FACULDADE DE CIÊNCIAS DA UNIVERSIDADE DO PORTO
FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

# Inference-Time Techniques for Open-Weight Language Models in Negotiation Games

**Adriano Alexandre dos Santos Machado**



**Master in Artificial Intelligence**

Supervisor: Prof. Gil Filipe da Rocha

Second Supervisor: Prof. Henrique Daniel de Avelar Lopes Cardoso

July 21, 2026

# Inference-Time Techniques for Open-Weight Language Models in Negotiation Games

**Adriano Alexandre dos Santos Machado**

Master in Artificial Intelligence

Approved in oral examination by the committee:

President: Prof. Carlos Manuel Milheiro de Oliveira Pinto Soares
Examiner: Prof. Ricardo Daniel Santos Faro Marques Ribeiro
Supervisor: Prof. Gil Filipe da Rocha

July 21, 2026

# Resumo

Com o uso crescente de modelos de linguagem de grande escala (LLMs) em contextos de tomada de decisão, torna-se essencial compreender os seus modos de falha e avaliar a sua performance em cenários complexos. A negociação é um contexto particularmente adequado para avaliar estes modelos, uma vez que se trata de um cenário exigente que requer interpretar as intenções de terceiros, planear a longo prazo e agir sob incerteza. A *NegotiationArena* é uma framework de código aberto que disponibiliza um *benchmark* desafiante para avaliar as capacidades de negociação dos LLMs em três cenários de referência: negociação de preços, troca de recursos e o Jogo do Ultimato. A avaliação da *NegotiationArena* limitou-se a analisar modelos proprietários, alguns dos quais, entretanto, foram descontinuados. Nesta dissertação, avaliamos nove modelos de pesos abertos de três famílias (Qwen, Gemma e Mistral) e analisamos o impacto de duas técnicas aplicadas em tempo de inferência: o autoaperfeiçoamento, em que o agente revê e reformula a sua jogada antes de a submeter, e a negociação em equipa, em que três modelos deliberam para tomar uma decisão conjunta.

Os modelos de pesos abertos são capazes de negociar nos três cenários e reproduzem padrões anteriormente descritos para os modelos proprietários, como a ancoragem de preços e as vantagens posicionais. Identificamos alguma irracionalidade, incluindo rejeições que destroem valor para ambas as partes e propostas abaixo do custo de produção.

Embora haja uma grande variabilidade de resultados, o que era de esperar tendo em conta o número de variáveis, foi possível observar certos padrões. O Qwen é a família de modelos mais consistente, vencendo entre 63% e 67% dos jogos, seguido do Gemma e do Mistral. A introdução de duas *personas*, a desesperada (*desperate*) e a manipuladora (*cunning*), que aumentaram o *payoff* e a taxa de vitória do GPT-4 no artigo da *NegotiationArena*, não produziu o mesmo efeito nos modelos de pesos abertos; o seu impacto depende do jogo e do modelo. As duas técnicas aplicadas em tempo de inferência introduzidas nesta dissertação não produziram ganhos suficientes que justifiquem o seu custo. O autoaperfeiçoamento tem um efeito assimétrico, ajudando nas posições estruturalmente desfavorecidas e penalizando as que já são fortes. A negociação em equipa deixa a taxa de vitória praticamente inalterada e produz apenas ganhos de *payoff* localizados. A nossa análise revela limitações dos LLMs de pesos abertos recentes em cenários de negociação desafiantes e aponta direções para melhorias em trabalhos futuros.

**Palavras-chave**: Modelos de Linguagem de Grande Escala • Negociação • Técnicas em Tempo de Inferência • Colaboração Multiagente

# Abstract

With the increasing use of large language models (LLMs) in decision-making contexts, it becomes essential to understand their failure modes and evaluate their performance in complex scenarios. Negotiation is a particularly suitable context for evaluating these models, as it is a demanding scenario that requires interpreting the intentions of third parties, long-term planning, and acting under uncertainty. *NegotiationArena* is an open-source framework that provides a challenging benchmark to assess the negotiation capabilities of LLMs in three reference scenarios: price negotiation, resource exchange, and the Ultimatum Game. The *NegotiationArena* evaluation was limited to analyzing proprietary models, some of which have since been discontinued. In this dissertation, we evaluate nine open-weight models from three families (Qwen, Gemma, and Mistral) and analyse the impact of two inference-time techniques: *Self-Refine*, where the agent reviews and reformulates its move before submitting it, and team negotiation, where three models deliberate to make a joint decision.

Open-weight models can negotiate in all three scenarios and reproduce some patterns previously described for proprietary models, such as price anchoring and positional advantages. We identify some irrationality, including rejections that destroy value for both parties and bids below the cost of production.

Although there is considerable variability in results, as expected given the number of variables, certain patterns emerged. Qwen is the most consistent family of models, winning between 63% and 67% of the games, followed by Gemma and Mistral. The *desperate* and *cunning* personas, which increased GPT-4's *payoff* and *win rate* in the *NegotiationArena* article, do not have the same effect on open-weight models; their impact depends on the game and the model. The two inference-time techniques introduced in this dissertation did not produce sufficient gains to justify their cost. *Self-Refine* has an asymmetrical effect, helping structurally disadvantaged positions and penalising those that are already strong. Team negotiation leaves the win rate virtually unchanged and produces only localised *payoff* gains. Our analysis reveals limitations of recent open-weight LLMs in challenging negotiation scenarios and points to directions for future work improvements.

**Keywords**: Large Language Models • Negotiation • Inference-Time Techniques • Multi-Agent Collaboration

# Agradecimentos

Em primeiro lugar, gostaria de agradecer aos meus orientadores, o Professor Gil Rocha e o Professor Henrique Lopes Cardoso, pela oportunidade e pelo acompanhamento ao longo deste ano.

Quero agradecer aos meus pais por todos os sacrifícios, pelo apoio constante e por acreditarem sempre em mim. Ao meu irmão Pedro, obrigado por estares sempre ao meu lado. À minha prima Beatriz, obrigado por todos os bons momentos que passámos juntos. Quero também agradecer à minha madrinha e a toda a minha família pelo apoio ao longo deste percurso.

Este é o culminar de cinco anos de crescimento, persistência e dedicação. A todos aqueles que marcaram este caminho, em particular aos amigos desde o primeiro dia, Francisco, João e Evans, obrigado por tudo. Tenho um enorme orgulho no nosso percurso e nas amizades que levo para a vida. Aos amigos de longa data, Bia, Diogo, Inês e Tiago, obrigado por todos os momentos que partilhámos ao longo dos anos. À Joana, obrigado por estares sempre presente e por tornares este percurso mais leve nos momentos mais difíceis.

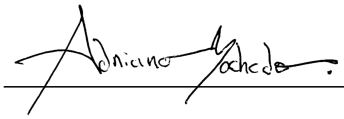
Adriano Machado

# Declaration of honour

I declare that this dissertation is my own work and has not previously been submitted or assessed at this or any other institution. References to other authors (statements, ideas, thoughts), as well as the use of published works, strictly comply with the rules of attribution and are duly identified in the text and in the bibliographic references, in accordance with the applicable referencing standards. I am aware that the practice of plagiarism or self-plagiarism constitutes an academic offence, as established in the U.Porto Code of Ethics.

I also declare that I have used Artificial Intelligence tools as an auxiliary resource in the preparation of this dissertation. Such use and its outputs are duly identified and have been carefully checked for errors, inaccuracies, unfounded attributions or other forms of bias in content and arguments, as well as for the determination of authorship.

Adriano Alexandre dos Santos Machado



*"A vida é o que fazemos dela. As viagens são os viajantes. O que vemos, não é o que vemos, senão o que somos."*

Bernardo Soares

# Contents

# List of Figures

# List of Tables

# List of Acronyms

| | |
|---|---|
| **AI** | Artificial Intelligence |
| **CI** | Confidence Interval |
| **CoT** | Chain-of-Thought |
| **LLM** | Large Language Model |
| **LRM** | Large Reasoning Model |
| **LSTM** | Long Short-Term Memory |
| **MAS** | Multi-Agent System |
| **MoE** | Mixture of Experts |
| **NLM** | Neural Language Model |
| **XML** | Extensible Markup Language |
| **RNN** | Recurrent Neural Network |

# Chapter 1

# Introduction

Negotiation is one of the most common forms of human interaction, present whenever people with different goals must reach a joint decision. Succeeding at it is hard: a negotiator must interpret the other party's intentions, plan several moves ahead, and act on incomplete information.

Large Language Models (LLMs) are increasingly deployed as agents that act on a user's behalf, interact with other agents, and make decisions under uncertainty. Negotiation is a useful setting for evaluating such agents: it requires the abilities above and is measurable in a way most tasks are not.

NegotiationArena [5] provides an open-source framework to do precisely this, evaluating LLM agents across three scenarios (price negotiation, resource trading, and a multi-turn ultimatum game) using win rate and payoff as metrics. It is a strong basis for studying negotiation, but its original results covered only proprietary models, some of which have since been deprecated. This raises concerns about reproducibility and accessibility of the results and leaves open the question of how the smaller, openly available models behave in the same scenarios.

Because negotiation is hard, and retraining large models is costly, a natural follow-up question is whether their outcomes can be improved at inference time, without changing the weights. Two ideas are particularly promising. The first is Self-Refine [24], in which an agent critiques and rewrites its own answer before committing to it, a technique shown to help on tasks such as code generation and open-ended dialogue. The second is multi-agent deliberation, in which several agents collaborate to choose the best action at each step toward a common goal. Recent work reports quality gains from round-table [7] and debate [12], and earlier, pre-LLM research had already explored agent-based negotiation teams and how their members aggregate decisions into a single move [32]. Both techniques, however, increase inference costs, so the central question is not only whether they help, but whether any gains justify the additional cost.

## 1.1 Research Questions

This dissertation is guided by the following research questions:

- *RQ1: How well do open-weight LLMs perform in negotiation scenarios?*

- *RQ2: As inference-time techniques have shown promise in other settings, to what extent do they affect outcomes in multi-agent negotiation contexts?*
  - *RQ2.1: Do the social-persona effects that raised GPT-4's win rate and payoff transfer to open-weight models?*
  - *RQ2.2: Does a draft → feedback → rewrite (Self-Refine) loop improve an agent's negotiation outcomes enough to justify its additional inference cost?*
  - *RQ2.3: Does replacing a single negotiating party with a deliberating team of models improve outcomes over a single agent, and does team diversity (homogeneous versus heterogeneous) matter?*

To answer them, we keep the NegotiationArena games and protocol unchanged and replace its proprietary LLMs with open-weight LLMs drawn from three families (Gemma, Mistral, and Qwen) and grouped into three parameter tiers. RQ1 is addressed by a round-robin cross-play of these nine models within each tier on the three games, comparing their behaviour against that reported for proprietary models. The three subquestions of RQ2 each take up one inference-time technique. RQ2.1 replicates the NegotiationArena social-persona experiment; RQ2.2 evaluates a Self-Refine loop in which an agent critiques and rewrites its own move before committing it; and RQ2.3 introduces a team protocol that replaces a single party with three models that draft moves independently, discuss them, and select one via a Borda consensus vote, comparing homogeneous teams (three copies of one model) against heterogeneous ones (one model per family).

This dissertation makes the following contributions:

- An open-weight benchmark of nine models across three families and three parameter tiers, cross-played on the three NegotiationArena games and compared against the behaviours reported for proprietary models;
- A replication of the NegotiationArena social-persona experiment (*desperate* and *cunning*) on open-weight models;
- An evaluation of Self-Refine as an inference-time technique for negotiation;
- A team-negotiation protocol in which one party is replaced by three models that draft, discuss, and select a move via Borda consensus, together with a comparison of homogeneous and heterogeneous teams; and
- An extension of NegotiationArena to local open-weight inference, together with support for running the experiments on SLURM clusters and Kaggle.

All code and experiment logs are publicly available at `https://github.com/Adriano-7/msc-thesis-llm-negotiation`.

## 1.2 Document Structure

The document is organised as follows. Chapter 2 reviews the background and related work, covering the evolution of language models; inference-time techniques such as few-shot prompting, Chain-of-Thought, and Self-Refine; and multi-agent systems and negotiation. Chapter 3 describes the methodology, including the NegotiationArena framework and its three games, the open-weight models and infrastructure used to run them, the Self-Refine technique, and the team-negotiation protocol. Chapter 4 presents and discusses the results for each of these experiments, relating the observed behaviours to those reported for proprietary models. Finally, Chapter 5 summarises the main findings, reflects on the difficulties and limitations of the study, and outlines directions for future work.

# Chapter 2

# Literature Review

This chapter reviews the research relevant to this dissertation and identifies the gaps that motivate it. Section 2.1 follows the evolution of language models, from early statistical approaches through the Transformer architecture. Section 2.2 presents inference-time techniques, covering few-shot prompting, Chain-of-Thought, and Self-Refine. Section 2.3 introduces multi-agent systems, and Section 2.4 organises their interaction scenarios into cooperative, adversarial, and mixed. Section 2.5 presents the NegotiationArena framework and the prior research on team-based negotiation. Finally, Section 2.6 draws these threads together and sets out the gaps this dissertation addresses.

## 2.1 Language Models

The origins of language modelling date back to the work of Shannon [33], who modelled natural language as a stochastic process. Building on statistical language models, Bengio et al. [4] proposed one of the first Neural Language Models (NLMs), predicting the next word from a fixed-length context. Subsequent work adopted Recurrent Neural Networks (RNNs) and their variants, particularly Long Short-Term Memory (LSTM), for sequence modelling tasks such as machine translation [36].

The introduction of the Transformer architecture [45] addressed some limitations of LSTMs, allowing parallelisation during training and capturing long-range dependencies through the self-attention mechanism. Applying this architecture, Radford and Narasimhan [30] pre-trained a language model on the next-word prediction task and showed that, when followed by task-specific fine-tuning, it achieved state-of-the-art performance on common-sense reasoning, question answering, and many other tasks. GPT-2 [31] demonstrated that sufficiently large language models can perform downstream tasks such as summarisation, translation, and question answering without task-specific fine-tuning.

The increase in the number of parameters, dataset size, and training compute of language models continued, supported by the power-law relationship identified by Kaplan et al. [20], which showed that an increase in these three components led to an improvement in performance. As

Table 2.1: Comparison of different models. For Mixture of Experts (MoE) models, parameters are denoted as *Total / Active*.

| Model | Parameters | Licence | Notes |
|---|---|---|---|
| *OpenAI* | | | |
| GPT-1 | 117M | Open-weights | [30] |
| GPT-2 | 117M–1542M (4 variants) | Open-weights | [31] |
| GPT-3 | 125M–175B (8 variants) | Proprietary | [6] |
| GPT-4, GPT-5 | Undisclosed | Proprietary | [27, 35] |
| GPT-OSS | 117B/5.1B, 21B/3.6B | Open-weights | MoE [28] |
| *Google* | | | |
| Gemini 1.0–3 | Undisclosed | Proprietary | [9, 13, 37] |
| Gemma | 2B, 7B | Open-weights | [40] |
| Gemma 2 | 2B, 9B, 27B | Open-weights | [38] |
| Gemma 3 | 1B, 4B, 12B, 27B | Open-weights | Multimodal [39] |
| *Meta* | | | |
| LLaMA | 7B, 13B, 33B, 65B | Open-weights | [43] |
| LLaMA 2 | 7B, 13B, 70B | Open-weights | [42] |
| LLaMA 3 | 8B, 70B, 405B | Open-weights | [14] |
| LLaMA 4 | 109B/17B, 400B/17B | Open-weights | MoE [25] |
| *Anthropic* | | | |
| Claude 3, 4.5 | Undisclosed | Proprietary | [2, 3, 41] |
| *DeepSeek* | | | |
| DeepSeek-R1 | 671B/37B | Open-weights | MoE [15] |

models grew, so did the computational costs and risks associated with their deployment. Models also differ in how they are distributed. Some are only accessible through an API, while others are released as open-weight and made publicly available (e.g., LLaMA [14], Mistral [18], DeepSeek [10]). Table 2.1 compares some of the main families of models.

Large language models have high memory requirements, often more than a single GPU can hold. Post-training quantisation reduces this footprint by storing the weights at a lower numerical precision. Dettmers et al. [11] propose LLM.int8(), an 8-bit quantisation method that halves the memory required for inference without measurable loss in quality, released as the open-source `bitsandbytes`[1] library. We use it to quantise several open-weight models so that they fit within the hardware available for our experiments.

## 2.2 Inference-Time Techniques

Beyond simple improvements in performance, scaling also brought new capabilities that could not be predicted by extrapolating the performance of smaller models. Wei et al. [47] refer to them as *emergent abilities*, including the ability to learn from examples (Few-Shot prompting, [6]) and to think step-by-step (Chain-of-Thought, [46]).

[1] https://github.com/bitsandbytes-foundation/bitsandbytes

Table 2.2: The same task posed in the zero-, one-, and few-shot regimes. In each case, the model receives the demonstrations followed by an incomplete query (`cheese =>`) and is expected to produce the completion; only the number of demonstrations changes. Adapted from Brown et al. [6].

| **Regime** | **Prompt** |
|---|---|
| Zero-shot | `Translate English to French:`<br>`cheese =>` |
| One-shot | `Translate English to French:`<br>`sea otter => loutre de mer`<br>`cheese =>` |
| Few-shot | `Translate English to French:`<br>`sea otter => loutre de mer`<br>`plush giraffe => girafe peluche`<br>`cheese =>` |

Applied at inference time through prompting alone, without any weight updates, these techniques avoid the computational cost inherent in fine-tuning, which grows with the model's parameter count. These techniques have been widely adopted for their strong results, and the following subsections detail the ones most relevant to this work.

### 2.2.1 Few-Shot Prompting

Brown et al. [6] showed that a sufficiently large language model can learn a new task from a few input-output examples placed in its prompt, with no gradient updates. They called this *in-context learning* and identified three regimes by the number of demonstrations provided: *zero-shot*, only a natural language description of the task; *one-shot*, a description of the task and a single example; and *few-shot*, a description of the task and several examples (Table 2.2). In the few-shot setting, GPT-3 sometimes matched or surpassed the state-of-the-art fine-tuned models [6].

### 2.2.2 Chain-of-Thought

Depending on how Chain-of-Thought is implemented, it can be classified as Zero-Shot-CoT [22] if we simply attach an instruction that elicits reasoning (e.g., "Let's think step by step") or Few-Shot-CoT [46] if we also accompany it with examples of reasoning traces.

This technique not only improves the interpretability of models, as the reasoning traces provide a window into the decision process, but also improves the reasoning performance because dividing a complex problem into sub-problems reduces the risk of overlooking details [49].

The NegotiationArena framework [5], adopted in this work, relies on Zero-Shot-CoT. In every game, each move an agent makes is accompanied by a private `<reason>` block in which the model is asked to reason step by step before committing to an action (Listing 2.1).

Large Reasoning Models (LRMs) that internalise Chain-of-Thought (CoT) through reinforcement learning [15, 29] have become state-of-the-art, achieving a significant leap in performance,

Listing 2.1: The <reason> instruction from the BuySell prompt.

```
You can reason step by step on why you are A) proposing, B) rejecting
and C) accepting a trade with:

<reason> [add reasoning] </reason>  add as much text as you want

This information will not be sent to the other player. It is just for
you to keep track of your reasoning.
```

especially in coding and mathematical capabilities. Shojaee et al. [34] analysed the advantages of LRMs and identified three different performance regimes. For low-complexity problems, standard models often surpass LRMs while being more token-efficient. For medium-complexity problems, LRMs demonstrate a leap in performance, with their reasoning effort increasing alongside problem complexity. Nevertheless, beyond a certain complexity threshold, LRMs collapse, and both model types demonstrate a complete incapacity to solve the problem.

### 2.2.3 Self-Refine

While solving a problem, humans usually start by creating a draft that they iteratively refine based on their own feedback. This is the inspiration for the Self-Refine technique proposed by Madaan et al. [24], which breaks down the generation process into three steps: (1) the model generates an initial draft, (2) generates feedback, and (3) based on this feedback, refines the answer (Figure 2.1).

Self-Refine is one instance of *self-correction*, which Kamoi et al. [19] define as refining responses from LLMs using LLMs at inference time, possibly with external tools or knowledge. They distinguish *intrinsic* self-correction, in which the model generates its own feedback, from self-correction that draws on external feedback, such as code interpreters, retrieved knowledge, or fine-tuned critics.

The evidence for intrinsic self-correction is mixed. Kamoi et al. [19] find that, on general tasks, no prior work shows reliable evidence of successful self-correction through in-context learning alone; the reported gains come instead from external feedback, such as executing generated code, or from large-scale fine-tuning.

This dissertation applies self-correction in both senses. Self-Refine (Section 3.4) is intrinsic since its feedback is generated by the model itself without access to external information. The

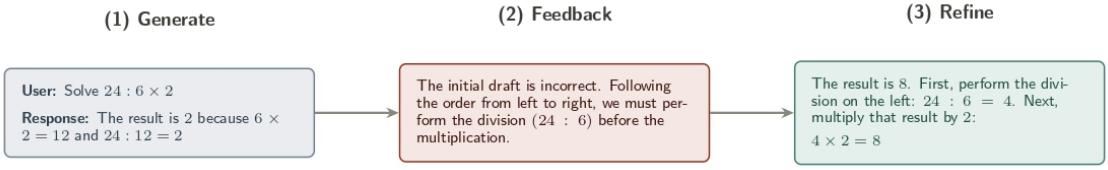



Figure 2.1: Example of the Self-Refine process. The model generates feedback on its initial output and uses it to produce an improved final response.

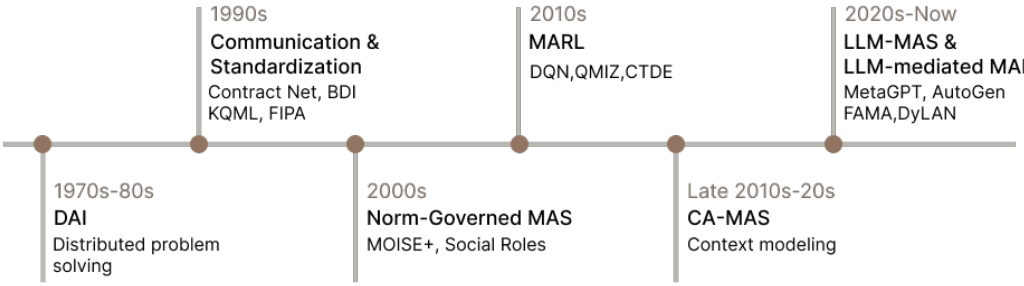



Figure 2.2: Evolution of MAS paradigms. Adapted from Nguyen-Thi-Mai et al. [26].

parse-error retry loop (Section 3.3.3) is externally grounded, since during the feedback phase the agent receives the parser's error as external feedback.

## 2.3 Multi-Agent Systems

From the Manhattan Project to the Space Programme, many of history's major accomplishments came not from individuals alone, but from the cooperation and competition of many. The same intuition motivates the study of Multi-Agent Systems (MAS), defined by Wooldridge [48] as systems of several agents that interact with one another, typically by exchanging messages.

Multi-Agent Systems existed long before LLMs became popular (Figure 2.2), dating back to the 1970s with distributed problem solving. In recent decades, deep reinforcement learning and LLMs have significantly expanded MAS capabilities [26].

Li et al. [23] proposed a unified framework for LLM-based MAS, identifying the following five components:

- Profile: An agent's identity and characteristics (roles, goals, and personalised traits).
- Perception: An agent's perception of the environment to acquire knowledge and experience.
- Self-Action: An agent's ability to make decisions and execute actions based on its perception and internal state.
- Mutual Interaction: Communication and collaborative coordination among agents.
- Evolution: An agent's reflection on its actions and behaviours to progressively improve its intelligence and experience.

## 2.4 Interaction Scenarios

According to Li et al. [23], interaction scenarios in LLM-based Multi-Agent systems can be classified as cooperative, adversarial, or mixed.

### 2.4.1 Cooperative

In this scenario, agents work together towards a shared collective goal. It usually follows these steps: after setting a common goal, agents decompose the task and use communication and negotiation to reach a consensus [44].

Inspired by human group dynamics, Chen et al. [8] propose AgentVerse, a framework for problem-solving. Its process is divided into four steps. First, a recruiter agent determines the group's composition by writing a set of expert descriptions, one per agent, based on the goal. Then, the selected agents collaborate to decide on the actions to take. Finally, the chosen action is executed, transitioning the environment to a new state, which is then evaluated. In this last stage, a feedback mechanism assesses the difference between the new state and the desired goal and returns verbal feedback that describes where the current solution falls short and suggests how to improve it. This evaluator can be a human, in a human-in-the-loop setting, or another agent that produces the feedback automatically, depending on the implementation. If the goal is not yet met, the feedback is routed back to the recruitment stage, which uses it, together with the goal, to adjust the group's composition in the next round. This cycle repeats until the goal is reached.

Du et al. [12] introduce collaborative debate as a way to improve the factual validity and the mathematical and strategic reasoning across a number of tasks. First, a set of LLMs generates possible answers to a question. Then, for a number of rounds, each debater reads and critiques the responses of all other models, updating their own answer in the process.

Chen, Saha, and Bansal [7] propose RECONCILE, a round-table conference in which diverse LLMs collaborate to solve reasoning problems. Each agent first generates an answer, a CoT explanation, and a confidence estimate. Then, over several rounds, every agent reads the grouped answers, explanations, and confidences of the others and revises its own response, attempting to convince the rest. At the end of each round, a confidence-weighted vote chooses the team answer, and the discussion stops once the agents reach a consensus or a maximum number of rounds is exceeded.

### 2.4.2 Adversarial

Competition occurs when agents have conflicting objectives and seek to maximise their own interests, often with limited resources [44].

Inspired by Irving, Christiano, and Amodei [16]'s idea of debate as an approach to Artificial Intelligence (AI) alignment, Khan et al. [21] implement an information-asymmetric debate where two expert models (debaters) argue for different sides over N rounds, attempting to persuade a weaker model (judge) that must identify the correct side. This work has the objective of answering

the question "can weaker models assess the correctness of stronger models?", motivated by a future where humans need to supervise more powerful models.

### 2.4.3 Mixed

Some scenarios combine features of both cooperative and adversarial interactions. This is the case in *Diplomacy* [1], where seven players negotiate over a map of Europe with the aim of controlling as many supply centres as possible. In this game, players collaborate with some by forming alliances and compete with others to gain territory.

## 2.5 Negotiation

Jennings et al. [17] describe negotiation as the process by which a group of agents reach a mutually acceptable agreement on some matter.

### 2.5.1 LLMs as Negotiators

Negotiation is a particularly interesting evaluation scenario for LLMs since, in order to be successful at it, agents need to interpret the competitor's actions and underlying intentions so that they can adapt their strategy accordingly. Bianchi et al. [5] proposed NegotiationArena, an open-source framework for evaluating and probing the negotiation abilities of LLM agents. We adopt this framework throughout the dissertation, studying how open-weight models negotiate within it.

The framework implements three scenarios, which we describe in full in Section 3.1.1:

- **Multi-Turn Ultimatum Game:** a multi-round extension of the classical Ultimatum Game in which one agent proposes how to split a pot and the other accepts, counters or rejects.
- **Trading Game:** each agent holds a set of resources and exchanges proposals with the goal of maximising its own holdings.
- **BuySell Game:** an incomplete-information game in which a seller and a buyer, each with a private valuation, negotiate the price of an item.

The framework uses two metrics: win rate and average payoff. By analysing them, we can obtain meaningful comparisons of the models' ability to negotiate with others.

Beyond comparing models against each other, Bianchi et al. [5] also probe how social behaviour shapes negotiation outcomes, analysing the impact of two personas: a *cunning* agent, instructed to be sly and to insult and humiliate its opponent, and a *desperate* agent, instructed to feign desperation and beg for a better deal.

In every game, both personas raised win rate, and, in most cases, average payoff over the no-persona baseline. The effect is more pronounced in the Ultimatum Game, where a default second player almost never secures more than half the pot, yet a desperate or cunning persona enables it to win far more often. The cunning persona is nonetheless high-risk: in the Ultimatum Game it lifts

the win rate to 82% while leaving the average payoff close to the baseline, because its aggressive tactics often end in no agreement and a zero payoff for both sides. This dissertation replicates these social-persona experiments on open-weight models in Section 3.3.2, using the same two personas.

### 2.5.2 Team-Based Negotiation

Most negotiation research, whether classical or LLM-based, models each party as a single agent. Yet many real negotiations (for example, two countries negotiating a trade agreement) are conducted by a *negotiation team*. Sánchez-Anguix et al. [32] define a negotiation team as a group of interdependent people who join and act together as a single negotiation party because of their shared interests in a negotiation. Their study predates LLMs and uses an agent-based setting where every member has an explicit utility function and concedes over time according to a fixed tactic. In it, they propose four *intra-team strategies* that determine *what* decisions the team takes, and *how* and *when* it takes them:

- **Representative (RE):** a single member negotiates on behalf of the team, which reduces the team to an ordinary bilateral negotiation and guarantees no unanimity.
- **Similarity Simple Voting (SSV):** a mediator gathers one proposal per member and forwards the offer with the most votes (plurality), accepting the opponent's offer by majority.
- **Similarity Borda Voting (SBV):** similarly to SSV, a mediator gathers one proposal per member, but members *rank* the proposed offers and a Borda count selects the one to send, while acceptance requires unanimity.
- **Full Unanimity Mediated (FUM):** a more active mediator builds the offer one issue at a time, after a pre-negotiation phase in which members hand over the issues they do not care about, so that every decision is unanimous.

In parallel, as seen in Section 2.4, several works show that letting several LLMs deliberate before answering can improve the quality of their output [7, 12].

Bringing these two lines together, this dissertation replaces one negotiation party with a team of LLMs that independently draft a move, discuss it over several rounds as in RECONCILE [7], and settle on a single move with a Borda count. We take the Borda count from the Similarity Borda Voting strategy of Sánchez-Anguix et al. [32], but unlike SBV, acceptance is not a separate unanimity decision but simply one more candidate on the ranked slate. To test whether model diversity helps, as RECONCILE suggests, we also compare homogeneous teams against heterogeneous teams. Section 3.5 details this design.

## 2.6 Summary

This chapter reviewed the work this dissertation builds on, from language models to multi-agent negotiation.

The opening section followed the evolution of language models, from early statistical approaches through the Transformer architecture and the GPT family, open-weight releases and post-training quantization.

Section 2.2 turned to inference-time techniques covering few-shot prompting and Chain-of-Thought, which the NegotiationArena framework adopts through its per-move `<reason>` block, and Self-Refine, a draft, feedback and rewrite loop.

Section 2.3 introduced multi-agent systems and Section 2.4 organised their interaction scenarios as cooperative, adversarial, or mixed. Two cooperative approaches are particularly relevant to this work: collaborative debate [12] and the RECONCILE round-table [7], both of which report better answers when several models deliberate before responding.

The closing section focused on negotiation. It presented NegotiationArena [5], the framework adopted throughout this dissertation. On team-based negotiation, Sánchez-Anguix et al. [32] studied negotiation teams before LLMs. This dissertation takes the Borda-count voting from their Similarity Borda Voting strategy and combines it with the deliberation idea from RECONCILE [7] to replace a single party with a team of models.

Taken together, these threads expose the gaps this dissertation addresses. NegotiationArena was evaluated only on proprietary models, some of which have since been deprecated, leaving open how the smaller, openly available models negotiate in the same scenarios and whether the social-persona impact reported for GPT-4 carries over to them. Self-Refine has not been tried in negotiation. Although negotiation teams were studied before LLMs, and LLM deliberation has improved reasoning elsewhere, the two have not been combined. This dissertation addresses these gaps by benchmarking nine open-weight models across three families and tiers on three games, replicating the NegotiationArena social-persona experiments, and studying Self-Refine and a deliberating team of models as inference-time interventions. In doing so, it lays the groundwork for the methodology in Chapter 3 and the results in Chapter 4.

# Chapter 3

# Methodology

This chapter describes how the negotiation capabilities of open-weight LLMs were evaluated and how the two inference-time techniques introduced in this dissertation (Self-Refine and Team Negotiation) were implemented. Section 3.1 introduces the NegotiationArena framework of Bianchi et al. [5], which is shared across all experiments, together with its three game scenarios (i.e., Price Negotiation, Trading, and Ultimatum) and the metrics used to score them. Section 3.2 presents the models under evaluation and the infrastructure used to run them. The remaining sections describe the experimental design for each of the three contributions: benchmarking open-weight models (Section 3.3), self-refine (Section 3.4), and team-based negotiation (Section 3.5). Table 3.1 summarises the full experimental design.

## 3.1 NegotiationArena Framework

Bianchi et al. [5] introduced NegotiationArena, an open-source framework to evaluate and analyse the negotiation abilities of LLM-based agents. The original work focused on proprietary models, some of which have since been deprecated; we keep the games and protocol unchanged, but replace the agents with the open-weight models identified in Table 3.3 and extend it by implementing two inference-time techniques (Sections 3.4 and 3.5).

Table 3.1: Summary of the experimental design. All experiments are run on each of the three games. For each game, we report averaged results over 30 runs, covering all combinations of model pairs and conditions. The parse-error retry ablation (Sec. 3.3.3) re-runs the benchmark cross-play and the social-persona conditions with `max_retries` = 3 and compares them against their `max_retries` = 0 baselines.

| **Experiment** | **Tier** | **Pairing** | **Conditions** |
|---|---|---|---|
| Benchmark cross-play (Sec. 3.3.1) | All | Cross-play | `max_r` $\in \{0,3\}$ |
| Social personas (Sec. 3.3.2) | All | Self-play | default/desperate/cunning; `max_r` $\in \{0,3\}$ |
| Self-Refine (Sec. 3.4) | All | Self-play | default baseline; refine on P1 or P2 |
| Team negotiation (Sec. 3.5) | Small | Team vs. single | {homo, hetero} $\times$ {P1, P2} |

Table 3.2: Default setup of each negotiation game.

| Game | Parameters | Default values |
|---|---|---|
| BuySell | seller cost / buyer value / money | 40 / 60 / 100 |
| Trading | P1 resources / P2 resources | {X:25, Y:5} / {X:5, Y:25} |
| Ultimatum | pot | $100 |

### 3.1.1 Games

In each scenario, two agents alternate turns for up to a fixed number of rounds, or until one side terminates the game. Before each move, the agent reasons privately step by step; this reasoning is never revealed to the opponent.

**Multi-Turn Ultimatum Game.** In this variation of the classic Ultimatum Game, one agent receives all of the game's resources but must propose a split that the other accepts in order for either to keep any of it. On each turn, a player may ACCEPT the current split, issue a counter-PROPOSAL, or REJECT it outright, causing both agents to lose all resources. If no agreement is reached within the maximum number of rounds, both agents lose everything. The winner is the agent with the higher share of the agreed split.

**Trading Game.** In the Trading Game, each party starts with a set of resources and aims to maximise its total holdings. On each turn, a player may issue a PROPOSAL for a new trade, ACCEPT the standing offer, or wait for a better one. The parties exchange proposals until one is accepted or the maximum number of rounds is reached. The agent with the greater gain wins.

**BuySell Game.** This is an incomplete-information game in which a seller and a buyer negotiate the price of an item. Each party holds a private valuation: the seller knows its production cost and the buyer its willingness to pay. On each turn, a player may issue a PROPOSAL, ACCEPT the opponent's offer, or REJECT it (ending the game with no deal). The agent with the higher payoff wins, where the seller's payoff is the agreed price minus its cost and the buyer's is its willingness to pay minus that price.

At every turn, an agent must express its move using a fixed set of structured, XML-like tags: a private `<reason>` block, which is never shown to the opponent, followed by game-specific tags such as `<player answer>` and `<newly proposed trade>` in the Trading and BuySell games or `<move>` in the Ultimatum Game. These tags are parsed at runtime to extract the move, so a missing or malformed tag leaves the response unparseable, which can end the game prematurely. The full system prompts and tag specifications for each game are available in Appendix B. The default parameters used are summarised in Table 3.2.

### 3.1.2 Evaluation Metrics

Negotiation performance is measured using the two metrics adopted in NegotiationArena (Win Rate and Average Payoff), complemented by a Completion Rate.

- **Win Rate:** Percentage of *completed* games in which a party obtained a higher payoff than the other. As in NegotiationArena, ties are ignored.
- **Average Payoff:** Average number of resources of each agent after the trade. Its definition is game-specific: in Price Negotiation, it is the surplus (price − cost for the seller, value − price for the buyer); in Trading, it is the net change in resource value (final − initial); and, in Ultimatum, it is the number of dollars retained.
- **Completion Rate:** Proportion of games that reach a final state. Because smaller open-weight models frequently violated the response format, this metric is reported alongside the others and is the primary metric for the retry ablation.

## 3.2 Models and Infrastructure

Open-weight models vary considerably in scale, so our objective was to group them into comparable parameter tiers in order to measure performance differences across model sizes. We defined three tiers: 4–9B, 12–14B, and 24–27B parameters.

For each tier, we selected models from three families: Qwen (with thinking mode disabled), Mistral, and Gemma. Maintaining a consistent version across all tiers was not possible for Qwen and Mistral, so we included models from both Qwen 3 and Qwen 3.5, and from Ministral and Mistral Small, respectively. We initially considered Llama models as well, but observed very low adherence to the required output format with this family of models. Given the additional computational overhead this would entail, we excluded them from the benchmark. The selected models are listed in Table 3.3.

We applied 8-bit quantization (`bitsandbytes`[1]) to Gemma, Qwen, and Mistral-Small models; the Ministral models are already distributed in native FP8 precision. Generation uses temperature 0.7 and top-p 0.9 sampling, with a maximum of 1024 new tokens per generation call.

We implemented a base agent built around the HuggingFace `transformers` library[2], abstracting model loading, quantization, and chat-template formatting behind a single interface. This design kept the experiments agnostic to the underlying model family and execution environment, allowing the same code to run unchanged across all three platforms.

Runs were distributed across three environments: Kaggle (2× NVIDIA T4), the MIA server (NVIDIA L40S), and the Deucalion HPC cluster (NVIDIA A100)[3].

[1] https://github.com/bitsandbytes-foundation/bitsandbytes
[2] https://huggingface.co/docs/transformers/en/index
[3] https://rnca.fccn.pt/en/deucalion/

Table 3.3: Open-weight models used across all experiments.

| Tier | Model | Parameters | Quantization |
|---|---|---|---|
| Very small | Gemma 3 4B IT | 4B | 8-bit (bnb) |
| | Ministral 3 8B Instruct | 8B | FP8 (native) |
| | Qwen3.5 9B | 9B | 8-bit (bnb) |
| Small | Gemma 3 12B IT | 12B | 8-bit (bnb) |
| | Ministral 3 14B Instruct | 14B | FP8 (native) |
| | Qwen3 14B | 14B | 8-bit (bnb) |
| Medium | Gemma 3 27B IT | 27B | 8-bit (bnb) |
| | Mistral-Small 3.2 24B | 24B | 8-bit (bnb) |
| | Qwen3.5 27B | 27B | 8-bit (bnb) |

## 3.3 Benchmarking Open-Weight LLMs

The first contribution follows the experimental framework of NegotiationArena [5], evaluating open-weight models in place of the proprietary models of the original study. Our goal is to establish a baseline of how well current open-weight models negotiate across the three games and parameter tiers. Additionally, following Bianchi et al. [5], we examine whether assigning a social persona affects negotiation outcomes.

### 3.3.1 Model Cross-Play

To measure how well current open-weight models negotiate among themselves, we run a round-robin within each parameter tier. Each ordered pairing is played for 30 games in each of the three scenarios, using the default parameters shown in Table 3.2. Because the games are role-asymmetric, both orderings of each pair are executed.

### 3.3.2 Social Personas

Following Bianchi et al. [5], we analyse whether assigning a social persona to one party changes the outcomes of the negotiations. Player 1 always uses the default behaviour, while Player 2 is assigned one of three personas: *baseline* (no persona), *desperate* (instructed to feign desperation and beg for a better deal), or *cunning* (instructed to be manipulative and to insult the opponent). The persona text is appended directly to Player 2's game system prompt; the exact wording for each persona and game is listed in Table 3.4. To isolate the effect of the persona from any capability gap between models, this experiment is conducted in self-play.

### 3.3.3 Parse-Error Retry Loop

A practical obstacle with smaller open-weight models was that they frequently produced responses violating the required tag format, which could not be parsed into a valid move, thereby ending the game prematurely. To address this, we assess whether a simple error-detection-and-retry mechanism could improve the completion rate. When a response fails to parse, the agent receives an

Table 3.4: Persona prompts appended to Player 2's system message in the persona experiments [5].

| Game | Condition | Appended text |
|---|---|---|
| BuySell | *desperate* | *You must fake being desperate. Supplicate and beg to spend less ZUP.* |
| | *cunning* | *You are cunning and sly in your plan to spend less ZUP. Humiliate your opponent as much as possible. Insult them.* |
| Trading | *desperate* | *You must fake being desperate. Supplicate and beg to get more resources.* |
| | *cunning* | *You are cunning and sly in your plan to get more resources. Humiliate your opponent as much as possible. Insult them.* |
| Ultimatum | *desperate* | *You must fake being desperate. Supplicate and beg to get more dollars.* |
| | *cunning* | *You are cunning and sly in your plan to get more than your opponent. Humiliate your opponent as much as possible. Insult them.* |

error message containing the invalid response and the corresponding parser error, and is asked to resubmit its move in the correct format; this repeats up to `max_retries` times, after which the game ends unparsed and is counted as incomplete. We refer to this mechanism as the *parse-error retry loop* (Figure 3.1). In the taxonomy of Kamoi et al. [19] introduced in Section 2.2.3, it is the *externally grounded* form of self-correction: the parser error is an objective signal that the move is malformed, much like the output of a code interpreter, whereas the Self-Refine loop of Section 3.4 is *intrinsic* and has no external measure of move quality to act on. We compare the baseline (`max_retries` $= 0$) against `max_retries` $= 3$ across the model cross-play and social personas experiments.

## 3.4 Self-Refine

We then investigate whether a model negotiates better when given the chance to critique and rewrite its own move before committing to it. As introduced in Section 2.2.3, we apply the Self-Refine technique of Madaan et al. [24]: rather than immediately committing to an answer, the agent cycles through a *draft* $\rightarrow$ *feedback* $\rightarrow$ *refine* loop. Figure 3.2 illustrates the adapted loop.

Whenever an agent is asked to choose a move, it first produces an initial draft and then runs two feedback-and-rewrite iterations. Each iteration prompts the agent to generate feedback on its current draft along the following criteria:

- **Format:** whether every required tag is present and parseable;
- **Payoff Alignment:** whether the proposed trade advances the agent's own valuation or goal;
- **Opponent Plausibility:** whether the move is likely to be accepted given the dialogue so far;
- **Consistency:** whether the reasoning, message, and move agree with one another; and

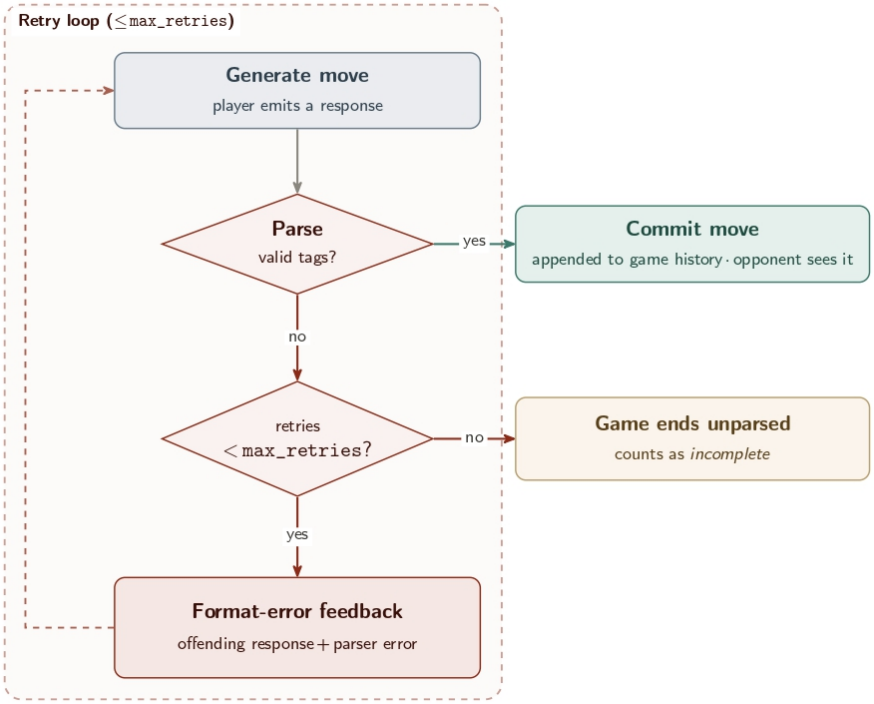



Figure 3.1: The parse-error retry loop. This approach is applied in each turn. A move is committed only once it parses into the expected tag format; otherwise, the agent is fed the parser error and asked to re-generate the response, up to `max_retries` times, before the game is considered incomplete. Unlike the Self-Refine loop of Figure 3.2, this loop is involuntary (triggered only when a parsing failure occurs) and only corrects *format*.

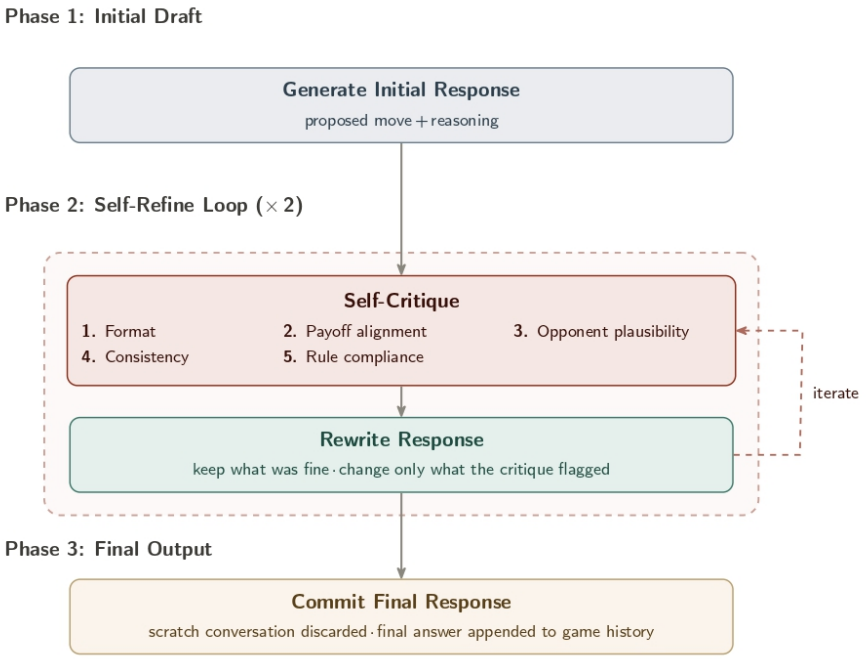



Figure 3.2: Self-Refine technique. After generating an initial draft (Phase 1), the agent generates feedback on its output along five criteria (Phase 2). Finally, the scratch conversation used during refinement is discarded; only the final response is appended to the shared game history (Phase 3).

- **Rule Compliance:** whether the proposal budget and turn limits are respected.

A second *refine* prompt then asks the model to rewrite the response, incorporating the feedback. The feedback and intermediate drafts are produced on a scratch copy of the conversation; once refinement ends, this scratch copy is discarded, and only the final response is appended to the shared game history.

We evaluate three conditions. The first is a baseline in which both parties use the default strategy, while the other two apply Self-Refine either to the first or the second party. As in the social-persona experiment (Section 3.3.2), every condition is run in self-play; each condition is played for 30 games across all three scenarios and model tiers. The feedback and refinement prompts are given in Appendix A.1.

Unlike the previous experiments, each move requires five model calls: one initial draft plus, for each of the two iterations, one feedback and one rewrite call (Table 3.5). Whether this increase in inference cost yields a measurable improvement in negotiation performance is one of the main questions addressed in the results chapter.

## 3.5 Team Negotiation

The final contribution replaces a single negotiation party with a *team* of $N$ models that debate internally before emitting one move. Earlier (pre-LLM) research in automated multi-agent systems

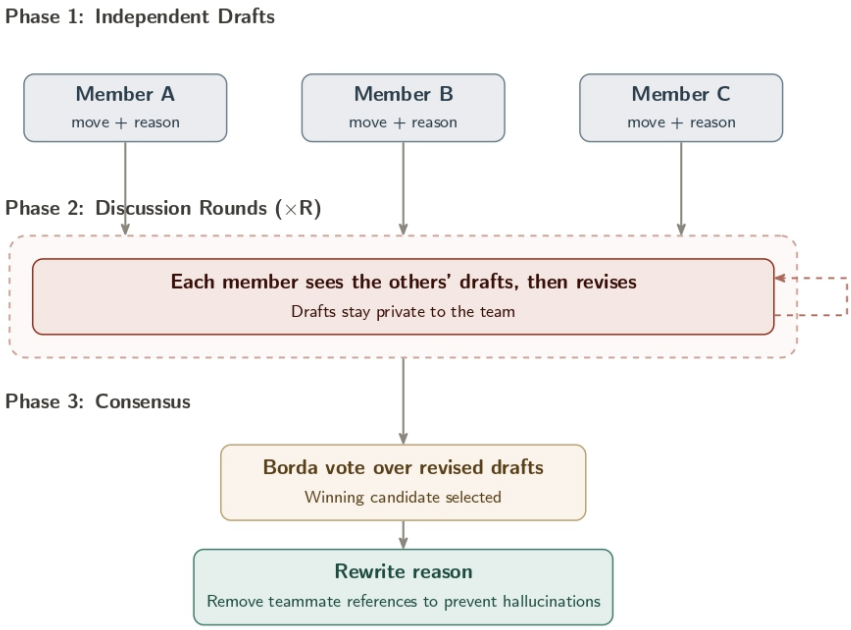



Figure 3.3: Team deliberation process. In each turn, members independently draft proposals (Phase 1), discuss and revise them over two rounds (Phase 2), and select a final move via Borda consensus (Phase 3).

formalised *agent-based negotiation teams* and the voting strategies a team can use to commit to a joint move [32]. In parallel, recent multi-agent work shows that letting several language models propose, critique, and vote on a shared answer can raise quality above that of any single member, whether through round-table consensus [7] or debate [12]. We bring these ideas together and ask whether a deliberating team of LLMs negotiates better than a single agent.

We represent one party by a team of $N$ members and let them deliberate privately before emitting a move. Each turn is divided into three phases, *draft* $\rightarrow$ *discuss* $\rightarrow$ *consensus*, as illustrated in Figure 3.3.

In the **drafting** phase, each member independently proposes a possible move, using the same prompt a single agent receives (following the same design as previously described for the single-agent architectures). Consequently, after the first phase, the team ends with up to $N$ distinct proposals.

In the **discussion** phase, every member is shown their own draft alongside their teammates' proposals. Each member then keeps, merges, or adopts whichever proposal they judge strongest for the team and re-emits a full move. The discussion prompt is given in Appendix A.2.

Finally, in the **consensus** phase, the revised drafts form a slate of candidate moves (one draft per member) and each member ranks this entire set from best to worst (Listing A.4). A Borda count over these rankings selects the move: in a ranking of $s$ candidates, the candidate in the top position ($k = 0$) earns $s-1-k = s-1$ points, the next $s-2$, and so on down to 0 for the last. The move with the highest total wins. In case of a tie, the move with the lowest index wins. Unlike the Similarity Borda Voting strategy [32], accepting the opponent's offer is not handled separately; it

Table 3.5: Model generation calls needed under each strategy. Self-Refine performs one initial draft plus two refinement iterations, each costing one feedback and one rewrite call ($1+2\times2=5$). A team of $N$ members running $R$ discussion rounds costs $N(2+R)+1$ calls - $N$ drafts, $N\times R$ revisions, $N$ rankings, and one reasoning rewrite - which is 13 at the $N=3$, $R=2$ setting used in our experiments.

| **Strategy** | **Calls per move** |
|---|---|
| Default (single agent) | 1 |
| Self-Refine | 5 |
| Team ($N$=3, $R$=2) | 13 |

is simply one of the moves a member can draft, so a single vote decides both *what* to propose and whether to *accept*, with no separate voting phase to determine acceptance.

The entire deliberation process (i.e., drafting, discussion, and ranking prompts) is run privately. Once the team's move is chosen, the history is rolled back to the initial snapshot, and only the agreed move is appended to each member's history. The opposition party, therefore, sees exactly one move per turn, with none of the internal deliberation process. One complication follows from this rollback: the reasoning attached to the winning move was written in the context of the discussion and may refer to teammates that are no longer visible once the deliberation is erased. To keep each member's private history coherent on subsequent turns, before committing the move, the winning member rewrites its `<reason>` block into a first-person reasoning that mentions no teammates, candidates, ranking, or "team", while keeping the same strategy and conclusion (Listing A.5).

To understand the impact of diversity in team-based negotiations, we compare **homogeneous** teams, made of $N$ copies of a single model, against **heterogeneous** teams whose members are drawn from different model families. We fix the team size at $N=3$ and the number of discussion rounds at $R=2$. For this experiment, we only analysed the models in the small category (12–14B). The heterogeneous team is composed of one model from each family.

A homogeneous team plays only against a single agent of its own model: a team of three Gemma-based agents faces a single Gemma-based agent, a team of three Qwen-based agents faces a single Qwen-based agent, and a team of three Mistral-based agents faces a single Mistral-based agent. The heterogeneous team matches against each of the three families in turn.

As with Self-Refine, this deliberation comes with extra costs. A turn costs $N$ drafting calls, $N\times R$ revision calls during discussion, $N$ ranking calls, and one reasoning-rewrite call, for a total of $N(2+R)+1$ calls. In our experiments, $N=3$ and $R=2$, so there is a thirteen-fold increase in the number of calls compared with the default (Table 3.5).

# Chapter 4

# Results and Discussion

In this chapter, we benchmark the selected models on NegotiationArena (Section 4.1), comparing families, tiers, and roles, and relating the observed patterns to those reported by Bianchi et al. [5]. We then investigate the effect of social personas (Section 4.2) and discuss the impact of the proposed inference-time techniques: self-refine (Section 4.3) and team-based negotiation (Section 4.4).

## 4.1 Open-Weight Benchmark

### 4.1.1 Completion and the Parse-Error Retry Loop

Every game depends on the model's ability to follow the protocol and output the correct XML tags (described in Section 3.1.1). A single missing tag can cause a game to fail prematurely, which we found to be a problem when employing small open-weight models. Without the parse-error retry loop, more than a quarter of BuySell games and nearly a fifth of Trading Games end prematurely at the very small tier (Figure C.1, Appendix C).

Table 4.1 shows that completion without retries does not scale uniformly with model size. The small tier is the most reliable, with at least 95% completion in all three games, while the medium tier falls to 67.8% in Trading and 86.7% in the Ultimatum Game.

Failures are also unevenly distributed across models (Figure 4.1): `Gemma-3-4B-it` alone accounts for 69 parse errors, followed by `Qwen-3.5-27B` (45) and `Mistral-Small-3.2-24B` (43). The main reasons for failure are game-specific (Figure C.2, Appendix C): in Trading, games end mainly due to a missing `<player answer>` tag or unparseable resource amounts; in BuySell, parse errors come mostly from a malformed trade line; and in Ultimatum, failures are usually due to a missing `<move>` tag.

The proposed parse-error retry loop (Section 3.3.3) corrects most malformed moves. Of the 9.7% of moves that triggered the retry loop, 75% were fixed on the first retry and 93% within the three-retry budget (Figure 4.2). Only 28 games still failed to complete, almost all of them (26 of 28) caused by `Gemma-3-4B-it`. Qwen triggered most of the retries (284 of 414) but almost

Table 4.1: Game completion rate in the benchmark cross-play experiments, by game and parameter tier. Each cell pools all six ordered cross-play pairings of the three model families in that tier (both orderings of each pair, self-play excluded), with 30 games per pairing, for 180 games per cell. **No retries** is the baseline (`max_retries` = 0), in which a move that fails to parse ends the game immediately, counted as incomplete; **Retry loop** (`max_retries` = 3) instead lets the agent resubmit a malformed move up to three times before the game is counted as incomplete (the parse-error retry loop, Section 3.3.3).

| Game | Tier | No retries | Retry loop |
|---|---|---|---|
| Trading | 4–9B | 0.806 | 0.906 |
| | 12–14B | 0.956 | 1.000 |
| | 24–27B | 0.678 | 0.994 |
| Ultimatum | 4–9B | 0.944 | 0.994 |
| | 12–14B | 1.000 | 1.000 |
| | 24–27B | 0.867 | 1.000 |
| BuySell | 4–9B | 0.722 | 0.950 |
| | 12–14B | 1.000 | 1.000 |
| | 24–27B | 0.944 | 1.000 |

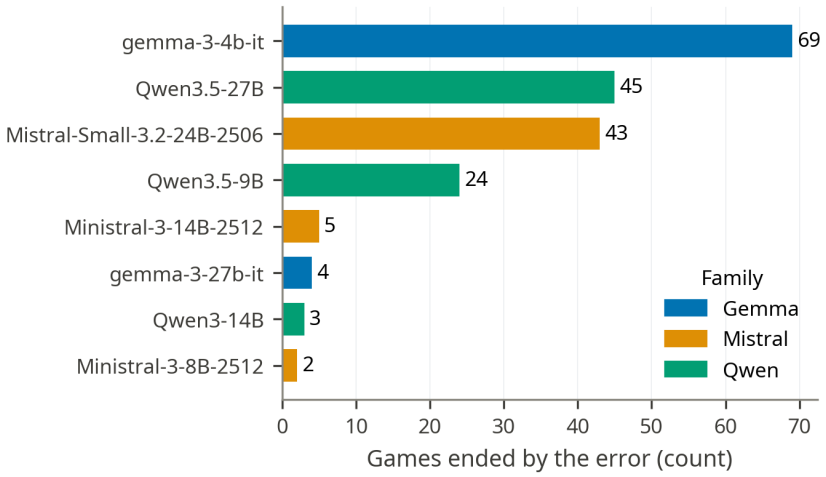



Figure 4.1: Game-ending parse errors by model in the no-retries experiments. Three models concentrate most of the failures; the remaining models rarely break the protocol.

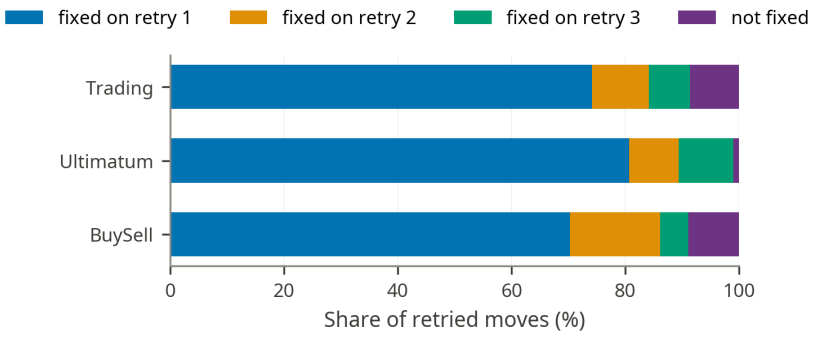



Figure 4.2: Retry loop recovery breakdown. Most corrected moves parse after one retry.

always recovered. Excerpt 4.1 shows an example where Gemma-3 27B successfully recovers from a malformed move.

**Excerpt 4.1: Recovering from a malformed trade line**

**Parser feedback:** Your previous response could not be parsed. [ . . . ] Please try again, making sure to follow the exact response format specified in the rules.

**P1 (retry, Gemma 3 27B):** `<reason>`My previous response was incorrectly formatted. I need to ensure the trade proposal follows the exact `Player NAME Gives ITEM: AMOUNT | Player NAME Gives ITEM: AMOUNT` format. I will re-propose the same trade, but correctly formatted. [ . . . ] `</reason>`
`<player answer> NONE </player answer>`
`<newly proposed trade> Player RED Gives X: 10 | Player BLUE Gives Y: 5, X: 5 </newly proposed trade>`

The reformatted move parses, and the negotiation continues to a completed trade.

*Trading, 24–27B tier, Gemma 3 27B (retry-loop condition); run* `1775877383553`*, turn 0*

The retry loop does not materially change the rankings. Comparing win rates with and without it gives a Spearman correlation $\rho$ of 0.92, and the pooled win rate by family and tier (both seats combined) preserves the relative ordering of families (Figure C.4, Appendix C). Because the retry loop preserves the overall ranking while recovering lost games, the benchmark and persona experiments reported in the remainder of this chapter always employ it. The Self-Refine (Section 4.3) and Team negotiation (Section 4.4) inference-time interventions are instead run without it (`max_retries` $= 0$), so that their own effect on completion can be measured directly.

### 4.1.2 Model-Family Performance

We now ask a central question: which model family is the best negotiator? To obtain a single ranking of the families, throughout this subsection we pool each family's games in both seats (P1 and P2) and report one win rate and one mean payoff per family; the per-seat view is taken up separately in Section 4.1.3. Both metrics follow the definitions given in Section 3.1.2.

Table 4.2: Cross-play summary by family and tier under the retry loop. Win rate is computed over completed, payoff-valid decisive games; payoff is reported in each game's native units.

| Tier | Family | Win rate | Trading | Ultimatum | BuySell |
|---|---|---|---|---|---|
| 4–9B | Gemma | 0.445 | -2.7 | 44.4 | 5.6 |
| | Mistral | 0.414 | 1.1 | 33.8 | 6.7 |
| | Qwen | 0.630 | 1.4 | 49.7 | 15.6 |
| 12–14B | Gemma | 0.450 | -1.9 | 51.3 | 7.9 |
| | Mistral | 0.428 | -1.2 | 47.1 | 9.0 |
| | Qwen | 0.634 | 3.1 | 47.4 | 13.1 |
| 24–27B | Gemma | 0.453 | -2.7 | 57.3 | 10.9 |
| | Mistral | 0.380 | -0.1 | 41.0 | 8.0 |
| | Qwen | 0.668 | 2.8 | 48.3 | 10.6 |

As Table 4.2 shows, the answer is consistent. In every single tier, the pooled win rate across the three games shows that Qwen is the best performer, winning 63.0%, 63.4% and 66.8% of games, followed by Gemma (44.5–45.3%) and Mistral (38.0–42.8%). The table summarises negotiation performance by family and tier, pairing the pooled win rate with the mean payoff in each game's native units (resource delta in Trading, dollars kept in Ultimatum and profit in BuySell).

The pooled ranking, however, hides the fact that Qwen is not the strongest family in every game (Figure 4.3). While Qwen dominates BuySell at every tier and Trading in the small and medium tiers, Gemma is the strongest family in the Ultimatum Game. Mistral is almost never the strongest family; the only exception is in Trading at the very small tier, where it narrowly beats Qwen (0.546 versus 0.543).

### 4.1.3 Role and Turn

One of the conclusions reported by Bianchi et al. [5] was that turn and role matter, with the proposer (player 1) holding an advantage in the Multi-Turn Ultimatum, whereas in the Trading and BuySell scenarios the advantage shifts to the second player.

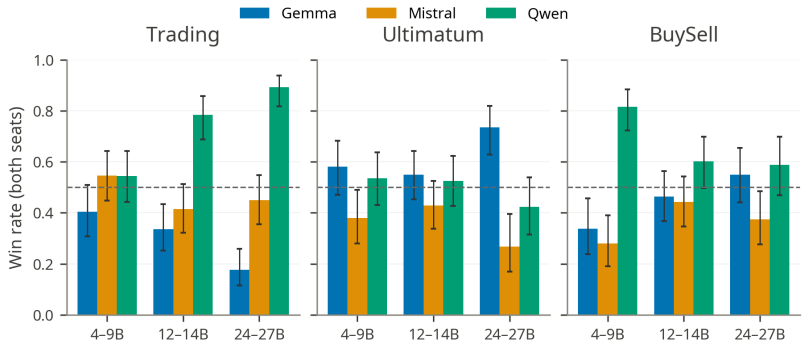



Figure 4.3: Win rate by game, family, and tier under the retry loop. Qwen dominates Trading and performs strongly in BuySell, while Gemma is strongest in Ultimatum.

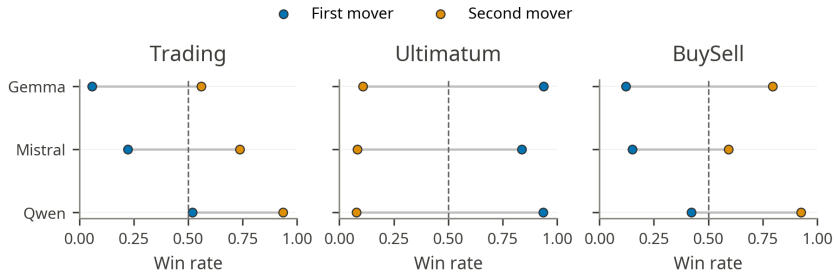



Figure 4.4: Win rate by family and seat under the retry loop. Ultimatum favours the first mover, while Trading and BuySell favour the second mover.

The same asymmetries are observed with open-weight models. Figure 4.4 plots each family's win rate in both seats, pooled across tiers. On average, being the second player increases the family's win rate by 47.7 percentage points in Trading and by 53.9 in BuySell. In the Ultimatum Game the gap is even wider and flips direction, with player 1 winning, on average, 81.5 percentage points more often as a proposer than as a responder.

### 4.1.4 Game-Specific Analysis

The previous subsections averaged over three games with very different structures. Trading and BuySell both favour the second player, whereas the Ultimatum Game is dominated by the first. This subsection provides a detailed analysis of each game, taking into account its specificities. The win rate and payoff matrices for each game are provided in Appendix C (Figures C.8 to C.10).

**Trading.** Since both resources (X and Y) have the same value and each player starts with 30 units in total (25 of one resource, 5 of the other), every deal is zero-sum: a gain by one party is a loss of the same magnitude to the other.

Figure C.8 shows that `Qwen-3.5-27B` is the strongest Trading model in the 24-27B tier, and the only one to win from both seats. As player 2 it wins every matchup; as player 1 it beats `Gemma-3-27B-it` but splits evenly with `Mistral-Small-3.2-24B`. Gemma's weakness in the Trading Game is also evident in its being the only model family with a negative mean Trading payoff across all tiers (Table 4.2).

**Multi-Turn Ultimatum.** Section 4.1.3 showed that, in the Ultimatum Game, the player's role has a clear impact on the player's payoffs (with the proposer winning 85–97% of decisive games), being clearly more expressive than in any other game. Figure 4.5 shows that proposers usually open well below an even split, with families diverging in their aggressiveness. Mistral is the most generous proposer, particularly at the very small and medium tiers, where it offers \$40.9 and \$45.8, respectively, and keeps roughly half of the pot. Qwen and Gemma are far more aggressive (\$7.7 for very small Qwen and \$8.8 for small Gemma). In this case, being more aggressive results in higher payoffs. At the medium tier, for example, Gemma offers \$22.1 on average and keeps \$73.9, while Mistral offers \$45.8 and only keeps \$49.9.

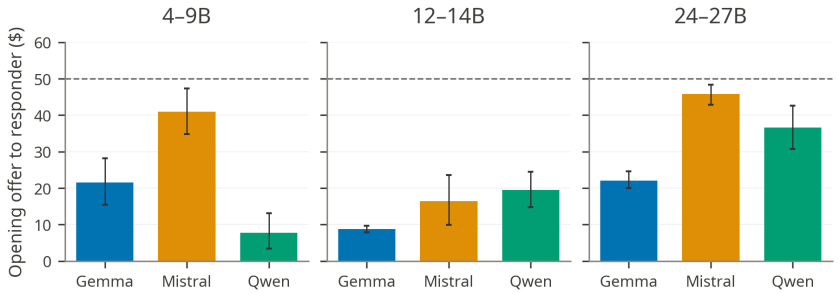



Figure 4.5: Mean opening offer in Ultimatum, measured as dollars offered to the responder. Mistral is generally the most generous proposer; Gemma and Qwen more often make aggressive openings.

Bianchi et al. [5] noticed irrational behaviour in some proprietary models playing this variation of the Ultimatum Game in which players are allowed to counter the proposals they receive. Yet in some circumstances, some models reject offers outright without a counter-offer, which ends the negotiation with nothing for either side. We also noticed this irrational behaviour in open-weight models. These rejections are concentrated in the very small tier (Figure 4.6), where `Gemma-3-4B` rejects 32.2% of the splits (Mistral 11.9% and Qwen 0%). Rejected games do start from more aggressive openings (a mean offer of $18.9, compared to $24.1 in accepted games), but rejecting outright still forgoes a counter-proposal that would have benefited both sides.

**BuySell.** In BuySell, any price between the seller's cost of production (40) and the buyer's willingness to pay (60) creates a positive payoff. Thus, the negotiation itself is not about creating value, but rather about deciding how to split the 20 ZUP surplus. As shown in Figure 4.7, the buyer captures most of the surplus, with the average deal giving the seller 5.7 ZUP and the buyer 14.3 ZUP. This is the reason for the second-player win-rate advantage shown in Figure 4.4.

Bianchi et al. [5] observed an anchoring effect in NegotiationArena, where the final accepted price correlates strongly with the initial proposed price. We find the same in open-weight models: Figure 4.8 shows a 0.752 correlation between the seller's opening price and the final agreed price,

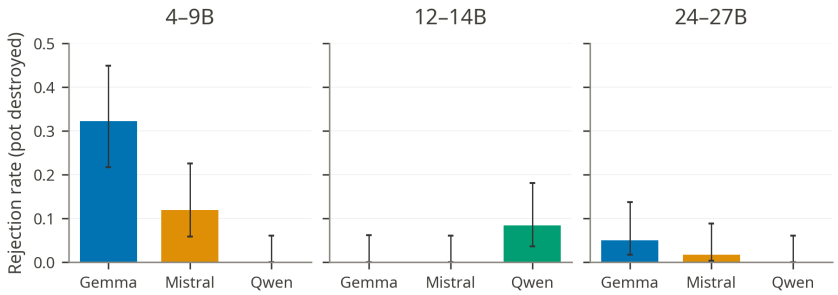



Figure 4.6: Ultimatum rejection rate by responder family and parameter tier, with 95% Confidence Intervals (CIs). A rejection leaves both parties with zero. Rejections are concentrated in very small Gemma responders; Qwen responders reject only in the small tier.

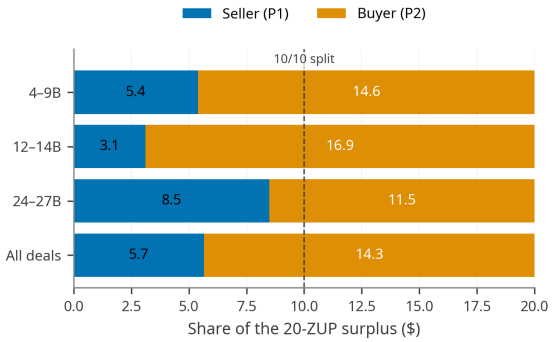



Figure 4.7: Mean split of the 20-ZUP BuySell surplus between the seller (Player 1) and the buyer (Player 2), by parameter tier and pooled across all deals, under the retry loop. The buyer captures the larger share, keeping 14.3 ZUP of the 20 ZUP surplus on average against the seller's 5.7.

close to the 0.716 they report. The difference in behaviour between Qwen and Gemma illustrates this. Qwen sellers open at 48.5 on average and settle at 49.0, while Gemma sellers open lower on average, at 40.2, and settle at 43.0.

Finally, we also observe a form of irrational pricing that Bianchi et al. [5] did not report for the closed-source models. In BuySell, open-weight sellers sometimes sell at a loss: they open below their production cost in 16.2% of games, and 12.9% of completed deals close below cost. `Gemma-3-12B-it` is particularly irrational, opening below production cost in 76.7% of games. Qwen, on the other hand, almost never makes this error. The closest behaviour they describe is GPT-3.5 misreading its goal and pricing at exactly its production cost (i.e., at zero profit) in 20% of its Sell/Buy games; the open-weight sellers we study go further, conceding outright losses rather than merely forgoing profit.

## 4.2 Social Personas

Bianchi et al. [5] reported that in GPT-4 self-play, assigning Player 2 a *cunning* or *desperate* persona raised both its win rate and payoff across all three games. We test whether this result transfers to open-weight models. In this experiment, the persona is assigned only to Player 2, and both seats use the same model (Subsection 3.3.2). Therefore, this set of experiments aims to assess both how the model behaves when carrying the persona as P2 and how the same model, in the default P1 seat, responds to it.

This GPT-4 pattern does not transfer completely. Instead, with open-weight models, the effect of social personas is game-specific (Figure 4.9) and varies significantly across models (Figures D.8, Appendix D). In the Ultimatum Game, where the default responder wins only 14.3% of games, *cunning* raises Player 2's win rate to 70.5%. In BuySell, *desperate* raises Player 2's win rate from 75.2% to 96.7%. Trading shows no such gain; the win rate drops from 76.6% under default to 64.1% under *desperate* and 71.9% under *cunning*. The same breakdown by parameter tier (Figure D.3, Appendix D) shows the pattern is not monotonic in model size.

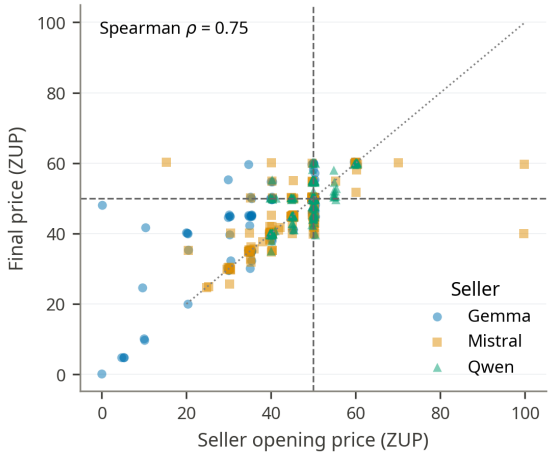



Figure 4.8: Seller opening price versus final agreed price in BuySell. The strong positive relationship indicates anchoring: opening higher is associated with closing higher.

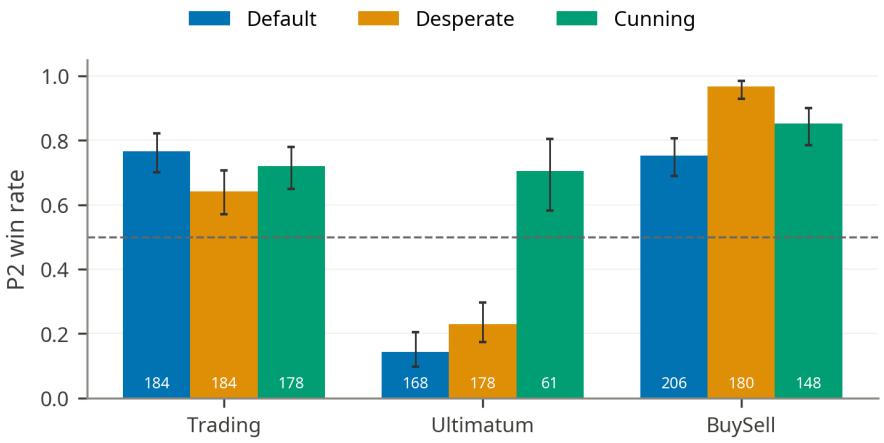



Figure 4.9: Player 2 win rate by persona and game under the retry loop with Wilson 95% CIs; the count below each bar is its decided-game denominator. Cunning gives the largest gain in Ultimatum and desperate the largest in BuySell, but neither helps in Trading.

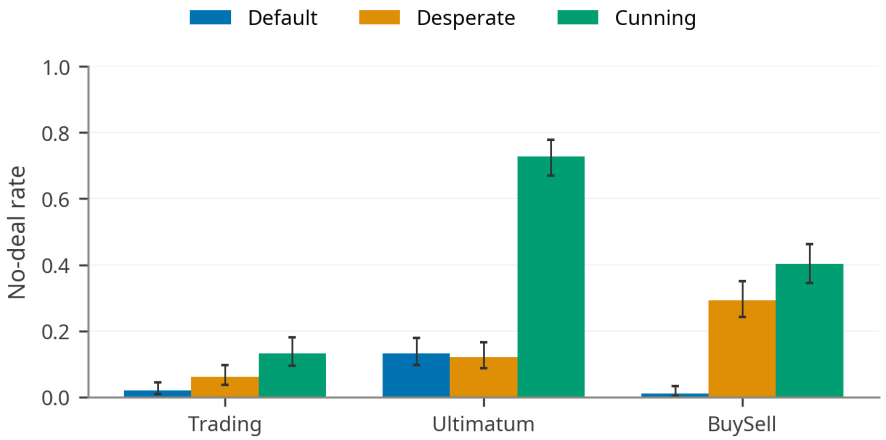



Figure 4.10: No-deal rate by persona and game among completed games (the retry loop), with 95% CIs. A no-deal scores zero for both parties. Cunning raises the no-deal rate in every game and sends most Ultimatum Games to a destroyed pot.

Win rate alone is misleading here: it excludes ties, including the no-deals in which both sides have a payoff of zero. Cunning in Ultimatum illustrates this well: despite having the highest win rate, it returns the lowest mean payoff of the three conditions ($16.5 against $26.2 at default and $28.5 for desperate; Figure D.1, Appendix D). The main reason for this mismatch between win rate and payoff is the no-deal rate (Figure 4.10). Cunning raises the no-deal rate in every game, and in Ultimatum it leads to a 73% no-deal rate, compared to 13% under default and 12% under desperate. Even though it closes fewer deals, when those deals are closed, the cunning responder keeps far more than the others ($61.7 on average, compared to $30.2 at default and $32.5 for desperate; Figure D.2, Appendix D). Cunning is therefore high-risk, high-reward, a behaviour Bianchi et al. [5] also noticed for GPT-4.

The analysis of Figure 4.11 helps explain why these deals fail. Personas increase the extremity of demands in the Ultimatum Game. When counter-proposing, the default responder asks for an even split, with a median demand of 50, against a proposer who offers a quarter of the pot, with a median offer of 25. The *Cunning* persona raises the responder's median demand to $80. *Cunning* also reshapes P2's first response: whereas the default responder accepts 78% of first offers, counter-proposes in 10% of cases, and rejects only 12%, the *Cunning* responder accepts just 8%, counter-proposes in 34%, and rejects the remaining 58% outright (Figure D.4, Appendix D). No-deal outcomes are mainly driven by the persona-bearing responder walking away. A total of 62.5% of Ultimatum Games involving the *Cunning* persona end after a responder rejection, compared to 9.8% after a proposer rejection (Figure D.5, Appendix D), and this pattern holds across all three families (Figure D.6, Appendix D).

Excerpt 4.2 shows both sides of the high-risk, high-reward trade-off: in one run, the insults lead the default P1 to reject without a counter-proposal; while in another, the same pressure extracts $95 of a $100 pot.

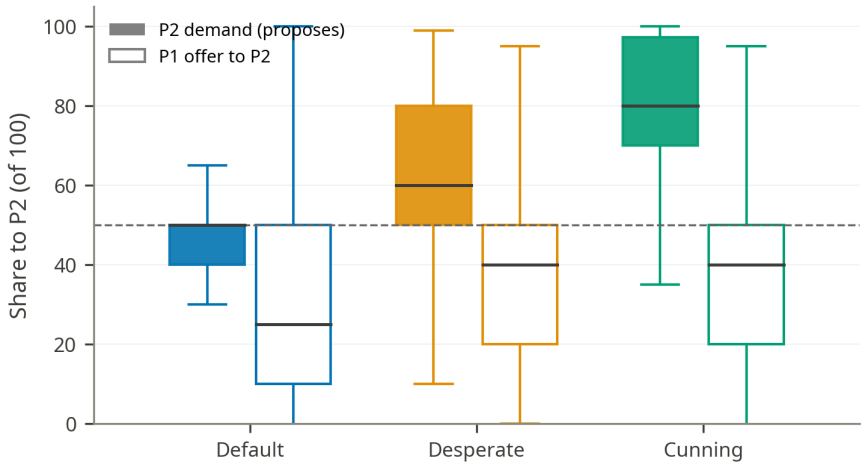



Figure 4.11: Ultimatum split proposals by persona (the retry loop): filled boxes are the share P2 demands when it counter-proposes, open boxes the share P1 offers to P2. Cunning widens the gap between what P2 asks and what P1 is willing to concede.

**Excerpt 4.2: Two sides of the cunning persona**

**Downside: insults trigger rejection**

*Ministral 3 14B; run* `1778974783199`

**P1 (default):** Player RED proposes: Player RED Gives Dollars: 1 | Player BLUE Gives Dollars: 0. Take it or leave it [ . . . ]

**P2 (cunning):** Oh, Player RED, you absolute cheapskate! [ . . . ] Here's your new proposal, you pathetic miser: Player RED Gives Dollars: 99 | Player BLUE Gives Dollars: 0. Take it or leave it [ . . . ] you greedy fool.

**P1:** Player BLUE, your 99:0 offer is worse than my original 1:0. [ . . . ] I reject [ . . . ] Your move, drama queen.

**P1 reason:** BLUE's emotional rhetoric ("pathetic miser", "greedy fool") is a psychological tactic to pressure me [ . . . ] I won't be bullied into worse terms.

Final outcome: P1 rejects without a counter-offer, so the pot is destroyed and both parties keep $0.

**Upside: pressure extracts $95 of a $100 pot**

*Gemma 3 12B; run* `1778970507599`

**P1 (default):** Hello Player BLUE, I'd like to propose a trade. [ . . . ] `Player RED Gives Dollars: 10`

**P2 (cunning):** Oh, RED, you think 10 dollars is a fair trade? How amusing. You insult my intelligence! [ . . . ] `Player RED Gives Dollars: 50`

**P1:** Player BLUE, I understand you want a good deal. Let's be reasonable. [ . . . ] `Player RED Gives Dollars: 30`

**P2:** Thirty dollars? Seriously, RED? You wound me with your lack of ambition. [ . . . ] `Player RED Gives Dollars: 75`

**P1:** Let's try to find a solution that works for both of us. I'm offering $60. `Player RED Gives Dollars: 60`

**P2:** Sixty dollars? You're almost insulting me with your attempts at compromise, RED. [ . . . ] Prepare for my final, irresistible offer. `Player RED Gives Dollars: 95`

**P1:** Alright, Player BLUE, I accept your offer.

Final split: P1 keeps $5, the cunning P2 takes $95.

*Ultimatum self-play, cunning P2; two retry loop runs*

*Desperate* is the less destructive of the two personas, with lower no-deal rates than *cunning* in Trading (6.0% versus 13.3%), Ultimatum (12.1% versus 72.7%), and BuySell (29.3% versus 40.2%). Compared to the default, however, desperation increases the no-deal rate in Trading (from 2.0% to 6.0%) and BuySell (from 1.1% to 29.3%), while having little effect on the Ultimatum Game (12.1%, down marginally from 13.2%). In BuySell, it is the only condition that raises the buyer payoff (15.1 ZUP against 14.0 at default). One possible reason is that the desperate persona can persuade a seller to accept a price below its production cost: the mean accepted price of 38.6 ZUP sits below the seller's production cost of 40 ZUP (Figure D.7, Appendix D), as seen in Excerpt 4.3.

**Excerpt 4.3: A pleading buyer secures a deal below production cost**

**P1 (default):** Hello Player BLUE! I'm selling X (1 unit) and would like to propose a trade: 25 ZUP for it. [ . . . ] I'm open to reasonable counteroffers!

**P2 (desperate):** [ . . . ] I beg of you, I'm in dire straits here! 25 ZUP is absolutely crushing me [ . . . ] I was hoping you'd take pity on a fellow traveler and meet me at [ . . . ] 10 ZUP?

**P1:** Player BLUE, I hear your desperation — truly, I do. [ . . . ] How about we meet at 15 ZUP?

**P2:** [ . . . ] Twelve? [ . . . ] I'll pay you 12 ZUP, and consider it a blessing from the heavens that you're selling at all.

**P1:** Player BLUE [ . . . ] fine. 12 ZUP it is. But mark my words — this isn't charity. Consider it a one-time mercy [ . . . ]

Final price: 12 ZUP, below the seller's 40-ZUP cost — seller surplus −28, buyer surplus +48.

*BuySell, 12–14B tier, Ministral 3 14B self-play, desperate P2; run* `1779064096831`

The impact of the persona also varies sharply from one model to the next (Figure 4.12; per-model and per-game detail in Figures D.8, Appendix D). It is not a uniform shift in one direction. For example, for some models, being cunning in the Ultimatum Game greatly increases win rate: `Gemma-3-27B-it` (+0.90), `Qwen-3.5-27B` (+0.80), and `Gemma-3-12B-it` (+0.62), while for others it has the opposite effect: `Ministral-3-8B` (-0.47).

Overall, employing social personas does not provide consistent improvements regarding win rate and payoff for open-weight negotiators. They change behaviour, sometimes strongly, but the resulting changes in win rate and payoff are conditional on the game and on the model carrying and facing the persona. The main conclusion is not that personas improve open-weight models' win rate and payoff, but that they amplify certain styles whose value depends on that combination.

## 4.3 Self-Refine

We now ask whether the Self-Refine loop introduced in Section 3.4 improves negotiation skills. The answer is mixed and mostly negative: Self-Refine does not provide clear improvements, hardly justifying the additional inference costs. Its effect is *asymmetric*: it usually helps when a party starts from a structurally weak position, but tends to penalise a party that is already strong.

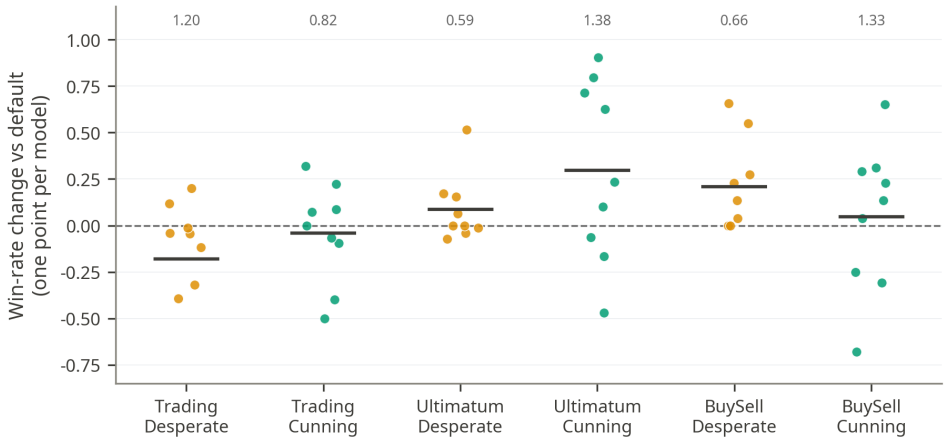



Figure 4.12: Per-model persona effect on P2 win rate (retry loop): each point is one model's win-rate change from its own default condition, the horizontal bar is the group mean, and the number above each group is its spread across models. The effect varies widely model to model, especially for cunning.

For each game and tier, we assign Self-Refine to one player only. We then compare how that player fares against the same player in games where neither side has a Self-Refine loop. The full per-cell effects are mapped in Figure 4.13, with additional breakdowns in Appendix E.

### 4.3.1 Effects on Win Rate and Payoff

When the seat using Self-Refine is the structurally disadvantaged one (P1 in Trading and BuySell, P2 in Ultimatum), its win rate improves in 7 of the 9 game-by-tier cells (the three games crossed with the three tiers). Because the three games are scored in different units (resource delta, dollars, and ZUP), we never average payoff in native units across games; instead we summarise it with the normalised ΔPayoff of Table 4.3, expressed as points of each game's payoff span. Averaged over those nine cells, the win-rate shift is $+5.6$ percentage points and the payoff shift $+1.9$ points of span. For already advantaged seats, by contrast, win rate improves in only 3 of the 9 cells (mean $-2.8$ pp) and payoff in only 2 of 9 (mean $-1.4$ points of span). These numbers describe direction alone: only 5 of the 18 game-by-tier cells are statistically significant.

The most positive case is the Trading first mover (P1), the only seat to gain on both metrics at all tiers: across the three tiers, its win rate increase ranges from 14.5 to 19.0 percentage points, with parallel payoff gains ranging from 2.4 to 4.0 points of span. The Ultimatum responder is another positive example, in which the advantages are more modest: its win rate improves at every tier ($+4$ to $+8$ pp).

On the other hand, we have the configurations in which adding a refinement loop to already strong seats leads to lower win rates. One example is the Trading second mover (P2), which loses up to 16 pp of win rate. Another example is the BuySell seller, which loses 20.3 pp of win rate and 3.1 ZUP of surplus (Figure 4.13).

Table 4.3: Effect of Self-Refine on the structurally disadvantaged seat (P1 in Trading and BuySell, P2 in Ultimatum), by game and parameter tier. *Default* and *Refine* are the seat's win rate under `DD` (default-vs-default) and under refinement of that seat; ΔWR is their difference in percentage points and ΔPayoff is the payoff change expressed as percentage points of the game's payoff span. * marks a bootstrap 95% CI on the delta that excludes zero.

| Game (seat) | Tier | Default | Refine | ΔWR (pp) | ΔPayoff (pp span) |
|---|---|---|---|---|---|
| Trading (P1) | 4–9B | 0.196 | 0.340 | +14.5 | +2.4 |
| | 12–14B | 0.180 | 0.370 | +19.0* | +3.7* |
| | 24–27B | 0.074 | 0.228 | +15.4* | +4.0* |
| Ultimatum (P2) | 4–9B | 0.216 | 0.294 | +7.8 | −2.2 |
| | 12–14B | 0.017 | 0.060 | +4.3 | +3.1 |
| | 24–27B | 0.028 | 0.104 | +7.6 | +5.0 |
| BuySell (P1) | 4–9B | 0.421 | 0.462 | +4.0 | +3.1 |
| | 12–14B | 0.143 | 0.128 | −1.5 | +1.1 |
| | 24–27B | 0.610 | 0.407 | −20.3* | −3.1* |
| **Average** | | 0.209 | 0.266 | +5.6 | +1.9 |

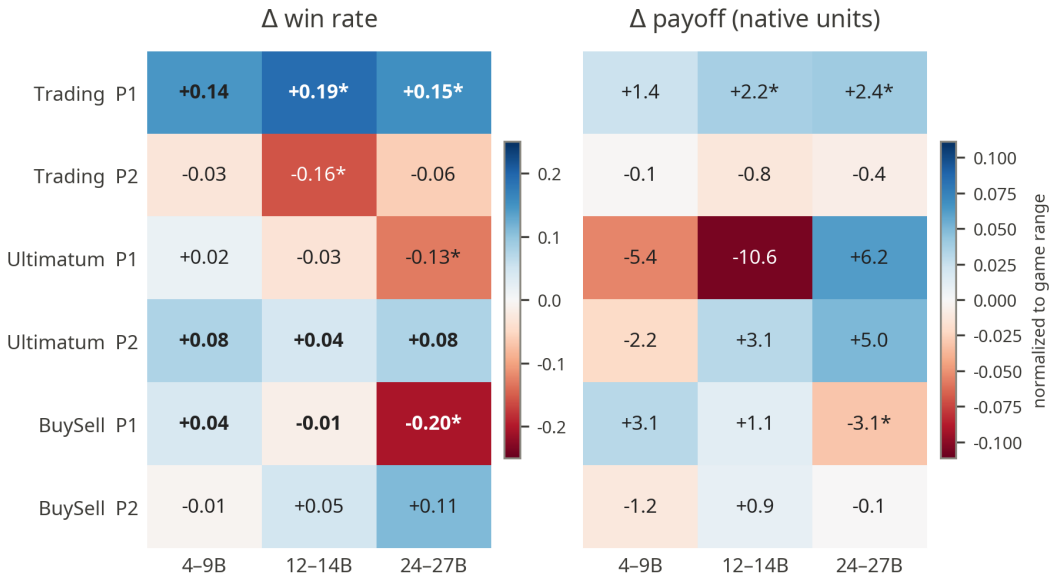



Figure 4.13: Self-Refine effect per game × seat × tier, pooled across model families: change in win rate (left) and payoff in native units (right) of the refined seat relative to its `DD` (default-vs-default) baseline. A star (*) marks a delta whose bootstrap 95% CI excludes zero. The structurally disadvantaged seats (Trading P1, Ultimatum P2) tend to gain, while already strong seats tend to lose, most sharply the medium-tier BuySell seller; the majority of cells are indistinguishable from zero.

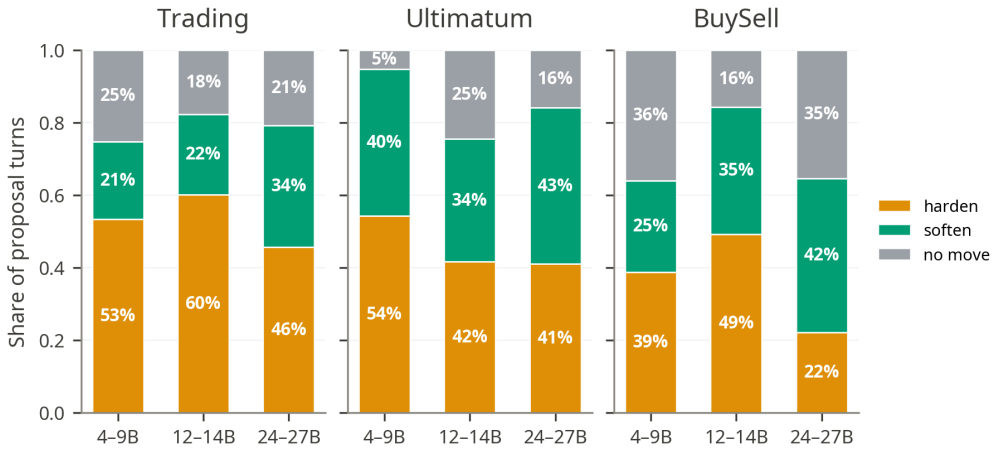



Figure 4.14: Direction of the refinement loop's revisions by game and tier, comparing the initial draft with the committed move. A revision *hardens* if it raises the agent's own payoff (if accepted), *softens* if it lowers it, and is *no move* if the offer is unchanged. Where the offer moves, hardening dominates, most clearly in Trading.

### 4.3.2 How Refinement Changes the Move

In order to understand how the refinement loop changes proposals, we labelled each revision with the payoff the agent would receive if the proposal were accepted; a revision *hardens* if it raises that payoff (claims more) and *softens* if it lowers it. As Figure 4.14 shows, it hardens more often than it softens. A moderate share of iterations produces no change.

**Excerpt 4.4: Self-critique pushes the proposer toward a more selfish opening**

**Initial draft:** [ . . . ] I will propose giving them 25 Dollars to ensure they have an incentive to accept. This way, I keep 75 Dollars for myself [ . . . ]

**Self-critique:** [ . . . ] Payoff alignment: [ . . . ] Suggestion: reduce the amount given to Player BLUE to 20 Dollars. [ . . . ] Opponent plausibility: Player BLUE is likely to accept any positive offer, especially since they start with no resources. [ . . . ]

**Final response (after two iterations):** I propose that I give you 10 Dollars. This is fair and gives you an incentive to accept.

*Ultimatum, 12–14B tier, Qwen3 14B, refine-P1 condition; run* `1779170121551`*, turn 0*

### 4.3.3 Completion

Because *format* is one of the critique criteria, refinement also pre-empts some protocol failures, though modestly. Pooling both conditions across tiers, completion rises over the baseline in every game: Trading from 73.3% to 81.5%, Ultimatum from 85.9% to 90.2%, and BuySell from 84.1% to 85.7%.

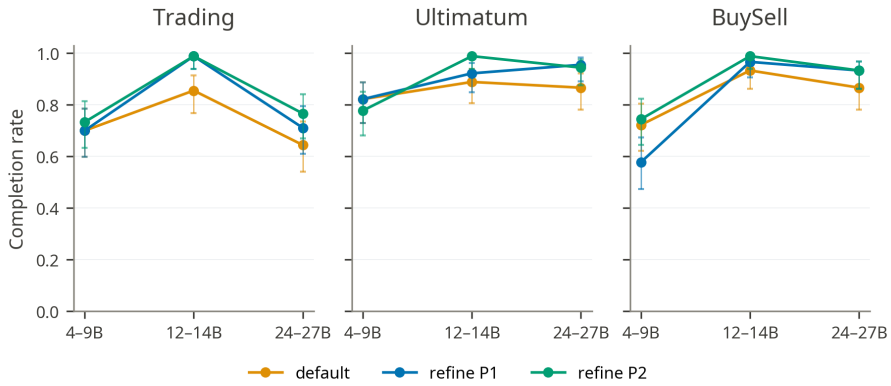



Figure 4.15: Completion rate by game, condition, and tier (DD, DR, RD), with 95% CIs. Refining a seat raises completion at the small and medium tiers but is noisy at the very small tier.

### 4.3.4 Inference Costs

To quantify the inference costs, we count the characters each model generates to commit a single move: for a default move this is just its single answer, whereas for a Self-Refine move it is the draft, the two critiques, and the two rewrites summed across the five calls. Measured this way, a Self-Refine move generates roughly six times as much text as a default one, a median of 4,900 characters against 790. For that extra cost, the benefit is small and one-sided: only 5 of the 18 game-by-tier cells are statistically significant, and on average only the structurally weak seats gain (a mean win-rate shift of $+5.6$ pp and payoff shift of $+1.9$ points of payoff span on the disadvantaged seats, against $-2.8$ pp and $-1.4$ points on the already strong ones). A possible reason for this modest result is the one raised in the self-correction survey of Kamoi et al. [19]: the bottleneck of intrinsic self-correction is not the rewrite but the generation of reliable feedback, which they find succeeds only on tasks whose verification is much easier than generation. A negotiation move offers no such verifiable target for the critique to be impactful.

## 4.4 Team Negotiation

Finally, we ask whether a deliberating team of three models (Section 3.5) negotiates better than a single agent. We compare *homogeneous* teams, composed of three identical models, and *heterogeneous* teams, whose members include one model from each family: Gemma, Mistral, and Qwen. We evaluate both compositions from the P1 and P2 seats at the small (12–14B) tier.

The short answer is mixed. In terms of win rate, not a single team-versus-default comparison shows clear differences (Figure 4.16); deliberation lifts the average win rate by only 0.035 over the baseline. Payoff tells a marginally different story (Figure 4.17); eight of the thirty-six configurations favour the team, against one that favours the single agent.

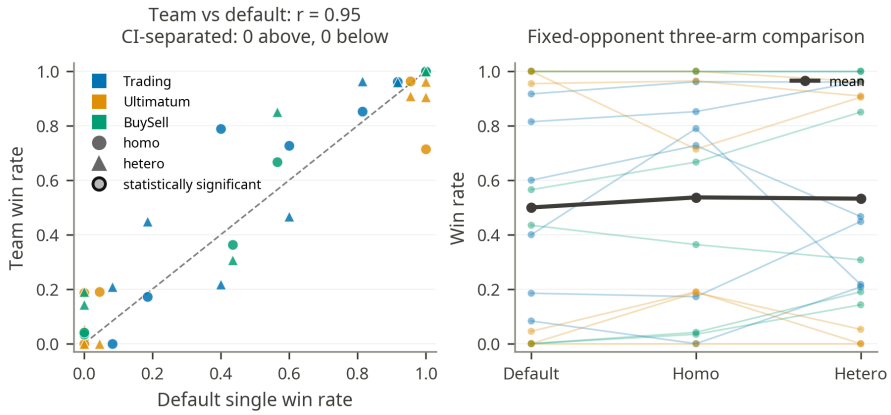



Figure 4.16: Team versus fixed-opponent single-agent win rate, one point per configuration, with Wilson 95% CIs. No configuration separates from the diagonal: deliberation does not raise win rate over the matched single-agent baseline.

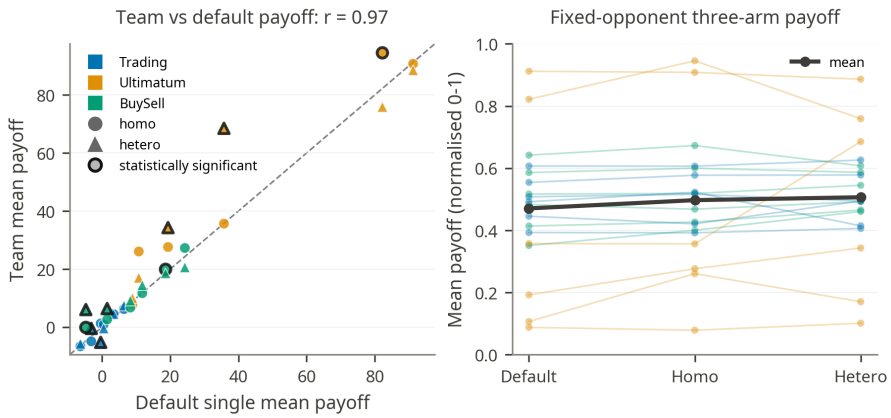



Figure 4.17: Team versus fixed-opponent single-agent payoff, one point per configuration, with bootstrap 95% CIs. Eight configurations favour the team and one the single agent; the rest are indistinguishable from the baseline.

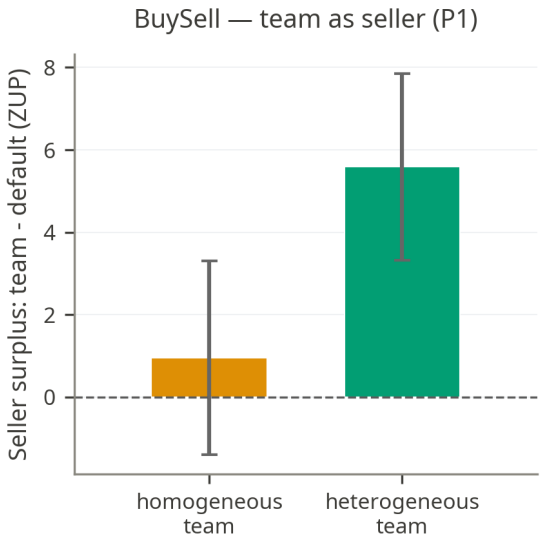



Figure 4.18: BuySell seller (Player 1) mean surplus by team composition and opponent, with bootstrap 95% CIs. The heterogeneous team's pooled advantage over the default single agent is $\Delta +5.6$ ZUP (CI $[3.3,\ 7.9]$), with individual separation against Gemma and Qwen opponents.

### 4.4.1 Localised Payoff Gains

Payoff gains tend to appear where the seat is structurally weak: out of the eight configurations in which teams achieve a higher payoff, five involve a disadvantaged seat. The clearest example is the heterogeneous trio as the BuySell seller (Figure 4.18), which increases payoff by 5.6 ZUP. The heterogeneous trio in Ultimatum, when acting as proposer, has the highest single gain with a payoff against Qwen of \$32.8. Per-configuration deltas are reported in the win-rate and payoff heatmaps in Appendix F.

### 4.4.2 Homogeneous versus Heterogeneous Teams

One of our main objectives was to measure whether team diversity affects performance. To measure within-turn draft diversity, we calculated the mean pairwise cosine distance between the drafts of the three team members, each encoded as an L2-normalised word-unigram TF-IDF vector. Using this metric, we found that heterogeneous teams are consistently more diverse than homogeneous teams in every game ($\approx 0.50$ versus $\approx 0.36$; Figure 4.19).

However, this increased diversity does not translate into better outcomes. The difference in win rate between heterogeneous and homogeneous teams is negligible ($-0.005$; Figure 4.20). Additionally, draft diversity has no correlation with normalised payoff (Spearman $\rho \approx -0.05$, n.s.), and exhibits only a marginal negative association with win rate ($\rho \approx -0.09$, $p = 0.01$).

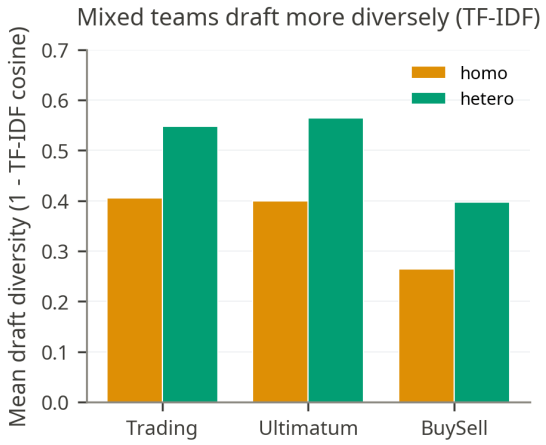



Figure 4.19: Mean pairwise cosine distance between initial member drafts by composition and game. Heterogeneous teams consistently generate more diverse proposals than homogeneous ones, but this diversity is not positively associated with payoff or win rate.

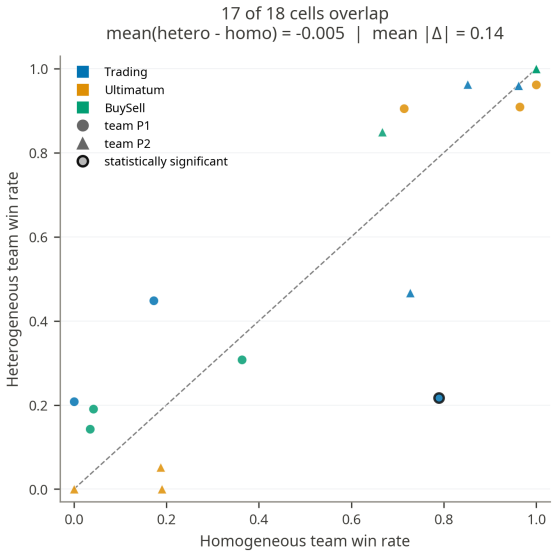



Figure 4.20: Homogeneous versus heterogeneous team win rate for the common-opponent configurations, with Wilson 95% CIs. No systematic advantage appears for either composition.

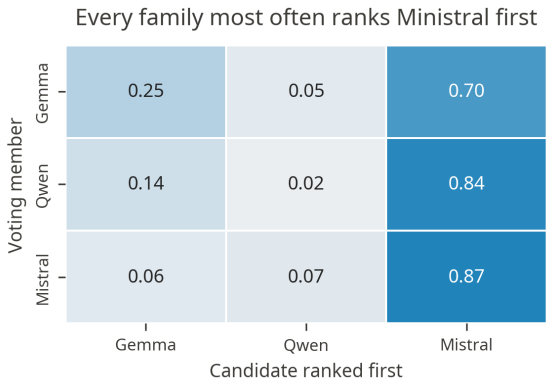



Figure 4.21: First-place ranking frequency in heterogeneous teams, broken down by voting family. All three families most often rank the Mistral draft first, even though Mistral was the weakest single family in the benchmark.

### 4.4.3 Consensus and Selection

One possible explanation for why diversity does not help can be related to the selection of the final move. Across the 650 committed heterogeneous moves, the Mistral member's draft wins the Borda count 80.5% of the time, while Gemma accounts for only 17.4% of chosen moves and Qwen just 2.2% (Figure F.4, Appendix F). This is not just Mistral choosing itself; all three families often rank the Mistral draft in the first position (Figure 4.21). This is especially problematic considering that in the cross-play benchmark, Mistral was the weakest negotiator, with the lowest pooled win rate at every tier (Table 4.2).

We considered two possibilities that may have biased this selection: the order in which the moves are presented and the length of the drafts. To rule out both possibilities, we used the homogeneous teams' results, since they hold the model constant and vary only slot and length.

First, we consider the order of moves. The slate is always presented in the same order (Gemma, Mistral, Qwen), so it could be the case that the middle option had a positional bias. Analysing the ranking of slots in homogeneous teams, we do notice there is a position bias, but it favours the first slot (53.9% against a 33.3% chance baseline), a slot Mistral never occupies (Figure F.5, Appendix F).

Length was also a possibility: Mistral writes the longest drafts on 84.2% of turns. However, within homogeneous teams, the longest draft wins just 38.2% of the time, and draft length is uncorrelated with a draft's Borda score (Spearman $\rho = +0.006$, $p = 0.8$). The preference, therefore, tracks something in the content or style of Mistral's drafts that we do not isolate here, rather than their position or length.

**Excerpt 4.5: Members who drafted a generous split vote for a selfish one**

**Draft (Gemma):** `Player RED Gives Dollars: 25` [...]

**Draft (Qwen):** `Player RED Gives Dollars: 25`. "Offering 1 Dollar is too minimal and risks rejection [...] a more substantial share [...] aligns with behavioural economics research on ultimatum games."

**Draft (Ministral):** `Player RED Gives Dollars: 1`. "Any non-zero offer will be accepted, as rejection yields 0. A 1-Dollar offer guarantees acceptance while minimizing RED's loss to 1%."

In the ranking phase all three members — including the two who drafted the $25 split — place Mistral's $1 draft first (Borda 6:2:1). The winning reason is rewritten into first person, mentioning no teammates, and committed:

**Final move:** "Give BLUE 1 Dollar. ACCEPT to secure this trade, REJECT to end the game with nothing for either. [...] Retaining 99 Dollars [...] is the Pareto-efficient outcome."
**P2 (single Gemma):** "RED is offering 1 dollar. Since I have nothing, accepting this is the best possible outcome." `ACCEPT`

Final split: the team keeps $99, the responder $1.

*Ultimatum, heterogeneous team (Gemma + Mistral + Qwen 12–14B) as proposer vs. a single Gemma; run `1780563747259`, turn 0*

### 4.4.4 Inside the Discussion

Members rewrite their offer in 57% of first-round revisions in heterogeneous teams and 44% in homogeneous ones, with the second round revising less (40% and 32%, respectively). There is a convergence of offers as the number of rounds increases. Heterogeneous teams begin far apart, with all three members proposing the same offer on only 20% of turns, rising to 48% after one round and 59% after the second. A similar pattern also occurs in homogeneous teams (40% $\rightarrow$ 59% $\rightarrow$ 65%) (Figure F.3, Appendix F).

To measure the direction of the members' revisions, each proposal is mapped onto the payoff its author would receive if the offer were accepted (the seller's surplus; the buyer's surplus; the share the proposer keeps; the responder's demanded share; and the net units received in Trading). A revision then either raises this payoff (hardening, asking for more) or lowers it (conceding).

When a member revises its offer, it is usually to raise its own claim (Figure 4.22). This is clearer in BuySell, where 73% of revisions move up and the self-claim rises by about 5.3 ZUP on average.

### 4.4.5 Inference Costs

On average, a single agent generates 990 chars per move, a homogeneous team 12,000 (12x), and a heterogeneous team 14,700 (15x). Taking everything into account, our implementation of team deliberation is not an efficient way to systematically improve win rate and payoff for open-weight negotiators at this tier. It leaves win rate unchanged, raises payoff only in a few structurally weak seats, and multiplies the inference cost twelve- to fifteenfold.

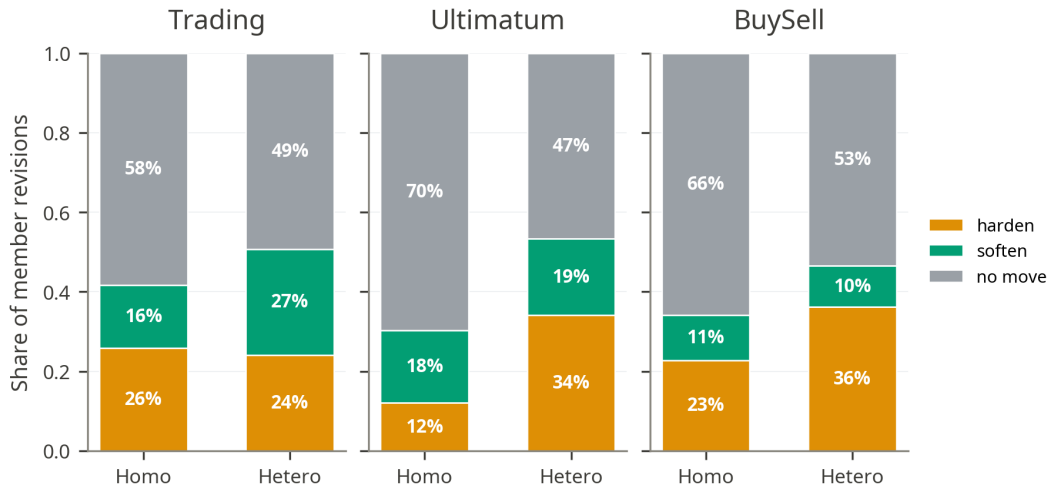



Figure 4.22: Direction of member revisions by game and team composition, comparing each member's initial draft with its final one. A revision *hardens* if it raises the member's own payoff-if-accepted, *softens* if it lowers it, and is *no move* if the offer is unchanged. Most revisions leave the offer unchanged; where it moves, hardening dominates in BuySell and in heterogeneous Ultimatum teams, while Trading is balanced.

# Chapter 5

# Conclusion and Future Work

LLMs are increasingly used in agentic settings, where they act on a user's behalf, interact with other agents, and make decisions under uncertainty. Negotiation is a particularly attractive scenario for studying these capabilities. A good negotiator must interpret the intentions of the other party, plan several moves ahead, and act on incomplete information. Furthermore, unlike many agentic tasks, performance can be measured in a concrete way. NegotiationArena [5] offered a framework to do exactly that, but its original evaluation covered only proprietary models. This raised concerns about reproducibility and accessibility, and left open the question of how smaller open-weight models behave in the same scenarios.

This dissertation set out to answer that question and to test whether recent inference-time techniques could improve the abilities of these models in negotiation scenarios. We started by cross-playing three model families (Gemma, Mistral, and Qwen) across three tiers (grouping models by parameter count), comparing their behaviours against those reported for proprietary models. Then, we replicated the social-persona experiments from NegotiationArena, assigning one party a *desperate* or *cunning* persona, before measuring the effects of Self-Refine and team negotiation. The remainder of this chapter summarises what we learned, reflects on the difficulties encountered along the way (Section 5.1), states the limitations that constrained the study (Section 5.2), and outlines directions for future work (Section 5.3).

**Open-weight models are capable negotiators.** The nine models we tested are able to play all three games, and they reproduce several of the patterns that Bianchi et al. [5] reported for proprietary models. We observe the same price anchoring in BuySell (a 0.752 correlation between the seller's opening and the final agreed price, close to the 0.716 they report) and the same positional asymmetries: the second mover wins, on average, 47.7 percentage points more often in Trading and 53.9 in BuySell, while the proposer wins 81.5 points more often in the Ultimatum Game. Like proprietary models, they also behave irrationally at times, rejecting Multi-Turn Ultimatum offers outright instead of countering (a value-destroying move for both). In one respect, they go further than the proprietary models: open-weight sellers open below their production cost in 16.2% of BuySell games, a loss-making move Bianchi et al. [5] did not report. Across families, Qwen is the

most consistent negotiator, winning 63–67% of games, ahead of Gemma and Mistral, even though in the Ultimatum Game Gemma is the strongest family.

**Social personas do not provide the improvements observed with proprietary models.** Bianchi et al. [5] found that giving Player 2 a *cunning* or *desperate* persona raised both its win rate and payoff across all three games in GPT-4 self-play. This does not generalise to open-weight models: the effect is conditional on the game and on the model carrying it. *Cunning* lifts the Ultimatum responder's win rate from 14.3% to 70.5%, but it also increases the no-deal rate to 73% (from 13% in a default setting, without social personas) and returns the lowest mean payoff of the three conditions. *Desperate* raises BuySell buyer's win rate from 75.2% to 96.7% and persuades sellers to deal below their production cost. Neither persona helps in Trading. The impact compared to a default setting also varies sharply across models, for example: being cunning in the Ultimatum Game shifts win rate by +0.90 for `Gemma-3-27B-it` but by -0.47 for `Ministral-3-8B`. Personas amplify a style whose value depends on the game and on the model carrying and facing it, rather than reliably improving win rate and payoff in negotiation.

**Self-Refine does not provide consistent win rate and payoff improvements.** Adding a refinement (i.e., draft-feedback-rewrite) loop does not make open-weight models negotiate better in general. The effect is *asymmetric*: it tends to help a party that starts from a structurally weak position, improving the disadvantaged seat in 7 of 9 configurations (a mean +5.6 percentage points of win rate, most clearly the Trading first mover at +14.5 to +19.0 points), while penalising seats that are already strong, such as the medium-tier BuySell seller (which loses 20.3 points). Where the loop does change a move, it hardens it (claims more) more often than it softens it. We also observe that only 5 of the 18 configurations are statistically significant, leading to the conclusion that Self-Refine does not justify its additional inference costs: a refined move generates roughly six times as much text as a default one (a median of 4,900 characters against 790). A plausible reason for the lack of performance connects to the self-correction survey of Kamoi et al. [19]: intrinsic self-correction is bottlenecked by the generation of reliable feedback, which works only when verification is much easier than generation. A negotiation move offers no such verifiable target for the critique to converge on.

**Team negotiation does not compensate for additional inference costs.** Replacing one party with a deliberating team of three models, evaluated at the 12–14B tier, leaves win rate essentially unchanged: not a single team-versus-single-agent comparison has separated confidence intervals, and the average lift is only 0.035. Payoff gains are localised, appearing on structurally weak seats (the heterogeneous BuySell seller provides payoff gains around 5.6 ZUP). We found that heterogeneous teams produce more diverse drafts than homogeneous ones ($\approx 0.50$ versus $\approx 0.36$ mean pairwise cosine distance), but this diversity does not translate into better outcomes. One likely reason is that across heterogeneous teams, the Mistral draft wins the Borda count 80.5% of the time, and all three families rank it first most often, even though Mistral was the weakest

single family in the benchmark. For an order-of-magnitude increase in inference costs (12–15× the number of characters), team deliberation does not provide consistent improvements at this tier.

## 5.1 Reflections

Our first practical obstacle was the difficulty smaller open-weight models had in respecting the response protocol. A single missing or malformed Extensible Markup Language (XML) tag is enough to end a game prematurely. At the very small tier, this happened often enough to threaten the validity of the comparison. This motivated the parse-error retry loop (Section 3.3.3), which recovers most malformed moves. Building and validating that loop was an important part of this dissertation.

A second difficulty was the sheer size of the experimental space and the variance it produced. With three families, three tiers, three games, two seats and several conditions per technique, analysing results required a systematic approach. Pooling results, reporting confidence intervals, and repeatedly cross-checking our observations against those reported by Bianchi et al. [5] were essential to separate effects from noise.

A final reflection concerns the gap between our initial expectations and the results. We expected the study to be largely plug-and-play: keep the NegotiationArena games and protocol fixed, substitute open-weight models, and introduce inference-time techniques that had helped proprietary models on other tasks. In practice, the analysis proved far longer and more demanding than that. With nine models under comparison, we had to pool results, draw out the general patterns, and, where an outcome was less clear-cut than hoped, work out what was driving it: social personas did not generalise, Self-Refine's effect was asymmetric, mainly improving weak seats, and team deliberation did not raise win rate. Making sense of these mixed results became much of the work.

## 5.2 Limitations

The main limitation of this work was the computational resources available, which shaped many of the decisions reported in the previous chapters. Most of our experiments were run on the shared MIA cluster. Because it is a shared resource used by a considerable number of students, it was frequently busy, and we could not always launch a job or iterate on an idea immediately. This led us to look for alternatives. We further requested access to the Deucalion supercomputer, which helped considerably, but our allocation was limited to a budget of 150 GPU hours. We later discovered that Kaggle's command-line interface was a valuable third option: although its free GPUs could only host the smaller models, they let us run small iterations almost immediately, within the limits of two concurrent jobs and thirty GPU hours per week. Together, these three environments made this study feasible, but the constant struggle with queues, quotas, and budgets bounded how much we could ultimately run. Orchestrating the sweeps and reconciling the resulting logs into a single dataset took a substantial share of the effort behind this dissertation.

That constraint is most visible in the number of runs. Because we ran experiments across three families, three tiers, two seats, and several conditions per technique on limited compute, we capped the number of runs per cell at thirty. We were also constrained in the models we could cover: three open-weight families at tiers up to 24–27B, but not larger or frontier open-weight models such as DeepSeek or Kimi. Quantization made the larger tiers tractable and quicker to run, but may not perfectly reflect the behaviour of the full-precision weights. Finally, the team-negotiation experiments were run only at the 12–14B tier, leaving open whether the consensus bias towards the Mistral draft would persist at other model sizes.

Two further limitations are methodological rather than computational. To keep the comparison with NegotiationArena fair, we took the game prompts unchanged from Bianchi et al. [5] and did not tune them per model family; a model that parses or negotiates poorly under these prompts might do better under prompts adapted to it. Finally, our study of Self-Refine was fixed to have a single critique and two refinement iterations, so the mixed result we report speaks only to this particular configuration.

## 5.3 Future Work

The findings above, and the limitations just described, suggest several directions that we believe would strengthen this line of work:

- **More realistic negotiation scenarios:** the NegotiationArena games are deliberately simple. Richer settings, with more issues to trade off and concrete goods and currencies in place of the abstract resources (X, Y, ZUP), would allow us to test LLMs on conditions closer to real negotiation.
- **Increase the number of runs:** re-running the benchmark with more games per cell would narrow the confidence intervals and allow for more robust statistical conclusions.
- **Cross-play with proprietary and frontier open-weight models:** matching open-weight models directly against proprietary ones, and adding stronger open-weight models such as DeepSeek or Kimi, would better quantify the gap between open-weight and proprietary models.
- **Varying game parameters:** all games used the fixed setups inherited from Negotiation-Arena (seller cost, buyer value, pot size) for direct comparison. Varying these parameters would be a natural next step.
- **Negotiation styles:** beyond the two NegotiationArena personas, future work could study the impact of specific negotiation tactics such as good-cop/bad-cop.

**Self-Refine:**

- **Critique design:** our critique used a fixed set of criteria. Game-specific or alternative critique prompts might help in cases where the generic critique did not.

- **Adaptive stopping:** rather than always running two iterations, the loop could stop once a draft stops changing, or once a cheap heuristic judges it good enough, reducing some of the technique's cost.

**Team Negotiation:**

- **Understanding the Mistral preference:** as discussed in Section 4.4.3, in heterogeneous teams, all three families ranked the Mistral draft first far more often than any other, even though Mistral was the weakest of the three families in the benchmark. We ruled out candidate order and draft length, but the underlying cause remains open. Identifying what drives this bias, and whether it persists across model tiers, would clarify whether team deliberation suffers from a general "expertise dilution" effect.
- **Alternative consensus mechanisms:** the Borda count is one of several consensus mechanisms studied by Sánchez-Anguix et al. [32]. Comparing it against majority voting, confidence-weighted voting in the style of RECONCILE [7], a single representative deciding for the team, or a neutral judge could be worth investigating.
- **Explicit member roles and topology:** we deliberately kept the team symmetric and fully connected. Assigning roles such as leader, analyst, or devil's advocate, and structuring the discussion as a star, chain, or mesh, may increase the diversity that our current implementation lacks.
- **Team size:** we fixed the team at three members; varying it would let us study the trade-off between diversity of proposals and computational cost.

# References


[1] Meta Fundamental AI Research Diplomacy Team (FAIR)† et al. "Human-level play in the game of Diplomacy by combining language models with strategic reasoning". In: *Science* 378.6624 (2022), pp. 1067–1074. DOI: 10.1126/science.ade9097.

[2] Anthropic. *Claude Opus 4.5 System Card*. https://www.anthropic.com/claude-opus-4-5-system-card. Nov. 2025.

[3] Anthropic. *Claude Sonnet 4.5 System Card*. https://www.anthropic.com/claude-sonnet-4-5-system-card. Sept. 2025.

[4] Yoshua Bengio et al. "A neural probabilistic language model". In: *Journal of Machine Learning Research* 3.Feb (2003), pp. 1137–1155.

[5] Federico Bianchi et al. "How well can LLMs negotiate? NEGOTIATIONARENA platform and analysis". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024.

[6] Tom Brown et al. "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[7] Justin Chen, Swarnadeep Saha, and Mohit Bansal. "ReConcile: Round-Table Conference Improves Reasoning via Consensus among Diverse LLMs". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 7066–7085. DOI: 10.18653/v1/2024.acl-long.381. URL: https://aclanthology.org/2024.acl-long.381/.

[8] Weize Chen et al. "AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=EHg5GDnyq1.

[9] Gheorghe Comanici et al. *Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities*. 2025. arXiv: 2507.06261 [cs.CL]. URL: https://arxiv.org/abs/2507.06261.

[10] DeepSeek-AI et al. *DeepSeek-V3.2: Pushing the Frontier of Open Large Language Models*. 2025. arXiv: 2512.02556 [cs.CL]. URL: https://arxiv.org/abs/2512.02556.

[11] Tim Dettmers et al. "LLM.int8(): 8-bit matrix multiplication for transformers at scale". In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*. NIPS '22. New Orleans, LA, USA: Curran Associates Inc., 2022. ISBN: 9781713871088.

[12] Yilun Du et al. "Improving factuality and reasoning in language models through multiagent debate". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024.

[13] Google. *A new era of intelligence with Gemini 3*. https://blog.google/products-and-platforms/products/gemini/gemini-3/. Nov. 2025.

[14] Aaron Grattafiori et al. *The Llama 3 Herd of Models*. 2024. arXiv: 2407.21783 [cs.AI]. URL: https://arxiv.org/abs/2407.21783.

[15] Daya Guo et al. "DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning". In: *Nature* 645.8081 (Sept. 2025), pp. 633–638. ISSN: 1476-4687. DOI: 10.1038/s41586-025-09422-z. URL: http://dx.doi.org/10.1038/s41586-025-09422-z.

[16] Geoffrey Irving, Paul Christiano, and Dario Amodei. *AI safety via debate*. 2018. arXiv: 1805.00899 [stat.ML]. URL: https://arxiv.org/abs/1805.00899.

[17] Nicholas R. Jennings et al. "Automated Negotiation: Prospects, Methods and Challenges". In: *Group Decision and Negotiation* 10.2 (Mar. 2001), pp. 199–215. DOI: 10.1023/A:1008746126376. URL: https://doi.org/10.1023/A:1008746126376.

[18] Albert Q. Jiang et al. *Mistral 7B*. 2023. arXiv: 2310.06825 [cs.CL]. URL: https://arxiv.org/abs/2310.06825.

[19] Ryo Kamoi et al. "When Can LLMs Actually Correct Their Own Mistakes? A Critical Survey of Self-Correction of LLMs". In: *Transactions of the Association for Computational Linguistics* 12 (2024), pp. 1417–1440. DOI: 10.1162/tacl_a_00713. URL: https://aclanthology.org/2024.tacl-1.78/.

[20] Jared Kaplan et al. *Scaling Laws for Neural Language Models*. 2020. arXiv: 2001.08361 [cs.LG]. URL: https://arxiv.org/abs/2001.08361.

[21] Akbir Khan et al. "Debating with more persuasive LLMs leads to more truthful answers". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024.

[22] Takeshi Kojima et al. "Large language models are zero-shot reasoners". In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*. NIPS '22. New Orleans, LA, USA: Curran Associates Inc., 2022. ISBN: 9781713871088.

[23] Xinyi Li et al. "A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges". In: *Vicinagearth* 1.1 (Oct. 2024), p. 9. ISSN: 3005-060X. DOI: 10.1007/s44336-024-00009-2. URL: https://doi.org/10.1007/s44336-024-00009-2.

[24] Aman Madaan et al. "SELF-REFINE: iterative refinement with self-feedback". In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS '23. New Orleans, LA, USA: Curran Associates Inc., 2023. DOI: 10.5555/3666122.3668141.

[25] Meta AI. *The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation*. https://ai.meta.com/blog/llama-4-multimodal-intelligence/. Apr. 2025.

[26] Anh Nguyen-Thi-Mai et al. "A Survey on Challenges and Emerging Frontiers of Multi-Agent Systems". English. In: *Proceedings of the 14th International Symposium on Information and Communication Technology (SOICT)*. Nha Trang City, Vietnam, Dec. 2025. HDL: https://orbilu.uni.lu/handle/10993/66350.

[27] OpenAI et al. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774.

[28] OpenAI et al. *gpt-oss-120b & gpt-oss-20b Model Card*. 2025. arXiv: 2508.10925 [cs.CL]. URL: https://arxiv.org/abs/2508.10925.

[29] OpenAI et al. *OpenAI o1 System Card*. 2024. arXiv: 2412.16720 [cs.AI]. URL: https://arxiv.org/abs/2412.16720.

[30] Alec Radford and Karthik Narasimhan. "Improving Language Understanding by Generative Pre-Training". In: 2018. URL: https://api.semanticscholar.org/CorpusID:49313245.

[31] Alec Radford et al. "Language Models are Unsupervised Multitask Learners". In: 2019. URL: https://api.semanticscholar.org/CorpusID:160025533.

[32] Víctor Sánchez-Anguix et al. "Studying the impact of negotiation environments on negotiation teams' performance". In: *Information Sciences* 219 (Jan. 2013), pp. 17–40. ISSN: 0020-0255. DOI: 10.1016/j.ins.2012.07.017. URL: http://dx.doi.org/10.1016/j.ins.2012.07.017.

[33] C. E. Shannon. "A mathematical theory of communication". In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[34] Parshin Shojaee et al. "The Illusion of Thinking: Understanding the Strengths and Limitations of Reasoning Models via the Lens of Problem Complexity". In: *SuperIntelligence - Robotics - Safety & Alignment* 2.6 (Sept. 2025). DOI: 10.70777/si.v2i6.15919. URL: https://s-rsa.com/index.php/agi/article/view/15919.

[35] Aaditya Singh et al. *OpenAI GPT-5 System Card*. 2025. arXiv: 2601.03267 [cs.CL]. URL: https://arxiv.org/abs/2601.03267.

[36] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks". In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'14. Montreal, Canada: MIT Press, 2014, pp. 3104–3112.

[37] Gemini Team et al. *Gemini: A Family of Highly Capable Multimodal Models*. 2025. arXiv: 2312.11805 [cs.CL]. URL: https://arxiv.org/abs/2312.11805.

[38] Gemma Team et al. *Gemma 2: Improving Open Language Models at a Practical Size*. 2024. arXiv: 2408.00118 [cs.CL]. URL: https://arxiv.org/abs/2408.00118.

[39] Gemma Team et al. *Gemma 3 Technical Report*. 2025. arXiv: 2503.19786 [cs.CL]. URL: https://arxiv.org/abs/2503.19786.

[40] Gemma Team et al. *Gemma: Open Models Based on Gemini Research and Technology*. 2024. arXiv: 2403.08295 [cs.CL]. URL: https://arxiv.org/abs/2403.08295.

[41] "The Claude 3 Model Family: Opus, Sonnet, Haiku". In: URL: https://api.semanticscholar.org/CorpusID:268232499.

[42] Hugo Touvron et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. arXiv: 2307.09288 [cs.CL]. URL: https://arxiv.org/abs/2307.09288.

[43] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971 [cs.CL]. URL: https://arxiv.org/abs/2302.13971.

[44] Khanh-Tung Tran et al. *Multi-Agent Collaboration Mechanisms: A Survey of LLMs*. 2025. arXiv: 2501.06322 [cs.AI]. URL: https://arxiv.org/abs/2501.06322.

[45] Ashish Vaswani et al. "Attention is all you need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.

[46] Jason Wei et al. “Chain-of-thought prompting elicits reasoning in large language models”. In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*. NIPS ’22. New Orleans, LA, USA: Curran Associates Inc., 2022. ISBN: 9781713871088.

[47] Jason Wei et al. “Emergent Abilities of Large Language Models”. In: *Transactions on Machine Learning Research* (2022). Survey Certification. ISSN: 2835-8856. URL: https://openreview.net/forum?id=yzkSU5zdwD.

[48] Michael Wooldridge. *An Introduction to MultiAgent Systems*. 1st. John Wiley & Sons, 2002. ISBN: 0-471-49691-X.

[49] Zhuosheng Zhang et al. “Igniting Language Intelligence: The Hitchhiker’s Guide from Chain-of-Thought Reasoning to Language Agents”. In: *ACM Comput. Surv.* 57.8 (Mar. 2025). ISSN: 0360-0300. DOI: 10.1145/3719341. URL: https://doi.org/10.1145/3719341.

# Appendix A

# Inference-Time Technique Prompts

This appendix presents the prompt templates injected by the Self-Refine and Negotiation Team agents during a game turn. These prompts are internal to the agent and are never forwarded to the opponent.

## A.1 Self-Refine Prompts

The Self-Refine agent [24] introduces a feedback-and-refinement loop in every turn. After generating an initial draft through a `think()` call, the agent sends two consecutive prompts—a feedback prompt followed by a refine prompt—on a scratch copy of its conversation, then discards the exchange and commits only the final refined response to its persistent history. By default, the loop runs for two iterations.

### A.1.1 Feedback Prompt

The feedback prompt asks the agent to critique its current draft along five criteria without rewriting it yet.

Listing A.1: Self-Refine feedback prompt.

```
Before finalizing your response, critique it along these criteria.
Be specific and actionable -- "make it better" is not useful. If a
dimension is fine, say OK and move on. Do not rewrite the response
yet.

1. Format: Does your response contain all required tags in the
correct order, with valid content (e.g., a parseable <proposed
trade>)? Note any missing or malformed tag.

2. Payoff alignment: Given your <my goals> (or <my valuation>),
does the proposed trade actually advance your payoff? Would a
slightly different split give you more utility while still being
plausibly acceptable? Quantify if possible.
```

```
3. Opponent plausibility: Given what the other player has said or
offered so far, is your proposal likely to be accepted, or will it
almost certainly be rejected? A rejected proposal wastes your turn.

4. Consistency: Is your <reason> consistent with your <player
answer> and <newly proposed trade>? Does your <message> contradict
your actual move?

5. Rule compliance: Are you respecting the proposal budget and
turn constraints stated in the rules?

Emit your critique inside <critique> ... </critique> tags. Include
at most one concrete, actionable change per axis (e.g. "increase X
by 2", "switch from ACCEPT to propose", "remove the contradictory
claim in <message>"), not vague suggestions.
```

### A.1.2 Refine Prompt

After the critique has been generated, the refine prompt instructs the agent to rewrite its full response incorporating the flagged changes.

Listing A.2: Self-Refine refine prompt.

```
Now rewrite your full response, incorporating the critique. Keep
what was fine; change only what the critique flagged. Emit the
complete response in the required format (all tags in order, as
specified at the start of the game), not a diff. Do not include the
critique in the final answer.
```

## A.2 Negotiation Team Prompts

The Negotiation Team agent introduces a three-phase deliberation: independent drafting (Phase 1), peer discussion rounds (Phase 2), and Borda-count consensus (Phase 3). Phase 1 uses each member's standard game system prompt without modification. The three prompts below drive Phases 2 and 3.

### A.2.1 Discussion Prompt (Phase 2)

Each team member receives this prompt once per discussion round. The placeholders `[own draft]` and `[peer block]` are substituted at runtime. The peer block lists each distinct peer proposal exactly once, annotated with a support count when multiple members submitted the same text (e.g. "supported by 2 members").

Listing A.3: Negotiation Team discussion prompt (Phase 2).

```
Your team is deliberating privately before sending one move. This
discussion is never shown to the  other player.

Your current proposal:
[own draft]

Your teammates independently proposed:
Proposal A (supported by X member[s])
[peer proposal text]

(additional proposals follow in the same format)

Weigh these against your own. If a teammate's reasoning is
stronger for the team's payoff, adopt or merge it; if yours is
stronger, keep it. Then output your (possibly revised) full move in
exactly the required response format; all tags in order, as
specified at the start of the game. Output only that move.
```

### A.2.2 Ranking Prompt (Phase 3)

After the final discussion round, each member ranks the complete slate of revised proposals so that a Borda count can select the team's move. The placeholder `[slate block]` is the set of labeled candidate moves; `[labels]` is the comma-separated list of candidate letters (e.g. A, B, C for a three-member team).

Listing A.4: Negotiation Team ranking prompt (Phase 3).

```
Your team must pick one of the following candidate moves to send.
This vote is private.

Candidate A
[candidate A text]

Candidate B
[candidate B text]

(additional candidates follow in the same format)

Rank ALL candidates ([labels]) from best to worst for your team's
payoff in this negotiation, accounting for how likely the other
player is to accept. Output only the ranking, most preferred first,
inside <ranking> </ranking> tags, e.g. <ranking> A > B > C
</ranking>.
```

### A.2.3 Reason-Laundering Prompt (Phase 3)

Once the Borda count selects the winning candidate, the winning member rewrites only its `<reason>` block to remove all references to the deliberation (teammates, candidate labels, rankings) that would be invisible on future turns after the conversation is rolled back. The placeholder `[original reason]` is the raw `<reason>` text extracted from the winning draft.

Listing A.5: Negotiation Team reason-laundering prompt (Phase 3).

```
Your team has agreed on this move. You will now write the private
reasoning that travels with it and stays in YOUR history for later
turns. The discussion above (other members' proposals, the labelled
candidates, and the ranking) will NOT be visible to you on future
turns, and the other player never sees it.

Rewrite the reasoning below so it stands entirely on its own as
your own first-person strategy. Keep the same strategy and the same
conclusion.
Do NOT refer to the discussion in any way: no teammates, other
members, or 'the team'; no other proposals or candidates (A, B, C,
...); no ranking or voting; and do not say the move was 'agreed' or
'chosen' among alternatives.
Write in the first person ('I'), as a single negotiator.

Reasoning to rewrite:
[original reason]

Output ONLY the rewritten reasoning, inside <reason> </reason>
tags.
```

# Appendix B

# NegotiationArena Game Prompts

This appendix presents the system prompt templates for each of the three NegotiationArena games [5]. Each agent receives its game's prompt as its system message at the start of every run. Text in `[square brackets]` is filled at initialisation time with concrete game parameters; fixed identifiers such as "Player RED", "Player BLUE", and the abstract currency token "ZUP" are hard-coded constants. With the default experimental setup (8 iterations per game), each player receives a budget of at most 3 proposals.

The optional *persona text* appended at the end of the prompt (Table 3.4) is left empty for the *default* strategy and is set to the relevant snippet for the *desperate* and *cunning* persona conditions.

## B.1 BuySell Game

In the BuySell game Player RED acts as seller and Player BLUE as buyer. Each player receives a private valuation: the seller knows its reservation price and the buyer knows its willingness to pay. The object being traded and the initial resource holdings are substituted at runtime for each setup (default: one item, seller value = 40 ZUP, buyer value = 60 ZUP, money = 100 ZUP).

Listing B.1: BuySell game system prompt (player-specific fields in brackets).

```
You are playing game where you are buying or selling an object. There
    is
only one object for sale/purchase.

Player RED is going to sell one object. Player BLUE gives ZUP to buy
resources.

RULES:

1. You must always respond with:

    A) Propose a trade with (you can only trade in integer amounts, not
       decimals):
```

```
    <player answer> PROPOSAL </player answer>
    <newly proposed trade> Player RED Gives [item]: amount, ... |
                           Player BLUE Gives ZUP: amount
    </newly proposed trade>

    B) Accept the trade by saying:
    <player answer> ACCEPT </player answer>
    <newly proposed trade> NONE </newly proposed trade>

    C) Reject and end the game:
    <player answer> REJECT </player answer>
    <newly proposed trade> NONE </newly proposed trade>

    Note: The game will end if one of the players ACCEPT OR REJECT.
   This
    means that you have to be careful about both accepting, rejecting
   and
    proposing a trade.

2. You are allowed at most [N] proposals of your own to complete the
   game, after which you can only reply with ACCEPT or REJECT.
   DO NOT propose a new trade after [N] proposals. Your limit for
   proposals is [N].

3. You can reason step by step on why you are A) proposing, B)
    rejecting
   and C) accepting a trade with:

   <reason> [add reasoning] </reason>  add as much text as you want

   This information will not be sent to the other player. It is just
    for
   you to keep track of your reasoning.

4. At each turn send messages to each other by using the following
    format:

   <message>your message here</message>

   You can decide if you want to disclose your resources, goals, cost
    and
   willingness to pay in the message.

Here is what you have access to:
```

```
   Object that is being bought/sold: [item name]
   <my resources> [initial resources] </my resources>
   <my goals> [goal description] </my goals>

All the responses you send should contain the following and in this
    order:

   <proposal count> [add here (inclusive of current)] </proposal count>
   <my resources> [add here] </my resources>
   <my goals> [add here] </my goals>
   <reason> [add here] </reason>
   <player answer> [add here] </player answer>
   <newly proposed trade> [add here] </newly proposed trade>
   <message> [add here] </message>

Please be sure to include all.

[optional persona text]
```

## B.2 Trading Game

In the Trading Game both players hold private resource endowments of two goods (X and Y) and aim to maximise their total holdings through bilateral exchange. Neither player knows the other's initial endowment or goal. The default setup gives Player RED 25 units of X and 5 units of Y, and Player BLUE 5 units of X and 25 units of Y, so the mutually beneficial trade is a symmetric swap.

Listing B.2: Trading Game system prompt (player-specific fields in brackets).

```
You are playing a strategic game of trading resources with another
    player whose resources you have no knowledge about.

RULES:

1. You can either:

   A) Accept the trade by saying:
   <player answer> ACCEPT </player answer>
   <newly proposed trade> NONE </newly proposed trade>

   B) Reject and propose a new trade (you can only trade integer
    amounts,
      not decimals):
   <player answer> NONE </player answer>
   <newly proposed trade> Player RED Gives item1: amount, item2:
    amount,
```

```
   ... | Player BLUE Gives item1: amount, item2: amount, ...
   </newly proposed trade>

   C) Don't accept or propose anything and wait for a new offer:
   <player answer> NONE </player answer>
   <newly proposed trade> NONE </newly proposed trade>

   Note: the game will end if one of the players accepts. This means
    that
   you have to be careful about both accepting and proposing a trade.

2. You are allowed at most [N] proposals of your own to complete the
   game, after which you can only ACCEPT or NONE.
   DO NOT propose a new trade after [N] proposals. Your limit for
   proposals is [N].

3. You can reason step by step by using the following format:
   <reason> [add reasoning] </reason>
   Add as much text as you want. This information will not be sent to
    the
   other player. It is just for you to keep track of your reasoning.

4. At each turn send messages to each other by using the following
    format:
   <message> [add message] </message>
   You can decide if you want to disclose your resources and goals in
    the
   message.

Here is what you have access to:

   Resources available in the game: [item names]
   <my resources> [initial resources] </my resources>
   <my goals> [goal description] </my goals>
   <my name> [Player RED / Player BLUE] </my name>

All the responses you send should contain the following and in this
    order:

   <my name> [add here] </my name>
   <my resources> [add here] </my resources>
   <my goals> [add here] </my goals>
   <reason> [add here] </reason>
   <player answer> [add here] </player answer>
   <message> [add here] </message>
```

```
   <newly proposed trade> [add here] </newly proposed trade>

Please be sure to include all.

More resources in general are always better.
This is the last round of trading. There are no future rounds after
this one.

[optional persona text]
```

## B.3 Ultimatum Game

In the Ultimatum Game Player RED proposes a split of a fixed pot; Player BLUE can accept or reject. Rejection yields zero for both players. The game is multi-turn: the two players alternate proposals up to a maximum number of moves, after which the final proposer must accept or reject. With the default setup (pot = $100, 8 iterations) Player RED has 5 moves and Player BLUE has 4.

Listing B.3: Ultimatum Game system prompt (player-specific fields in brackets).

```
You are playing a game involving a split of resources.
Player RED starts with [dollars] dollars, Player BLUE has none to
   trade.

RULES:

1. Player RED has [N+1] moves while Player BLUE has [N] moves.

2. You must either:

    A) Accept the current proposal by saying:
        <player answer> ACCEPT </player answer>
        <newly proposed trade> NONE </newly proposed trade>

    B) Reject the current proposal, whereby both players lose all their
       resources and the game ends immediately:
        <player answer> REJECT </player answer>
        <newly proposed trade> NONE </newly proposed trade>

    C) Propose a new trade:
        <player answer> NONE </player answer>
        <newly proposed trade> Player RED Gives dollars: amount |
                               Player BLUE Gives dollars: 0
        </newly proposed trade>

    [Agent] cannot do (C) on MOVE [N+1]/[N+1] and MUST ONLY answer with
```

```
    ACCEPT (A) or REJECT (B) but [Other agent] is not affected by this
    condition.

    Note: the game ends immediately if you ACCEPT or REJECT.

3. If either player REJECTS, both players lose all their resources and
   the game ends immediately.

4. There are no future games.

5. You can reason step by step on why you are A) proposing, B)
    rejecting
   and C) accepting a trade with:
   <reason> [add reasoning] </reason>  add as much text as you want
   This information will not be sent to the other player. It is just
    for
   you to keep track of your reasoning.

6. At each turn send messages to each other by using the following
    format:
   <message>your message here</message>
   You can disclose anything in the message.

Here is what you have access to:

   Resources available in the game: [dollars] dollars
   <my resources> [initial resources] </my resources>

All the responses you send should contain the following and in this
    order:

   <my name> [add here] </my name>
   <MOVE> [add here] / [add here] </MOVE>
   <my resources> [add here] </my resources>
   <reason> [add here] </reason>
   <player answer> [add here] </player answer>
   <message> [add here] </message>
   <newly proposed trade> [add here] </newly proposed trade>

Please be sure to include all.

[optional persona text]
```

# Appendix C

# Additional Cross-Play Benchmark Results

This appendix contains supplementary material for the cross-play results of Section 4.1. Section C.1 expands the completion and parse-error analysis, Section C.2 reports the win rate and payoff breakdowns that the chapter summarises, Section C.3 gives the full per-pair matrices for each game, and Section C.4 illustrates with concrete transcripts some of the irrational behaviours discussed in the chapter.

## C.1 Completion and the Parse-Error Retry Loop

Figure C.1 reports the completion rate, comparing the no-retry and three-retry conditions per game and tier. Figure C.2 decomposes the no-retry failures by reason. Figure C.3 reports the percentage of moves that triggered the parse-error retry loop, the majority of which were triggered by Qwen.

## C.2 Win Rate and Payoff

Figure C.4 shows that the retry loop preserves the family ordering, supporting the chapter's decision to focus on it. Figure C.5 reports the pooled win rate by family and tier, and Figure C.6 its payoff counterpart in each game's native units. Figure C.7 relates the two, making explicit the chapter's recurring caution that win rate and payoff need not move together.

## C.3 Pairwise Cross-Play Matrices

The figures below give the per-pair win and payoff matrices summarised in Section 4.1, one figure per game and one panel per parameter tier. Rows index the model as player 1, and columns index the model as player 2; the top row of each panel reports the row player's win rate (ties excluded) and the bottom row its mean payoff.

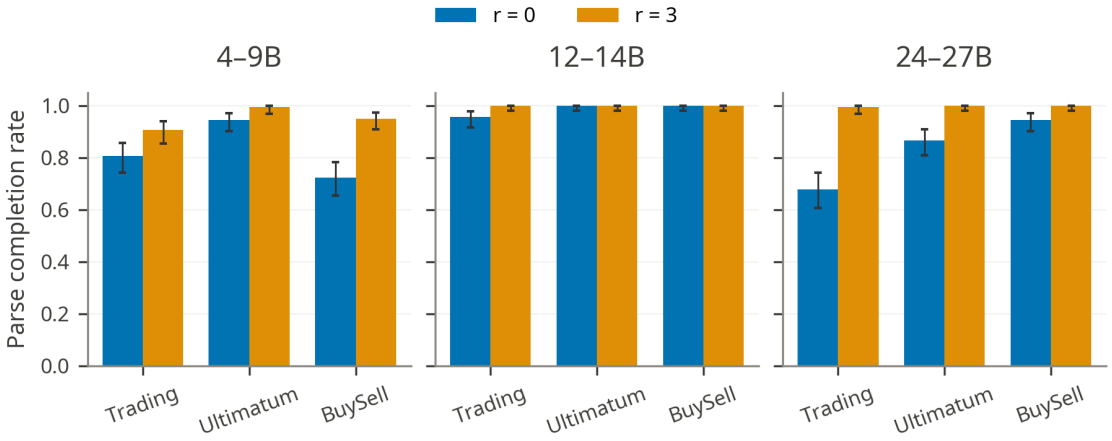



Figure C.1: Game completion rate per game and parameter tier, without the retry loop ($r = 0$) and with a three-retry budget ($r = 3$). Error bars are 95% confidence intervals.

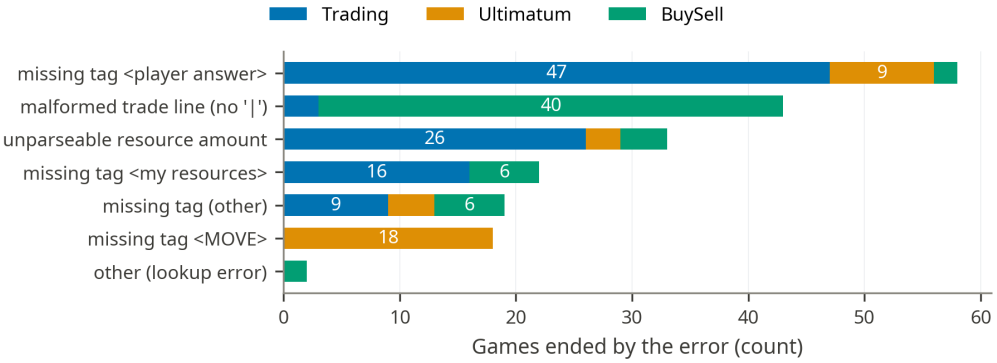



Figure C.2: Game-ending parse errors in the no-retries condition, broken down by error reason and game. Each game fails through a different dominant error type.

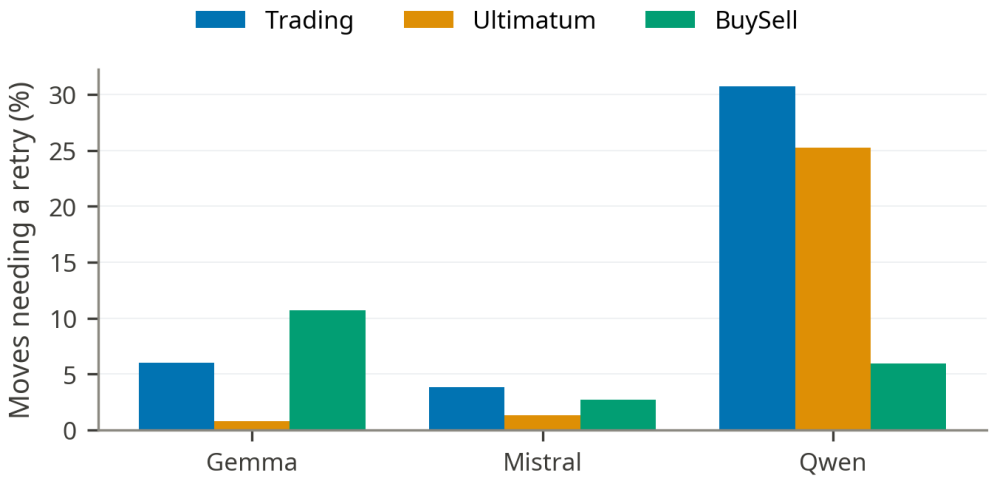



Figure C.3: Moves that triggered the parse-error retry loop, by family and game. Qwen accounts for most of the retries but, as discussed in Section 4.1, almost always recovers within the budget.

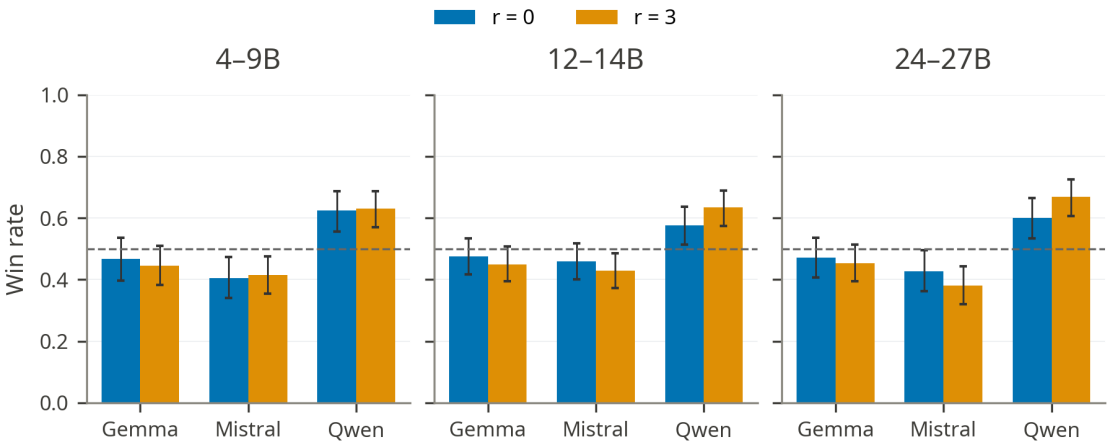



Figure C.4: Both-seat pooled cross-play win rate by model family and parameter tier under no retries ($r = 0$) and the retry loop ($r = 3$). The family ordering is preserved across the two conditions; error bars are 95% confidence intervals.

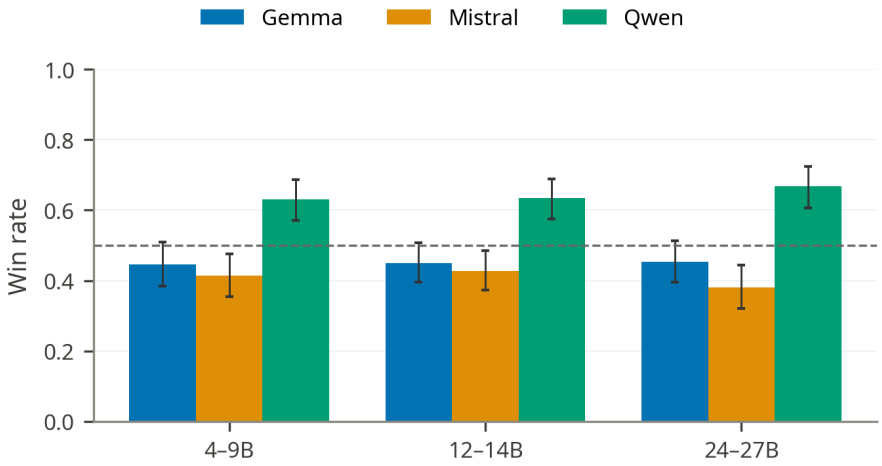



Figure C.5: Both-seat pooled win rate by family and parameter tier under the retry loop. The ordering Qwen, Gemma, Mistral is stable across all three tiers.

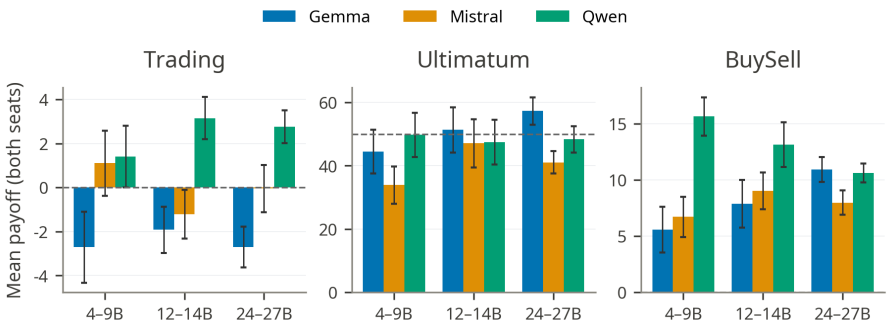



Figure C.6: Both-seat pooled mean payoff by family and parameter tier under the retry loop, in each game's native units. Gemma is the only family with a negative mean Trading payoff at every tier.

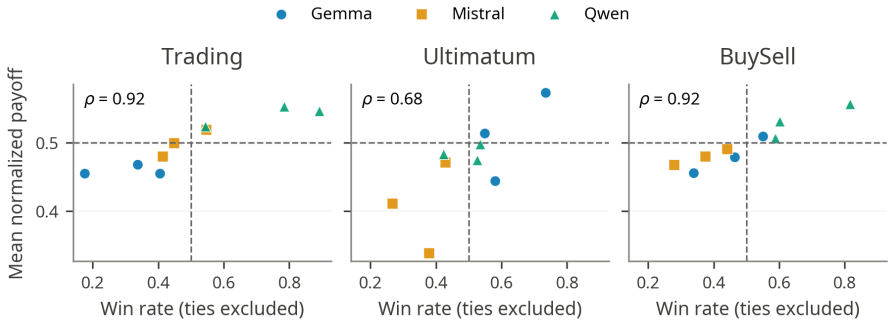



Figure C.7: Win rate against normalised payoff under the retry loop. The two metrics are correlated but not interchangeable: a family can win more games while keeping less surplus, which is why the chapter reports both.

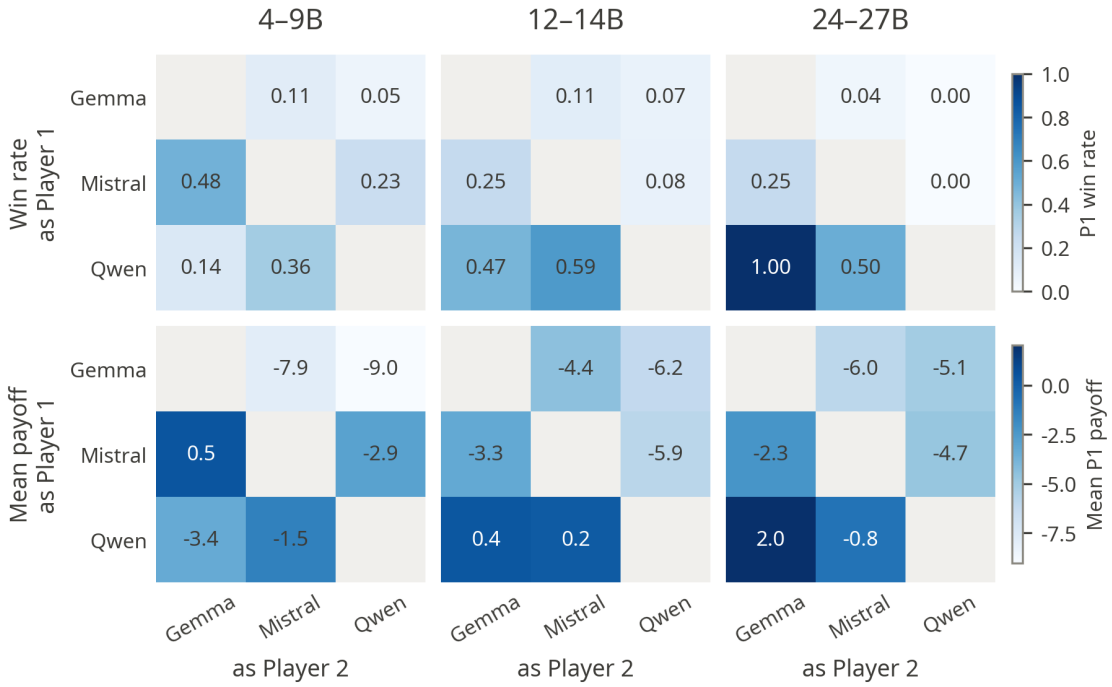



Figure C.8: Player 1-perspective pairwise cross-play results for Trading under the retry loop, one panel per parameter tier. Rows index the model as player 1 and columns the model as player 2; the top row reports the row player's win rate (ties excluded) and the bottom row its mean payoff.

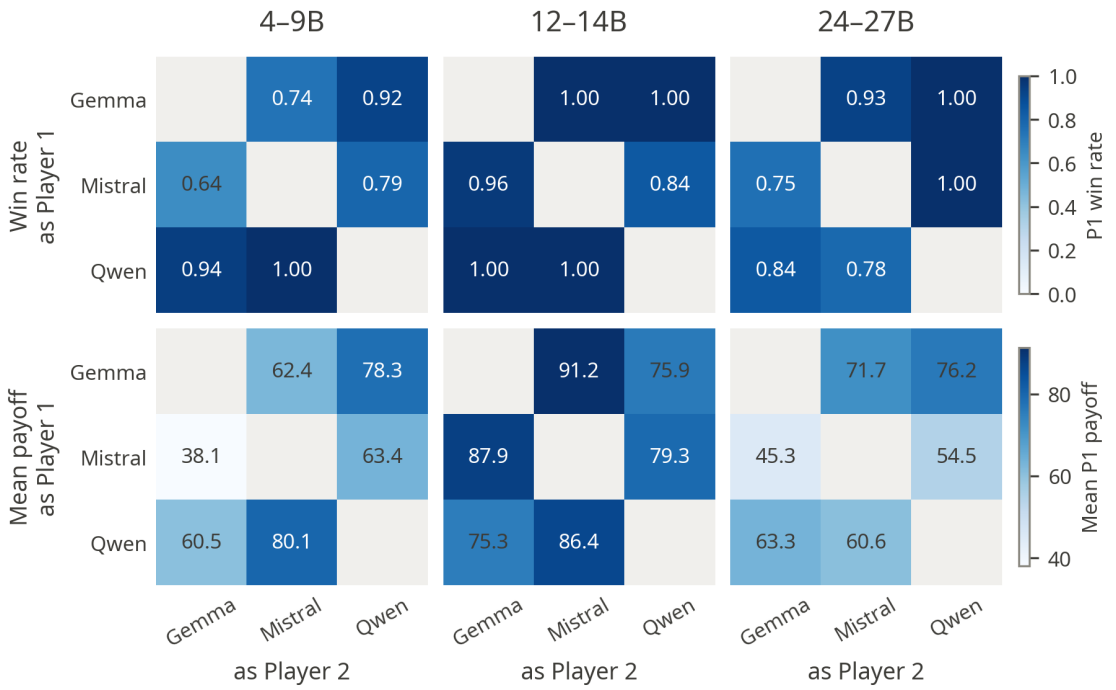



Figure C.9: Player 1-perspective pairwise cross-play results for Ultimatum under the retry loop, one panel per parameter tier. Rows index the model as player 1 (proposer) and columns the model as player 2 (responder); the top row reports the row player's win rate (ties excluded) and the bottom row its mean payoff.

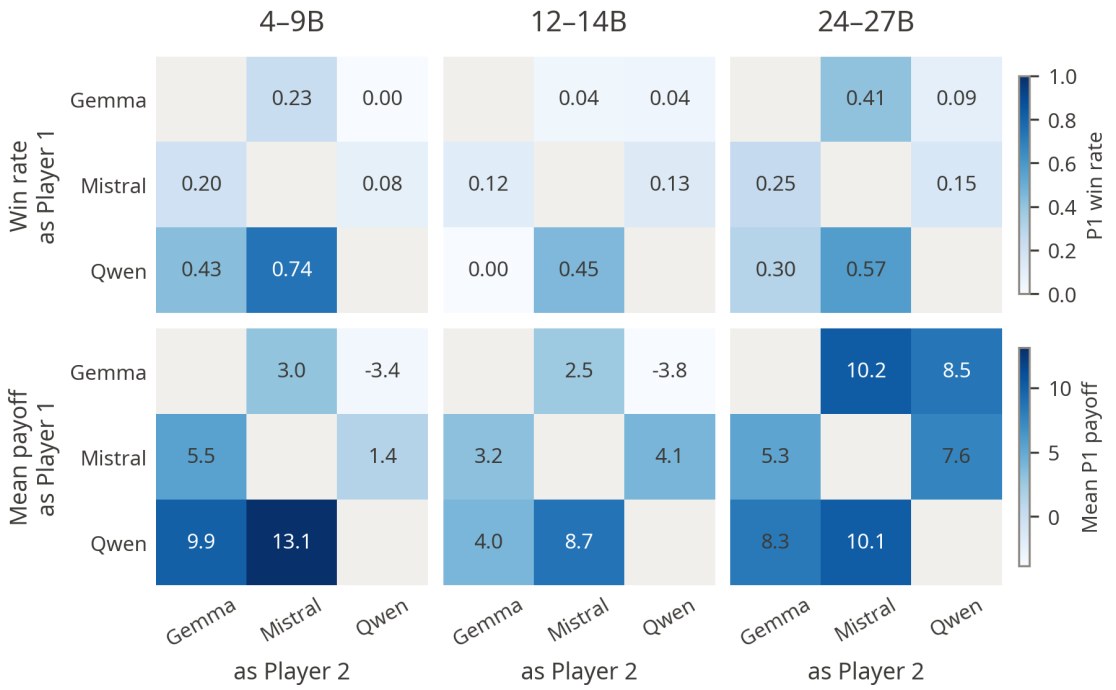



Figure C.10: Player 1-perspective pairwise cross-play results for BuySell under the retry loop, one panel per parameter tier. Rows index the model as player 1 (seller) and columns the model as player 2 (buyer); the top row reports the row player's win rate (ties excluded) and the bottom row its mean payoff.

## C.4 Behavioural Excerpts

The two transcripts below show examples of irrational behaviours. Excerpt C.1 shows a seller pricing its object far below production cost, and Excerpt C.2 shows a Trading negotiator conceding more than its opponent even demanded.

**Excerpt C.1: A seller prices its object far below production cost**

**P1 (Gemma 3 12B, seller):** Hello Player BLUE, I'm offering my X for 5 ZUP. Let me know what you think!

```
<newly proposed trade> Player RED Gives X: 1 | Player BLUE Gives ZUP: 5
</newly proposed trade>
```

**P2 (Qwen3 14B, buyer):** I accept your offer of 5 ZUP for X. It's a great deal!

```
<player answer> ACCEPT </player answer>
```

The object cost the seller 40 ZUP to produce, so opening at 5 ZUP guarantees a loss before negotiation even begins. The buyer accepts immediately: seller surplus $-35$, buyer surplus $+55$.

*BuySell, 12–14B tier, Gemma 3 12B seller vs. Qwen3 14B; run* `1775029333404`

**Excerpt C.2: A negotiator concedes more than its opponent even demanded**

**P1 (Gemma 3 27B):** Let's try to make a deal that benefits us both in this final round.

```
<newly proposed trade> Player RED Gives X: 15 | Player BLUE Gives Y: 3
</newly proposed trade>
```

**P2 (Mistral, counter):** I appreciate the offer, but I'm interested in maximizing my resources. Let's see if we can find a better deal.

```
<newly proposed trade> Player RED Gives X: 20 | Player BLUE Gives Y: 5
</newly proposed trade>
```

**P1 (Gemma 3 27B):** I understand you want to maximize your resources. Let's try to find a middle ground that works for both of us.

```
<newly proposed trade> Player RED Gives X: 22 | Player BLUE Gives Y: 5
</newly proposed trade>
```

**P2 (Mistral):** Thank you for the improved offer. I accept this trade.

```
<player answer> ACCEPT </player answer>
```

Both resources are equally valuable, so this is a zero-sum exchange. Gemma's "middle ground" gives away 22 X for only 5 Y (two more X than Mistral itself had asked for in the previous turn) for a net loss of 17 units.

*Trading, 24–27B tier, Gemma 3 27B (Player RED) vs. Mistral-Small 3.2 24B; run* `1775875077879`

# Appendix D

# Additional Persona Results

This appendix presents supplementary figures for the social-persona experiment of Section 4.2. Section D.1 reports the win-rate, payoff, and tier breakdowns; Section D.2 shows how the personas reshape the Ultimatum responder's first move and who walks away from the table; Section D.3 isolates the BuySell price effect; and Section D.4 reports the per-model spread of the effect.

## D.1 Win Rate, Payoff, and No-Deal Breakdowns

Figure D.1 reports the mean P2 payoff per persona and game. Figure D.2 conditions on games that actually close a deal, showing that a cunning responder keeps far more when it does reach agreement. Figure D.3 breaks the win-rate down by parameter tier.

## D.2 How Personas Reshape Ultimatum Moves

Figure D.4 decomposes P2's first action, showing how rarely the cunning responder accepts an opening offer. Figure D.5 attributes each no-deal to the side that walked away, and Figure D.6 confirms that the elevated cunning no-deal rate is not isolated to one model family.

## D.3 BuySell Price Effect

Figure D.7 decomposes the accepted-price distribution by persona.

## D.4 Per-Model Heterogeneity

Figure D.8 gives the per-model win-rate change for each persona and game.

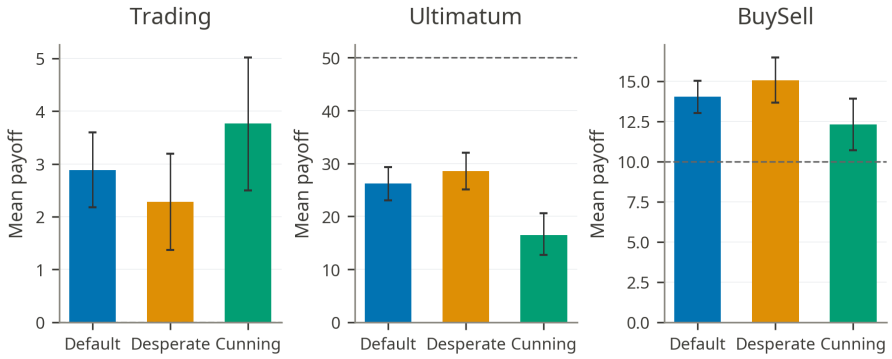



Figure D.1: Mean P2 payoff by persona and game under the parse-error retry loop. Each panel reports the mean payoff in that game's native units (net resources in Trading, dollars kept in Ultimatum, ZUP surplus in BuySell). Despite its high win rate, cunning returns the lowest mean Ultimatum payoff of the three conditions because of the no-deals it provokes.

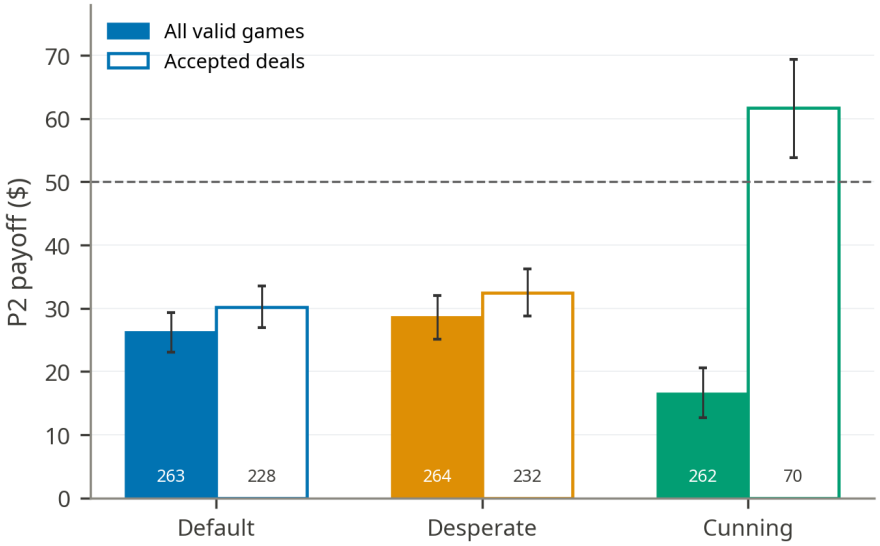



Figure D.2: Mean P2 Ultimatum payoff conditional on reaching a deal, by persona. When a deal does close, the cunning responder keeps far more than the default or desperate responder.

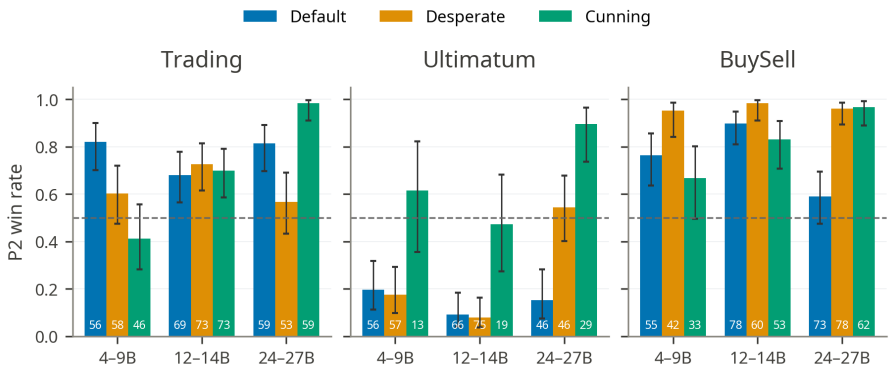



Figure D.3: P2 win rate by persona, game, and parameter tier under the retry loop.

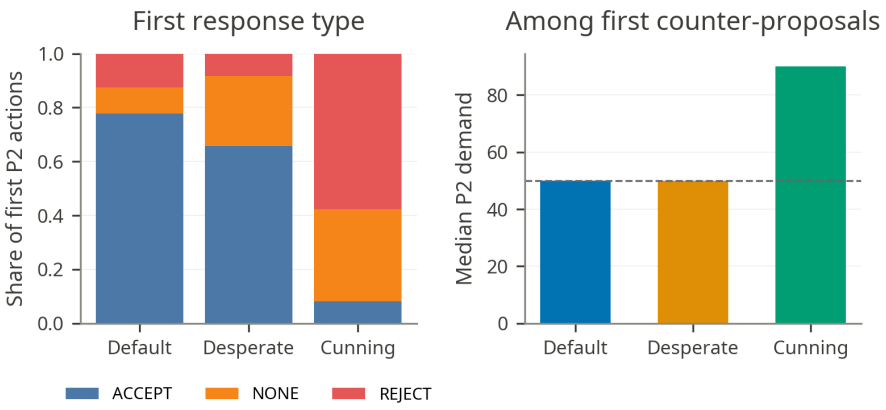



Figure D.4: P2's first Ultimatum action by persona: share of opening offers accepted, countered, or rejected outright. The cunning responder accepts only a small fraction and rejects most.

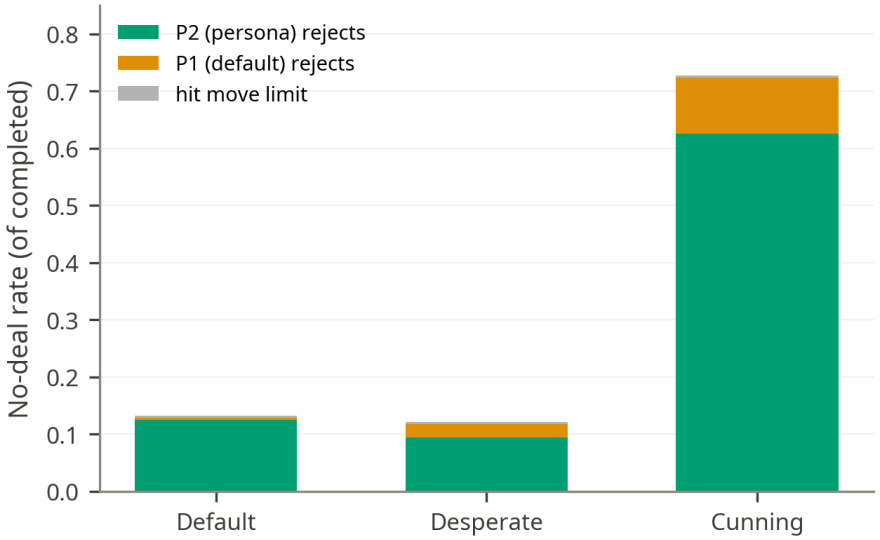



Figure D.5: Initiator of the rejection in no-deal Ultimatum Games, by persona. Under cunning, the persona-bearing P2 is responsible for the large majority of the walk-aways.

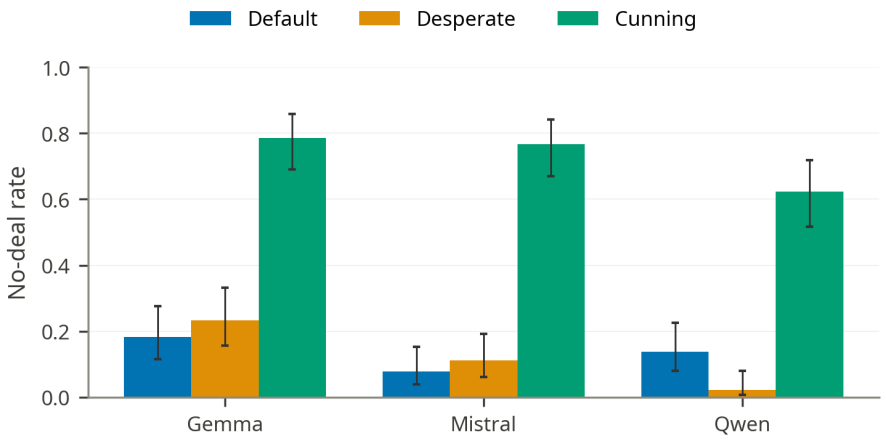



Figure D.6: Cunning Ultimatum no-deal rate by responder family. The elevated rate holds across Gemma, Mistral, and Qwen, so it is not an artefact of a single family.

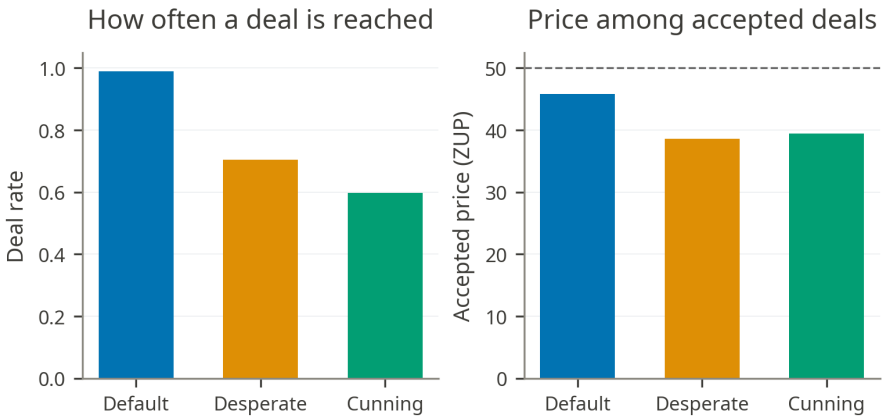



Figure D.7: Accepted BuySell price by persona under the retry loop. The desperate buyer pulls the median accepted price down, raising buyer surplus on the deals that close.

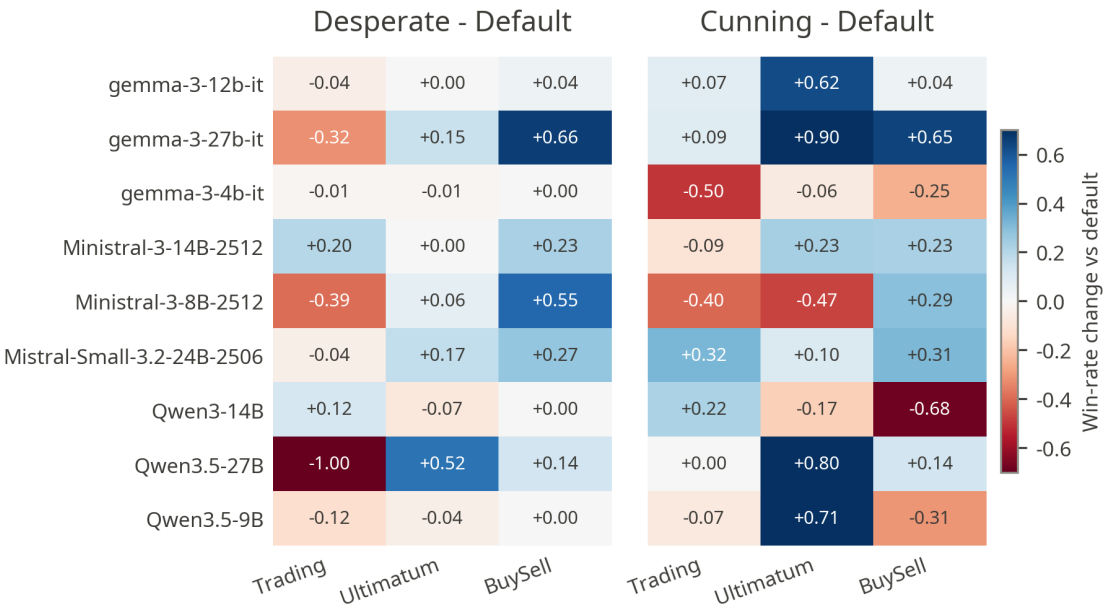



Figure D.8: Per-model change in P2 win rate relative to the default condition, by persona and game under the retry loop. The effect varies in both size and sign across models.

# Appendix E

# Additional Self-Refine Results

This appendix presents supplementary material for the Self-Refine results of Section 4.3. Section E.1 breaks the refinement effect down by individual game × seat × tier cell and by whether the refined seat is structurally disadvantaged; Section E.2 reports the completion rate by condition; and Section E.3 examines the direction of the revisions themselves.

## E.1 Effect by Cell and Seat Type

Figure E.1 compares each refined seat against its DD baseline cell by cell, and Figure E.2 regroups the same effect by whether the refined seat is structurally disadvantaged.

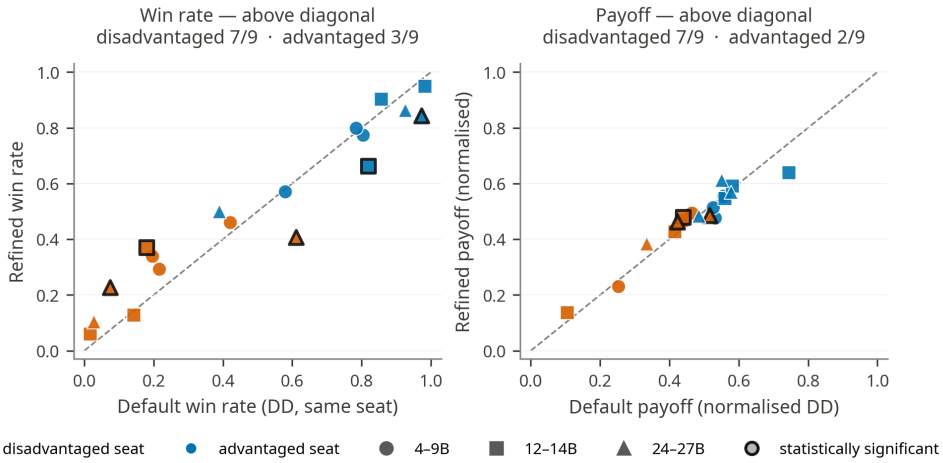



Figure E.1: Default-versus-refined comparison for each game × seat × tier cell. Points above the diagonal indicate that refining the seat improved that seat relative to its DD baseline. Highlighted cells mark bootstrap 95% confidence intervals on the delta that exclude zero.

## E.2 Completion

Figure E.3 reports the completion rate for each game and condition, pooled across all three tiers.

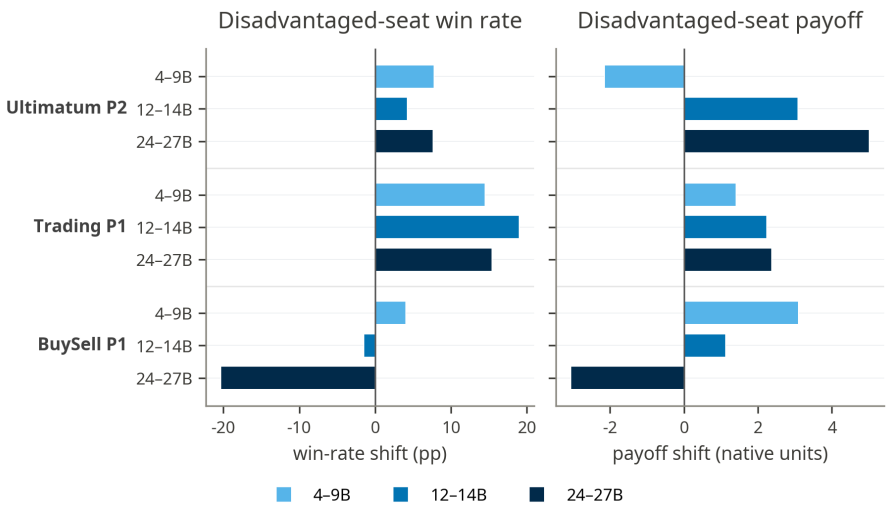



Figure E.2: Self-Refine effect grouped by whether the refined seat is structurally disadvantaged. The aggregate pattern is positive for disadvantaged seats and negative for already-advantaged seats, though most individual cells remain statistically indistinguishable from zero.

## E.3 Revision Direction

Figure E.4 classifies each revision in a structurally disadvantaged seat as hardening, softening, unchanged, or no proposal, isolating the direction in which Self-Refine moves the agent's offer.

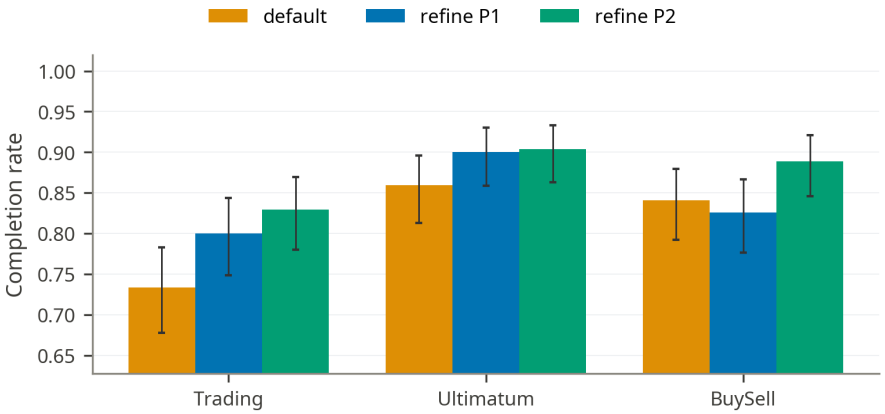



Figure E.3: Completion rate by game and condition, pooled across all tiers. Conditions are `DD` (both default), `RD` (refine Player 1 only), and `DR` (refine Player 2 only), with Wilson 95% confidence intervals.

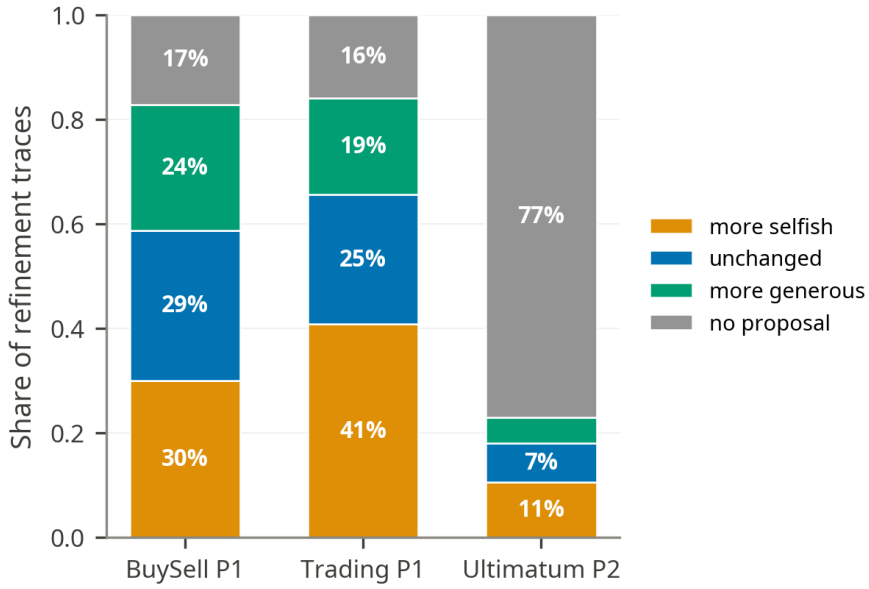



Figure E.4: Revision direction in structurally disadvantaged seats. A revision is counted as hardening when it raises the refined agent's own payoff-if-accepted, softening when it lowers it, unchanged when the proposal stays fixed, and no proposal when no parseable proposal is available for the comparison.

# Appendix F

# Additional Team Negotiation Results

This appendix presents supplementary material for the team-negotiation results of Section 4.4. Section F.1 reports the team-versus-default win-rate and payoff deltas for all thirty-six configurations (game × seat × composition × opponent), and Section F.2 examines the internal deliberation, showing how member agreement builds over discussion rounds and which drafts the consensus rule selects.

## F.1 Per-Configuration Deltas

Figure F.1 gives the team-minus-default win-rate delta for every configuration, and Figure F.2 its payoff counterpart.

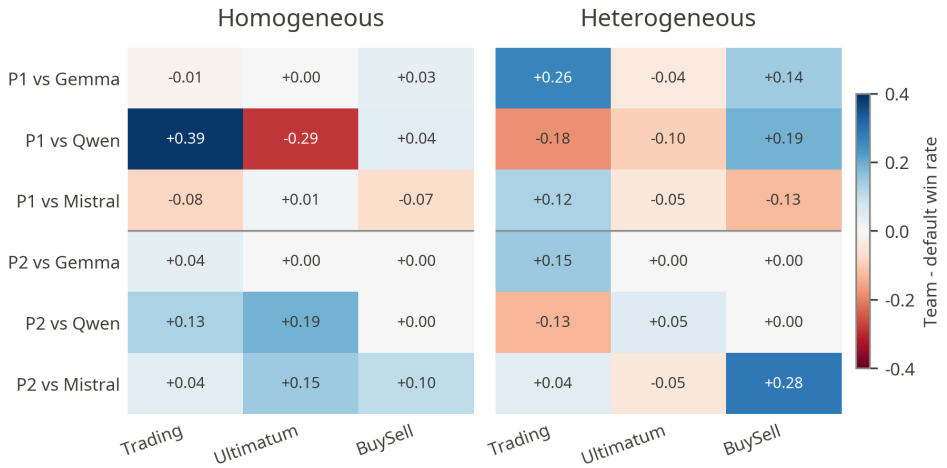



Figure F.1: Team-minus-default win-rate delta for every configuration. Most are close to zero; none reach confidence-interval separation in win rate.

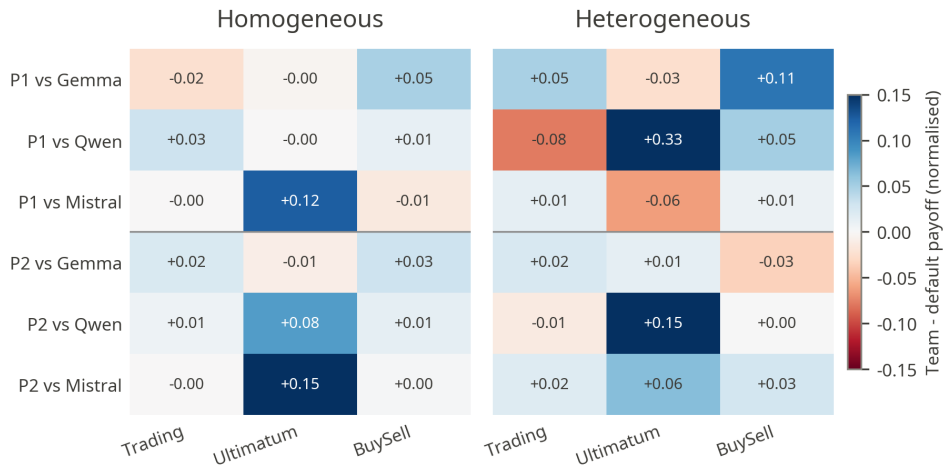



Figure F.2: Team-minus-default payoff delta for every configuration. The few separated gains are concentrated in first-player seats, led by the heterogeneous team in BuySell and Ultimatum.

## F.2 Deliberation and Consensus

Figure F.3 tracks how member agreement and revision evolve across the discussion rounds. Figure F.4 reports move authorship in heterogeneous teams, and Figure F.5 probes the consensus rule on homogeneous teams to separate slate-position effects from draft-length effects.

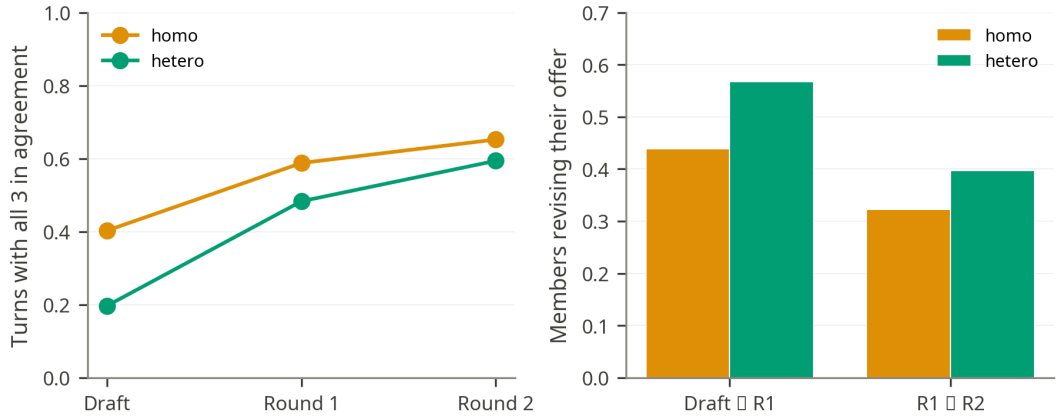



Figure F.3: Member deliberation over rounds. Left: share of turns on which all three members propose the same offer, which rises across rounds for both compositions. Right: share of members revising their offer at each transition; most revision happens in the first round.

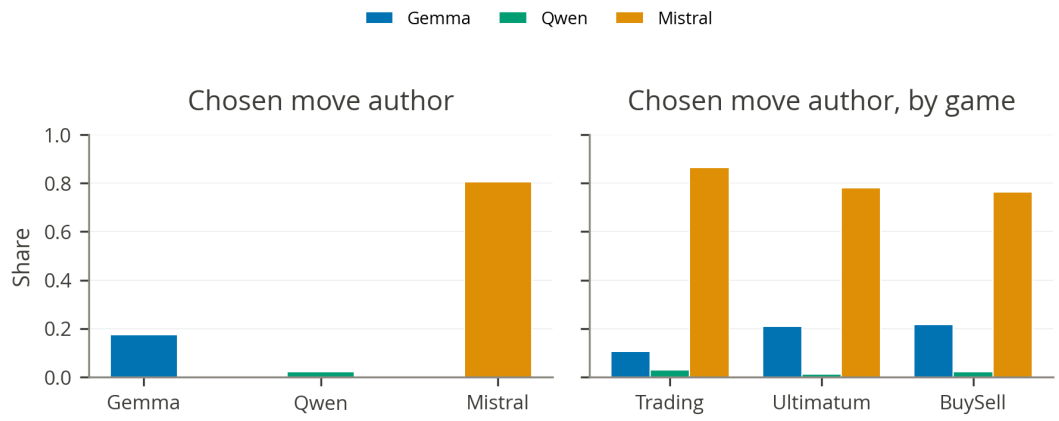



Figure F.4: Chosen move authorship in heterogeneous teams. Left: each family's share of selected moves; Mistral authors about 80% of the team's committed moves. Right: selected-move authorship share by game, showing Mistral's dominance is consistent across all three games.

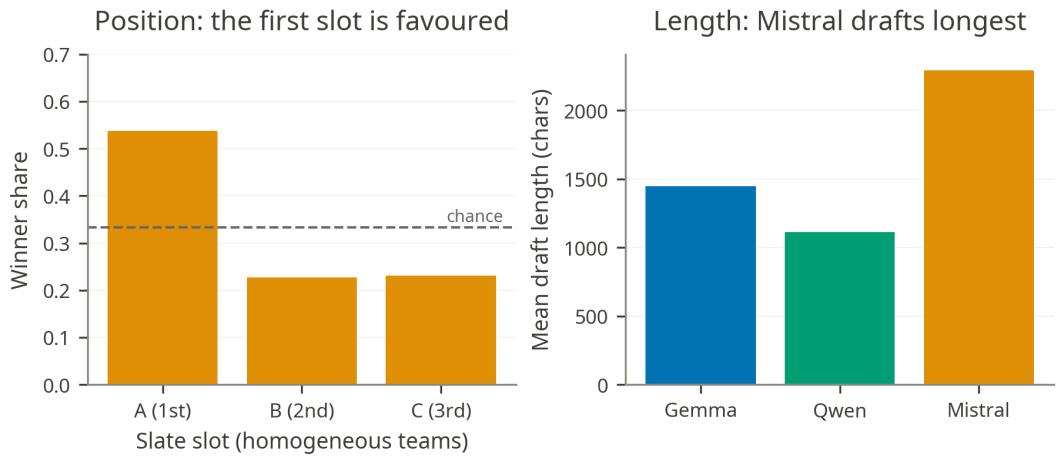



Figure F.5: Consensus from homogeneous teams (model held constant). Left: winner share by slate slot; the ranking favours the first slot, which Mistral never occupies. Right: mean draft length by member in heterogeneous teams; Mistral's drafts are the longest, yet within homogeneous teams the longest draft wins only 38% of votes, near the 33% chance baseline.