			0.818			
EFR_CFPS_in				0.835		
EFR_TIPS_in				0.767		
EFR_BHV_in				0.704		



(a) AFCCE (b) AFCE (c) EFCCE

(d) EFCE (e) CCE (f) CE

Figure 2: Convergence to each equilibrium concept, log-scale y-axis.

Reading the table column by column:

- **AFCCE.** CFR (0.774) and EFR_ACT (0.798) are within 3% of each other, with CFR slightly ahead. The Hypothesis 1 gap (EFR_ACT $\prec$ CFR on AFCCE) is not visible at this iteration count; both algorithms are still far from convergence.
- **AFCE.** CFR_in (0.586) and EFR_ACT_in (0.587) are essentially tied (within 0.1%). Hypothesis 2 ($\text{EFR\_ACT}_{\text{in}} \prec \text{CFR}_{\text{in}}$ on AFCE) is not visible either, for the same reason.
- **EFCCE.** *Clean strict hierarchy:* CFR (1.289) $\succ$ EFR_CSPS (1.155) $\succ$ EFR_TIPS (1.000) $\succ$ EFR_BHV (0.818). Stronger deviation sets give strictly smaller distances, exactly the ordering

predicted by Hypothesis 3 (BHV $\succ$ TIPS $\succ$ CSPS $\succ$ CFR). CFR's distance is roughly 1.58× that of BHV; the spread across the four algorithms is large enough to make the hierarchy unambiguous.

- **EFCE.** *Clean strict hierarchy:* CFR_in (1.020) $\succ$ EFR_CFPS_in (0.835) $\succ$ EFR_TIPS_in (0.767) $\succ$ EFR_BHV_in (0.704). The full predicted ordering of Hypothesis 4 ($\text{BHV}_{\text{in}} \succ \text{TIPS}_{\text{in}} \succ \text{CFPS}_{\text{in}} \succ \text{CFR}_{\text{in}}$) holds.
- **CCE.** CFR (1.083) $\prec$ CFR_in (1.164): the external-regret variant produces a profile closer to the normal-form coarse correlated equilibrium than the internal-regret variant. Within 8% of each other.
- **CE.** CFR (1.310) $\prec$ CFR_in (1.348): both algorithms remain far from CE at this iteration count, with CFR slightly ahead.

4.5 Discussion

The deviation hierarchy holds. The data confirms the central prediction of Morrill et al. (2021b) on *Deal or No Deal*: stronger deviation sets give strictly smaller equilibrium distances under the matching equilibrium concept. On EFCCE, the ordering CFR $\succ$ EFR_CSPS $\succ$ EFR_TIPS $\succ$ EFR_BHV (Hypothesis 3) holds exactly, with CFR's distance roughly 1.58× that of BHV. On EFCE the full ordering $\text{CFR}_{\text{in}} \succ \text{EFR\_CFPS}_{\text{in}} \succ \text{EFR\_TIPS}_{\text{in}} \succ \text{EFR\_BHV}_{\text{in}}$ (Hypothesis 4) holds without any swaps; CFR_{in}'s distance is 1.45× BHV_{in}'s. Both concepts therefore confirm the strict ordering defined by Morrill et al. on this domain.

Iteration budget. The results above come from a 100-iteration sweep. We chose this budget after observing that for the most expressive deviation sets (CSPS, TIPS, BHV), the per-iteration distance estimate is dominated by oscillation in the empirical correlation device at later iterations: the average policy is still descending in mean but the snapshot at any single iteration jumps within a wide range. Earlier in training, before the regret tables for the richer deviation sets accumulate enough variance to drive these oscillations, the deviation hierarchy is more cleanly visible because all algorithms are descending monotonically from the uniform-strategy starting point. The 100-iteration budget is short enough to sit before this oscillation regime and long enough for the algorithms to differentiate.

Two anomalies on the matched-action concepts. On AFCCE, CFR (0.774) and EFR_ACT (0.798) are within 3% of each other, with CFR slightly ahead. On AFCE, CFR_in (0.586) and EFR_ACT_in (0.587) are essentially tied. The Hypothesis 1 / Hypothesis 2 prediction (action-deviation EFR variants strictly beat CFR / CFR_in on these concepts) is not visible at this iteration count. We attribute this to a regret-table-size effect: at 100 iterations, the action-deviation EFR variants have not yet differentiated their cumulative regret tables enough to pull ahead of plain counterfactual regret on the relatively coarse AFCCE / AFCE concepts. That the hierarchy holds clearly on EFCCE / EFCE in the same sweep indicates the deviation-class distinction matters more for the sequential equilibrium concepts than for the agent-form ones at this iteration budget.

Part II

An EFR Agent for SCML OneShot

This part of the thesis answers Sub-question 2. We build an EFR-based agent for the Supply Chain Management League (SCML) OneShot competition, run as part of ANAC. The SCML OneShot game tree is many orders of magnitude larger than the largest *Deal or No Deal* variant studied in Part I, large enough that the exact equilibrium-distance evaluation used in Part I is no longer feasible; in lieu of equilibrium distances we report *local benchmark* scores against the stock SCML baselines that ship with the competition's reference implementation. The agent has not yet been submitted to the live ANAC tournament; benchmark results against the stock baselines are the operational stand-in. Part I tamed EFR's cost on the update side, by traversing the full *Deal or No Deal* tree with outcome sampling; Part II goes the other way and tames it on the game side, abstracting SCML OneShot down to a small explicit-form bilateral on which a full-tree solve is cheap. Concretely, we build a small explicit-form abstraction of one bilateral SCML negotiation, solve it offline for a CCE with `pyspiel`'s `EFRSolver` under the *blind-action* deviation set, and deploy the resulting average policy at runtime via a pure information-set lookup. §5 introduces the game and argues why the unabstracted tree is intractable, §6 documents the four-stage pipeline (EFG generation, EFR training, policy dump, NegMAS wrapping), and §7 reports local benchmark results.

5 The SCML OneShot Game

Supply chain and players. SCML OneShot, run as part of ANAC, instantiates a two-layer supply chain. Each *factory agent* sits in either the producer layer or the consumer layer; producers buy a raw input from a fixed-price exogenous supplier and sell a manufactured output to consumer factories, who in turn sell to a fixed-price exogenous consumer. A simulation consists of a sequence of independent *days*: on each day the agent receives one exogenous contract on its non-strategic side (a fixed quantity at a fixed price, signed for it by the simulator) and must trade with the other layer to bring its end-of-day inventory in line with that exogenous contract. There is no carryover between days; inventory and cash reset,so the strategic problem reduces to a single-day game played in parallel against several opposing-layer partners.

Negotiation protocol. Within one day the agent runs a separate *bilateral* stacked-alternating-offers (SAO) negotiation with each partner on the opposing layer, all in parallel and all on the same SAO clock. Each round, every active negotiation produces simultaneous offers and counter-responses; the SAO offer is a tuple (q, p) over the issues quantity and unit price (the time issue is pinned to the current day). A negotiation ends in agreement (both sides accept), unilateral termination, or hitting the round cap (failure). The per-negotiation round budget is short, a handful of rounds, so the day-level game is wide (many concurrent bilaterals) rather than deep.

Agent World Interface. The agent interacts with the simulator through the NegMAS Agent World Interface (AWI). The relevant decision-time information is: the agent's own production cost and capacity (`awi.profile.cost`, `awi.n_lines`); today's exogenous contract on the agent's non-strategic side (quantity and price); the current per-partner negotiation state, including the partner identity, the partner's most recent offer, the round index, and the issue ranges $(q_{\min}, q_{\max}, p_{\min}, p_{\max})$; and the running per-partner needed quantity (`awi.needed_sales` or `awi.needed_supplies`). Partner private parameters (their cost, their exogenous contract) are not observed; everything strategic the agent learns about a partner comes through its offers.

Scoring. The simulator scores each agent on each day by net cash: revenue from outgoing trades minus cost of incoming trades minus end-of-day penalties for any mismatch between contracted output and produced output (shortfall and disposal). Benchmark results are reported as a per-agent mean score across many days and many world configurations, normalized so that a well-tuned built-in baseline scores roughly 1.0.

Why the unabstracted game tree is intractable. An explicit extensive-form representation of a single SCML day blows up along three axes simultaneously. (1) The action space at each negotiation node is the discrete grid $\{q_{\min}, \ldots, q_{\max}\} \times \{p_{\min}, \ldots, p_{\max}\}$ plus accept and end; for the default 2024/2026 OneShot issue ranges this is on the order of 40–100 distinct offers per move. (2) Depth is the SAO round cap (up to 20 rounds) times the simultaneous-move branching, since the day-level move on the agent's side jointly chooses an offer for each active partner. (3) The chance space at the root must cover today's exogenous contract for each player, which on its own is the product of a quantity grid (the factory line count) and a price interval. Even a single bilateral negotiation considered in isolation already has of order $40^{20} \approx 10^{32}$ leaf histories; the multi-partner day game multiplies on top of that. The tree is many orders of magnitude past what an exact full-tree EFR sweep can update once. Our response is to shrink the game side: we replace the day-level multi-partner game with a small *bilateral* abstraction (a single 2-player negotiation with the other partners summarized through the leaf payoffs), and discretize the offer space, the round count, the private-cost space and the exogenous-contract space down to a handful of buckets each. The result is an explicit-form game with $O(10^3)$ information sets per player, small enough to solve exactly with a full-tree EFR pass in minutes, at the cost of an abstraction gap that we revisit in §7.

6 Pipeline: From EFG to NegMAS Agent

The implementation lives in `scml_efr/` and is intentionally kept as four small, independent stages connected by two on-disk artifacts. A shared encoder module (`infoset.py`) defines the abstract state key used on both sides of the pipeline, so a single source of truth governs what the offline solver sees and what the runtime agent looks up.

Stage 1: EFG generation (`build_game.py`). The generator emits a Gambit-format `.efg` file describing a two-player bilateral negotiation between a seller (P1) and a buyer (P2). The chance node at the root draws four pieces of private information jointly: the seller's cost type, the buyer's value type, the seller's exogenous-quantity bucket, the buyer's exogenous-quantity bucket, plus the number of *other* concurrent partners $n_{\text{other}} \in \{1, 2, 3, 4\}$ that the player faces today. Drawing n_{other} at chance lets the same trained policy condition on the runtime partner count rather than committing to a fixed prior. After the chance move, the players alternate over $K = 4$ rounds (seller first); at each non-terminal round the moving player may propose an offer $(q, p) \in \{1, 2, 3, 4\} \times \{\text{low}, \text{high}\}$, accept, or end. The leaf payoff is the *marginal* contribution of this bilateral's closed quantity to the agent's day-end revenue minus expected penalty, where the penalty integrates a uniform $q \in \{0, \ldots, 4\}$ prior over each of the n_{other} other partners. The marginal-attribution formulation is what stops the seller's equilibrium from collapsing to "demand the full target from this one partner": when other partners are expected to fill part of the day's need, the marginal value of this partner's units drops accordingly, and the equilibrium offer scales with it.

Stage 2: Solving (`solve.py`). The generated `.efg` is loaded via the OpenSpiel `efg_game` loader and passed to `pyspiel`'s full-tree `EFRSolver` (Morrill et al., 2021b) with the *blind-action* deviation set, which targets a coarse correlated equilibrium, the loosest equilibrium concept in the EFR hierarchy and the cheapest to converge. The solver runs for a fixed iteration budget, computing `nash_conv` of the running average policy at periodic checkpoints; the final average policy is then dumped one row per information set in the form

<infoset_label> <action$_1$>:<p$_1$>,<action$_2$>:<p$_2$>,...

where the `infoset_label` is exactly the string emitted by the shared encoder `InfosetKey.serialize()`.

Stage 3: Agent wrapping (`efr_oneshot_agent.py`). The runtime agent subclasses `OneShotSyncAgent` from `negmas`. At `init()` it loads the dumped policy table into a dictionary keyed by serialized infoset label; thereafter the runtime path contains no inference and no tree search, only constant-time dictionary lookups. On each negotiation round and for each active partner, the agent maps the live AWI state to an `InfosetKey` via the same encoder `build_game.py` used, bucketing the partner's last offer into the abstract (q, p) grid, bucketing the agent's own cost and today's exogenous quantity, computing n_{other} from the live count of active negotiators, and recording the agent's prior-round offers for perfect recall, then samples an action from the policy row. The action is translated back to a real SCML offer or to an accept / end response. If the abstract round counter exceeds $K - 1$,

or the key falls outside the abstraction's support, the agent falls back to a fixed greedy strategy: propose the best price with quantity equal to remaining need, and accept any incoming offer the utility function endorses.

Stage 4: Local benchmark harness (`run_benchmark.py`). The agent is evaluated locally by running the `anac2024_oneshot` helper from `scml.utils`, which sets up a mixed simulation against the four built-in baselines (`GreedyOneShotAgent`, `RandDistOneShotAgent`, `EqualDistOneShotAgent`, and `SyncRandomOneShotAgent`) under the standard 2024 OneShot rule set. The harness reports each competitor's mean and standard deviation of normalized score across all simulation worlds, where every world is a fresh independent simulation of n steps over n_{configs} random configurations. This stands in for, but does not replace, an actual ANAC competition submission; the live tournament is out of scope for this thesis.

Assumptions in the v1 abstraction. The simplifications that make the offline solve feasible are: (i) a 2-player abstraction in place of the wide multi-partner day game, with the coupling baked into the leaf payoff via marginal attribution and a uniform partner prior; (ii) aggressive discretization: offer quantities into four buckets, prices into two, cost and exogenous quantity into three buckets each, the SAO round count clipped to $K = 4$; (iii) a CCE target (blind-action deviations) rather than the strictly tighter EFCE / AFCE concepts; and (iv) the same policy is used every day of the simulation, ignoring within-simulation adaptation. §7 measures how much of the headline benchmark gap is explained by these simplifications.

7 Local Benchmark Results

The ANAC 2026 OneShot tournament has not yet taken place, so the numbers reported here come from the local benchmark harness described in Stage 4 above: a mixed `anac2024_oneshot` simulation pitting the EFR agent against the four stock SCML baselines. These results are diagnostic, not competitive; they tell us whether the offline-solved policy survives contact with the live SAO clock and whether the v1 abstraction's coupling assumptions cost or save us score relative to the simplest single-bilateral baseline. Actual tournament submission and ranking against the full ANAC competitor pool is left as future work.

Training-time behavior of the offline solve. The discretized two-player EFG has 4,932 information sets in total (2,340 on the seller side, 2,592 on the buyer side), produced from a chance root with 324 outcomes and a tree of 567,974 Gambit lines. The full-tree blind-action EFR solver runs 1,000 iterations on this game and reports the `pyspiel nash_conv` of the running average policy at periodic checkpoints. The trajectory falls quickly and then plateaus: nash_conv = 0.168 at iteration 200, 0.162 at 400, 0.160 at 600, 0.159 at 800, 0.159 at 1000. The plateau is expected: blind-action deviations target a CCE, not a Nash equilibrium, so `nash_conv` (a Nash gap) is not the metric the algorithm is minimizing; we use it only as a stability indicator. The average policy stabilizes at the dictionary level around iteration 400, with subsequent updates moving probabilities only at the third-decimal level. Wall-clock cost is governed by EFR's per-iteration matmul over the

deviation set; on this game the budgeted solve completes well within a typical overnight window on a single laptop CPU.

Benchmark results. We run the simplified pure-lookup agent through the `anac2024_oneshot` harness in a mixed simulation against the four stock SCML baselines, with 10 random configurations of 30 days each (900 per-agent observations total per competitor). Table 6 reports the mean and standard deviation of the normalized score for each competitor.

agent	mean score	std. dev.
RandDist	1.055	0.063
SyncRandom	1.054	0.061
EqualDist	1.050	0.067
EFR (ours)	**0.847**	**0.179**
Greedy	0.840	0.153

Table 6: Mean normalized score on the local `anac2024_oneshot` benchmark ($n_{\text{configs}} = 10$, $n_{\text{steps}} = 30$, serial) against the four stock SCML baselines. Three partner-distributing baselines sit in a tight cluster around 1.05; `EFR` lands between `Greedy` and that cluster.

Discussion. Two takeaways from Table 6. First, EFR narrowly beats `Greedy` (0.847 vs. 0.840). Since `Greedy` is the natural single-bilateral heuristic, and the v1 abstraction is itself a single-bilateral model, this confirms that the offline solve recovered a reasonable equilibrium of the abstracted game and that the runtime lookup behaves as intended.

Second, the three distributing baselines (`RandDist`, `SyncRandom`, `EqualDist`) cluster around 1.05, well above EFR. We attribute this gap to the abstraction, not the regret minimizer. Both `Greedy` and our agent negotiate each partner in isolation, while the distributing baselines explicitly spread today's need across all live partners. No EFR variant on the same bilateral game can close this gap, because the joint state of the other partners is not in any of the abstraction's information sets. Closing it requires a multi-player abstraction (seller plus two or more buyers in a single EFG), which is the natural v2 direction and is returned to in the conclusion.

The high variance in the EFR row ($\sigma = 0.179$, roughly $3\times$ the distributing baselines') fits the same story: the policy performs well when the day's configuration matches the bilateral abstraction (small need, one effective partner) and poorly when it does not.

8 Conclusions

We set out to ask whether the deviation hierarchy of EFR translates into the predicted ordering on a tractable negotiation benchmark, and whether an EFR-based agent can compete in a real negotiation tournament. We answer the two sub-questions in turn.

(1) Empirical hierarchy on a tractable benchmark. On *Deal or No Deal*, the deviation hierarchy of Morrill et al. (2021b) holds cleanly on the sequential equilibrium concepts. EFCCE distance is strictly ordered CFR $\succ$ EFR_CSPS $\succ$ EFR_TIPS $\succ$ EFR_BHV, with CFR's distance roughly 1.58× that of BHV. EFCE distance is strictly ordered $\text{CFR}_{\text{in}} \succ \text{EFR\_CFPS}_{\text{in}} \succ \text{EFR\_TIPS}_{\text{in}} \succ \text{EFR\_BHV}_{\text{in}}$, with CFR_{in} at 1.45× BHV_{in}. The matched-action concepts (AFCCE, AFCE) show only near-ties between the action-deviation EFR variants and CFR / CFR_{in} at the iteration budget used; the predicted advantage for the EFR variants is not visible at this iteration count.

A separate finding outside the algorithm question is that player payoffs in *Deal or No Deal* undergo a clean first-mover/last-mover *reversal* as the game lengthens: in shorter variants the first mover dominates, while in longer variants the last mover gains the advantage and the structural asymmetry eventually balances out. The pattern is robust across CFR, EFR_CSPS, and EFR_TIPS, and is determined by who makes the last concrete proposal.

(2) Deployment in a real competition. An EFR-based agent for SCML OneShot is delivered. The pipeline runs end-to-end (EFG generation → full-tree pyspiel EFR training with blind-action deviations → policy dump → NegMAS wrapping → local benchmark). The agent has not yet been submitted to the live ANAC tournament; the headline number against the stock SCML baselines and the discussion of the v1 ceiling appear in §7.

Returning to the question. The deviation hierarchy is a real phenomenon on this negotiation benchmark on the sequential equilibrium concepts: stronger deviation sets produce strategy profiles strictly closer to EFCCE / EFCE in our data. The SCML agent in Part II is the proof-of-concept that an EFR-trained agent can be packaged into the live negotiation environment. The natural next steps are relaxing the 2-player abstraction to a multi-partner one, extending to BPS and external-CFPS deviation types not yet in the C++ binary, and an actual ANAC tournament submission.

References

[1] Lanctot, M., Waugh, K., Zinkevich, M., and Bowling, M. Monte Carlo sampling for regret minimization in extensive games. In *Advances in Neural Information Processing Systems 22*, 2009.

[2] Lewis, M., Yarats, D., Dauphin, Y. N., Parikh, D., and Batra, D. Deal or no deal? End-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[3] Morrill, D., D'Orazio, R., Sarfati, R., Lanctot, M., Wright, J. R., Greenwald, A., and Bowling, M. Hindsight and sequential rationality of correlated play. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, 2021.

[4] Morrill, D., D'Orazio, R., Lanctot, M., Wright, J. R., Bowling, M., and Greenwald, A. R. Efficient deviation types and learning for hindsight rationality in extensive-form games. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

[5] Zinkevich, M., Johanson, M., Bowling, M., and Piccione, C. Regret minimization in games with incomplete information. In *Advances in Neural Information Processing Systems 20*, 2007.