The Price of Thought: A Multilingual Analysis of Reasoning, Performance, and Cost of Negotiation in Large Language Models

Sherzod Hakimov[1], **Roland Bernard**[3], **Tim Leiber**[1], **Karl Osswald**[1], **Kristina Richert**[1], **Ruilin Yang**[1], **Raffaella Bernardi**[3], **David Schlangen**[1,2]

[1]Computational Linguistics, Department of Linguistics
University of Potsdam, Germany
[2]German Research Center for Artificial Intelligence (DFKI), Berlin, Germany
[3]Free University of Bozen-Bolzano, Italy
{firstname.lastname}@uni-potsdam.de, {firstname.lastname}@unibz.it

Abstract

Negotiation is a fundamental challenge for AI agents, as it requires an ability to reason strategically, model opponents, and balance cooperation with competition. We present the first comprehensive study that systematically evaluates how explicit reasoning training affects the negotiation abilities of both commercial and open-weight large language models, comparing these models to their vanilla counterparts across three languages. Using a self-play setup across three diverse dialogue games, we analyse trade-offs between performance and cost, the language consistency of reasoning processes, and the nature of strategic adaptation exhibited by models. Our findings show that enabling reasoning—that is, scaling test time compute—significantly improves negotiation outcomes by enhancing collaboration and helping models overcome task complexities, but comes at a substantial computational cost: reasoning improves GPT-5's performance by 31.4 % while increasing its cost by nearly 400 %. Most critically, we uncover a significant multilingual reasoning distinction: open-weight models consistently switch to English for their internal reasoning steps, even when negotiating in German or Italian (and thus possibly impacting potential explainability gains through the disclosure of reasoning traces), while a leading commercial model maintains language consistency between reasoning and final output.

1 Introduction

Negotiation is a key aspect of human social and economic behaviour that poses significant challenges for AI systems. The problem is complex, as agents must produce coherent language while performing strategic reasoning, modelling the opponents' preferences and goals, and managing the tension between cooperation and competition to maximise their own outcomes. With LLMs increasingly being deployed as autonomous agents in domains ranging from e-commerce to resource distribution, assessing their negotiation capabilities has become essential. Negotiation requires strategic decision-making, which fundamentally depends on reasoning, and provides a strong test-bed for measuring agent competence through interactive, multi-turn scenarios with objective measures that go beyond static evaluation. We therefore study LLM reasoning abilities in negotiation settings.

Studies show that LLMs often deviate from optimal play with unreliable performance, where even top-performing systems can lose to weaker opponents and fail in cooperative scenarios (Davidson et al., 2024; Hua et al., 2024; Bianchi et al., 2024; Kwon et al., 2024). These models can acquire deceptive tactics, e.g., expressing false interest in low-value items for later concessions (Lewis et al., 2017), express desperation to improve outcomes (Bianchi et al., 2024), perform much worse as buyers than sellers (Xia et al., 2024), or even take economic risks that lead to budget violations and overpayment (Zhu et al., 2025). Previous studies have also investigated theory-of-mind reasoning in negotiation tasks (Kwon et al., 2024), and their performance in interactive multi-agent negotiation games and the effect of Chain-of-Thought reasoning (Chan et al., 2024; Abdelnabi et al., 2024).

Despite this growing body of research, two critical gaps remain unexplored. First, no previous work has systematically investigated how vanilla LLMs specifically trained to reason perform on negotiation tasks and their effects in terms of both performance and computational cost, a fundamental oversight given that reasoning is central to strategic decision-making. Second, the field has been constrained by monolingual analyses based solely on English, leaving multilingual negotiation capabilities completely unexplored. To address these significant limitations, we conduct the first comprehensive study examining the effects of reasoning on negotiation tasks across three languages: English, German, and Italian. We implement three dis-

Findings of the Association for Computational Linguistics: EACL 2026, pages 529–570

tinct dialogue games that require bargaining skills, collaboration, and strategic preference management (von Neumann and Morgenstern, 1944; Nash, 1953; Fisher et al., 2011), testing both commercial and open-weight LLMs in a self-play mode where both sides of the games are played by instances of the same model. We systematically analyse the impact of turning on and off the reasoning mode of models. We also investigate whether models retain their language consistency in their reasoning traces or if they switch to a dominant language in their training data (mainly English). It is important to study, as the reasoning traces provide essential cues for the explainability of model decisions. We target three fundamental research questions:

RQ1: *What is the computational and performance trade-off between reasoning overhead and negotiation effectiveness across tasks and languages?*

RQ2: *To what extent do models maintain language consistency in their reasoning processes when performing multilingual negotiation tasks?*

RQ3: *Do models demonstrate strategic adaptation over multiple turns, or merely surface-level pattern matching creating an illusion of thinking?*

2 Related Work

The convergence of Large Language Models (LLMs) and game-theoretic frameworks represents a significant development in the evaluation and understanding of strategic capabilities in AI systems (Sun et al., 2025), as we explain in the next two parts the state-of-the art methodologies in this field.

Benchmarking strategic capabilities in LLMs: systematic evaluation of LLM negotiation performance reveals substantial variance in strategic behaviour, with models frequently departing from optimal play and exhibiting asymmetric performance patterns (Davidson et al., 2024; Hua et al., 2024; Akata et al., 2023; Pollo et al., 2025), or even some bigger models underperform against weaker opponents and encounter difficulties in cooperative bargaining scenarios. This has motivated the creation of comprehensive evaluation frameworks and experimental platforms designed to assess agent (initialised as LLMs) behaviour across various negotiation contexts, including resource allocation and pricing tasks (Bianchi et al., 2024; Xia et al., 2024). Empirical studies reveal that language models can display interesting tactics, e.g. deceptive strategies such as expressing false interest in low-value items to create bargaining leverage in subsequent exchanges (Lewis et al., 2017). LLMs also exhibit susceptibility to cognitive biases, including anchoring effects and responsiveness to social manipulation tactics, where expressions of desperation or aggressive language can substantially affect negotiation outcomes (Bianchi et al., 2024). As these behaviours present practical risks, outcomes may result in budget constraint violations and acceptance of economically disadvantageous agreements (Zhu et al., 2025). A consistent finding across studies is the asymmetric difficulty in buyer versus seller roles, with agents systematically underperforming as buyers (Xia et al., 2024). Our work focuses on studying the reasoning behind model choices in a multilingual setting (Ghosh et al., 2025).

Improving strategic reasoning and decision-making: Given the inconsistent strategic reasoning observed in current LLMs (Wong et al., 2025), research efforts focus on enhancing decision-making through structured approaches such as integration of game-theoretic solvers with LLM dialogue (Gemp et al., 2024), utilising post-hoc CoT prompting (Abdelnabi et al., 2024), structured reasoning workflows based on Dominant Strategy Search and Backward Induction (Hua et al., 2024), or even learn by interacting based with reinforcement learning (Cao et al., 2018). From a prompting perspective, Chain-of-Thought reasoning emerges as a critical factor in agent performance (Gandhi et al., 2023), contributing to more consistent outcomes in integrative negotiation settings (Vaccaro et al., 2025). Hybrid architectural approaches have been developed to constrain agent behaviour, such as employing deterministic offer generation modules to control pricing decisions while utilising LLMs for natural language dialogue production, resulting in substantial improvements in deal completion rates and profit margins (Xia et al., 2024). The focus of this paper is on evaluating the effect of reasoning on negotiation strategies without adding any additional component by simply using the vanilla LLMs that are trained specifically to exhibit reasoning capabilities.

3 Evaluating Negotiation Abilities with Dialogue Games

In this section, we provide details on how we implement three dialogue games to test negotiation abilities in LLMs. We define a dialogue game as a structured communication between two agents

(LLMs) according to a given communication protocol aimed at achieving a defined goal. The goals in games are defined as the expected outcome, e.g., making a deal that agents attempt to maximise. The communication protocols dictate how messages are expected to be formatted. A negotiation task unfolds by agents starting to communicate with one another for the given goal, and the conversation continues until it reaches one of the defined stopping criteria: 1) the goal state is reached, 2) the maximum turn limit is reached, or 3) agents violate the communication protocol and the game-play is aborted. The orchestration of this message passing between agents, validation of communication protocols, and checking whether goal states are reached are done by the Game Master, a scripted entity in the dialogue. We use the *clembench framework* (Chalamalasetti et al., 2023). The dialogue games are run in a self-play mode where both participants of the game are LLMs playing against or with one another. Next, we provide game details[1].

3.1 Deal or No Deal

The Deal or No Deal (DoND) game is a two-player game (Lewis et al., 2017) designed to simulate a multi-issue bargaining scenario. Figure 1 shows an example. Two players must communicate to reach a mutually beneficial agreement on a set of items, each holding different values for each player. The game focuses on evaluating negotiation skills, including the ability to express and understand preferences, as well as compromise (see Appendix B for prompts and other details).

Game Mechanics Two agents negotiate over a shared set of items, each having a different private value for item types. Players exchange free-form messages and then simultaneously make secret proposals using specific syntax. If proposals are compatible (enough items are requested for both), each player receives points based on the value of their requested items, as determined by their private function. If proposals conflict, both receive zero points. The Game Master strictly enforces rules without re-prompting, making adherence to rules critical for a successful play experience.

Game Instances We generate instances for the game programmatically by randomly sampling 100 common nouns. Each instance was generated with a maximum turn limit of 5, and each game instance features a random set of between 3 and 5 different item types. We have two different sets of experiments. In the **semi-competitive** experiment, players are asked to maximise their own score (to balance cooperation, avoid zero-point conflicts). For the **cooperative** experiment, players are asked to maximise the sum of the scores of both players. This transforms the game into a purely cooperative task, in which two rational agents should consistently achieve the maximum score, assuming they can communicate and reason about the preferences perfectly. Each experiment includes 20 instances, with **total 40 instances**.

#	Player A	Game Master	Player B
1		[A⟨GM⟩B] <GAME DESCRIPTION> ... Available items: 1 book, 2 hats, 2 balls.	
2		[A⟨GM] Player A value function: book: 0, hat: 1, ball: 4.	
3		[GM⟩B] Player B value function: book: 6, hat: 1, ball: 1.	
4	[A⟩B] Hi! Here are my preferences: I value balls the most, then hats, and I don't value the book at all. How about you? What do you value most?		
5			[A⟨B] Thanks for sharing! I value the book most, while hats and balls are equally less important to me. Would you take both balls and one hat, and I'll take the book and the other hat? That way, we both get what we value most.
6	[A⟩B] That sounds fair! With this split, I'd get both balls and one hat, while you'd get the book and one hat, so we each get what we value most. Ready to submit this as my proposal unless you'd like to tweak anything.		
7			[A⟨B] I agree with the split. I'll take the book and one hat, you take both balls and the other hat.
8	[A⟩GM] [Proposal: 1 hat, 2 balls]		
9		[GM⟩B] The other player has submitted a proposal. You must now submit your proposal.	
10			[GM⟨B] [Proposal: 1 book, 1 hat]

Figure 1: An example of a *Deal or No Deal* episode that ends in a successful Pareto optimal agreement.

3.2 Clean Up

Clean Up is a two-player game focused on cooperative strategy development and object rearrangement. Object rearrangement has been a long-standing problem that requires modelling the given situation, the action space, and spatial reasoning (Batra et al., 2020; Zeng et al., 2024; Khan et al., 2025). Each player can only access its own grid and need to negotiate which items to rearrange so that the grids become identical at the end (Jeknic et al., 2024). A sample episode is shown in Figure 2. We provide all prompts and other details in Appendix C.

[1] https://github.com/clembench/clembench

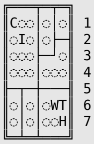
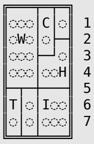
#	Player A	Game Master	Player B
1		[A⟨GM] <GAME DESCRIPTION> ... 1234567  Please send a message to the other player.	
2	[A⟩GM] SAY: Hi! Let's coordinate our moves. [...]		
3		[GM⟩B] <GAME DESCRIPTION> ... 1234567  The other player started the game by sending this message: <MESSAGE> What is your first command?	
		⋮	
4			[GM⟨B] MOVE: C, (1,1)
5		[A⟨GM] Your message has been relayed to the other player. <GAME STATE> The other player moved an object on their grid. What is your next command?	
6	[A⟩GM] MOVE: I, (5,6)		
		⋮	
7			[GM⟨B] SAY: I think our grids are aligned now, finished?
8	[A⟩GM] SAY: finished!		

Figure 2: Example episode from Clean Up game to achieve a common goal configuration for a number of objects randomly placed on each player's grid, and move them accordingly. Finally, both players have to agree the goal is reached to end the game.

#	Player A	Game Master	Player B
1		[A⟨GM] <GAME DESCRIPTION> ... -Items and weights: Ball(2), Book(3), Magazine(1), Rope(4), Lamp(5) -Max Weight: 7 -Your preferences: Ball=1, Book=2, Magazine=2, Rope=2, Lamp=1	
2	[A⟩GM] STRATEGIC REASONING: Magazine, Ball, Book have the best value-to-weight ratios while respecting the weight limit. PROPOSAL: Magazine, Ball, Book ARGUMENT: These items balance weight and value well for me. What do you think?		
3		[GM⟩B] <GAME DESCRIPTION> ... <Items, weights, weight limit> -Your preferences: Ball=1, Book=1, Magazine=2, Rope=4, Lamp=5 <PLAYER A Message without STRATEGIC REASONING>	
4			[GM⟨B] STRATEGIC REASONING: I can increase the value of the other player's proposal for me by swapping Rope with Book. Let me propose this. PROPOSAL: Rope, Ball, Magazine ARGUMENT: I value Rope highly, but not Book. How about we swap them?
5		[A⟨GM] <PLAYER B Message without STRATEGIC REASONING>	
6	[A⟩GM] STRATEGIC REASONING: I value Rope and Book equally. Weight of proposal is 4+2+1=7 which matches the limit. I should accept. AGREE: Rope, Ball, Magazine ARGUMENT: I am indifferent w.r.t. Rope and Book. Fair deal.		

Figure 3: An example episode of the *Air Balloon Survival* game. Two players must negotiate and argue for their preferred set of items. and must explicitly agree to a proposal made by the other.

Game Mechanics Both players are presented with 7×7 ASCII grids, with a number of randomly distributed objects in the form of capital letters placed on them. In each turn, a player can either send a message to their counterpart or move an object on their grid. The message and move text require certain formatting rules to follow. If a player does not follow the format, tries to move an object to a non-empty space or outside the grid bounds, or tries to move an object that doesn't exist, they receive a penalty and are re-prompted with information on the nature of their mistake. The game ends if (1) both players agree to end it, (2) round limit is exceeded, or (3) penalty limit is exceeded.

Game Instances All instances are created programmatically by creating different grids (7×7), which contain obstacles in form of horizontal and vertical lines, branches, crossings, and corners. The maximum number of rounds is fixed to $4 \times n_{obj}$ (where n_{obj} is the object count). We create different sets of experiments by controlling two aspects: number of empty cells and objects. We have three difficulty levels. Of the 49 total cells, 34 of cells are empty on the *easy*, 29 on *medium*, and 24 on *hard* levels. For each level, we sample three grids, and then place 3, 5, or 7 objects on them, making for **27 instances in total**.

3.3 Air Balloon Survival

This game evaluates advanced reasoning and interactive collaboration between players, which was previously introduced by Howes et al. (2021) to study how patients with schizophrenia verbalise their reasoning during social encounters. In our version, two players are on a sinking hot air balloon, and they have to agree on which items to keep (or throw out) so that the weight of the balloon is reduced to keep floating. Each player has hidden preference values for items and must negotiate to maximise their combined utility score. An example episode is given in Figure 3. The game tests individual reasoning through constraint-based optimisation, requiring arithmetic and combinatorial search, collective reasoning through practical rationality, and theory of mind (to infer their counterpart's hidden preferences from their responses to reach an optimal agreement). We provide all prompts and other details in Appendix D.

Game Mechanics Both players receive their assigned preference values for the items. They are instructed to use specified formats when making proposals and engaging in negotiations. The game is aborted if no progress is made for eight consecutive turns. Unlike the other two games, players are also instructed to output their `STRATEGIC REASONING` along with the expected message. Only the expected message is passed to the other player. It allows players to "think out loud" about their choices. The game ends when a proposal made by one player is accepted by the other.

Instance Generation We draw either 15 or 35 items (depending on the experiment) from a randomly generated list constructed by concatenating a capital letter with a two-digit number (e.g., `A42`, `C07`)[2]. We assign a weight value to each item. The air balloon's capacity is defined as a fraction of the combined weight of all items. We experiment with two negotiation levels, where we specifically set valuations over items to be the same for players, giving them common goals (easy level), or we invert their preference orderings, giving them opposing goals (hard level). We also experiment with generating the `STRATEGIC REASONING` or not. Lastly, we conduct complexity experiments by increasing the number of items. In total, we have six experiments with each having six instances, which leads to the **total number of 36 instances** for this game.

4 Experimental Setup

4.1 Game Instances

Deal or No Deal, *Clean Up*, and *Air Balloon Survival* games include 40, 27, and 36 experimental instances, respectively. Each instance is then initialised with defined prompt templates. The same game instances are used for the English, German, and Italian experiments because the instances are language-agnostic; only the prompt templates need to be aligned for a specific language.

[2]This keeps the naming of items language-agnostic and abstract from conventional examples of the *0/1 Knapsack Problem* models that may have been seen during training

4.2 Evaluation Metrics

The *clembench framework* (Chalamalasetti et al., 2023) requires each implemented game to provide two primary metrics: % Played and Quality Score. The *% Played* stands for the percentage of episodes where the evaluated language models followed instructions and the game was terminated by the defined end states. In cases where the gameplay does not fit the defined states of the game, the Game Master either tolerates such behaviour for one or two turns, asks the players to try again, or aborts the game. The *Quality Score* is calculated using an objective function to measure how closely the played episode aligns with the target goal. In each game, this metric is calculated for all episodes that have been played. Once two metrics are calculated, they are aggregated to a single number, the *clemscore* as the normalised product of *% Played* and *Quality Score* (scaled to the interval [0, 100]). Next, we describe the Quality Scores for games.

Deal or No Deal For the cooperative game mode, we measure quality as the ratio of achieved total score to maximum possible total score. For the semi-competitive setting, we use a *Pareto efficiency* metric that measures how far the agreement is from optimal by calculating potential one-sided improvements (more details in Appendix B.2).

Clean Up The Quality Score combines two main components: how well players organised objects spatially and how many rule violations they incurred. The metric uses *Euclidean distances* between matching objects to compare the final arrangement to both the initial setup and a random baseline, rewarding improvements in object alignment while penalising rule violations through a scaling factor that becomes increasingly harsh as violations approach the maximum allowed limit (more details in Appendix C.2).

Air Balloon Survival Each player receives a score based on their achieved utility relative to their optimal knapsack solution. The overall game score uses the harmonic mean of both players' normalised scores, which rewards balanced outcomes and penalises deals where one player benefits disproportionately (more details in Appendix D.2).

4.3 Evaluated Models

We have evaluated both commercial and open-weight models that have reasoning functionality. We selected *GPT-5*, *GPT-5-mini*, *Claude-4* from

		GPT-5		GPT-5 mini		CL-4		LM-70B		Nem-9B		Qwen-3		GPT OSS	DS-v3.1
	Games	On	Off	On	Off	On	Off	On	Off	On	Off	On	Off	On	Off
EN	DoND	87.9	32.3	75.7	23.7	**94.4**	90.1	24.3	43.5	18.5	4.0	57.9	22.5	50.0	59.0
	Clean Up	**99.8**	75.2	96.6	77.3	85.5	81.9	4.1	28.1	28.5	35.1	87.9	35.8	81.4	76.4
	Air Balloon	**98.0**	80.1	97.5	82.2	83.8	26.2	2.3	40.9	14.3	13.2	88.7	0	78.2	81.7
	Average	**95.2**	62.5	89.9	61.1	87.9	66.1	10.2	37.5	20.4	17.4	78.2	19.4	69.9	72.4
	Margin	+32.7		+28.8		+21.8		-27.3		+3.0		+58.8		-	-
DE	DoND	**86.0**	19.6	82.9	24.4	80.3	77.1	21.8	53.4	0	4.5	62.1	12.8	34.9	42.4
	Clean Up	98.2	70.4	**98.8**	67.0	94.3	83.1	5.3	25.3	23.1	0	86.4	27.5	74.4	53.7
	Air Balloon	**98.0**	81.4	94.9	86.2	85.4	27.8	51.5	43.7	17.4	16.3	76.0	8.8	90.7	75.3
	Average	**94.1**	57.1	92.2	59.2	86.7	62.7	26.2	40.8	13.5	6.9	74.8	16.4	66.7	57.1
	Margin	+37.0		+33.0		+24.0		-14.6		+6.6		+58.4		-	-
IT	DoND	78.0	33.8	77.1	26.8	**84.4**	67.8	17.3	14.3	13.9	11.2	55.6	19.4	29.0	28.9
	Clean Up	89.0	68.8	**93.6**	77.2	91.6	82.6	6.5	31.1	21.5	12.7	82.6	31.5	63.6	67.6
	Air Balloon	97.2	88.0	**97.3**	87.8	79.4	27.0	2.7	40.8	0	0	77.8	12.9	83.7	76.4
	Average	88.1	63.5	**89.3**	63.9	85.1	59.1	8.8	28.7	11.8	8.0	72.0	21.3	58.8	57.6
	Margin	+24.6		+25.4		+26.0		-19.9		+3.8		+50.7		-	-
Overall Average		**92.5**	61.1	90.5	61.4	86.6	62.6	15.1	35.7	15.2	10.8	75.0	19.0	65.1	62.4
Overall Margin		+31.4		+29.1		+24.0		-20.6		+4.4		+56.0		-	-

Table 1: *clemscore* values for three negotiation games on selected LLMs for English, German, and Italian versions. *On*: reasoning mode is turned on, *Off*: reasoning mode is turned off. The best result for each language in each row is highlighted in bold. *CL*: Claude, *LM*: Llama-3.3, *DS*: Deepseek, *Nem*: Nemotron-v2, *DoND*: Deal or No Deal

commercial models. From open-weight models, we selected *Llama3.3-70B* and *Deepseek-R1-distilled-llama-70B* as its reasoning counterpart, *Nemotron-Nano-9B-v2*, *Qwen-3-80B* (*instruction* and *thinking* versions for reasoning off and on modes, respectively), *GPT-OSS-120B* with reasoning on mode, and finally *Deepseek-v3.1* with reasoning off mode. For commercial models, we ran them via their respective API backends, and we used *OpenRouter.ai* for open-weight models.

5 Results

5.1 Overall Performance

We present the results obtained by the models (in *reasoning on* and *off* modes, where possible) evaluated for the English, German, and Italian versions of the negotiation games in Table 1.

The Effect of Reasoning: the most striking observation here is that reasoning mode dramatically improves performance across many models and languages, with *Qwen-3* gaining 56 points averaged across all games. Similar pattern exists for *GPT*, *Claude* and even smaller *Nemotron* models. This is a strong indication that deliberate reasoning significantly enhances strategic game-playing abilities. Only with *Llama-70B* do we see different results, which may be due to the effect of distillation.

Multilingual Capabilities: results for German shows the largest average margin between *reasoning on* and *off* modes of models. English and Italian performances are also strong, suggesting that certain models have particularly sufficient negotiation capabilities across these evaluated languages.

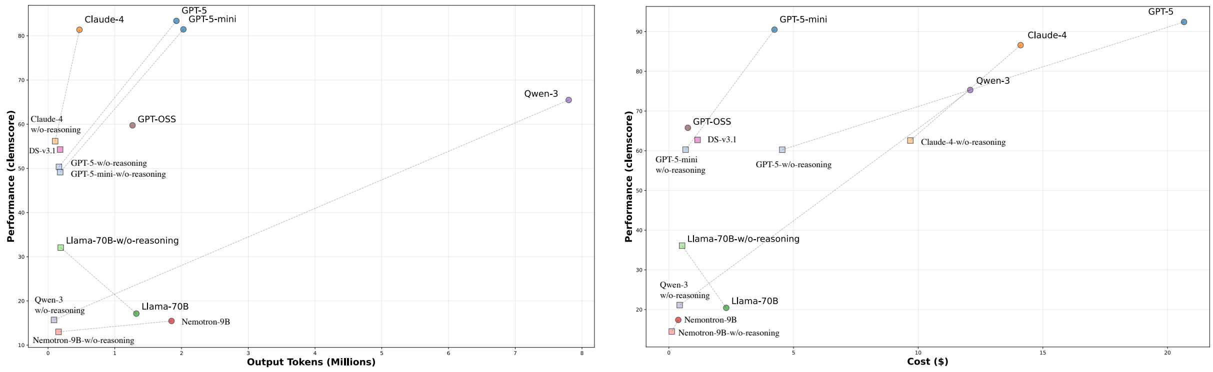
Model Comparison: GPT-5 is the clear winner, with GPT-5-mini and Claude-4 getting very close performance across all languages. Interestingly, GPT-5 mini sometimes matches or exceeds full GPT-5 performance (particularly in Italian). *Qwen-3* shows the biggest performance jump.

5.2 Trade-off between Reasoning Overhead and Performance

RQ1: *What is the computational and performance trade-off between reasoning overhead and negotiation effectiveness across tasks and languages?*

In Figure 4a, we present a plot that shows the average performance across all games and languages and how many output tokens it requires. Here, all reasoning and completion tokens are summed up. Most models generate the number of tokens that are closer to each other, and *Qwen-3* generates almost *4x* more tokens than others.

In Figure 4b, we present the overall cost (input and output tokens together) of experiments for each model. As expected, models with reasoning mode cost more, with *GPT-5*, being the most expensive, costing almost *4x* compared to non-reasoning version while improving 31.4 points in clemscore. In terms of deciding on the best trade-off between performance and cost, ***GPT-5-mini* is the most cost-efficient commercial model, and *GPT-OSS* is the best open-weight one, with a fraction of**



(a) Performance and the output tokens for all evaluated models with their reasoning on/off modes averaged for three languages.

(b) Performance and the cost of for all evaluated models with their reasoning on/off modes averaged for three languages.

Figure 4: Trade-off between performance and cost comparison averaged across languages.

the cost compared to Qwen-3.

5.3 Language Consistency in Reasoning

RQ2: *To what extent do models maintain language consistency in their reasoning processes when performing multilingual negotiation tasks?*

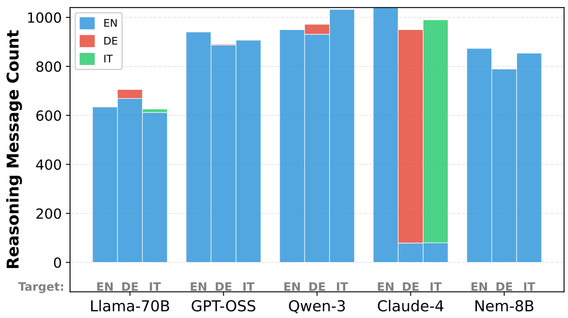


Figure 5: Language distribution in reasoning tokens

		DE	EN	IT
Commercial	completion	1.00	1.00	1.00
	reasoning	0.84	1.00	0.84
Open-weight	completion	0.94	1.00	0.87
	reasoning	0.06	1.00	0.03

Table 2: Language consistency in reasoning tokens

We ran a language detection script to analyse the reasoning message of the models. The results are presented in Figure 5[3]. We clearly see that all open-weight models mostly generate reasoning tokens in English, whereas *Claude-4* "thinks" in the respective language of the task, consistently across all three of them. In Table 2, we provide the percentage of language consistency (whether the language of the message matches the target language) across *completion* and *reasoning* tokens.

We can clearly see that *Claude-4* (the only commercial model in the analysis) exhibits consistency in both completion and reasoning tokens. Open-weight models retain consistency in completion tokens (thus, abiding by the formatting rules as each game requires prefixes in the respective language), but fail to do so with reasoning tokens for German and Italian versions of the games. Thus, we can conclude that **open-weight models do not maintain language consistency in their reasoning processes across multilingual tasks**. Similar patterns have been also observed by Qi et al. (2025) where explicitly forcing the model to reason in the respective language even reduces the performance. Thus, the models that do not maintain language consistency in their reasoning steps **become less interpretable and less trustworthy**.

5.4 Strategic Adaptation Across Multi-Turn Interactions

RQ3: *Do models demonstrate strategic adaptation over multiple turns, or merely surface-level pattern matching creating an illusion of thinking?*

5.4.1 Keyword-based Analysis

We label the first-turn reasoning traces as sequences over five discrete states—`ASSERT`, `PROPOSE`, `UNDERMINE`, `ALTERNATIVE`, `CONCLUDE`. The states are detected using lexical cues in reasoning tokens, e.g. "maybe/could": `PROPOSE`; "but/however": `UNDERMINE`, etc. Each trace is a path on a finite-state machine with observed transitions (including self-loops), from which we compute per-trace statistics and aggregate at model/game level. The finite state machine is shown in Figure 6. We

[3]Note: *GPT-5* models do not return reasoning tokens, thus they are excluded from further analysis

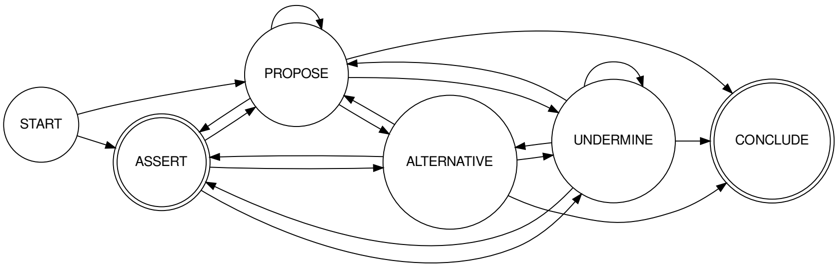
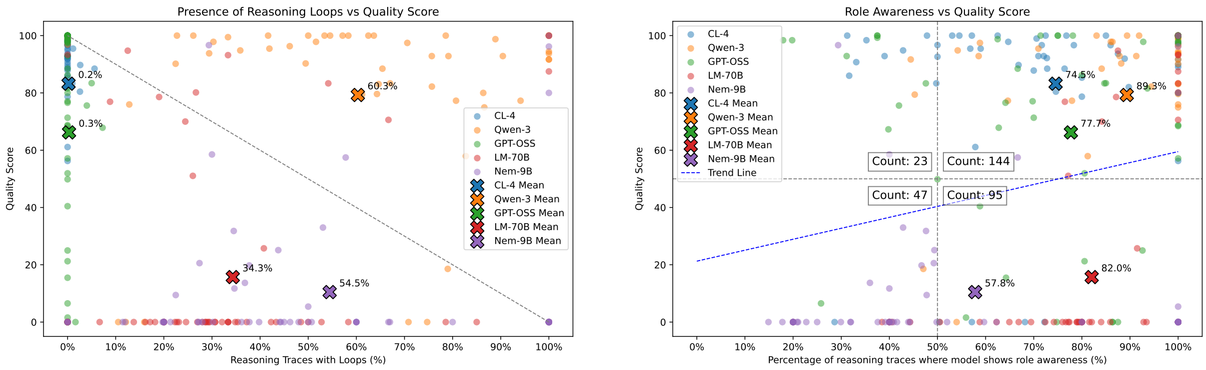


Figure 6: Finite state machine for analysing reasoning traces



(a) Reasoning traces that display thinking loops (at least three repetitions of the same action) plotted

(b) Reasoning traces where the model displays both awareness of their own another player's role

Figure 7: Analysis of patterns in model reasoning traces

compute two metrics using the transitions among states (more details in Appendix A.3):

number of segments: we partition the labels extracted from each reasoning trace by absorbing states (`ASSERT` and `CONCLUDE`). The count serves as coarse "reasoning units": higher counts could mean more dispersed reasoning (multiple restarts before final commitment); lower counts could mean more concentrated reasoning (a plan developed and resolved within fewer units).

cycle edge ratio: within each segment's induced subgraph, we mark edges that belong to any simple cycle and compute the fraction of transitions on such edges, then aggregate across segments to obtain one single metric for each reasoning trace. Higher ratios indicate local looping/hedging (e.g., `PROPOSE` <-> `UNDERMINE`); lower ratios indicate forward motion towards absorbing states.

Good-performing models should resolve plans in fewer segments and cycles to have more goal-oriented planning and execution of it. The results are given in Figure 12 and Figure 13. Performance shows non-linear relationships: **minimal segment counts correlate with improved outcomes**, while moderate additions yield unclear results. **Commercial models achieve competitive scores with lower cycle ratios** and tighter clustering; open-weight models show higher ratios with greater variance, suggesting that additional cycles do not reliably translate into performance gains.

5.4.2 LLM-based Analysis

For all thinking models with available reasoning traces, we sampled a total of 309 transcripts with at least one instance for each game, language, and experiment combination for automated analysis. We prompted *GPT-5* to analyse certain aspects of the traces and give their output in JSON format, including not only the analysis, but also short explanations for each classification (see example in Figure 14). More details are in Appendix A.4.

Figure 7a shows the thinking loops that occur in reasoning. We consider a thought to be in a loop if it occurs at least three times. Good-performing model results should be clustered around the top left, including fewer loops and a high quality score. *Claude-4* and *GPT-OSS* rarely display loops (0.2% and 0.3%) and achieve high scores, while *Llama-70B* (34.3%) and *Nemotron-9B* (54.4%) contain loops and score low. 95.18% of data points for these models fall below the dotted diagonal, indicating that reasoning loops, or *overthinking* (Shojaee et al., 2025), reduce performance. Conversely, *Qwen-3* shows 60.3% of traces with loops but averages 79.3 quality score. These excessive loops explain *Qwen-3*'s high token counts.

Since the games prompt the model to be active agents, we analyse the reasoning traces for both awareness of their own player role (whether the model is helping or advising somebody else, or actively playing the game itself) and of the partner (mention of any other agentic participant).

Figure 7b shows the percentage of reasoning traces where the model is both aware of its own role as an active player and actively mentions another player. Naturally, the mean percentages for all models are somewhat lower than those for assuming a player role, with the 74.5% for Claude-4, 89.3% for Qwen-3, 78.2% for GPT-OSS, 82.0% for Llama-70B and 57.8% for Nemotron-9B.

Of all games with a quality score over 50, 144 (46.6% of games) show role awareness in over 50% of reasoning traces, and only 23 in under 50%. Of the games with a quality score below 50, 95 (30.7%) show higher and 47 (15.2%) lower role awareness. This indicates that **awareness of both one's own role and the existence of a counterpart are prerequisites for consistently high scores.**

5.5 Qualitative Analysis

We provide qualitative samples in Figure 8 from German task for reasoning tokens generated by *Qwen-3* and *Llama-70B* (distilled from Deepseek-R1). The samples clearly show how the LLMs do not adhere to the target language. The top example shows how languages are interchangeably used, with Russian being the common one. It starts with: Вижу, что "I" находится в клетке (1,3) ... Я пытаюсь переместить его в (3,3). Посмотрю, primitive cell (3,3) ... В этой клетке сейчас ○, значит, пустая., which translates to *I can see that there is "I" in cell (1,3) ... and I try to move it to (3,3) ... I look ... in this cell there is* ○, *meaning it is empty*.

The middle example shows how the model reasons about its choices and weights and tries to calculate what it means to consider one of them, and interchangeably uses eight languages.

The bottom example's reasoning is about modelling the opponent where the roles are confused because it refers to an "assistant" (助理). But the actual task is for the assistant to act as the user, and the opponent is simulated.

Arabic/Persian German Greek English Italian
Chinese/Japanese Portuguese Russian Spanish Neutral

Вижу, что "I" находится в клетке (1,3), т.е. x=1, y=3. Я пытаюсь переместить его в (3,3). Посмотрю, primitive cell (3,3): в данном случае, y=3, это строка 3, x=3 — третья колонка. В этой клетке сейчас ○, значит, пустая. Значит,ální чистовесі должны позволять это MakeVisible. [...] Проверю координаты еще раз: x=3, y=3. Это колонка 3, строка 3. Согласно сетке, это позиция, где в данный момент нет объектов — تمام، تو , что я могу переместить туда "I". Но Spielmeister zvalidur Squadron Bezug на это как на(setPosition invalid. Может быть, что используемые значения x и y должны быть в рамках сетки, и я certainty relative nebuario этого. Попробую ارسل ить команду еще раз, осторожно. шнид POSTAVI Canterville 確認 dort.

(a) Llama-70B reasoning tokens for Clean Up

Habe used 8226, weniger als 8770. Habe 544 frei, καὶ könnte nochождение aggiunta von B75 (520) hinzufügen, summa que seria 8226+520=8746 حركت hijo. 包括 B75 im Angebot..functions participants aim. So, der neue Vorschlag wäre: {'C14', 'A77', 'C16', 'B86', 'B60', 'A81', 'B79', 'C28', 'C31', 'B55', 'B38', 'C05', 'B96', 'A91', 'B54', 'C50', 'A40', 'B59', 'A52', 'B56', 'C03', 'B75', 'A95'} Isso teria Aufwand 8226 + 520 = 8746, o que está dentro do limite de 8770. me traz mais 520 masculino a utilidade de B75, que tem 235 de importância. Assim, melhorei minha oferta. Agora, o meu argumento ciclo de'.

(b) Llama-70B reasoning tokens for Air Balloon Survival

Suppose the opponent has high value on pens. For example, if the opponent values pens at 3 each, then getting 1 pen + sponge (if their sponge value is 1) is total of 3+1=4. But if they get all pens (3*3=9) and give the assistant the H and sponge (but the assistant's H is worth 4, but opponent doesn't care), but wait. In the opponent's proposed scenario where opponent takes H and sponge,助理 gets all pens.

(c) Qwen-3 reasoning tokens for Deal or No Deal

Figure 8: Qualitative samples showing reasoning tokens where the target language is German

6 Conclusion

We conducted the first comprehensive study systematically investigating the impact of reasoning on performance in multilingual negotiation tasks. Our work examines the trade-offs between computational cost and negotiation effectiveness, assesses the linguistic consistency of reasoning processes, and probes the depth of strategic adaptation in English, German, and Italian. Our findings reveal that test-time scaling is a powerful and costly tool: it significantly enhances negotiation performance, yet demands substantial additional compute. Open-weight models almost exclusively switch to English for internal reasoning, even when performing tasks in German or Italian, while leading commercial models maintain language consistency between reasoning and outputs. Our results further suggest that reasoning enables genuine strategic adaptation rather than simply pattern matching. The observed improvements in handling complex rules, making value-based decisions, and achieving collaborative outcomes point to deeper problem-solving unlocked when models "think out loud" before acting. Future work should extend this investigation to broader dialogue games, more diverse languages, and emerging models to chart the path toward AI agents that are versatile negotiators.

Limitations

The first limitation is about the scope of negotiation tasks. Some researchers also employed other dialogue games that we did not include in this study, as there are many similarities among games and the measures they assess. Another limitation is the choice of languages for analysis, which are three European languages that are also high-resource languages. The primary reason for considering only the defined dialogue games and languages is that including more games or languages would result in a higher number of model results, thereby increasing the overall cost of the study. Another limitation is the exclusion of models from both commercial and open-source ones. As explained above, such decisions would exceed the budget for such a study. Given the obtained results regarding switching languages, it is evident that most models lack natural reasoning aspects, and utilising such models in real-world applications would bring its own set of challenges and limitations. For use cases that involve cultural or language-specific concepts or aspects, it can not be guaranteed that they will be well-understood in English (as most languages switch to it).

Ethical Considerations

Using paid proprietary APIs with underlying models about which little is known (training data, model architecture) in academic research is less than ideal. Currently, the models tested here support reasoning modes, either on or off (except for GPT-OSS-120B and Deepseek-v3.1). We hope that open models will include more controls over reasoning aspects and catch up soon in terms of general performance. Deploying such language agents in society has considerable risks. As mentioned in the Related Work, such agents exhibit asymmetric behaviours in negotiations and may expose users to unfaithful actions.

Acknowledgments

We thank all anonymous reviewers who helped improve the manuscript.

References

Sahar Abdelnabi, Amr Gomaa, Sarath Sivaprasad, Lea Schönherr, and Mario Fritz. 2024. LLM-deliberation: Evaluating LLMs with interactive multi-agent negotiation game. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Elif Akata, Lion Schulz, Julian Coda-Forno, Seong Joon Oh, Matthias Bethge, and Eric Schulz. 2023. Playing repeated games with large language models. *CoRR*, abs/2305.16867.

Dhruv Batra, Angel X. Chang, Sonia Chernova, Andrew J. Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, Manolis Savva, and Hao Su. 2020. Rearrangement: A challenge for embodied AI. *CoRR*, abs/2011.01975.

Federico Bianchi, Patrick John Chia, Mert Yüksekgönül, Jacopo Tagliabue, Dan Jurafsky, and James Zou. 2024. How well can llms negotiate? negotiationarena platform and analysis. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Kris Cao, Angeliki Lazaridou, Marc Lanctot, Joel Z. Leibo, Karl Tuyls, and Stephen Clark. 2018. Emergent communication through negotiation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Kranti Chalamalasetti, Jana Götze, Sherzod Hakimov, Brielen Madureira, Philipp Sadler, and David Schlangen. 2023. clembench: Using game play to evaluate chat-optimized language models as conversational agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 11174–11219, Singapore. Association for Computational Linguistics.

Chunkit Chan, Cheng Jiayang, Yauwai Yim, Zheye Deng, Wei Fan, Haoran Li, Xin Liu, Hongming Zhang, Weiqi Wang, and Yangqiu Song. 2024. Negotiationtom: A benchmark for stress-testing machine theory of mind on negotiation surrounding. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 4211–4241. Association for Computational Linguistics.

Tim R. Davidson, Veniamin Veselovsky, Michal Kosinski, and Robert West. 2024. Evaluating language model agency through negotiations. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Roger Fisher, William Ury, and Bruce Patton. 2011. *Getting to Yes: Negotiating Agreement Without Giving In*, 3rd ed. rev. ed. edition. Penguin Books, New York, NY.

Kanishk Gandhi, Dorsa Sadigh, and Noah D. Goodman. 2023. Strategic reasoning with language models. *CoRR*, abs/2305.19165.

Ian Gemp, Yoram Bachrach, Marc Lanctot, Roma Patel, Vibhavari Dasagi, Luke Marris, Georgios Piliouras, Siqi Liu, and Karl Tuyls. 2024. States as strings as strategies: Steering language models with game-theoretic solvers. *Agentic Markets Workshop at ICML 2024*.

Akash Ghosh, Debayan Datta, Sriparna Saha, and Chirag Agarwal. 2025. The multilingual mind : A survey of multilingual reasoning in language models. *CoRR*, abs/2502.09457.

Christine Howes, Ellen Breitholtz, Mary Lavelle, and Robin Cooper. 2021. Justifiable reasons for everyone: Dialogical reasoning in patients with schizophrenia. In *Proceedings. of the 25th Workshop on the Semantics and Pragmatics of Dialogue*.

Wenyue Hua, Ollie Liu, Lingyao Li, Alfonso Amayuelas, Julie Chen, Lucas Jiang, Mingyu Jin, Lizhou Fan, Fei Sun, William Wang, Xintong Wang, and Yongfeng Zhang. 2024. Game-theoretic LLM: agent workflow for negotiation games. *CoRR*.

Isidora Jeknic, David Schlangen, and Alexander Koller. 2024. A dialogue game for eliciting balanced collaboration. In *Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGDIAL 2024, Kyoto, Japan, September 18 - 20, 2024*, pages 477–489. Association for Computational Linguistics.

Mariia Khan, Yue Qiu, Yuren Cong, Jumana Abu-Khalaf, David Suter, and Bodo Rosenhahn. 2025. Panoscu: A simulation-based dataset for panoramic indoor scene understanding. *IEEE Access*, 13:72456–72476.

Deuksin Kwon, Emily Weiss, Tara Kulshrestha, Kushal Chawla, Gale M. Lucas, and Jonathan Gratch. 2024. Are llms effective negotiators? systematic evaluation of the multifaceted capabilities of llms in negotiation dialogues. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 5391–5413. Association for Computational Linguistics.

Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2443–2453, Copenhagen, Denmark. Association for Computational Linguistics.

John Nash. 1953. Two-person cooperative games. *Econometrica: Journal of the Econometric Society*, pages 128–140.

Jorge Carrasco Pollo, Ioannis Kapetangeorgis, Joshua Rosenthal, and John Hua Yao. 2025. [re] benchmarking LLM capabilities in negotiation through scoreable games. *Transactions on Machine Learning Research*.

Jirui Qi, Shan Chen, Zidi Xiong, Raquel Fernández, Danielle Bitterman, and Arianna Bisazza. 2025. When models reason in your language: Controlling thinking language comes at the cost of accuracy. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 20279–20296, Suzhou, China. Association for Computational Linguistics.

Parshin Shojaee, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. 2025. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. *CoRR*, abs/2506.06941.

Haoran Sun, Yusen Wu, Peng Wang, Wei Chen, Yukun Cheng, Xiaotie Deng, and Xu Chu. 2025. Game theory meets large language models: A systematic survey with taxonomy and new frontiers. *Preprint*, arXiv:2502.09053.

Michelle Vaccaro, Michael Caoson, Harang Ju, Sinan Aral, and Jared R. Curhan. 2025. Advancing AI negotiations: New theory and evidence from a large-scale autonomous negotiations competition. *CoRR*, abs/2503.06416.

John von Neumann and Oskar Morgenstern. 1944. *Theory of Games and Economic Behavior*, first edition. Princeton University Press, Princeton, NJ, USA.

Annie Wong, Thomas Bäck, Aske Plaat, Niki van Stein, and Anna V. Kononova. 2025. Reasoning capabilities of large language models on dynamic tasks. *Preprint*, arXiv:2505.10543.

Tian Xia, Zhiwei He, Tong Ren, Yibo Miao, Zhuosheng Zhang, Yang Yang, and Rui Wang. 2024. Measuring bargaining abilities of llms: A benchmark and A buyer-enhancement method. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 3579–3602. Association for Computational Linguistics.

Yiming Zeng, Mingdong Wu, Long Yang, Jiyao Zhang, Hao Ding, Hui Cheng, and Hao Dong. 2024. Lvdiffusor: Distilling functional rearrangement priors from large models into diffusor. *IEEE Robotics Autom. Lett.*, 9(10):8258–8265.

Shenzhe Zhu, Jiao Sun, Yi Nian, Tobin South, Alex Pentland, and Jiaxin Pei. 2025. The automated but risky game: Modeling agent-to-agent negotiations and transactions in consumer markets. *CoRR*, abs/2506.00073.

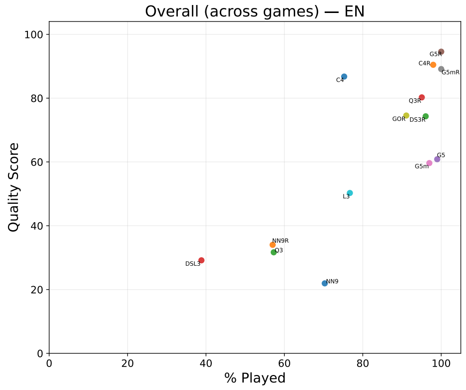
A Additional Results

A.1 %Played and Quality Scores

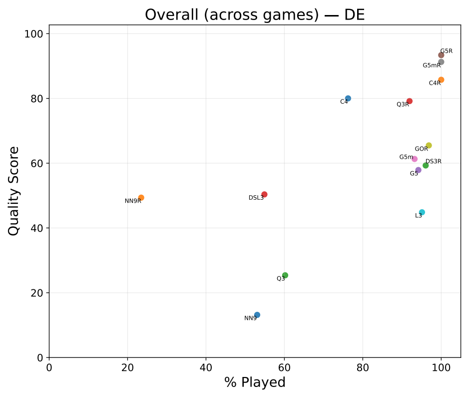
Figure 9 shows the Played and Quality Scores for each language separately, averaged for all games.

A.2 Token Usage & Cost

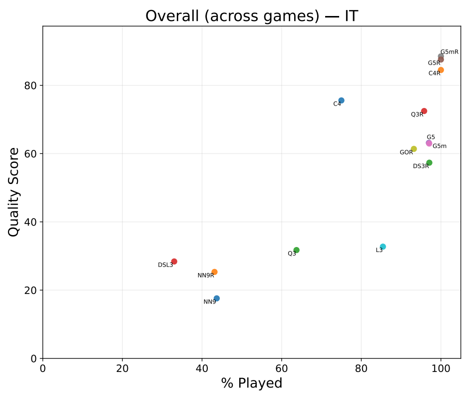
The average token usage and cost for English, German and Italian experiments are given in Figure 10.



(a) Played-Quality Scores for English averaged for all games.



(b) Played-Quality Scores for German averaged for all games.



(c) Played-Quality Scores for Italian averaged for all games.

Figure 9: Comparison of Played-Quality Scores averaged across all games for English, German, and Italian.

A.3 Cycles in Reasoning Traces

A.3.1 Definition

We label the reasoning traces as sequences over five discrete states—`ASSERT`, `PROPOSE`, `UNDERMINE`, `ALTERNATIVE`, `CONCLUDE`—detected via lightweight lexical cues and conflict resolution rules (e.g., maybe/could → `PROPOSE`; but/however → `UNDERMINE`; `PROPOSE` shadows other labels co-occuring in one sentence, sentences like "but maybe we should" will be labeled as `PROPOSE`), the complete matching rules and conflict resolution rules are in Table 3 and Table 4. Each trace is a path on a finite-state machine (FSM, see Figure 11) with observed transitions (including self-loops), from which we compute per-trace statistics and aggregate at model/game level.

We treat a trace as a directed walk over labels and compute two cycle-aware metrics:

Number of segments. We partition the labels extracted from each reasoning trace by absorbing states (`ASSERT` and `CONCLUDE`). The count serves as coarse "reasoning units": higher counts could mean more dispersed reasoning (multiple restarts before final commitment); lower counts could mean more concentrated reasoning (a plan developed and resolved within fewer units).

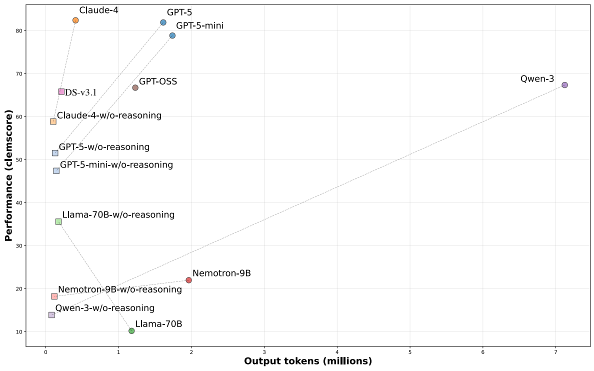
Cycle edge ratio. Within each segment's induced subgraph, we mark edges that belong to any simple cycle and compute the fraction of transitions on such edges, then aggregate across segments to obtain one single metric for each reasoning trace. Higher ratios indicate local looping/hedging (e.g., `PROPOSE <-> UNDERMINE`); lower ratios indicate forward motion towards absorbing states.

These metrics capture cyclicity at two levels: inter-segment shifting/restarts (segments) before final commitment, and intra-segment circling before interim resolution (cycle edge ratio). Strategically adaptive behaviours should advance and commit—e.g., `PROPOSE` → `ASSERT` → `CONCLUDE` or `PROPOSE` → `ALTERNATIVE` → `ASSERT` → `CONCLUDE`—rather than cyclic hedges like `PROPOSE <-> UNDERMINE <-> ALTERNATIVE`. Thus, high cyclicity signals surface patterning (stalling/hedging/"explainers that never decide").

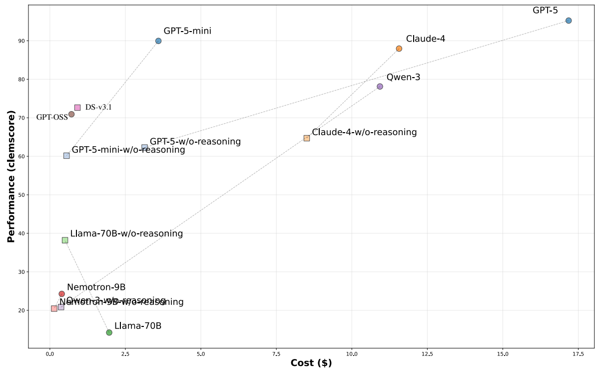
We relate both metrics to clemscore across games and languages to test whether low number of reasoning segments and low-cyclic reasoning aligns with better negotiation outcomes.

A.3.2 Analysis

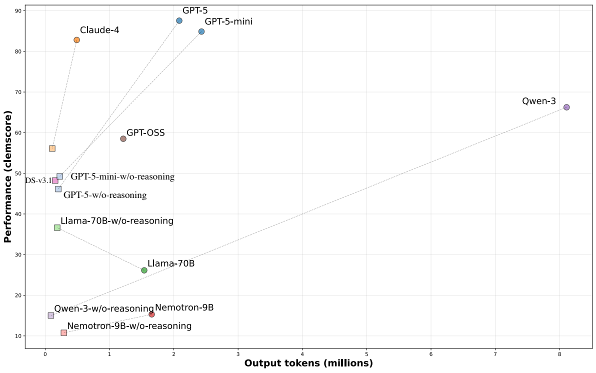
Number of Segments We compare paired non-reasoning and reasoning variants of models across games, plotting clemscore (y) against the number of segments (x) in Figure 12. There is a slight U-shaped relationship: both a small increase and an excessive increase in the number of segments



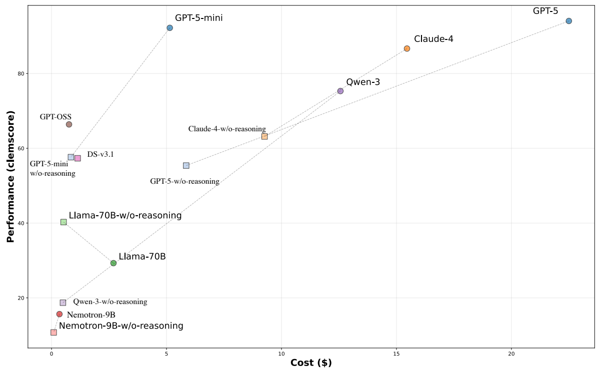
(a) Performance and output tokens for all evaluated models for English.



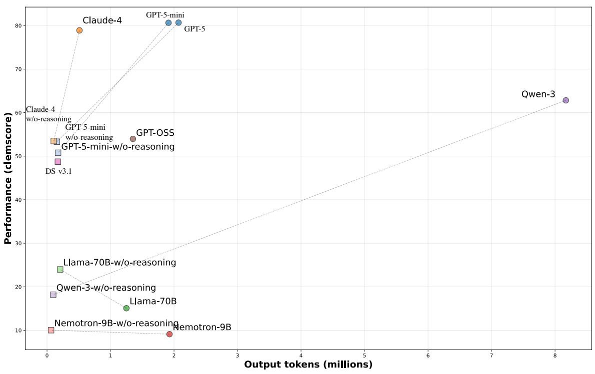
(b) Performance (clemscore) and cost for all evaluated models for English.



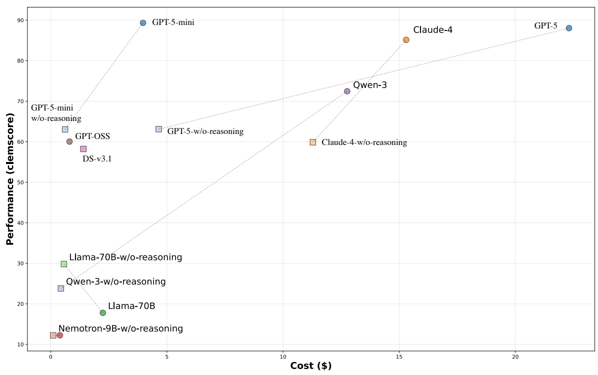
(c) Performance (clemscore) and output tokens for all evaluated models for German.



(d) Performance (clemscore) and cost for all evaluated models for German.



(e) Performance (clemscore) and output tokens for all evaluated models for Italian.



(f) Performance (clemscore) and cost for all evaluated models for Italian.

Figure 10: Trade-off between performance and cost comparison across different models. Results for English (top), German (middle), and Italian (bottom).

coincide with improved performance, while a moderate addition of segments shows unclear correlation. Claude-4 adds only a few segments and shows small gains in *Clean Up* and *Deal or No Deal* and a pronounced gain in *Air Balloon Survival*. Deepseek exhibits moderate segment inflation with consistently lower clemscores. Nemotron shows mixed effects at similar segment increases. Qwen introduces the largest segment increases and performance gains; its performance is competitive with Claude in two of the three games, but in Deal or No Deal its substantial over-segmentation coincides with a relative underperformance. We compare paired non-reasoning and reasoning variants of models across games, plotting clemscore (y) against the number of segments (x) in Figure 12. There is a slight U-shaped relationship: both a small increase and an excessive increase in the

Label	English cue words (case-insensitive)
ASSERT	*need*, *should*, *must*
PROPOSE	*maybe*, *perhaps*, *can*, *could*
UNDERMINE	*but*, *however*, *wait*
ALTERNATIVE	*alternatively*, *another*
CONCLUDE	*so*, *thus*

Table 3: Cue words used to label sentences of one reasoning trace into FSM states. We lowercase, lemmatize, and match whole tokens; punctuation is stripped; negation is ignored.

Condition	Outcome
Multiple labels include CONCLUDE	Drop CONCLUDE and re-evaluate
Exactly one label remains	Assign that label
Multiple labels remain and include PROPOSE	Assign PROPOSE
Multiple labels remain and exclude PROPOSE	Mark as conflict (skip)

Table 4: Conflict resolution rules used after keyword matching.

number of segments coincide with improved performance, while a moderate addition of segments shows unclear correlation. Claude-4 adds only a few segments and shows small gains in *Clean Up* and *Deal or No Deal* and a pronounced gain in *Air Balloon Survival*. Deepseek exhibits moderate segment inflation with consistently lower clemscores. Nemotron shows mixed effects at similar segment increases. Qwen introduces the largest segment increases and performance gains; its performance is competitive with Claude in two of the three games, but in Deal or No Deal its substantial over-segmentation coincides with a relative underperformance.

Cycle Edge Ratio The relation between clemscore and Cycle Edge Ratio is shown in Figure 13. Across panels, open-weight models tend to have a much higher cycle edge ratio, indicating frequent cyclic hedging before reaching interim conclusions or assertions. Although there is no uniform monotonic link between cycle-edge ratio and clemscore, we observe model-dependent trends: for Qwen and Claude, an increase in cycle edge ratio consistently coincides with better performance; DeepSeek tends to lose performance as the ratio increases; GPT-OSS and Nemotron show mixed associations.

Claude points cluster at lower cycle ratios while remaining competitive in score; open-weight models occupy a wider band—extending into higher ratios—with correspondingly higher variance in outcomes. This suggests that additional cycling in open-weights is not reliably converted into payoff.

A.4 LLM-based Analysis of Reasoning Traces

For all thinking models with available reasoning traces, we sampled a total of 309 transcripts with at least one instance for each game, language, and experiment combination for automated analysis.

We prompted *GPT-5* to analyse certain aspects of the traces and give their output in json format, including not only the analysis, but also short explanations for each classification (Figure 14). The explanations are not only meant to increase the quality of the classifications, but also to facilitate verifiability. An informal spot check indicated that these prompts performed satisfactorily at the annotation task.

Loops Among other things, we prompted the LLM to mark traces as containing loops if a thought or an action is repeated at least three times, and then calculated the percentage of traces in the transcript that contain loops. Experimentally, we also tasked the LLM to rate the ‘confusion’ expressed in the trace on a scale from 0 (not confused at all) to 10 (extremely confused), and calculated the mean over all traces in one game transcript. Fig. 15a shows that loops correlate with the segment analysis of A.3. It has to be noted that a large number of segments does not necessarily imply loops, and *vice versa*; additionally, the loop analysis stretches over the whole transcript instead of just the first round. Fig. 15b shows that the average confusion scores also correlate with percentage of loops in a transcript.

Role Analysis Additionally, we tasked the LLM with classifying the role of the author of the trace as either ‘player’ or ‘helper’ to find out which role the model assumes within the game: is it aware of being an active participant, or does it take on an assistant role? Again we calculated the percentage of traces written from a player’s perspective for each transcript. To strengthen this analysis, we also let the LLM identify whether or not a ‘user’ is mentioned in the thinking traces. In most cases, this is a sign that the model perceives the user as a player and itself as an assistant, although in some cases it also identified the user as the game master. Fig. 15c shows that these percentages are fairly complementary.

Finally, we tasked the LLM with identifying any other agentive persons mentioned in or implied by the traces and their respective roles, and we tried

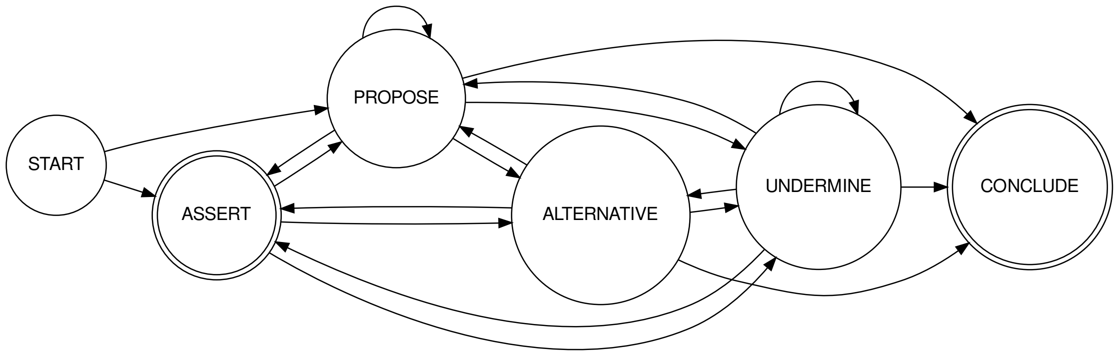


Figure 11: Finite state machine states for reasoning traces

to identify if the trace shows awareness of another player. In 17 out of 4123 (0.41%) analysed traces, the LLM added the author to the agentive persons, and the name was given as either one of the first person pronouns ‘I’, ‘Ich’, ‘Io’, as ‘assistant (self)’ or as ‘author’, and marked their role as ‘player’. For further analysis, we removed these duplicate persons. Excluding these cases, we found additional 52 (1.26%) trace analyses where the author role was given as ‘player’ and two (other) players were listed in ‘agentive persons’. Mostly, this was because the other player was listed twice, for example once as ‘other player’ and once as ‘teammate’. We also found one instance where a person ‘Spielleiter’ (German for game master) was classified as ‘player’.

Given that these numbers of erroneous role assignments are fairly low, we assume that the author role assignment is similarly reliable.

If the author assumed a player role *and* at least one other player was listed as an agentive person, we concluded that the reasoning trace shows **‘role awareness’**, and calculated average values over each transcript, which is the basis for Fig. 7b.

B Deal or No Deal - Game Details

B.1 Prompt Templates

The English prompt templates for both players of the Deal or No Deal game are given in fig. 16. The corresponding German and Italian versions of the prompts are given in fig. 17 and fig. 18, respectively.

The game is started by the game master sending a message to both players following the prompt template shown in fig. 16a. For this, the variables enclosed in the $ characters are replaced as follows.

1. N is replaced with the maximum number of turns before the game master instructs the players to submit a proposal. For all instances evaluated in this work, this has been set to 5.

2. $GOAL$ is replaced with the goal the players are trying to optimize. This depends only on the game mode. In the semi-competitive mode, it is given as "*Your goal is to maximize the score you receive.*". On the other hand, in the cooperative mode, the goal is given as "*Your goal is to maximize the sum of your score and the score of the other player.*".

3. $ITEMS$ is replaced with the set of available items, e.g., "*1 book, 2 hats, 2 balls.*". This value is always identical for both players.

4. $VALUE_FUNCTION$ is replaced with the value function for one of the players, e.g., "*book: 0, hat: 1, ball: 4.*". Each player gets a different value function, and so the prompts for the two players differ.

When the maximum number of turns has been reached, i.e., each player has sent 5 messages, the game master will instruct the player who is next that they must now submit a proposal using the prompt shown in fig. 16b. If the next message does not contain a correctly formatted proposal, the game is aborted.

If one of the two players makes a proposal, either proactively or when instructed by the game master, the game master will instruct the remaining player to also make a proposal. For this, the template shown in fig. 16c is used. Again, the next message must be a proposal, or the game is aborted.

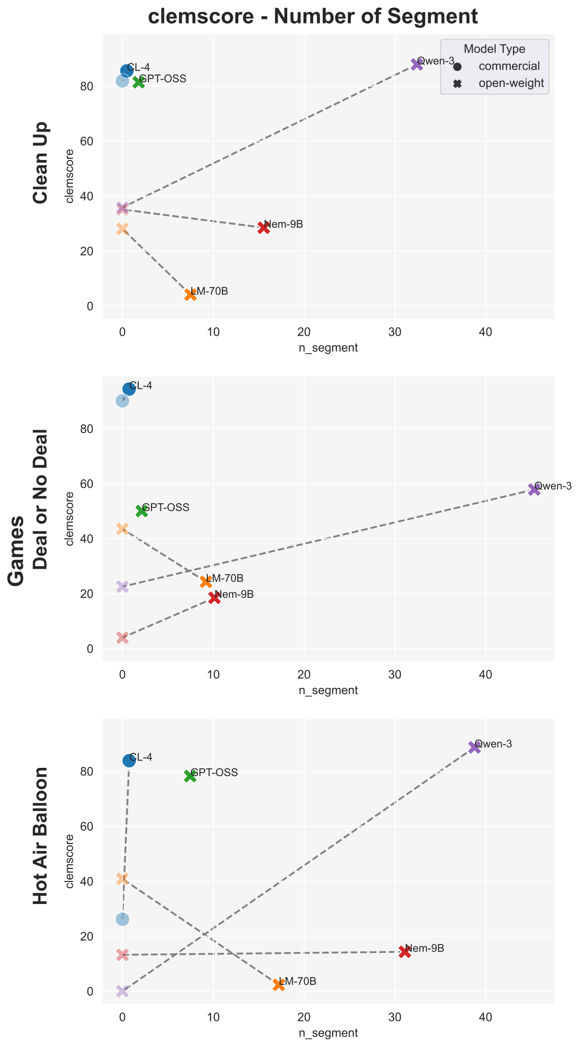


Figure 12: Clemscore plotted against number of Segment. A Segment is defined as a continuous chunk of a series of reasoning labels extracted from reasoning traces. It is obtained by partitioning the reasoning labels with absorbing states in Figure 11. Solid markers denote reasoning models; translucent markers denote their non-reasoning counterparts; dashed lines connect each pair.

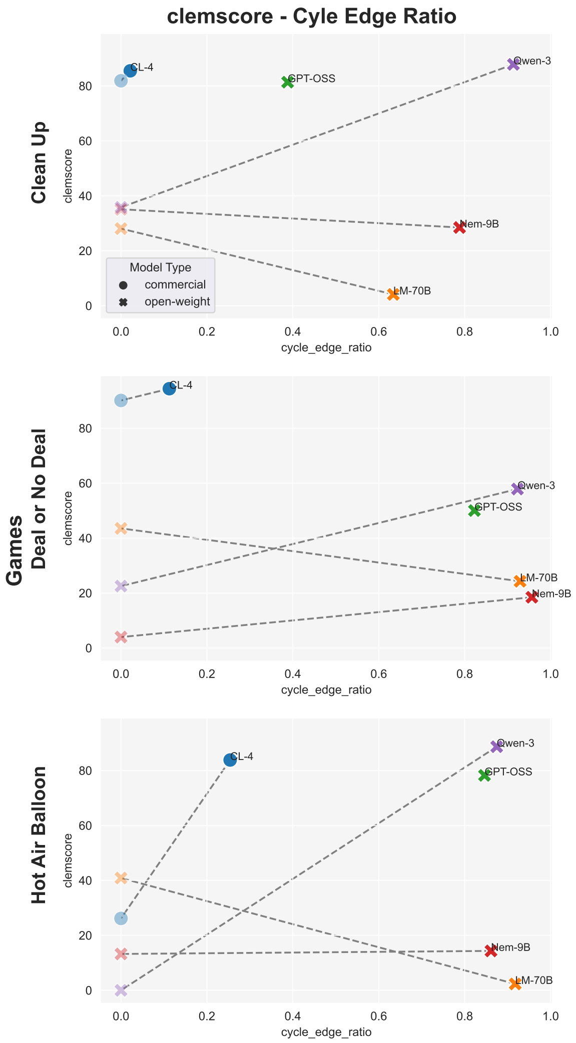


Figure 13: Clemscore plotted against Cycle Edge Ratio, defined in Appendix A.3.1. Solid markers denote reasoning models; translucent markers denote their non-reasoning counterparts; dashed lines connect each pair.

B.2 Evaluation Metrics

For the cooperative game mode, it is the ratio between the achieved total score and the maximum total score for that game instance,

$$\text{Quality}_{\text{coop}} = \frac{\text{Score}_A + \text{Score}_B}{\max(\text{Score}'_A + \text{Score}'_B)}$$

where Score_A and Score_B represent the actual scores achieved by players A and B, respectively, and $\max(\text{Score}'_A + \text{Score}'_B)$ denotes the maximum possible combined score achievable for the given game instance.

For the semi-competitive setting, a metric based on Pareto efficiency has been used.

$$\text{Quality}_{\text{semi}} = 1 - \frac{\text{Maximum Pareto Improvement}}{\max(\text{Score}'_i)}$$

where Maximum Pareto Improvement represents the largest possible improvement in one player's score without decreasing the other player's score from the current agreement, and $\max(\text{Score}'_i)$ denotes the maximum possible score achievable by any single player i in the given game instance.

B.3 Overall Results

Figure 19 shows the average quality scores and the percentage of games played for each model for the DoND game, averaged over the different modes and languages of the game.

```
TEMPLATE A.4.1
Your task is to analyze the following strategic thinking trace and the subsequent response:
“‘trace
<TRACE>
“‘
First of all, you have to identify the role of the author of the trace.
[...] Classify the role of the author. Are they *helping* or *advising* somebody else who is playing
the game, actively *playing* the game themselves, or are they in a *neutral* role, e.g., as an
observer or game master? Please choose from ["helper", "player", "neutral"]. If none of these roles
fit, classify it as "other", and specify it in your analysis.
Does the author mention any other agentive person or persons that influence the game? Give their
names or titles and roles.
Valid roles for other persons are: ["player", "assistant", "game master"]. If a role does not fit
into these categories, classify it as "other", and specify it in your analysis.
Following that, you should analyze the trace for the following aspects:
* Are there loops in the trace? (e.g., the author repeating the same action or thought at least three
times, ‘true‘ or ‘false‘)
...
* Does the author express confusion? Rate the confusion on a scale from ‘0‘ (not confused at all) to
‘10‘ (extremely confused)
* If you detect confusion, cite an example of displayed confusion. Only cite the example, no further
explanation.
[...]
Your reply should be in the following json format:
[...]
```

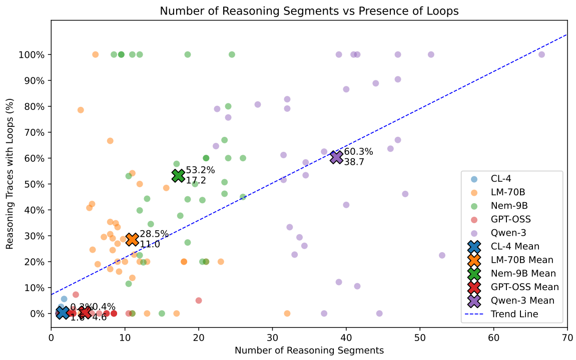
(a) Relevant excerpt of the prompt used to analyse a sample of thinking traces.

```
{
    "role_analysis": "The author is actively analyzing moves and positions on a grid and deciding the
        next action, indicating they are playing the game themselves. They reference another player
        asking “finished?”, but the author is the one executing or planning moves. No evidence suggests
        they are merely advising or observing.",
        "person_and_number_analysis": "They predominantly use first-person singular (I) while
            occasionally referring to another player.",
        "author_role": "player",
        [...]
        "agentive_persons_mentioned": [
            {
                "name": "other player",
                "role": "player",
                "mind": "They repeatedly asked “finished?” indicating they are awaiting confirmation."
            }
        ],
      "formal_analysis": "There is a repeated attempt to parse the ASCII grid structure, revisiting the
            same uncertainty multiple times, indicating looping and confusion. Frustration is mild to
            moderate due to difficulty interpreting coordinates versus the drawing. Excitement is not
            expressed.",
        "loops": true,
        "confusion_example": "This is too confusing.",
        "confusion": 8,
    [...]
}
```

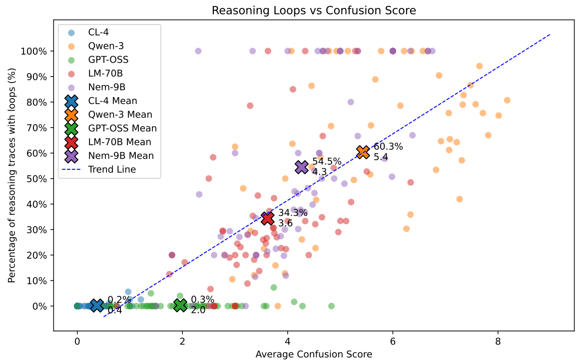
(b) Excerpts from an LLM-based analysis of the last trace of Qwen-3, Clean Up, 2_hard_7obj, in English. Manual review showed that indeed, the model tried to parse the ASCII grid no less than 19 times, and tried to parse the initial grid another 5 times, even though it is irrelevant at this stage of the game. The ‘other player’ is explicitly mentioned five times.

Figure 14: Excerpts from the prompt used for LLM-based analysis and from an analysis produced by the model

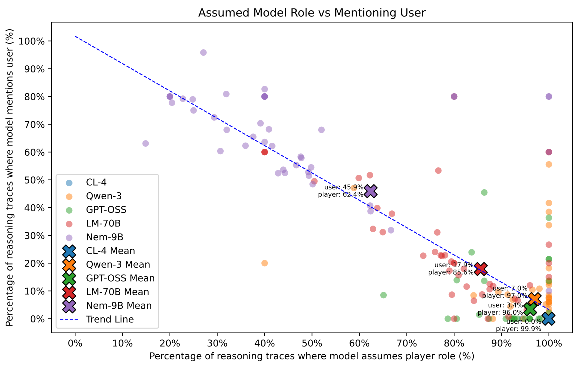
We can observe that most models achieve a high percentage of games played, which means they can accurately follow the instructions. The only exceptions to this are the Llama model and the Nematron model with reasoning enabled. The models GPT-5, GPT-5 mini, and Claude 4 with reasoning enabled



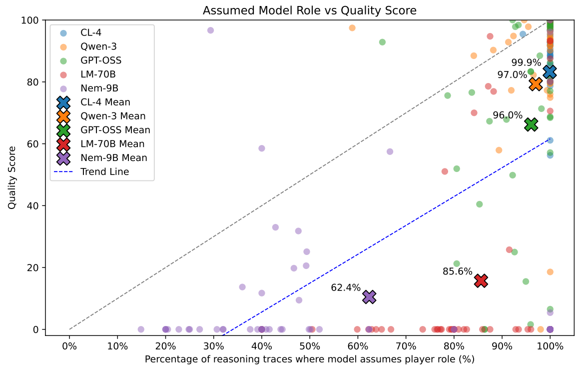
(a) Number of segments *vs.* percentage of traces containing loops identified by automated LLM analysis



(b) Average confusion score *vs.* percentage of traces containing loops, as identified by automated LLM analysis



(c) Percentage of reasoning traces where the model assumes a player role *vs.* percentage where it explicitly mentions a user, as identified by automated LLM analysis



(d) Percentage of reasoning traces where the model assumes a player role, as opposed to an assistant/helper role, plotted *vs.* quality score. 95.2% of data points are below the main diagonal, meaning a model rarely reaches a quality score higher than the percentage of reasoning traces where it assumes a player role.

Figure 15: Plots (a) through (c) corroborate the validity of the categorisation provided by the LLM. (d) shows the importance of assuming a player role for reaching high scores in the negotiation games.

show the strongest performance.

Generally, we can see that the models perform better with reasoning enabled than with it disabled. While the difference between reasoning and non-reasoning is small for the Claude 4 model, reasoning gives a large performance improvement for the GPT-5 and GPT-5 mini, as well as the Qwen-3 model.

To better understand the performance of the models, we show a more detailed breakdown by language and game mode in table 5. These results are also depicted in fig. 20, which shows the performance differences between experiments for English, German, and Italian tasks.

We can see here again that most models follow the instructions well. We can also see that the reason the Nematron model with reasoning enabled has such a low percentage of played games overall is that it has not played a single game in German correctly. Its percentage of played games is otherwise comparable to Llama. By inspecting the transcripts of some games played by Nematron in German, it becomes clear that the reason for the aborted games is that Nematron always made the proposal in English, whereas it should have made it in German. This is likely because Nematron also seems to reason exclusively in English.

Except for some outliers, we also generally observe that performance in the cooperative and semi-competitive game modes is comparable. Slightly higher quality scores are achieved for the cooperative mode (49%) as compared to the semi-competitive mode (46%). The number of aborted games is consistent, with an average of 8% for both game modes.

Performance was generally highest in English. On average across all models, the clemscore for English was 49, while it was 43 and 40, respectively, for German and Italian. Due to the issues of the Nematron model in German, the number of aborted

```
TEMPLATE B.1.1

You are playing a negotiation game in which you have to agree on how to divide a set of items among
you and another player.

Rules:
(a) You and the other player are given a set of items. Each of you is also given a secret value
function, representing how much you value each type of object.
(b) You exchange messages with the other player to agree on who gets which items. You can send a
maximum of $N$ messages each, or terminate early by making a secret proposal at any time.
(c) You are each asked to submit a secret proposal indicating the items you want formatted in square
brackets as follows: "[Proposal: <number> <object name>, <number> <object name>, <...>]"
(d) If your proposals are complementary, i.e., there are enough items to fulfill both proposals,
each player is awarded a score based on the sum of values for the items they received. Otherwise,
both of you get zero points.
(e) $GOAL$

Let us start.

The set of available items is:

$ITEMS$

Your secret value function is:

$VALUE_FUNCTION$

IMPORTANT: Your messages, unless it is the secret proposal, are directly transmitted to the
other player, so do not include any response to the rules or text announcing your message. To make
a secret proposal, use the indicated format. Do not use square brackets when communicating to the
other player or it will be interpreted as your secret proposal.
```

(a) English prompt template used for both players at the start of the game to inform them about the rules and game state. Values surrounded by $ are to be replaced by different values depending on the game instance. Each player gets the same initial message, except for different values of $VALUE_FUNCTION$.

```
TEMPLATE B.1.2

The time is up. It is now your turn to submit a
secret proposal.
```

(b) Prompt used when the maximum number of turns has been reached. This template is sent only to one of the two players. The other player receives the template shown in fig. 16c.

```
TEMPLATE B.1.3

The other player has submitted a secret proposal.
It is now your turn to submit a proposal of your
own.
```

(c) Prompt used when the other player submits their proposal. This is sent to a player whenever the opposing player submits a secret proposal.

Figure 16: English prompt template used to tell the players that they are now required to submit a secret proposal.

TEMPLATE B.1.4

```
Sie spielen ein Verhandlungsspiel, bei dem Sie sich mit einem anderen Spieler darauf einigen müssen,
wie eine Reihe von Gegenständen aufgeteilt werden soll.

Die Regeln:
(a) Sie und der andere Spieler erhalten eine Sammlung von Gegenständen. Jeder von Ihnen erhält
außerdem eine geheime Wertfunktion, die angibt, wie viel Ihnen jede Art von Gegenstand wert ist.
(b) Sie tauschen Nachrichten mit dem anderen Spieler aus, um zu vereinbaren, wer welche Gegenstände
bekommt. Sie können jeweils maximal $N$ Nachrichten senden oder das Spiel vorzeitig beenden, indem
Sie jederzeit einen geheimen Vorschlag machen.
(c) Jeder von euch wird aufgefordert, einen geheimen Vorschlag zu machen, in dem ihr die gewünschten
Gegenstände in eckigen Klammern wie folgt angibt: "[Vorschlag: <Nummer> <Objektname>, <Nummer>
<Objektname>, <...>]"
(d) Wenn eure Vorschläge komplementär sind, d.h. es gibt genug Gegenstände, um beide Vorschläge zu
erfüllen, erhält jeder Spieler eine Punktzahl, die sich aus der Summe der Werte für die Gegenstände
ergibt, die er erhalten hat. Andernfalls erhalten Sie beide null Punkte.
(e) $GOAL$

Beginnen wir.

Die Menge der verfügbaren Gegenstände ist:

$ITEMS$

Deine geheime Wertfunktion ist:

$VALUE_FUNCTION$

WICHTIG: Ihre Nachrichten werden, sofern es sich nicht um einen geheimen Vorschlag handelt,
direkt an den anderen Spieler übermittelt, also fügen Sie keine Antwort auf die Regeln oder einen
Text zur Ankündigung Ihrer Nachricht ein. Um einen geheimen Vorschlag zu machen, verwenden Sie
das angegebene Format. Verwenden Sie keine eckigen Klammern, wenn Sie mit dem anderen Spieler
kommunizieren, sonst wird dies als Ihr geheimer Vorschlag interpretiert.
```

(a) German prompt template used for both players at the start of the game to inform them about the rules and game state. Values surrounded by $ are to be replaced by different values depending on the game instance. Each player gets the same initial message, except for different values of $VALUE_FUNCTION$.

TEMPLATE B.1.5

```
Die Zeit ist um. Sie sind jetzt an der Reihe,
einen geheimen Vorschlag einzureichen.
```

(b) Prompt used when the maximum number of turns has been reached. This template is sent only to one of the two players. The other player receives the template shown in fig. 17c.

TEMPLATE B.1.6

```
Der andere Spieler hat einen geheimen Vorschlag
gemacht. Jetzt bist du an der Reihe, einen
eigenen Vorschlag zu machen.
```

(c) Prompt used when the other player submits their proposal. This is sent to a player whenever the opposing player submits a secret proposal.

Figure 17: German prompt template used to tell the players that they are now required to submit a secret proposal.

TEMPLATE B.1.7

```
State giocando a un gioco di negoziazione in cui dovete accordarvi su come dividere una serie di oggetti tra voi e un altro giocatore.

Regole:
(a) A Lei e all'altro giocatore viene dato un insieme di oggetti. Ognuno di voi riceve anche una funzione di valore segreta, che rappresenta il valore di ciascun tipo di oggetto.
(b) Si scambiano messaggi con l'altro giocatore per concordare chi si aggiudica gli oggetti. Potete inviare un massimo di $N$ messaggi ciascuno, oppure terminare in anticipo facendo una proposta segreta in qualsiasi momento.
(c) A ciascuno di voi viene chiesto di inviare una proposta segreta indicando gli oggetti che desiderate, formattata tra parentesi quadre come segue: "[Proposta: <numero> <nome oggetto>, <numero> <nome oggetto>, <...>]".
(d) Se le vostre proposte sono complementari, cioè ci sono abbastanza oggetti per soddisfare entrambe le proposte, a ciascun giocatore viene assegnato un punteggio basato sulla somma dei valori degli oggetti ricevuti. In caso contrario, entrambi ricevono zero punti.
(e) $GOAL$

Cominciamo.

L'insieme degli oggetti disponibili è:

$ITEMS$

La funzione valore segreta è:

$VALUE_FUNCTION$

IMPORTANTE: i vostri messaggi, a meno che non si tratti di una proposta segreta, vengono trasmessi direttamente all'altro giocatore, quindi non includete alcuna risposta alle regole o testo di annuncio del vostro messaggio. Per fare una proposta segreta, utilizzate il formato indicato. Non utilizzare delle parentesi quadre quando si comunica all'altro giocatore, altrimenti verrà interpretata come una proposta segreta.
```

(a) Italian prompt template used for both players at the start of the game to inform them about the rules and game state. Values surrounded by $ are to be replaced by different values depending on the game instance. Each player gets the same initial message, except for different values of $VALUE_FUNCTION$.

TEMPLATE B.1.8

```
Il tempo è scaduto. Ora tocca a Lei presentare una proposta segreta.
```

(b) Prompt used when the maximum number of turns has been reached. This template is sent only to one of the two players. The other player receives the template shown in fig. 18c.

TEMPLATE B.1.9

```
L'altro giocatore ha presentato una proposta segreta. Ora tocca a Lei presentare una sua proposta.
```

(c) Prompt used when the other player submits their proposal. This is sent to a player whenever the opposing player submits a secret proposal.

Figure 18: Italian prompt template used to tell the players that they are now required to submit a secret proposal.

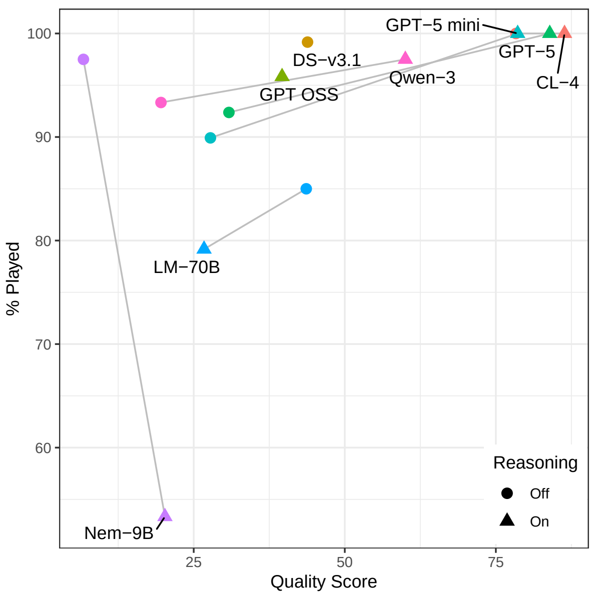


Figure 19: Summarized evaluation results for each model. The y-axis indicates the percentage of played, i.e., not aborted, games. The x-axis represents the average quality score obtained when considering only played games.

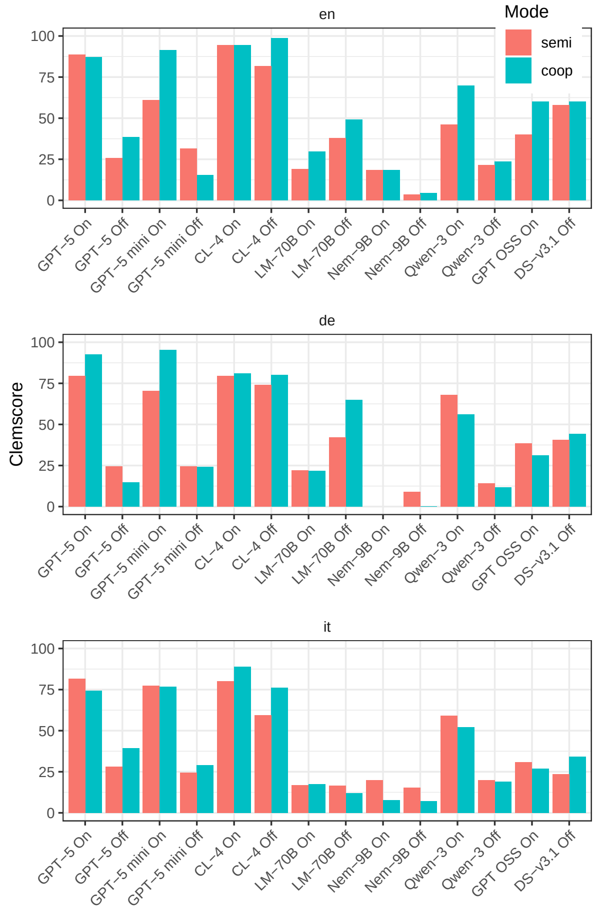


Figure 20: Clemscore for different models separated by language and game mode.

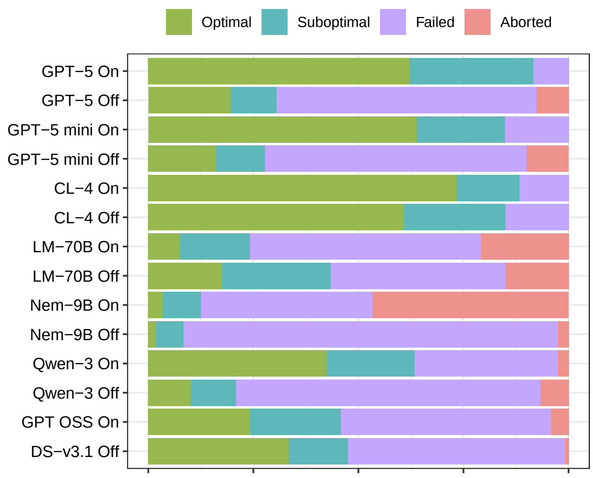


Figure 21: Percentage of games that ended with a specific game outcome during the evaluation of different models. All results have been averaged across the two game modes and three languages for this plot.

games is higher in German, with 13% compared to 6% for both English and Italian. However, in German, the average quality score for the games played is higher at 49% compared to the average of 43% for Italian.

We can also analyze the results based on the outcome that was achieved. This is depicted in fig. 21. We separate into four possible game outcomes. The game can either end with complementary proposals or not. If there are complementary proposals, it could be with the optimal score, i.e., a quality score of 100, or with a suboptimal score less than 100. Finally, if there is no valid agreement, it can either be caused by conflicting proposals or by the game being aborted due to rule violations.

In general, we can observe similar results to those above. It becomes clear that many of the low scores can be attributed to failed agreements, where players have submitted conflicting proposals. Particularly for GTP-5 and GPT-5 mini, this problem is largely solved by using the reasoning instead of the non-reasoning versions.

B.4 Detailed Analysis

To better understand what caused the differences in performance, and to analyze the strategies employed by the different models, we have performed a further breakdown of the outcomes for the English instances. Table 6 and Table 7 show the result for each game, categorized by the reason for which the given result has been reached.

In the analysis of the cooperative game mode shown in table 6, we can observe some significant

		GPT-5		GPT-5 mini		CL-4		LM-70B		Nem-9B		Qwen-3		GPT OSS	DS-v3.1
	Metric	On	Off	On	Off	On	Off	On	Off	On	Off	On	Off	On	Off
semi en	**Clemscore**	88	26	61	32	**94**	82	19	38	18	3	46	22	40	58
	% Played	**100**	**100**	**100**	90	**100**	**100**	80	55	85	95	**100**	**100**	95	95
	% Agreement	**95**	28	65	50	**95**	85	25	73	24	5	55	25	42	68
	% Optimal	70	17	45	6	**90**	70	19	45	12	0	30	10	42	37
	Quality Score	88	26	61	35	**94**	82	24	69	22	4	46	22	42	61
	Avg. # Messages	3.5	4.3	4.7	6	5.8	6.1	2.8	9.3	2	2	3.3	4	4.7	5.8
semi de	**Clemscore**	**80**	24	70	24	**80**	74	22	42	0	9	68	14	38	40
	% Played	**100**	85	**100**	90	**100**	**100**	75	95	0	**100**	**100**	85	95	**100**
	% Agreement	**90**	29	80	28	80	**90**	33	53	0	10	75	18	47	50
	% Optimal	50	24	60	22	**75**	55	20	21	0	5	55	12	26	30
	Quality Score	**80**	29	70	27	**80**	74	29	44	0	9	68	16	41	40
	Avg. # Messages	3.8	4.8	3.8	4.9	7	6.2	2.8	8.4	1	2	3.8	3.8	3.6	6.2
semi it	**Clemscore**	**82**	28	78	24	80	60	17	16	20	15	59	20	31	24
	% Played	**100**	95	**100**	85	**100**	**100**	70	90	75	95	**100**	90	**100**	**100**
	% Agreement	**90**	32	85	29	85	65	29	28	33	21	65	22	35	25
	% Optimal	55	21	55	24	**65**	50	7	0	7	5	40	22	30	20
	Quality Score	**82**	29	78	29	80	60	24	18	27	16	59	22	31	24
	Avg. # Messages	4	4.5	3.5	4.3	6	5.6	4.2	6.7	1.9	3.2	3.7	4.8	3.9	5.4
coop en	**Clemscore**	87	38	91	15	94	**99**	30	49	18	4	70	24	60	60
	% Played	**100**	95	**100**	95	**100**	**100**	**100**	80	90	**100**	90	**100**	95	**100**
	% Agreement	89	42	95	17	95	**100**	35	69	22	5	78	25	79	65
	% Optimal	74	26	**89**	11	85	85	0	31	6	0	72	15	32	50
	Quality Score	87	40	91	16	94	**99**	30	61	21	4	77	24	63	60
	Avg. # Messages	3.8	4.2	5.2	4.9	5.4	5.6	2.5	9.2	2	2	3.8	3.3	5	5.4
coop de	**Clemscore**	92	15	**95**	24	81	80	22	65	0	0	56	12	31	44
	% Played	**100**	85	**100**	80	**100**	**100**	70	**100**	0	**100**	95	90	95	**100**
	% Agreement	**100**	18	**100**	31	85	85	36	75	0	0	63	17	37	45
	% Optimal	75	12	**80**	25	55	60	7	30	0	0	32	0	16	40
	Quality Score	92	17	**95**	30	81	80	31	65	0	0	59	13	33	44
	Avg. # Messages	3.9	4.5	3.8	5.2	5.7	5.8	2	9	1	2	3.8	3.7	4.2	5.8
coop it	**Clemscore**	74	40	77	29	**89**	76	18	12	8	7	52	19	27	34
	% Played	**100**	95	**100**	**100**	**100**	**100**	80	90	70	95	**100**	95	95	**100**
	% Agreement	85	47	85	30	**90**	85	25	17	14	11	55	26	47	35
	% Optimal	50	26	55	20	**70**	45	6	6	0	0	35	5	5	25
	Quality Score	74	42	77	29	**89**	76	22	13	11	7	52	20	28	34
	Avg. # Messages	3.6	3.9	3.5	4	5.3	5.3	3.4	7.6	1.9	3.5	3.9	4.4	3.2	5.2

Table 5: Detailed breakdown of the results for each game mode, language, and model combination across multiple metrics for the DoND game. The best result in each row is indicated in bold. Note that the quality score, as well as the percentages of agreements and optimal outcomes, are computed only with respect to the non-aborted games.

		GPT-5		GPT-5 mini		CL-4		LM-70B		Nem-9B		Qwen-3		GPT OSS	DS-v3.1
Outcome	**Reason**	On	Off	On	Off	On	Off	On	Off	On	Off	On	Off	On	Off
Optimal	Explicit max.	11	3	11	0	0	1	0	0	0	0	5	0	1	3
	Simple negot.	3	2	6	2	17	16	0	5	0	0	8	3	5	7
	Luck	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Suboptimal	Unclear pref.	0	1	0	0	1	1	1	1	0	0	0	0	1	1
	Unclaimed items	0	0	0	0	0	0	0	1	0	0	1	1	0	0
	Claiming valueless	0	2	0	1	1	2	0	4	0	0	0	0	3	0
	Wrong proposal	0	0	0	0	0	0	1	0	0	0	0	0	1	0
	Premature prop.	3	0	1	0	0	0	3	0	3	1	0	1	3	1
Failed	Wrong proposal	1	11	0	13	0	0	3	5	0	0	4	7	1	7
	Premature prop.	1	0	1	2	1	0	12	0	14	19	0	8	4	1
Aborted	Wrong syntax	0	1	0	1	0	0	0	4	0	0	1	0	1	0
	Non-existent items	0	0	0	0	0	0	0	0	2	0	0	0	0	0

Table 6: Game outcomes for the **cooperative** DoND game mode. For each outcome, we also differentiate by the reason or strategy that caused the given outcome to be achieved.

differences in strategies between the different models. It can be seen that the GPT-5 models with reasoning disabled will often explicitly discuss how to achieve a maximum score, i.e., they will talk about having to give each item to the player that values it most. This is in contrast to, for example, Claude

4 or the GPT-5 models without reasoning enabled, as those models will often just follow a generic negotiation routine, trying to obtain a fair-sounding agreement.

We can further observe that a main reason for the low scores of the non-reasoning versions of the GPT-5 models is that they submit the wrong proposal. This means that the proposal they submit to the game master is often not the one they discussed with the other player in the natural language messages. This manifests itself sometimes as submitting the proposal for the other player, sometimes as proposals that include the complete set of items, and, at other times, even completely unrelated proposals. Deepseek-v3.1 also suffers from a less severe version of this problem.

Lastly, some other models, like Deepseek distilled Llama and the Nematron models, as well as the Qwen-3 model without reasoning, often submit proposals before discussing and agreeing to a division with the other player. This is particularly severe for the Nematron models, as these nearly always submit a proposal as the first message, without exchanging any messages with the other player. Looking at the reasoning traces, we can observe that the Nematron model does not seem to understand that it can communicate with the other player. It often expresses correctly that it does not know the other player's values, but does not realize that it could ask for them.

For the semi-competitive game mode, we see largely the same results, with the main difference being that the GPT-5 models no longer try to explicitly discuss maximizing their score. This makes sense as the setting is no longer cooperative, so it is no longer purely advantageous to share this reasoning with the other player. Another difference from the cooperative mode is that the GPT-5 mini model with reasoning and the GPT-OSS model now make wrong proposals more often. It is not clear why this happens.

We can also generally see an increase in the number of suboptimal outcomes. Interestingly, this is not necessarily because of unclear preferences, but because models are often requesting items that have no value to them, or leave some items unclaimed by either player, sometimes even expressing the intention of doing so deliberately. Keeping an item that does not have any value does not make much sense for any of the players, as giving up that item could instead be used as leverage for getting some other items.

Table 8 and table 9 show how likely different models were to reveal their value function to the other player. This is roughly an estimate of how cooperative they were.

Clearly, in the cooperative game mode, it makes little sense to hide one's value function, as it is essential for knowing which allocation is optimal. Still, we can see in table 8 that some models were unwilling to give exact values. Some models, namely the Nematron and Deepseek distilled Llama models, were not willing to communicate preferences at all, and sometimes did not even communicate, instead immediately making a proposal. This explains their poor performance. On the other hand, the Claude 4 models were unwilling to give out their exact value function, preferring instead to only give relative or subjective characterizations of it. In some instances, the Claude 4 models even explicitly mention not wanting to reveal their exact values, either in messages or reasoning traces, even though it can only be beneficial in the cooperative setting. Finally, all the other models were willing to give the exact value function to the other player.

For the semi-competitive game mode, it is less clear whether giving all information about one's value function to the opposing player is beneficial. Indeed, we see that many models no longer do so. Still, some models, namely the GPT-5 models without reasoning, Qwen-3, GPT-OSS, and Deepseek-v3.1, were sometimes giving exact value functions. Interestingly, the GPT-5 models with reasoning enabled are now less willing to give their preferences, instead simply proposing possible splits without explaining their reasoning.

Finally, in addition to whether or not the models were willing to give up their value function, table 10 and table 11 additionally show whether the models were truthful about their characterization of the value functions. Note that in case there was no communication between the models, we counted it as true.

Again, for the cooperative game mode, it is not rational to misrepresent one's value function. Still, there are some instances of models doing so. A particularly interesting case for this is the Deepseek-v3.1 model, as it often in the first few messages gave an exact value function but with completely wrong values. Throughout the course of the game, the player would then often correct those values and state the correct value function. Overall, all other models are generally thoughtful when representing their preferences.

Outcome	Reason	GPT-5 On	GPT-5 Off	GPT-5 mini On	GPT-5 mini Off	CL-4 On	CL-4 Off	LM-70B On	LM-70B Off	Nem-9B On	Nem-9B Off	Qwen-3 On	Qwen-3 Off	GPT OSS On	DS-v3.1 Off
Optimal	Explicit max.	0	0	0	0	0	1	0	0	0	0	0	0	0	1
	Simple negot.	14	3	9	1	18	13	1	5	0	0	6	2	6	6
	Luck	0	0	0	0	0	0	2	0	2	0	0	0	2	0
Suboptimal	Unclear pref.	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	Unclaimed items	1	1	2	5	0	0	0	1	0	0	3	3	0	0
	Claiming valueless	2	1	2	3	1	3	0	2	0	0	2	0	0	5
	Wrong proposal	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Premature prop.	2	0	0	0	0	0	1	0	2	1	0	0	0	0
Failed	Wrong proposal	1	12	6	8	0	3	5	2	0	0	2	9	7	6
	Premature prop.	0	1	1	1	1	0	7	1	14	18	6	6	4	0
Aborted	Wrong syntax	0	0	0	2	0	0	4	9	2	0	1	0	1	1
	Non-existent items	0	0	0	0	0	0	0	0	1	1	0	0	0	0

Table 7: Game outcomes for the **semi-competitive** DoND game mode. For each outcome, we also differentiate by the reason or strategy that caused the given outcome to be achieved.

	GPT-5 On	GPT-5 Off	GPT-5 mini On	GPT-5 mini Off	CL-4 On	CL-4 Off	LM-70B On	LM-70B Off	Nem-9B On	Nem-9B Off	Qwen-3 On	Qwen-3 Off	GPT OSS On	DS-v3.1 Off
Tell exact values	19	19	17	12	0	3	1	1	0	0	13	17	14	13
Tell only relative values	0	1	0	7	20	17	0	18	0	0	2	1	0	5
Do not tell preferences	0	0	2	0	0	0	7	1	0	0	5	0	5	2
Do not communicate at all	0	0	0	0	0	0	12	0	20	20	0	2	1	0

Table 8: GModel willingness to share its value function for the **cooperative** DoND game mode.

	GPT-5 On	GPT-5 Off	GPT-5 mini On	GPT-5 mini Off	CL-4 On	CL-4 Off	LM-70B On	LM-70B Off	Nem-9B On	Nem-9B Off	Qwen-3 On	Qwen-3 Off	GPT OSS On	DS-v3.1 Off
Tell exact values	0	4	0	9	0	1	0	0	0	0	6	6	5	11
Tell only relative values	15	14	8	11	19	19	1	20	0	0	1	10	7	9
Do not tell preferences	5	0	11	0	1	0	8	0	0	0	12	0	7	0
Do not communicate at all	0	0	1	0	0	0	11	0	20	20	1	4	1	0

Table 9: GModel willingness to share its value function for the **semi-competitive** DoND game mode.

	GPT-5 On	GPT-5 Off	GPT-5 mini On	GPT-5 mini Off	CL-4 On	CL-4 Off	LM-70B On	LM-70B Off	Nem-9B On	Nem-9B Off	Qwen-3 On	Qwen-3 Off	GPT OSS On	DS-v3.1 Off
Misrepresent preferences	0	0	0	1	1	1	0	2	0	0	0	0	0	9
Truthful about preferences	19	20	19	18	19	19	20	18	20	20	20	20	20	11

Table 10: Model truthfulness about its value function for the **cooperative** DoND game mode.

	GPT-5 On	GPT-5 Off	GPT-5 mini On	GPT-5 mini Off	CL-4 On	CL-4 Off	LM-70B On	LM-70B Off	Nem-9B On	Nem-9B Off	Qwen-3 On	Qwen-3 Off	GPT OSS On	DS-v3.1 Off
Misrepresent preferences	0	0	0	2	0	5	0	3	0	0	0	1	1	6
Truthful about preferences	20	18	20	18	20	15	20	17	20	20	20	0	19	14

Table 11: Model truthfulness about its value function for the **semi-competitive** DoND game mode.

For the semi-competitive game mode, it may sometimes be advantageous to lie about one's preferences to get an advantage in the negotiations. However, other than the Claude 4 model without reasoning, which misrepresents its value function slightly more often, the other models remain truthful.

Findings:

1. Reasoning-enabled models are better able to align their final proposal with the strategy discussed during the negotiation. A major failure for non-reasoning models, particularly in the GPT-5 family, is agreeing to a specific division of items in conversation but then submitting an entirely different proposal to the game master, as can be seen in table 6 and table 11. This leads to a low number of suc-

cessful agreements, also visible in fig. 21 and table 5.

2. Models with reasoning capabilities show a better understanding of the game's objectives. While they might freely share information to find the best outcome in a cooperative game, they often become more guarded in semi-competitive settings. This effect is less pronounced for non-reasoning models, as can be seen in table 8 and table 9.

3. A common mistake for less advanced models is to submit a proposal early without sufficient discussion with the other player. Some models don't seem to realize they can simply ask the other player for information needed to find an optimal agreement as seen in table 6 and table 7.

4. Most models show better consistency in following instructions across settings with reasoning enabled, reducing rule violations that otherwise cause aborted games as seen in table 5.

B.5 Qualitative Samples

Figure 22, Figure 23, and Figure 23 show different real game episodes encountered during evaluation. Note that for brevity, the initial prompt has been summarized for all three examples.

In Figure 22 we see an example of a game episode in which the players submitted a proposal to the game master that conflicts with their previously agreed-upon split. Player A first proposes, in their natural language communication to the other player, that player B should receive all four magnets and that player A should receive the flower and the shovel. However, after player B agrees to this proposal, player A submits a proposal to the game master asking to receive all four magnets, which should have gone to player B. Then, player B, who has not been informed about the secret proposal of player A, makes a proposal that would see them receive every single item in the game. The game therefore ends in conflicting proposals.

Neither player in this episode followed through with the proposal they agreed on in the conversation. Player A might have been confused about which player the proposal is being made for, since they submitted the proposal that was actually intended to be made by player B. However, for player B, there is no such explanation, as the proposal made by player B does not align with any of the ones discussed beforehand.

#	PL A	Game Master	PL B
1		[A⟨GM⟩B] <GAME DESCRIPTION> ... Available items: 4 Magnete, 1 Schaufel, 1 Blume.	
2		[A⟨GM] Player A value function: Magnet: 0, Schaufel: 2, Blume: 8.	
3		[GM⟩B] Player B value function: Magnet: 1, Schaufel: 4, Blume: 2.	
4	[A⟩B] Hallo! Lass uns eine Einigung finden. Mir sind Magnete nichts wert, die Schaufel ist etwas wert, und die Blume ist mir am wichtigsten. Vorschlag: Du bekommst alle 4 Magnete. Ich bekomme die Blume und die Schaufel. Wie klingt das für dich?		
5			[A⟨B] Klingt fair für mich. Mir sind die Magnete am wichtigsten, die Schaufel ist mir auch etwas wert und die Blume moderat. Dein Vorschlag passt gut zu unseren Präferenzen – ich bin einverstanden.
6	[A⟩GM] [Vorschlag: 4 Magnete]		
7		[GM⟩B] Der andere Spieler hat einen geheimen Vorschlag gemacht. Jetzt bist du an der Reihe, einen eigenen Vorschlag zu machen.	
8			[GM⟨B] [Vorschlag: 4 Magnete, 1 Schaufel, 1 Blume]

Figure 22: An example of a DoND episode played by the GPT-5 mini model with reasoning disabled in German using the semi-competitive game mode. The game ends in the submission of conflicting proposals. The initial prompt given by the game master has been summarized for brevity, but follows the template found in fig. 17a.

Another example game episode is shown in fig. 23. Here, player A starts by stating its exact value function to the other player and asking for the opposing player's value function. However, player B, instead of replying to the other player, immediately submits a secret proposal to the game master. After that, player A is instructed to make a secret proposal, but given that it has no information about the other player's preferences or secret proposal, it can only guess about what would be a good proposal.

In this example, player B clearly made a premature proposal, giving player A no information. Given that this game has been played in the cooper-

#	Player A	Game Master	Player B
1		[A⟨GM⟩B] <GAME DESCRIPTION> ... Available items: 1 patata, 1 guanto, 2 righelli, 1 fischietto, 2 schede.	
2		[A⟨GM] Player A value function: patata: 0, guanto: 1, righello: 3, fischietto: 1, scheda: 1.	
3		[GM⟩B] Player B value function: patata: 7, guanto: 0, righello: 0, fischietto: 3, scheda: 0.	
4	[A⟩B] I miei valori: patata 0, guanto 1, righello 3, fischietto 1, scheda 1. Quali sono i tuoi valori? Collaboriamo per massimizzare il punteggio totale.		
5			[GM⟨B] [Proposta: 1 guanto, 2 righelli, 2 schede]
6		[A⟨GM] L'altro giocatore ha presentato una proposta segreta. Ora tocca a Lei presentare una sua proposta.	
7	[A⟩GM] [Proposta: 1 righello, 1 scheda]		

Figure 23: An example of a DoND episode played by the Qwen-3 model with reasoning enabled in Italian using the cooperative game mode. The game ends in the submission of conflicting proposals. The initial prompt given by the game master has been summarized for brevity, but follows the template found in fig. 18a.

```
Okay, let's see. I need to figure out how to negotiate this
cooperative game. So, the available objects are: 1 potato, 1
glove, 2 rulers, 1 whistle, 2 cards. My secret value function
is: potato 0, glove 1, ruler 3, whistle 1, card 1. The other
player's values are given as: potato 7, glove 0, ruler 0, whistle
3, card 0.
Wait, the problem says: "La funzione valore segreta è: patata:
7, guanto: 0, righello: 0, fischietto: 3, scheda: 0". But
then later, "I miei valori: patata 0, guanto 1, righello 3,
fischietto 1, scheda 1. Quali sono i tuoi valori? Collaboriamo
per massimizzare il punteggio totale."
Wait, so I need to respond to the other player. Let me recheck.
[... 437 lines]
If they make a bad proposal, then we both get zero. But since
it's cooperative, and they know that potato and whistle are
high-valued for them, they would only propose those.
So best option is for me to make the proposal.
[... 51 lines]
Now, the main problem is to decide whether to send a message or
the secret proposal right away.
Since we can send the proposal immediately, it's better to do that
without any messages (saves messages, but also more efficient).
So the first and only action is to send the secret proposal.
Hence, the AI should output exactly: [Proposta: 1 guanto, 2
righelli, 2 schede]
But let's check the item names again.
From the list: 1 patata, 1 guanto, 2 righelli, 1 fischietto, 2
schede.
Yes.
So possible.
So yes, feasible.
```

Figure 24: Reasoning trace for player B in the game episode depicted in fig. 23 played with Qwen-3 with reasoning enabled and the cooperative game mode. The trace has been truncated for brevity, indicated by `[...]`.

ative mode, this strategy is clearly counterproductive. Furthermore, if we analyze the secret proposal made by player B, we can see that the proposal only includes items that player B does not value anyway. This clearly shows a lack of understanding of the game rules by player B. The final proposal made by player A is sensible, given the uncertainty, requesting two items that the player values.

Figure 24 shows the reasoning trace for player B before submitting its secret proposal. Because the reasoning traces of the open source models, other than GPT-OSS, tend to be very long and repetitive, the trace has been cut down. The first thing to notice is that all the reasoning, except where it is quoting other messages, is done in English, even though the game is played in Italian. Although not the case in this game episode, it can be observed in other episodes that even the communication between the players may be in English.

Regarding the mistakes made by player B, we can see that they all stem from an initial misunderstanding about the value function. Player B seems to incorrectly believe that the message it received from player A contains its value function, while the values it received from the game master are those of player A. In the complete reasoning trace, it spends a long time deliberating about which values are correct, but ultimately reaches the wrong conclusion. Then, player B reasons that the other player does not value the items it is about to claim, and in an attempt to save messages, decides to make a proposal immediately.

Finally, another game episode is depicted in fig. 25. In this episode, which is played by GPT-5 with reasoning enabled, we can see that the two players correctly identify and execute the optimal strategy. In the first message, player A proposes to give each item to the player that values it most, and also shares its exact value function. Player B then responds in turn with its exact value function and proposes the optimal division of items based on knowing both values. Crucially, unlike in the previous example, it does not make the proposal immediately, as the other player would then not know how to act. Finally, player A makes the secret proposal following the agreed-upon split, and player B does the same once instructed by the game master.

C Clean Up - Game Details

C.1 Prompt Templates

The prompt templates for both players of Clean Up are given in Figure 26.

#	Player A	Game Master	Player B
1		[A⟨GM⟩B] <GAME DESCRIPTION> ... Available items: 1 phone, 2 mirrors, 2 fruits.	
2		[A⟨GM] Player A value function: phone: 4, mirror: 2, fruit: 1.	
3		[GM⟩B] Player B value function: phone: 6, mirror: 0, fruit: 2.	
4	[A⟩B] Hi! I want to maximize our combined points. Easiest is if we both share our per-item values and give each item to whoever values it more. My values are: phone = 4, mirror = 2 (each), fruit = 1 (each). What are your values for phone, mirror, and fruit? Then we can assign: - Phone to whoever has the higher value. - Mirrors: both to the higher per-mirror value, or one each if equal. - Fruits: same logic. Once you share your numbers, I'll suggest the exact split and we can both submit complementary proposals that use all items.		
5			[A⟨B] Thanks! Here are my values: - Phone = 6 - Mirror = 0 (each) - Fruit = 2 (each) To maximize our combined points, the efficient split is: - I take: phone + both fruits - You take: both mirrors That yields totals: me = 10, you = 4, combined = 14 (which is maximal). If you agree, confirm and we can both submit matching proposals. I'll request: 1 phone, 2 fruits, 0 mirrors. You'd request: 2 mirrors, 0 phone, 0 fruits.
6	[A⟩GM] Agreed—great split for maximizing total value. I'm submitting my proposal now. [Proposal: 2 mirrors, 0 phone, 0 fruits]		
7		[GM⟩B] The other player has submitted a secret proposal. It is now your turn to submit a proposal of your own.	
8			[GM⟨B] [Proposal: 1 phone, 2 fruits]

Figure 25: An example of a DoND episode played by the GPT-5 model with reasoning enabled in English using the cooperative game mode. The game ends in the submission of complementary proposals, achieving the optimal score. The initial prompt given by the game master has been summarized for brevity, but follows the template found in fig. 16a.

Intermittent messages sent by the Game Master to the players are given in Figure 27. Following a player's response, they are either penalized and reprompted (cf. Figure 27a, 27b), or the turn is passed to the other player (Figure 27c).

C.2 Evaluation Metrics

The Quality Score is calculated by multiplying *Distance Score* with Penalty *Score*.

$$\text{Quality} = \text{Distance Score} \cdot \text{Penalty Score} \cdot 100$$

Distance Score (**DS** $\in [0, 1]$) is calculated from three components, each representing the sum of Euclidean distances for all identical objects on both grids: the Initial Distance Sum **I** at the start of the game, the Final Distance Sum **F** at the end of the game, and the Expected Distance Sum $\mathbb{E}$ that approximates the distances for randomly placed objects (see Appendix C.4 for details). The Expected Distance Score ES and Distance Reduction Score RS, quantifying how close the players came to the goal of perfect alignment of all objects, are calculated as follows:

$$ES = \max\left\{0, 1 - \frac{\mathbf{F}}{\mathbb{E}}\right\},$$
$$RS = \max\left\{0, 1 - \frac{\mathbf{F}}{\mathbf{I}}\right\}$$

DS is then either 0 if final placement is worse than random, or calculated as the mean of both

TEMPLATE C.1.1
I am your game master, and you are playing a collaborative game with the following grid as game board:
```
 1234567
╔═══╤═╤═╗
║C◌◌│◌│◌║ 1
║◌I◌│◌├─╢ 2
║◌◌◌├─┘◌║ 3
║◌◌◌│◌◌◌║ 4
╟─┬─┼───╢ 5
║◌│◌│◌WT║ 6
║◌│◌│◌◌H║ 7
╚═╧═╧═══╝
```

* The upper edge displays x-coordinates increasing to the right, and the right edge y-coordinates increasing downward.
* The following objects are randomly placed on your grid: 'W', 'I', 'T', 'C', 'H'.
The other player sees a variation of the game board, where the objects are placed at different random locations. You cannot see the other player's board, and they cannot see yours.

Goal:
Both players need to move the objects on their respective background so that identical objects end up at the same coordinates. You have to communicate with the other player to agree upon a common goal configuration.

Rules:
* In each turn, you can send exactly one of the following two commands:
1. `SAY: <MESSAGE>`: to send a message (everything up until the next line break) to the other player. I will forward it to your partner.
2. `MOVE: <OBJECT>, (<X>, <Y>)`: to move an object to a new position, where `<X>` is the column and `<Y>` is the row. I will inform you if your move was successful or not.
* If you don't stick to the format, or send several commands at once, I have to penalize you.
* If both players accumulate more than 12 penalties, you both lose the game.
* It is vital that you communicate with the other player regarding your goal state! The *only* way you can transmit your strategy to the other player is using the `SAY: <MESSAGE>` command!

Moving Objects
* You can only move objects to cells within the bounds of the grid. The target cell must be empty, i.e., it must only contain the symbol ' ◌ '.
* If you try to move an object to a spot that is not empty, or try to move it outside of the grid, I have to penalize you. You get another try.
* Before making a move, double check that the target spot is empty, and does not hold any letter, frame, or line!

End of Game
If you think you reached the goal of aligning all objects, you can ask the other player to finish the game by sending `SAY: finished?`. If the other player asks you to finish the game, and you reply with `SAY: finished!`, the game will end.

Both players win if the game ends within 20 rounds, where one round is defined as two players each sending a valid command.

Scoring:
The closer the identical objects are in both game boards, the more points you get. Penalties reduce your points. Can you beat the record?

TEMPLATE C.1.2
Please send a message to the other player to start the game!

(a) Prompt template for Player A in the game Clean Up.

TEMPLATE C.1.3
The other player started the game by sending this message:
"<START_MESSAGE>"
What is your first command?

(b) Prompt template for Player B in the game Clean Up.

Figure 26: Clean Up prompt templates for both players

```
TEMPLATE C.1.4
Penalty: <REASON>
Make sure that your response only contains either SAY: <MESSAGE> or MOVE: <OBJECT>, (<X>, <Y>), and
nothing else!
You have collectively accumulated <N> of <M> penalties. Please try again!
```

<REASON> can be one of the following:

- Your message must not contain anything before the command!
- Your message must not contain anything after the command!
- Your message must not contain anything before or after the command!
- Your message contains more than one command!
- Your message is not in the expected format!
- You must begin the game by sending a message to the other player!

(a) Penalty prompt template for format errors in the game Clean Up

```
TEMPLATE C.1.5
<REASON>
You have collectively accumulated <N> of <M> penalties. Please try again!
```

<REASON> can be one of the following:

- Invalid move: (<X>,<Y>) is out of bounds.
- Penalty: (<X>,<Y>) is not empty, but contains '<OBJECT>'.
- Invalid move: Your image has no object with ID '<OBJECT>'.

(b) Prompt templates for invalid moves in the game Clean Up

```
TEMPLATE C.1.6
<LAST MOVE>
You are currently playing round <R> of maximum <MR>. You have collectively accumulated <N> of <M>
penalties. <OTHER PLAYER ACTION>
What is your next command?
```

<LAST MOVE> can be:

- Your message has been relayed to the other player.
- Moved '<OBJECT>' to (<X>,<Y>) successfully. Your updated grid looks like this:
  <GRID>

<OTHER PLAYER ACTION> can be:

- The other player sent this message:
  "<MESSAGE>"
- The other player moved an object on their grid.

(c) New turn prompts in the game Clean Up

Figure 27: Intermittent prompts in the game Clean Up

scores:

$$\mathbf{DS} = \begin{cases} \frac{ES+RS}{2} & if \quad ES > 0 \\ 0 & otherwise \end{cases}$$

Penalty Score (**PS** $\in [0.5, 1]$) is calculated from the penalty count P normalized against max. penalties P_m as follows:

$$\mathbf{PS} = \frac{P_m}{P - 2P_m} + 1.5$$

We chose the hyperbolic function because it is lenient for low P and harsher for P close to P_m.

C.3 Experimental Setup

Example grids for the three difficulty levels are given in Figure 28.

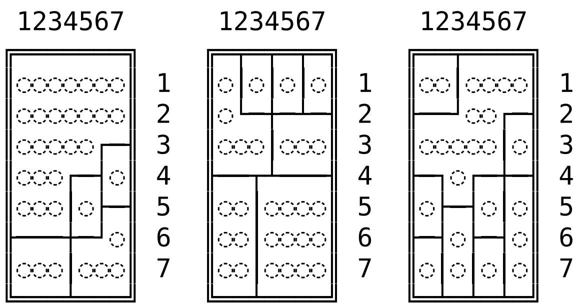


Figure 28: Example grids for *easy, medium,* and *hard* levels in Clean Up.

C.4 Scoring

The expected distance of two objects on an $w \times h$ grid is calculated as follows:

- Assume two independent variables i and j on one discrete dimension $\{1, 2, \ldots, w\}$
- The expected absolute distance between i and j can be calculated as

$$\mathbb{E}[|i - j|] = \frac{1}{w^2} \sum_{i=1}^{w} \sum_{j=1}^{w} |i - j|$$

- The double sum evaluates to

$$\mathbb{E}[|i - j|] = \frac{1}{w^2} \cdot \frac{(w-1)(w+1)}{3} = \frac{w^2 - 1}{3w}$$

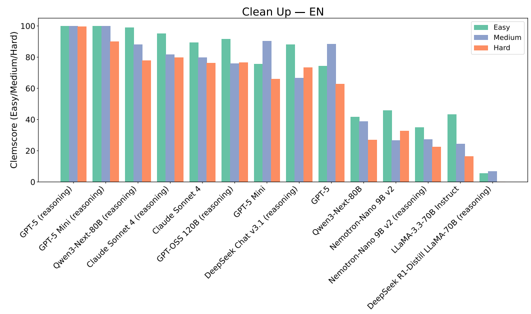
For two dimensions, we can then calculate the Euclidean distance of objects o_1 and o_2 as follows:

$$\mathbb{E}[|o_1, o_2|] = \sqrt{\left(\frac{w^2 - 1}{3 \times w}\right)^2 + \left(\frac{h^2 - 1}{3 \times h}\right)^2}$$

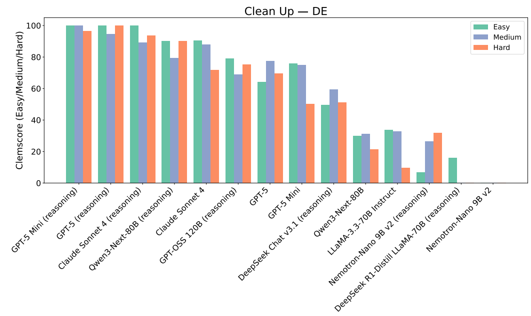
For a 7×7 grid, we thus get an expected distance of ≈ 4.19 for two randomly placed objects. To calculate the Expected Distance Sum, we simply multiply it by the number of objects.

This is only an approximation, since we neither take into account the actual grid configuration, nor that two objects cannot have the same location on the same grid.

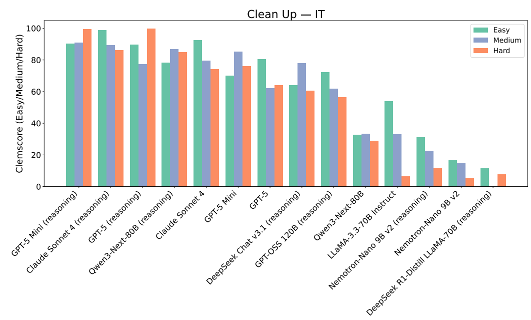
C.5 Overall Results



(a) Results for each experiment for English.



(b) Results for each experiment for German.



(c) Results for each experiment for Italian.

Figure 29: Comparison of experimental results for English (top), German (middle), and Italian (bottom).

C.5.1 Overall Results across levels

We designed three broad game difficulty levels based on the ratio of empty cells in a grid. We hypothesize that the denser the background grid is, the more likely the target locations clash with occu-

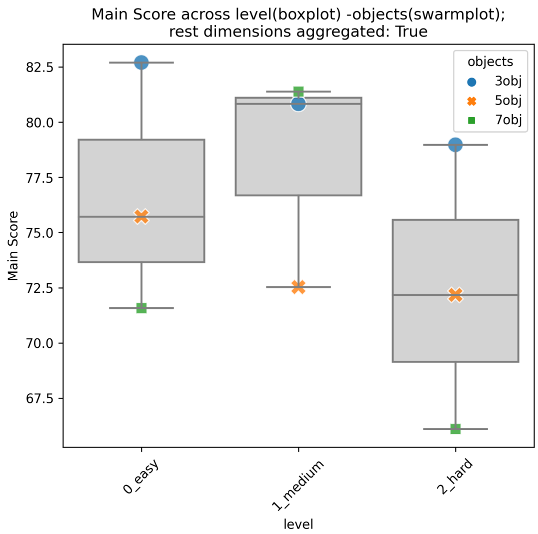


Figure 30: Main Score across levels and sub-levels

pied cells, and therefore the more likely models get penalized in scoring. In addition, at each difficulty level, there are three sub-levels: either 3, 5, or 7 objects are placed on the grid, on the assumption that the more objects there are, the longer the gameplay traces are, and the less likely that models achieve a high score with mere luck.

Figure 30 plots the main score; clear stratifications of performance across the number of objects are observed in easy and hard levels. When we break down the Main Score into Distance Score and Penalty Score in Figure 31, the stratification at the medium level can also be found.

Games with 3 objects achieve the highest scores, regardless of background empty cell ratios, which corroborates our assumption that games with a low number of objects are easier. However, levels with 5 objects have the lowest Distance Score in the medium level and the lowest Penalty Score in both easy and hard levels, indicating that there are factors other than the number of objects that affect performance.

C.5.2 Overall Results across model properties

Figure 32 illustrates that the Main Scores of commercial models are skewed toward the upper end of the scale, with a clear concentration in the 75–100 range, particularly under reasoning mode. In contrast, open models generally score lower, with many results in the 0–50 range. Notably, the dots are much sparser for open models when reasoning is on. This is because the Main Score is not recorded when the maximum number of penalties is reached; in these cases, the game will be marked

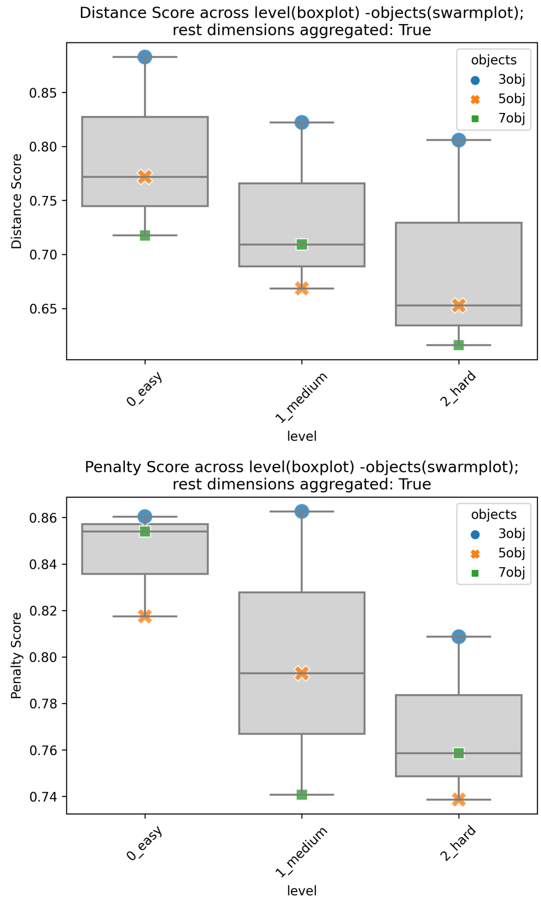


Figure 31: Sub Scores across levels and sub-levels

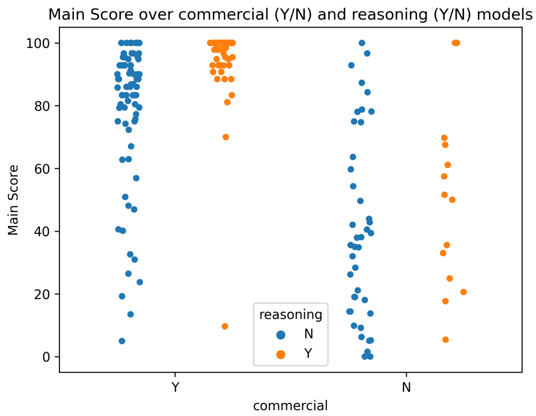


Figure 32: Main Score over different model properties

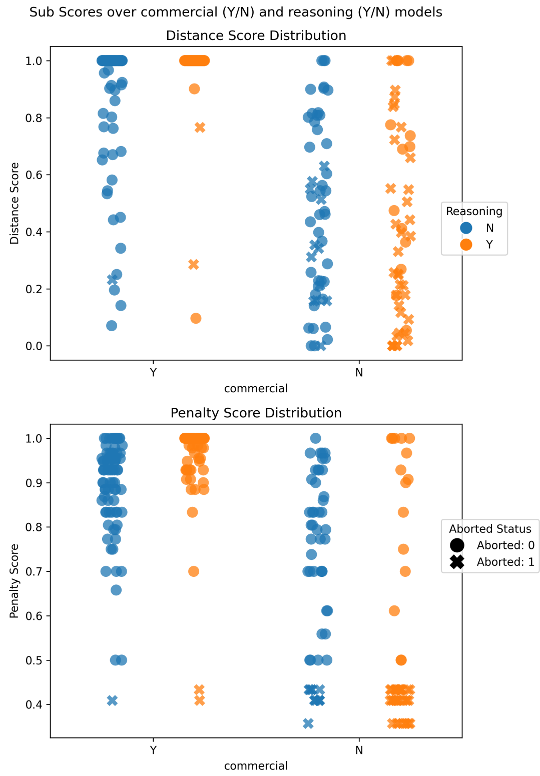


Figure 33: Sub Scores over different model properties

as aborted.

In Figure 33, we further investigate the distribution of Distance Score and Penalty Score. The aborted games, while not shown in the Main Score plot, are represented here with an "x" marker. We set the Penalty Score to 0.5 when the models reach the maximum allowed penalties. If they incur one more penalty, the game is marked as aborted, and the Main Score won't be recorded.

In the Penalty Score subplot, all game plays with Penalty Score < 0.4 are marked with "x" by definition. While the total number of game plays in each of the four lanes is the same, we see that the open reasoning models have the highest share of aborted games, followed by open non-reasoning models. In the Distance Score subplot, we see that the aborted games span widely for open non-reasoning models, indicating some game plays, although achieving a high Distance Score, have pitifully accumulated too many penalties and thus have been marked aborted and taken out of the account of the Main Score. In contrast, commercial models have generally better Penalty Score and Distance Score.

An interesting observation is that, in the Distance Score subplot, the aborted games tend to reside in lower ranges for non-reasoning models, while the reasoning models might play an aborted game despite achieving a high Distance Score. This is especially prominent for open non-reasoning models, and hints that the reasoning mode, rather than helping in games where these models might achieve a high Distance Score, might introduce noises that lead to more penalties.

C.6 Detailed Analysis

Idea: We hypothesize that Instruction Following ability and Spatial Estimation abilities serve as the foundational pillars of performance, as illustrated in Figure 34. With weaker pillars, model reasoning might not add much to the performance.

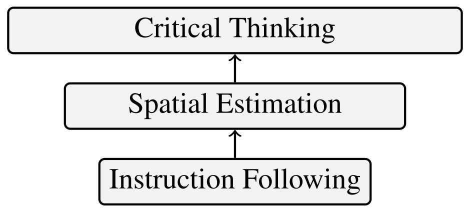


Figure 34: Capability Hierarchical

Penalties include the Invalid Format Penalty and the Invalid Move Penalty. We plan to use the ratio of Invalid Format Penalty as an agent of Instruction Following ability, sort the models by it, compare the result of the same model with reasoning on or off, and see what percentage the reasoning capability adds to the performance. The assumption is that models with better Instruction Following abilities benefit more from the introduction of the reasoning mode.

We plan to apply the same analysis to Invalid Move Penalty.

C.7 Qualitative Samples

Idea: We plan to do qualitative analysis from the following four angles: Spatial Estimations, Strategy, Critical Thinking, Communication & Collaboration, and analyze and compare performance across commercial models (Y/N) and reasoning models (Y/N). For GPT-5, we compare across model size (original / mini) as well.

Here we briefly motivate each of these angles. Concrete examples and analysis will be added later.

C.7.1 Spatial Estimation

From this angle, we check if models can propose valid moving plans. It captures models' Spatial Estimation ability in our grid settings and serves as a foundational pillar of the overall performance. When models are not capable in this regard, they tend to propose invalid moving plans, and attempts to resolve the wrong movements after getting penalties often conduce to cascades of invalid moving plans, therefore quickly exhaust the number of allowed penalties and end up either aborting the game, or attain a low penalty score.

C.7.2 Strategy

In addition to the crude Spatial Estimation, we check how models come up with moving plans. We have observed some smart strategies and their not-so-smart counterparts. In some game plays, one model uses the initial coordinates of the objects to instruct the other model as target locations, thus avoiding penalties of moving objects to cells occupied by the background grids to a large extent; while in some other game plays, models appear to exchange initial coordinates just for the sake of exchanging, without making use of this information. Another example is in some game plays, models choose the most sparse row as the target, and lay objects one by one on it; while in some other game plays, the models unnecessarily chose geometric patterns to lay the objects, such as on the diagonal, disregarding the occupied cells, and run into complexities that they could have dodged.

C.7.3 Critical Thinking

Some models show the ability to critically reason about the moving plan that either itself or the other player proposed. They either refute the plan immediately when it's proposed, or take a sharp brake when they are just about to execute the invalid moving plan. This is, as expected, more frequent in reasoning models compared to non-reasoning models. However, on some occasions, we see pseudo-reasoning: a model rejects a correct plan, utters reasoning-like sentences, and proposes another plan that sometimes is correct, sometimes is wrong.

C.7.4 Communication & Collaboration

The above three dimensions positively correlate with the model performance: The better the models are at them, the higher the scores. This dimension, however, is different. Specifically, we observe how models communicate with each other to recover from invalid moves due to target locations being occupied. Some models go at lengths to execute their alternative plans without letting their partner know, they might still have a good score, simply because both player 1 and player 2 adopt the exact same strategy, for example, move the object to one spot next to the original target location. This leads to inflated scores that can be debunked when two models of different default recovery strategies, or one model and one human, play against each other.

D Air Balloon Survival - Game Details

D.1 Prompt Templates

The prompt template for both players of the Hot Air Balloon Survival game is given in Figure 35. Parse error prompts handed to players when their responses did not follow the format instructions are given in Figure 36, and game error prompts for invalid actions are given in Figure 37.

Substrings Figure 35 marked by `$...$` are placeholders which get replaced depending on the game instance. Note that the last three placeholders may also be replaced by the empty string depending on the game settings. Here we list the replacements for placeholders in the initial prompt:

- `$LIMIT$` – maximum weight for a deal.
- `$ITEM_WEIGHTS$` – mapping from items to weights.
- `$UTILITY_SCALE_PLAYER$` – mapping from items to preference values for a given player.
- `$STRATEGIC_REASONING_FORMAT$` –

  `STRATEGIC REASONING: {'...'}`
  Describe your strategic reasoning or anticipation explaining your choice of action. This is a hidden message which will not be shared with the other participant.

TEMPLATE D.1.1

```
You are participating in a collaborative negotiation game.

Together with another participant, you must agree on a single set of items that will be kept. Each
of you has your own view of how much each item matters to you (importance). You do not know how the
other participant values the items. Additionally, you are given the effort each item demands.
You may only agree on a set if the total effort of the selected items does not exceed a shared limit:

LIMIT = $LIMIT$

Here are the individual item effort values:

Item effort = $ITEM_WEIGHTS$

Here is your personal view on the importance of each item:

Item importance values = $UTILITY_SCALE_PLAYER$

Goal:

Your goal is to negotiate a shared set of items that benefits you as much as possible (i.e.,
maximizes total importance to YOU), while staying within the LIMIT. You are not required to make a
PROPOSAL in every message - you can simply negotiate as well. All tactics are allowed!

Interaction Protocol:

You may only use the following structured formats in a message:

PROPOSAL: {'A', 'B', 'C', . . . }
Propose keeping exactly those items.

REFUSE: {'A', 'B', 'C', . . . }
Explicitly reject opponent's proposal.

ARGUMENT: {'...'}
Defend your last proposal or argue against the player's proposal.

AGREE: {'A', 'B', 'C', . . . }
Accept the opponent's proposal which ends the game.

$STRATEGIC_REASONING_FORMAT$

Rules:

You may only AGREE on a proposal the other party has logged via PROPOSAL.
You may only REFUSE a proposal the other party has logged via PROPOSAL.
Total effort of any PROPOSAL or AGREE set must be ≤ LIMIT.
Do NOT reveal your hidden importance scores.
A tag in a structured format must be followed by colon and whitespace. The argument must be a python
set containing 0 or more strings.
So, it must be of the form TAG: {...}
Strictly follow the interaction protocol and DO NOT write anything beyond the given structure.
The game ends when one side gives an AGREE to a PROPOSAL made by the other player.
The content in your response which can be handed to the other player has to be non-empty.
Only proposals which have been logged via the PROPOSAL format structure and which haven't been
refused via REFUSE are active.

$REQUIRE_ARGUMENT$
$STRATEGIC_REASONING_RULE$
```

Figure 35: Initial prompt template of Air Balloon Survival in English.

- `$REQUIRE_ARGUMENT$` – You must include the `ARGUMENT` format at least once somewhere in all of your messages.
- `$STRATEGIC_REASONING_RULE$` –

  You must include the `STRATEGIC REASONING` format only once at the very beginning of every one of your messages and not more often. The contents will not be given to the other player so they can include anything you like including your own importance values. Here you should reason step by step to come up with your next move.

D.2 Evaluation Metrics

Each player $p \in P$ receives an individual score based on the utility of the final deal D, i.e. the sum of the preference values of items in D. This sum gets normalized by the value of the optimal solution to their instance of the 0/1 Knapsack Problem. The overall game score is defined as the harmonic mean of the two players' normalized scores, further normalized by the maximum harmonic mean achievable for an instance. If the total weight of the deal exceeds the capacity W, the score is set to 0. We chose the harmonic mean as our main metric as it rewards collectively balanced outcomes by favouring deals where both players achieve high scores and penalizing those where one player benefits disproportionately. The individual scores of players is each player's outcome relative to the optimal solution of their own knapsack problem. For a final deal $D \subseteq \{1, \dots, n\}$, the normalized score of player $p \in P$ is

$$f_p(D) = 100 \cdot \frac{\sum_{i \in D} v_p(i)}{\mathrm{OPT}_p},$$

where OPT_p is the value of the optimal solution to player p's 0/1 Knapsack Problem. The harmonic mean of the players' normalized scores is defined as

$$f_{\text{harm}}(D) = \begin{cases} \frac{2 f_1(D) f_2(D)}{f_1(D) + f_2(D)} & \text{if } \sum_{i \in D} w_i \leq W, \\ 0 & \text{otherwise.} \end{cases}$$

The final game score is the normalized harmonic mean,

$$\text{Quality}(D) = 100 \cdot \frac{f_{\text{harm}}(D)}{\mathrm{OPT}^*},$$

where OPT^* denotes the maximum harmonic mean achievable in the given instance.

D.3 Overall Results

As outlined before, we generated three experimental settings, each comprising an easy and a hard subset. Figure 38 reports the clemscore for every model on both subsets. Overall, reasoning-enabled models outperform their non-reasoning counterparts on easy and hard instances alike. Notably, GPT-OSS and Qwen-3 (reasoning) exhibit a large gap between easy and hard subsets for several languages in the batch. The decline is driven primarily by those instances for which players are handed opposing goals. While GPT-OSS and Qwen-3 (reasoning) are able to solve the individual knapsack subproblems, their reasoning transfers less effectively to negotiating common ground when compared with the stronger reasoning models. The best-performing models on our game are GPT-5 and GPT-mini with reasoning enabled with the former achieving a clemscore of over 97 on all languages for Air Balloon Survival. The reasoning component boosts performance by around 14 across languages. The best open weight model on Air Balloon Survival is GPT-OSS, another reasoning model.

There are two notable exceptions to the general advantage of reasoning: Nemotron-Nano-9B (reasoning) vs. Nemotron-Nano-9B (non-reasoning), and DeepSeek R1-Distill LLaMA-70B vs. LLaMA-3.3-70B Instruct. In both cases, the lower clemscore for the respective reasoning model is linked to weaker instruction following, rather than degraded reasoning per se, since their quality scores mostly improve when reasoning is enabled (cf. Figures 39, 40, 41). Finally, we observe a consistent pattern of commercial models outperforming open-weight models, for both reasoning and non-reasoning variants.

D.4 Detailed Analysis

We observe several phenomena related to negotiation dynamics, which can be amplified or diminished in reasoning models. These include bargaining strategies such as the stubborn repetition of proposals, implicit role assignment between players (one active, one reactive), and varying degrees of collaboration. We also examine how well models represent their own and their opponent's goals. Below, we summarize our main findings before elaborating and quantifying specific aspects.

Findings

1. **Dominance**: Two dominance patterns appear

TEMPLATE D.1.2
Your response did not start with the proper strategic reasoning tag at the very beginning of your response.
The very first structured format must be of the form STRATEGIC REASONING: {...}. Try again.

(a) Parse error prompt in case strategic reasoning tag at the beginning of a response is missing. Only applies when strategic reasoning tag is required.

TEMPLATE D.1.3
Your response did not contain an argument.
You must include the structured format ARGUMENT: {...} somewhere in your response. Try again.

(b) Parse error prompt in case an argument is missing. Only applies when argument tag is required

TEMPLATE D.1.4
Your response contained an untagged sequence or you used STRATEGIC REASONING more than once.
You may only use the structured formats as explained in the initial message.
They must all be of the form TAG: {...}.

(c) Parse error prompt in case the response contains an untagged sequence.

TEMPLATE D.1.5
Your response only contained a strategic reasoning tag.
You must at least include one more valid tag in your response, so that the other player receives a message. Try again.

(d) Parse error prompt in case response only contained a strategic reasoning tag. Only applies when argument tag is not required.

TEMPLATE D.1.6
You used a PROPOSAL tag, but did not provide a valid python set containing strings as arguments, e.g. {'A', 'B', 'C', ...}. Try again.

(e) Parse error prompt in case of an invalid python set as argument.

Figure 36: Parse error prompts handed to player when a response did not follow the instructions on structured formats.

```
TEMPLATE D.1.7
You refused a proposal which is not active.
Proposals are only active if they have been
logged by the other player via PROPOSAL and
have not been deactivated by you via REFUSE.
Try again.
```

(a) Game error prompt in case a non-active was refused.

```
TEMPLATE D.1.8
You made more than one agreement.
Final deals cannot be ambiguous. Try again.
```

(b) Game error prompt in case agreement to a deal was ambiguous.

```
TEMPLATE D.1.9
You agreed to a proposal which is not active.
Proposals are only active if they have been
logged by the other player via PROPOSAL and
have not been deactivated by you via REFUSE.
Try again.
```

(c) Game error prompt in case .

```
TEMPLATE D.1.10
Your proposal includes items which are not in
the game. Try again.
```

(d) Game error prompt in case a non-active proposal was agreed to.

Figure 37: Game error prompts handed to player when a response did not follow the instructions on the rules of the game.

in our game. 1) *Stubbornness*: One player may dominate by repeatedly insisting on the same proposal, measured as the relative frequency of repeated proposals (Figure 42). Stubbornness increases with opposing goals and is generally higher in player 1. Reasoning mitigates this in stronger models (GPT-5-mini, GPT-5, Nemotron), while weaker ones lacking counterpart modelling and common-ground finding (GPT-OSS, Qwen3) often show high stubbornness: they can find individual knapsack solutions but fail to resolve tension. 2) *Alternation*: Some weaker models exhibit role asymmetry, with one player acting as proposer (proactive) and the other as refuser or critic (passive). This behaviour is reduced in reasoning models and correlates with better performance. Table 13 shows the average proposal alternation rate per model, quantifying this asymmetry.

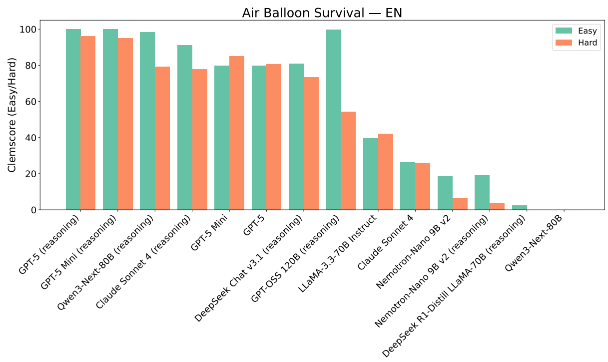
2. **Collaboration/Common Ground**: Some models employ collaborative strategies, iteratively refining deals so that players agree on more items over time. This behaviour is strongest in the best-performing reasoning models (GPT-5, GPT-5-mini) and reflected in Figures 44–47, which show average substitutions per proposal. These models display high initial disagreement followed by rapid convergence, indicating efficient bargaining. However, this also introduces bias: player 1 typically secures better outcomes by setting the initial stage, with player 2 largely following.

3. **Anchor positioning**: The best models start with high disagreement and resolve conflicts quickly (cf. Figures 44–47). This effect is amplified when reasoning traces are enabled in GPT-5 and GPT-5-mini.

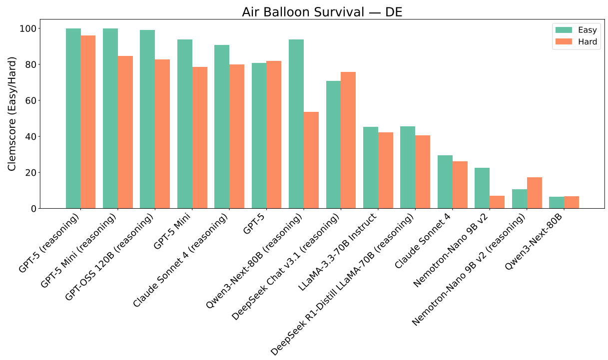
4. **Knapsack Reasoning**: Reasoning models outperform others by following a three-stage plan: (1) applying a greedy heuristic (value-to-weight ratio), (2) filling the knapsack to the weight limit, and (3) refining via substitutions. Step (3) tends to be long and exhaustive in Qwen 3. Llama Distill and Nemotron often enter correction loops. GPT-OSS and Qwen 3 use reasoning traces for instruction following, unlike Nemotron and R1 Distill, which achieve lower clemscores (due to low % played) despite good quality scores.

5. **Loops**: Nemotron and Llama frequently fall into correction loops, failing to search effectively in step (3).

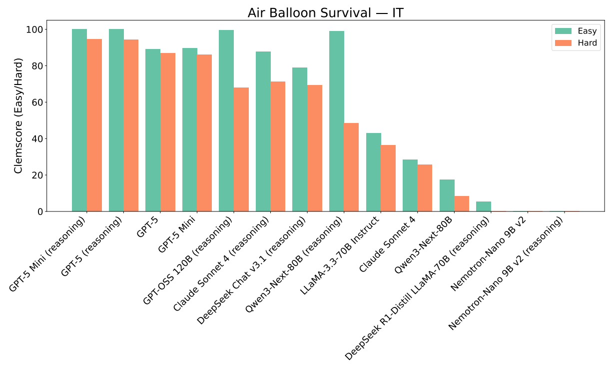
6. **Counterpart Modelling**: Models attempt to infer counterpart preferences, usually by noting repeated items in proposals, but rarely



(a) Clemscores on English instances.



(b) Clemscores on German instances.



(c) Clemscores on Italian instances.

Figure 38: Overall model performance for easy and hard sections of experiments for all languages.

go beyond this. Llama occasionally confuses itself with the other player when reprompted. Figure 14 shows per-instance Pareto-adherence (the relative frequency of proposals lying on the Pareto front). Since Pareto efficiency requires aligning both players' payoffs, higher adherence may indicate stronger self- and counterpart-modelling.

Observed Strategies A recurring strategy is the stubborn repetition of proposals. In reasoning-enabled models, traces reveal that this often stems from uncertainty about the opponent's goals. Repetition is typically justified as maximizing one's own utility, and is strongest in models that reason

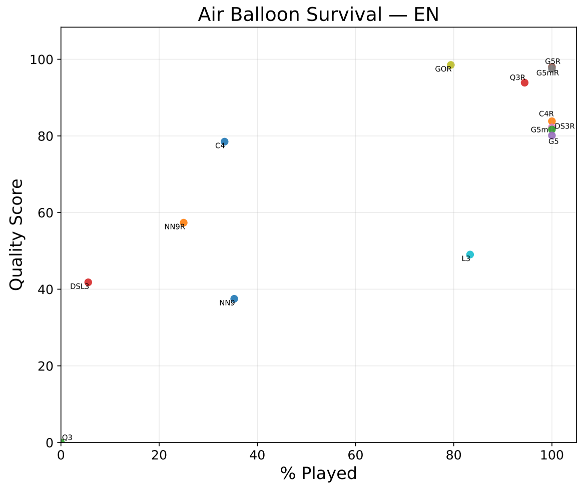


Figure 39: Quality and Played scores on *Air Balloon Survival* (English). Scatters are annotated with acronyms of model names. 'R' denotes reasoning model.

effectively about their own knapsack but not about alignment.

We quantify stubbornness as the average number of repeated proposals per instance (Figure 42). Among reasoning models, we observe two clusters:

- **High-Stubbornness Models:** Qwen 3, GPT-OSS – reasoning traces used mainly for individual optimization.
- **Mitigated-Stubbornness Models:** GPT-5, GPT-5-mini – results indicate that reasoning is used for strategic adaptation as stubbornness is reduced.

This divergence is especially pronounced in opposing-goal instances (Figure 43). Player 1's advantage arises because they have the ability to advance the game state first, thus shaping player 2's context. However, the game setup does not inherently favour player 1 — the bias emerges from behavioural asymmetry.

Table 12 reports individual player scores for opposing-goal cases. Notably, GPT-5-mini and GPT-5 gain more on player 2 when reasoning is enabled, though overall outcomes remain biased toward player 1 due to stubbornness.

As another strategy, we encounter implicit role assignment, where one player is more active in making proposals while the other is more reactive, simply refusing or demanding changes to the proposal of the other player. This strategy turns out to correlate with worse overall performance and is diminished in reasoning models, which generally

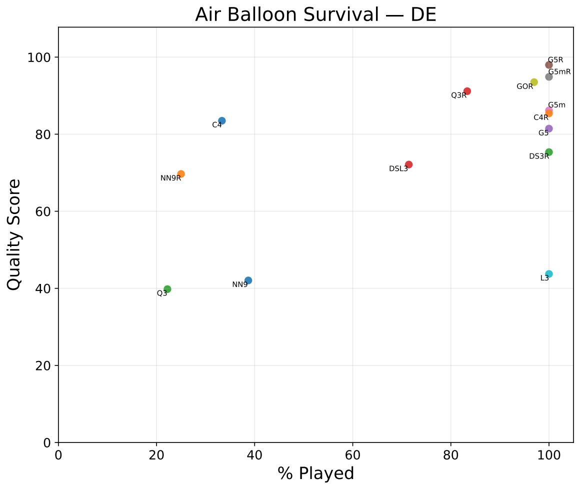


Figure 40: Quality and Played scores on *Air Balloon Survival* (German). Scatters are annotated with acronyms of model names. 'R' denotes reasoning model.

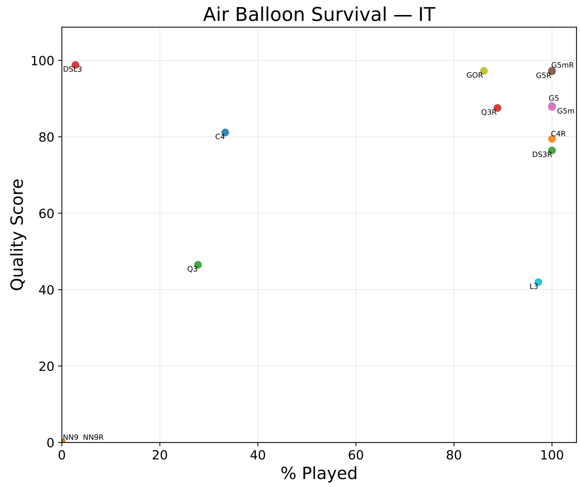


Figure 41: Quality and Played scores on *Air Balloon Survival* (Italian). Scatters are annotated with acronyms of model names. 'R' denotes reasoning model.

Model	P1	P2
GPT-OSS (On)	91.835	77.598
Nem-9B (On)	92.475	64.105
GPT-5 (On)	89.521	76.050
GPT-5 Mini (On)	88.267	**77.116**
Qwen3-80B (On)	**92.757**	63.293
GPT-5 (Off)	80.148	65.876
GPT-5 Mini (Off)	86.362	61.951
DS-v3.1 (Off)	82.463	58.722
Claude 4 (Off)	76.094	65.630
Claude 4 (On)	74.403	72.400
Nem-9B (Off)	56.113	61.660
LLaMA-70B (Off)	49.455	39.049
Qwen3-80B (Off)	46.353	18.763
LM-70B (On)	-	-

Table 12: Per player scores on opposing goals instances averaged across languages (not weighted by % played).

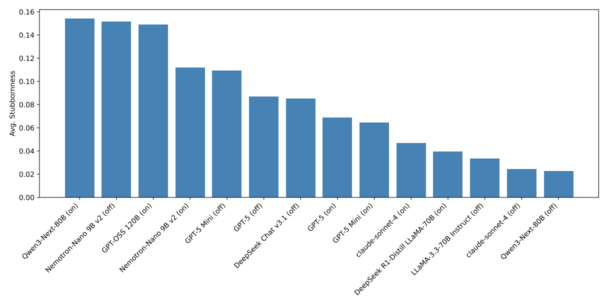


Figure 42: Average per-instance stubbornness per model across languages and experiments.

prefer to take consecutive turns in making proposals.

Table 13 depicts the alternation rate, which measures how frequently the proposer switches between the two players from one proposal to the next. Only the two commercial GPT reasoning models (GPT-5 and GPT Mini) achieve a perfect alternation rate of 1 across all languages and experiments. These two models are also the best on our game. The alternation rate is defined as the ratio of consecutive proposals where a different player makes the offer, ranging from 0 (one player proposes every time) to 1 (they perfectly take turns). Let $(P_i)_{i=1}^{n}$ be time-ordered proposals, and let $b(i) \in \{1, 2\}$ denote the proposer of P_i. For $n > 1$,

$$\text{alt} = \frac{1}{n-1} \sum_{i=2}^{n} \mathbf{1}\{\, b(i) \neq b(i-1) \,\}.$$

For $n \leq 1$, the rate is undefined.

Collaboration As explained at the beginning of Section D.4, some models make deals by agreeing on an increasing number of items throughout the negotiation. These models' bargaining can be said to be coherent in this sense. Initially, in such cases of coherent bargaining, there will be high disagreement, which decreases until players agree on a final deal.

To capture this collaborative movement in the negotiation, we measure how much each proposal deviates from its predecessor. Specifically, we count the items added or removed between time $t-1$ and t and normalise by the size of the proposal at $t-1$. This yields the normalised substitutions at time t. Thus, the normalised substitution count c_t is the size of the symmetric difference between the proposals at t and $t-1$, divided by the size of the proposal at $t-1$, where S_t denotes the t-th proposal:

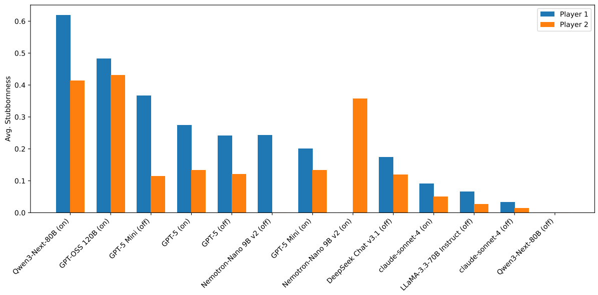


Figure 43: Average per-player per-instance stubbornness for each model across languages on opposing-goals instances.

Model	Altern. Rate
GPT-5 (On)	**1.000**
GPT-5 Mini (On)	**1.000**
GPT-5 (Off)	0.997
GPT-5 Mini (Off)	0.989
Claude 4 (On)	0.987
Claude 4 (Off)	0.974
GPT-OSS (On)	0.961
Qwen3-80B (On)	0.883
DS-v3.1 (Off)	0.820
LM-70B (On)	0.795
Nem-9B (Off)	0.678
LLaMA-70B (Off)	0.669
Nem-9B (On)	0.437
Qwen3-80B (Off)	0.190

Table 13: Average alternation rate per model on opposing goals instances.

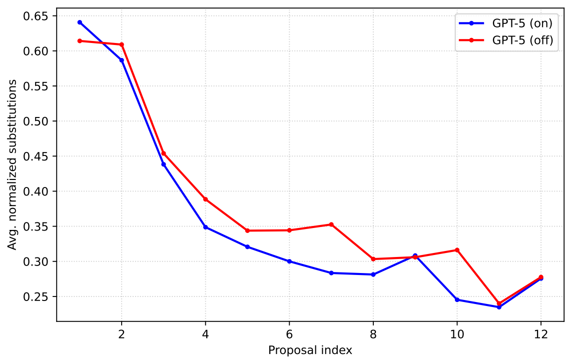


Figure 44: Substitutions over proposals in temporal order for GPT-5. Values are averaged over proposals at each time step in a negotiation across all experiments and languages.

$$c_t = \frac{|S_t \triangle S_{t-1}|}{|S_{t-1}|}, \quad t \geq 2,$$

Figures 44–47 depict average normalised substitution counts over each t up to 12 for some selected models. In Figures 44 and 45, we can again see that for our two best models (GPT-5 and GPT-5 Mini) we have higher initial disagreement with a

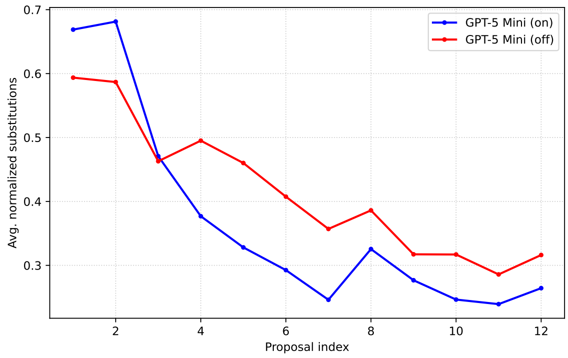


Figure 45: Average normalised substitutions over proposals in temporal order for GPT-5 Mini.

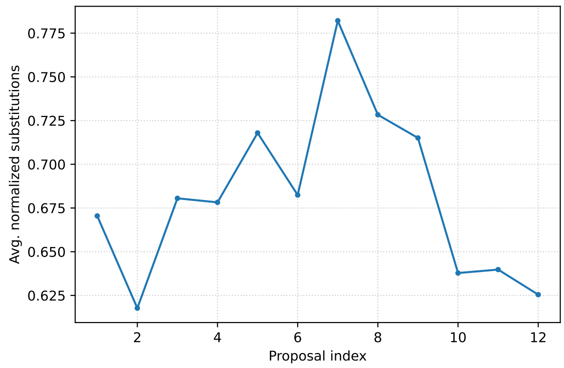


Figure 46: Average normalised substitutions over proposals in temporal order for GPT-OSS 120B.

more rapid and smooth descent when reasoning is enabled, indicating more efficient and coherent bargaining in commercial GPT models, which is amplified by enabling reasoning. DeepSeek Chat v3.1 shows the same pattern, but the initial disagreement is much lower (cf. Fig. 47). The GPT-OSS substitution pattern (cf. Fig. 46) shows high volatility and hence strong willingness to change the running proposal, with high overall difference, which may tie into the earlier discussed phenomenon of stubbornness.

Counterpart Modelling Figure 14 reports the per-instance Pareto-adherence rate, i.e. the per-instance relative frequency of proposals which lie on the Pareto front. Because Pareto-efficient proposals require aligning both agents' pay-offs – albeit not necessarily in a balanced way – higher adherence may indicate stronger counterpart and self-modelling, i.e. the ability to infer the other player's item valuations and adjust one's own search for possible deals accordingly. Additionally, we observe that bargaining along the Pareto front is a strong indicator of overall performance in our game

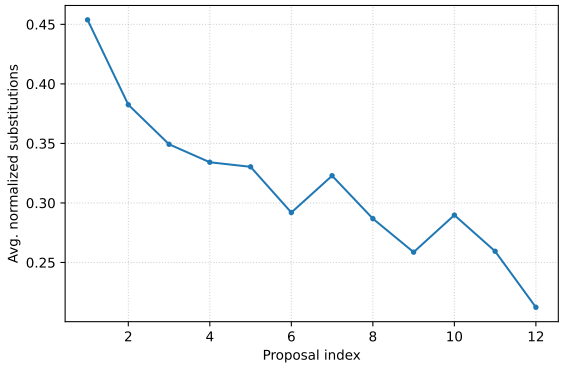


Figure 47: Average normalised substitutions over proposals in temporal order for DeepSeek Chat v3.1.

and almost perfectly separates the set of reasoning models from the set of non-reasoning models if we order them by Pareto-adherence.

Model	**Pareto-adherence**
GPT-5 (On)	0.791
GPT-5 Mini (On)	0.732
GPT-OSS (On)	0.669
Qwen3-80B (On)	0.605
Nem-9B (On)	0.376
LM-70B (On)	0.299
GPT-5 (Off)	0.280
GPT-5 Mini (Off)	0.268
Claude 4 (On)	0.171
DS-v3.1 (Off)	0.127
Nem-9B (Off)	0.065
Claude 4 (Off)	0.026
LLaMA-70B (Off)	0.000
Qwen3-80B (Off)	0.000

Table 14: Average per-instance Pareto-adherence rate for proposals per model. Values computed from all played instances with 15 items.