Relational Machines: Multi-Agent Coordination and Reasoning under Partial Information

Chelsea Zou

A THESIS

SUBMITTED TO THE DEPARTMENT OF

SYMBOLIC SYSTEMS

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

June 2026

To the Directors of the Program on Symbolic Systems: I certify that I have read the thesis of Chelsea Zou in its final form for submission and have found it to be satisfactory for the degree of Master of Science.

Signed Electronically

June 4, 2026
Noah Goodman, Principal Advisor
Department of Psychology & Computer Science

To the Directors of the Program on Symbolic Systems: I certify that I have read the thesis of Chelsea Zou in its final form for submission and have found it to be satisfactory for the degree of Master of Science.

Signed Electronically

June 4, 2026
Robert Hawkins
Department of Linguistics

Acknowledgements

I am deeply grateful for my experience at Stanford and the people I am surrounded by, who continually inspire me to stay curious, seek deeper understanding, and strive for excellence. None of this would have been possible without the unconditional support of my family, especially my mother.

I want to thank my super smart research collaborators and mentors, my advisor Noah Goodman for his valuable insight and guidance, Robert Hawkins for his advice and feedback, and the broader SymSys community for their unique modes of thinking: creative yet logically rigorous, playful yet technically grounded, and guided by a shared sense of awe toward life's strangest questions. I feel like I have finally found my academic tribe. I also want to thank my undergrad advisors Kenneth Kurtz, Shiqi Zhang, and Steven Lynn: I would not be here without the research I have done with them.

Finally, I appreciate all the new friends I've made here. Thank you for all our chaotic jokes, late-night intellectual rabbit holes, and spontaneous excursions. And to my poker degens: our games have somehow taught me as much about reasoning and decision making as any cognitive scientist could hope for.

I am incredibly excited for the future. To have gotten to where I am so far, I know I have been extremely **lucky**. Out of poker habit, though, I shall continue to misattribute it entirely to skill.

Abstract

Human intelligence is deeply interactive. Not only do we build models of the world, but we also build models of other people: what they know, what they want, and what they are likely to do next. Solving real-world problems involves not only individual reasoning but strategic interaction with others, making coordination and communication central to achieving collective goals. This research investigates language-model agents' capacity for strategic reasoning and coordination within multi-agent environments characterized by partial information. Two scenarios are explored: adversarial interactions through poker, and cooperative coordination via calendar scheduling. An agentic poker environment qualitatively assesses agents' strategic reasoning, opponent modeling, and adaptation capabilities. A calendar scheduling task quantitatively evaluates cooperative multi-agent coordination by examining task success, efficiency, fairness, and privacy preservation. Results reveal that agents possess sophisticated local inferential reasoning but struggle with sustained global adaptation. In poker, agents proficiently infer hidden states yet often fail to dynamically update opponent models. Similarly, in calendar scheduling, agents can schedule tasks but inadequately optimize collective outcomes or communicate precisely. These findings underscore the importance of multi-agent signals as essential training resources for developing adaptive and robust AI agents suitable for complex real-world interactions.

Contents

1 Introduction

Humans reason within inherently social and dynamic environments. We constantly interpret signals from others, adapt to changing contexts, and respond strategically to others' behaviors. Today, AI is entering similar settings, transitioning from isolated reasoning tasks to interactive multi-agent settings. Hence, evaluating and enhancing agents' abilities to reason, communicate, and coordinate with each other under partial information becomes crucial. The measure of model intelligence therefore expands beyond individual task performance, emphasizing how effectively agents reason and act among other agents.

The central motivation for my work is: how do language model agents reason and coordinate in strategic, partial information, multi-agent environments where success depends on other's private information, beliefs, and actions? I focus on two environments that make this question observable. Poker and calendar scheduling. Poker is a zero-sum game. The objective to maximize one's own gains, at the expense of another's loss. This environment stresses adversarial coordination and adaptation. Calendar scheduling, on the other hand, has one shared goal: to successfully schedule meetings that accommodates all participants' calendars. This stresses cooperative coordination and shared commitments.

While my research shows agents can perform sophisticated local reasoning, their capabilities become fragmented when interactions become more complex. In poker, agents can infer an opponent's likely range, reason about what their own actions represent, and choose moves designed to manipulate another agent's beliefs. However, they fail to update opponent models as the session unfolds. In calendar scheduling, agents can negotiate, exchange availability, reschedule commitments, and reach agreement under partial information. But they struggle to

optimize costs, distribute disruption fairly, and communicate at the right level of precision. To solve these challenges, agents need to adapt globally and reliably maintain, revise, and repair their narratives of other agents over time.

This thesis makes two main contributions toward understanding multi-agent reasoning in language models. 1) We design and build an agentic poker arena to *qualitatively* analyze model traces to study emergent strategic reasoning, opponent modeling, and adaptation failures in a dynamic adversarial environment. 2) We introduce CalBench, an agentic calendar scheduling benchmark that *quantitatively* evaluates distributed coordination under private information constraints using objective metrics for task success, coordination quality, fairness, communication efficiency, and privacy preservation. Finally, I briefly discuss the broader implications of these kind of environments to generate richer reward signals for multi-agent reinforcement learning.

Together, these contributions point to a broader view of language-model reasoning: agent intelligence measures should be extended beyond isolated problem solving. Real agents must act among other agents, infer hidden states, negotiate tradeoffs, preserve privacy, and revise their behavior as the environment changes. In my thesis, I aim to show that multi-agent reasoning is not just a performative extension of single-agent reasoning, but a core capability for any agentic system expected to operate in the real world. As AI shifts from research settings to production, the benchmark for intelligence transitions from what a model can solve alone to what they can accomplish together. The greatest human achievements have always depended on collective intelligence, and for AI to create at scale, it must learn the same primitives of coordination.

2 Background

2.1 Agentic and Multi-Agent LLM benchmarks

Recent benchmarks have begun to evaluate large-language model (LLM) agents in interactive settings rather than static question-answering tasks. AgentBench studies tool use and decision-making across web, game, and embodied environments [9]; TauBench and Tau2 evaluate tool-agent-user interaction in realistic service workflows [18, 1]; and NaturalPlan tests natural-language planning when the full task context is available to the model [19]. These benchmarks mark an important shift from answer evaluation to trajectory evaluation. However, they primarily study agents acting with access to the relevant task state, either through the prompt or through tools, rather than agents coordinating when information is distributed across multiple private actors.

Multi-agent benchmarks extend this direction by studying social intelligence, collaboration, competition, and communication structure [20, 12]. These settings expose new failure modes that are invisible in single-agent evaluation. Recent work shows that multi-agent systems can underperform single-agent baselines, suggesting that additional agents can introduce coordination overhead, noisy communication, and poorly calibrated delegation rather than automatic gains [8, 4, 7]. Partial-information environments such as CRAFT further show that communication ability does not necessarily translate into collective success [16]. An agent may produce coherent messages while still failing to reveal useful information, infer others' constraints, or adapt to group-level objectives.

Together, these results suggest that multi-agent evaluation should focus not only on whether agents interact, but on whether they coordinate. Real-world assistants will often represent different users with private

information, asymmetric preferences, and competing costs. Evaluating such systems requires metrics beyond task completion. This thesis builds on prior interactive and multi-agent benchmarks by treating coordination reasoning under partial information as the core object of study.

2.2 Poker, Imperfect-Information Games, and Strategic Reasoning

Poker requires decision-making under hidden information, stochastic outcomes, and strategic interaction. Unlike perfect-information games such as chess or Go, players must act under uncertainty and reason about how its own actions may reveal information that can change the opponent's beliefs. This makes poker a natural environment for studying theory-of-mind, deception, opponent modeling, and adaptation.

Classical intersections between poker and AI have largely approached this problem through computational game theory. Counterfactual regret minimization introduced a practical method for approximating equilibria in large extensive-form games with imperfect information [22]. Subsequent systems combined regret minimization, abstraction, and real-time search to achieve superhuman performance. DeepStack used recursive reasoning, depth-limited lookahead, and a learned value function to defeat professional players in heads-up no-limit Texas Hold'em [15]. Brown and Sandholm's Libratus further advanced this line by defeating top human professionals in heads-up no-limit Texas Hold'em, using a combination of blueprint strategy computation, subgame-solving during play, and self-improvement from observed weaknesses [2]. Their later system Pluribus extended superhuman poker AI to six-player no-limit Texas Hold'em, showing that strong play was possible even in multiplayer imperfect-information settings where equilibrium compu-

tation is substantially harder [3].

These systems demonstrate that poker can be solved to a high level by specialized algorithms with explicit game-theoretic machinery. My work asks a different question: whether general-purpose language model agents exhibit strategic reasoning in interactive poker settings. Rather than evaluating chip count or equilibrium approximation, I analyze model traces to study whether agents infer opponent ranges, reason about what their actions represent, bluff strategically, and update opponent models over time. This shifts the focus from solving poker as an optimization problem to using poker as an evaluation mechanism for multi-agent reasoning.

Recent work has begun to evaluate LLMs in game-theoretic and poker-like settings. GTBench evaluates strategic reasoning across a suite of games, including incomplete-information settings [5]. PokerBench specifically proposes poker as a benchmark for LLM decision-making, emphasizing that poker requires mathematics, planning, strategy, and psychological reasoning [21]. These benchmarks motivate poker as a test of strategic competence, but my work focuses on the qualitative structure of agentic reasoning traces: how agents represent opponents, how those representations guide actions, and where they fail to adapt over repeated interaction.

2.3 Calendar Scheduling and Distributed Constraint Optimization

Calendar scheduling is essentially a distributed constraint optimization problem (DCOP), which formalize multi-agent problems in which agents control local variables and coordinate to minimize a global cost subject to distributed constraints [13, 6]. Meeting scheduling has been an important application area for this framework. Maheswaran et al.

introduced DiMES, a distributed multi-event scheduling framework for real-world domains involving joint activities [11]. Modi and Veloso studied multi-agent meeting scheduling with rescheduling, emphasizing the difficulty of deciding when an existing commitment should be displaced to accommodate a new meeting [14].

The core structure of the problem is that calendar scheduling is not merely a planning task, but a distributed coordination problem under private information. CalBench builds on this DCOP tradition but changes the object of study. Instead of designing a specialized scheduling protocol, it evaluates whether general-purpose language model agents can coordinate through natural language under the same private-information pressures. The benchmark uses oracle solutions to measure avoidable displacement cost, baseline protocols to anchor the evaluations, and communication traces to measure whether agents share the right information. This lets us ask not only whether agents schedule meetings, but whether they do so efficiently, fairly, and privately.

3 Adversarial Multi-Agent Reasoning in Poker

> *"Poker is not a card game played by people. It's a people game played with cards"*

[1]Effective reasoning in multi-agent environments requires agents to condition onto the joint history of all other actions. We use poker as an empirical domain precisely because it makes these dependencies explicit and measurable. The game of poker involves complexities beyond computation and search. The optimal action is more than just dependent on the state of the game, it is dependent on the state of other players. It requires observing others, catching their mistakes, adapting to behavioral patterns, and exploiting recurring tendencies.

Modern poker solvers can approximate game-theoretic optimal strategies, however, the strongest plays are often exploitative where players should make dynamic adjustments (i.e., bluffing more against players who overfold, value-betting more against players who call too often, etc.). The environment is constantly changing, and good strategies require noticing subtle opponent tendencies and dynamically updating beliefs about them over time. The agent must therefore reason not only about the cards and probabilities, but also whether they can treat other agents as adaptive strategic actors.

3.1 Environment Design

We design a no-limit Texas hold'em environment and run 9 agents over 100 hands across 4 different models: GPT-5, Claude Sonnet 4.6, Gemini 3 Pro, and Gemini 3 Flash. At each decision point, the environment constructs an agent-specific observation containing the player's starting cards, stack size, table position, current street, pot size, visible board cards, and the public action history. Private cards belonging to other agents are withheld, while all public actions are serialized into the prompt so that agents must infer from incomplete information. Each model call is constrained to emit a structured poker action, with logged thinking traces of the agent at every event.

These traces are the main object of analysis. Objective metrics like final chip count is a weak signal because outcomes are subject to high variance in short poker runs. We therefore analyze how the agent represented the strategic situation, what inference it drew from the op-

[1]Conducted in collaboration with Vaibhav Tulsyan. I led project conception, design, implementation, experimentation, and analysis. Vaibhav contributed to development, evaluation, and review feedback. Study can be found at https://moltecarlo.com/

ponent's actions, what action it selected, and whether later evidence changed its internal reasoning.

Our environment setup further constructs agents with varying memory configurations. Some agents play without persistent memory and must rely only on the current context. Other agents are given persistent memory skills where they can optionally read/write opponent tendencies and strategic rules across hands. These memory-enabled agents maintain a compact strategy file across decisions and games. Before each action, the prompt instructs the agent to read its current skill file, apply a relevant rule or opponent read, optionally run poker calculations such as pot odds, choose an action, and then rewrite the skill file with updated strategic notes. This creates a lightweight longitudinal memory mechanism where agents can accumulate opponent-specific reads and adapt strategy across hands while still acting under the same private-information constraints. It allows us to separate two questions: whether a model can reason locally about a single strategic situation, and whether it can use past experience of other agents to change future play.

We focus on three capabilities. 1) We examine local strategic reasoning: whether the agent can condition its action on current information, public actions, and board texture. 2) We examine theory-of-mind reasoning: whether the agent reasons about the opponent's beliefs and about how its own actions affect those beliefs. 3) We examine adaptation: whether the agent can update its opponent model over time rather than repeatedly applying a fixed read.

3.2 Local Strategic Reasoning in Multi-agent Settings

An agent operating in a partially observable multi-agent setting must be capable of interpreting another agent's actions as evidence about hidden state. The traces from our study show that agents can engage in this form of inferential reasoning: they identify opportunities to act on inferred opponent weakness, suppress dominant responses in order to elicit information-revealing actions, and recognize when an opponent's behavioral sequence is more consistent with deception than with genuine state.

This capacity constitutes what we term local strategic competence: the ability to reason correctly about a specific interaction given the current information state. However, local competence is insufficient for robust multi-agent performance. A policy can produce individually justified decisions while remaining globally exploitable. The remainder of this section characterizes the systematic failure modes that emerge at this level.

3.3 Recursive Belief Modeling

A core challenge in adversarial multi-agent environments is that actions function simultaneously as decisions and as signals. An action changes the world state and updates other agents' beliefs about the acting agent's policy. This induces a recursive belief structure: each agent must model the other's private state, the other's model of its own policy, and the effect of each action on that mutual model.

The traces indicate clear instances of this reasoning, including cases where it succeeds and cases where its structural limits are exposed. In one poker hand, a Gemini Flash agent holds a very strong hidden hand

but faces an opponent who entered the round aggressively, signaling confidence. Rather than responding immediately, it deliberately plays passively not from weakness, but to keep the opponent committed to the narrative it has already built about its own advantage. This is k-level reasoning at the second order. The Gemini Flash agent is not just modeling what the opponent has, it is modeling what the opponent believes the other party has, then selecting an action specifically to exploit that belief.

This recursive structure can be made explicit in levels. At K=0, an agent evaluates only its own position. At K=1, it models what the opponent holds based on observable behavior. At K=2, it models what the opponent thinks they hold and exploits that belief. At K=3, it would model how the opponent models its own reasoning process, and adjust to preclude exploitation of its own behavioral regularities. The agents in this study exhibit reliable K=2 reasoning but show minimal evidence of stable K=3 reasoning.

A second case illustrates the structural vulnerability of stopping at K=2. One Claude Sonnet 4.6 agent holds a weak hand but attempts to deceive by behaving in a way that mimics a strong one, by remaining passive early in the round, then making an assertive move later, a sequence that typically signals genuine strength. Its opponent reasons carefully: the early passivity followed by a late aggressive move fits the pattern of a player who improved to a good hand mid-round. The opponent concludes it is beaten and gives up the best hand. The deception succeeded precisely because agents can construct simple narratives of other agent's beliefs.

This is the structural vulnerability of any fixed reasoning depth. An opponent's capacity for belief inference creates exploitable structure. An agent that predictably maps certain behavioral patterns to certain conclusions can be manipulated by an adversary that knows it will make

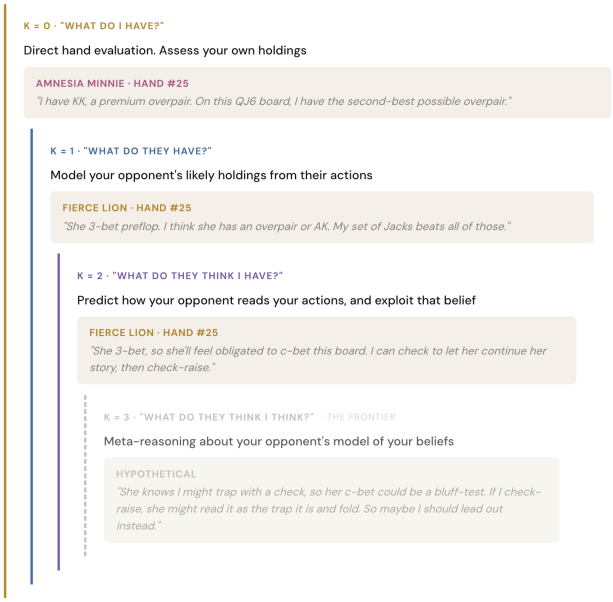

Figure 1: K-level Reasoning

those inferences. At level K+1, an opponent can construct signals that reliably produce the desired conclusion in a K-level reasoner. Therefore, it is advantageous for agents to be capable of higher-level cognitive modeling in adversarial situations.

3.4 Opponent Modeling as Belief State Estimation

The central failure mode identified in the study is the inability to maintain opponent models as dynamic belief states. Agents frequently construct characterizations early on in an interaction, and then later treat them as fixed labels, continuing to condition on them even as observed behaviors have changed. Across different contexts, agents cite the identical profile entry verbatim, even though the situation has materially changed. This is a failure of nonstationary reasoning. In single-agent settings, the environment is stationary: the distribution over observa-

tions does not depend on the agent's previous outputs. In multi-agent settings this assumption fails.

The study's traces also document the opposite failure mode in a different agent, GPT-5, which represents having no opponent model at all. Every decision is generated by applying a generic situational rule, a GTO play that maps the structural features of the current moment to a textbook action, with no reference to the specific opponents present or their observed tendencies. Despite memory being enabled, the GPT agent never once references what a particular opponent has been doing.

3.5 Memory as Policy Modification

Among the memory-enabled agents, some engaged with their memory system extensively, while others made zero reads, zero writes, and zero rule applications despite having memory access. We found that this was model dependent: Gemini models tend to use more of its tools, whereas Claude ignored them and relied on decisions in-context. Between those that did use their memory features, the availability of persistent memory is still not sufficient to produce adaptive behavior. What matters is whether the stored observations are converted into policy updates of the agent's action distribution. However, while some agents stored accurate observations, the notes never propagated into a varied action selection. For instance, one agent recorded that a given opponent tends to show fear when large bets were placed on early streets, yet they never tried to exploit that observation throughout the hands, even in spots where doing so was clearly warranted.

Another example: midway through the session, some of the agents were eliminated, reducing the field to fewer players. In poker, this has well-understood strategic implications. Hands that were marginal under a larger player size (where the likelihood of running into a stronger

holding was high enough to justify caution) become more profitable to play assertively in a smaller player pool. Optimal play requires recalibrating the action threshold accordingly: playing more hands, applying more pressure, and extracting value from a larger fraction of situations. None of the agents made this adjustment. Several flagged the change explicitly in their reasoning traces but maintained essentially identical behavior throughout, showing dissociation between stated situational awareness and policy output. Memory that accumulates accurate observations without affecting agent behavior is functionally useless.

3.6 Summary

The poker study shows that language-model agents can exhibit meaningful local strategic reasoning in competitive imperfect-information games. They can infer ranges, identify bluffing opportunities, trap, and reason about how actions affect opponent beliefs. However, these abilities are not sufficient for robust strategic behavior. The agents often treat opponent models as static, use memory as storage rather than policy update, and fail to reason about how their own reasoning policy becomes exploitable. The next chapter studies the cooperative analogue of this problem. Calendar scheduling removes adversarial incentives but still holds conditions to preserve private information, communication, and interdependence.

4 Cooperative Multi-Agent Coordination in Calendar Scheduling

[2]While poker tests whether agents can reason strategically under hidden information and adversarial incentives, calendar scheduling tests

whether agents can cooperate when the objective is shared. Here, no agent wins at another's expense. Instead, all agents must reach a consistent joint decision, placing each incoming meeting at the same time slot across every participant's calendar, despite holding private information about their own constraints, costs, and commitments. Success requires language-mediated coordination: agents must communicate selectively, revealing enough to reach agreement without leaking more than necessary.

This section presents CalBench, a controlled benchmark for evaluating multi-agent calendar scheduling under private information. We describe the environment design (§4.1), the formal evaluation metrics (§4.2), the baseline non-LLM reference protocols that anchor performance expectations (§4.3), experimental setup (§4.5), and empirical results across seven models (§4.6).

4.1 The CalBench Environment

Task Structure. Each task consists of N agents, T discrete time slots, and a stream of M incoming meetings. Agent i maintains a private calendar whose slots may be one of four types: free, occupied by a movable errand, occupied by a previously scheduled meeting, or blocked by an immovable commitment. Occupied entries carry a displacement cost and a private semantic context visible only to the owning agent. Each incoming meeting k has a participant set $P_k \subseteq \{1, \ldots, N\}$ and must be written to the *same* slot on every participant's calendar simul-

[2]Work done in collaboration with Yiheng Yao, Selena She, Noah Goodman, and Robert Hawkins. Yiheng and I equally co-led the study, environment design, implementation, evaluation, and writing. Selena contributed to analysis. Noah and Robert provided valuable insight and guidance. For full paper, see https://arxiv.org/abs/2605.09823

taneously. Because no agent can inspect another agent's calendar, coordination must proceed through language, via messages exchanged over configurable communication channels, rather than through any centralized planning mechanism. The task is structurally decentralized. A single agent with global visibility could solve it directly, but the multi-agent setup is the mechanism by which agents must selectively share information to coordinate while withholding irrelevant private details. This distinguishes CalBench from benchmarks where the multi-agent framing is incidental where a centralized agent with full problem state could solve the task equally well.

Calendar Generation. Calendars are generated using a design that guarantees feasibility. The generator first samples a hidden feasible witness assignment, a set of slot assignments for all incoming meetings that satisfies all calendar constraints. It then fills each agent's calendar with errands drawn randomly until the target density ρ_i is reached, while deliberately preserving absorbing slots so that displaced errands always have valid landing pads. This construction ensures that every generated task has at least one known feasible solution, making the oracle cost computable and the evaluation of excess cost meaningful. Calendar density, the fraction of slots pre-filled with errands, controls task difficulty. Some calendar entries are marked *blocked*, meaning they cannot be moved; agents must learn to distinguish flexible events from hard constraints through coordination rather than inspection. CalBench supports both shared and per-agent density settings, allowing tasks in which some agents represent users with substantially denser or sparser schedules than others. In the experiments reported here, starting configurations are balanced across models for fair evaluation.

Cost Structure. Two cost settings are studied. In the uniform-cost setting, every errand carries displacement cost 1, and displacing a previ-

ously scheduled meeting also costs 1 per participant calendar on which it must be moved. In the varied-cost setting, errands are assigned low, medium, or high disruption costs from a balanced set. Internally, these costs are evaluated as $\{1, 2, 3\}$, but agents observe them on a logarithmic display scale of $\{1, 10, 100\}$. The logarithmic presentation was adopted after pilot runs revealed that small linear differences were not behaviorally salient to language model agents; the displayed scale makes the intended preference ordering clearer, while evaluation uses the underlying linear costs. The varied-cost setting introduces a qualitatively harder coordination problem. Optimal scheduling sometimes requires moving a previously scheduled multi-agent meeting rather than displacing a local errand, which in turn requires agents to initiate a repair episode with external participants from prior meetings. This tests whether agents can recognize when cross-meeting coordination is cost-justified and execute it without leaking unnecessary private information in the process.

Private Semantic Contexts. Each errand and prior meeting is linked to a natural-language private context before it is shown to an agent. These labels are drawn from a label bank with three sensitivity tiers: *public* (routine low-stakes errands), *sensitive* (ambiguously private everyday logistics), and *very sensitive* (medical, legal, financial, or identity-related). Agents see their own private labels on every turn, but do not see the tier labels themselves. Non-participant agents see only the slot type and cost, not the semantic label. Critically, semantic context is never necessary to solve the scheduling problem. An agent may need to communicate its availability, whether a slot is occupied, but never needs to reveal *why*. This separation allows the benchmark to distinguish two forms of privacy loss: our main metric of structural leakage about calendar states (measured through the VPS framework, described in §4.4) and semantic leakage through unnecessary disclosure

of event descriptions.

Communication Protocols and Round Structure. Each game proceeds in rounds, one per incoming meeting. Every round passes through four phases:

Cheap-talk phase. Agents coordinate through configurable message channels. CalBench supports three channel types: private direct messages (DM) between any two agents, a participant groupchat visible only to the current meeting's participants, and an all-agent groupchat visible to every agent. Participants are active by default; non-participants become active only after receiving a direct message or observing an all-agent broadcast. This activation rule allows agents outside the current meeting to help resolve conflicts without granting them unsolicited access to coordination.

Voluntary phase. After cheap-talk closes, activated non-participants may submit rescheduling actions, for example moving their copy of a previously scheduled meeting that is blocking the current participants. This phase allows repair coordination across meeting boundaries.

Decision phase. Each current participant independently submits an atomic batch of actions. A batch may contain any number of RESCHEDULE actions followed by exactly one SCHEDULE action that places the incoming meeting. Every reschedule action must include a natural-language justification. The entire batch is validated against the agent's calendar and applied only if it passes all valid response rules.

Resolution phase. The environment checks that all meeting participants scheduled the incoming meeting to the same slot, and that any prior multi-agent meetings remain consistent across all participants' calendars. If these conditions are met, the meeting is

successfully scheduled; otherwise it is marked unresolved and the game proceeds to the next round.

Task Difficulty. Task difficulty is quantified using a CP-SAT feasibility score [17]. For each generated task, CP-SAT enumerates the number of distinct meeting-to-slot assignments F that satisfy all calendar constraints, and normalizes by the total number of possible injective assignments:

$$d = \frac{F}{\frac{T!}{(T-M)!}} \tag{1}$$

where T is the total number of calendar slots and M is the number of meetings. Lower d scores correspond to harder tasks because fewer joint schedules are feasible. Tasks are bucketed into easy, medium, and hard difficulty ranges based on this score, and the canonical evaluation suite balances tasks across these buckets.

4.2 Evaluation Metrics

CalBench reports five primary evaluation metrics for each model. Lower is better for all metrics except task success.

Task Success. Task success measures the fraction of meetings that are successfully scheduled. Let $\mathcal{G}_i$ be the set of games in which agent i appears, $M_{g,i}$ the number of meetings involving agent i in game g, and $M_{g,i}^{\text{sched}}$ the number of those meetings successfully scheduled. Task success is:

$$S_i = \frac{1}{|\mathcal{G}_i|} \sum_{g \in \mathcal{G}_i} \frac{M_{g,i}^{\text{sched}}}{M_{g,i}} \tag{2}$$

A meeting is successful only if all required participants write it to the same slot *and* all previously scheduled meetings remain calendar-

consistent. This is the basic feasibility metric: an agent team that cannot consistently place meetings has failed the core task regardless of cost or privacy behavior.

Coordination Cost. Coordination cost measures how much avoidable displacement cost the agents incur relative to an oracle scheduler. The oracle is computed via CP-SAT: given full information about all agents' calendars, CP-SAT finds the minimum-cost feasible assignment for the successfully scheduled meetings. Let $c_{g,i}^{\text{real}}$ be the realized displacement cost incurred by agent i in game g, and $c_{g,i}^{\text{oracle}}$ the CP-SAT oracle cost for the same meetings:

$$C_i \;=\; \frac{1}{|\mathcal{G}_i|} \sum_{g \in \mathcal{G}_i} \max\bigl(0, c_{g,i}^{\text{real}} - c_{g,i}^{\text{oracle}}\bigr) \tag{3}$$

Communication Efficiency. Communication efficiency measures how many messages are exchanged per successfully scheduled meeting:

$$E_i \;=\; \frac{1}{|\mathcal{G}_i|} \sum_{g \in \mathcal{G}_i} \frac{N_{g,i}^{\text{msg}}}{\max\bigl(M_{g,i}^{\text{sched}},\, 1\bigr)}, \tag{4}$$

where $N_{g,i}^{\text{msg}}$ counts all direct messages, participant groupchat messages, and all-agent broadcasts sent by agent i. Lower values indicate that agents reached outcomes with less communication overhead and fewer opportunities for privacy leakage. This metric is normalized to successful meetings only, so it does not penalize agents whose low communication correlates with coordination failure.

Fairness. Fairness measures whether the excess displacement burden is distributed evenly across agents within a game. Each agent's excess burden within a game is:

$$e_{g,i} \;=\; c_{g,i}^{\text{real}} - c_{g,i}^{\text{oracle}}. \tag{5}$$

The mean excess burden within the game is $\bar{e}_g = \frac{1}{N_g} \sum_j e_{g,j}$, and each agent's relative excess burden is $r_{g,i} = e_{g,i} - \bar{e}_g$. The fairness cost for agent i is:

$$F_i \;=\; \frac{1}{|\mathcal{G}_i|} \sum_{g \in \mathcal{G}_i} |r_{g,i}|. \tag{6}$$

Lower F_i indicates that agent i's excess burden typically stays close to the within-game average: the agent is neither systematically exploited nor systematically freed from disruption. Higher F_i indicates that the agent often absorbs substantially more or less than its fair share.

Privacy Cost. Privacy cost measures how much information agents leak about their private calendar states. The formal measurement framework is described in §4.4. The scalar reported per agent is the mean per-game calibrated VPS loss V_i, where lower values indicate better privacy preservation.

4.3 Non-LLM Baseline Protocols

To anchor the evaluation, CalBench includes four non-LLM scheduling protocols as reference points. These are simple, rule-based algorithms that reveal how well a scheduling task can be solved when agents follow a fixed information-sharing strategy. Together they define a frontier along the excess and privacy cost of scheduling meetings.

IMAP: Full Disclosure. One agent acts as the initiator and asks all other participants to report the cost of scheduling the meeting in each available slot. Cost is zero if the slot is free, a small displacement cost if an existing commitment could be moved, or infeasible if the slot cannot be cleared. The initiator picks the cheapest feasible slot across all participants and confirms it. This gives the best possible outcome

Protocol	Role
IMAP	Full disclosure baseline. Agents share all their costs openly, so the best possible slot is always chosen. Upper bound on scheduling efficiency.
SD-MAP	Minimal disclosure baseline. Agents only say yes or no to proposed slots, sharing no cost information. Shows what efficiency is lost by hiding costs.
DSM-Welfare	Partial disclosure baseline tuned toward coordination. Agents share coarse satisfaction scores over many candidate slots.
DSM-Private	Same mechanism as DSM-Welfare, but tuned toward privacy. Agents share scores over fewer slots, reducing information leakage at the cost of coordination.

Table 1: Four reference protocols spanning the disclosure-efficiency tradeoff, against which LLM agent behavior is benchmarked.

for the current meeting, but at the cost of full transparency: every participant's complete cost structure is exposed on every scheduling request.

SD-MAP: No Disclosure. The initiator proposes slots one at a time in order. Participants reply with a simple yes or no: they never share costs. The first slot everyone accepts is confirmed. If a participant's existing commitment could be moved to accommodate the new meeting, they say yes, and the rescheduling is handled later. This reveals almost nothing about participants' private calendars, but ignores costs entirely: a slot that inconveniences everyone equally looks the same as one that is free for all.

DSM: Partial Disclosure. Rather than sharing raw costs, participants report coarse satisfaction scores (0 = impossible, up to 11 = perfectly free) over a set of candidate slots. The initiator picks the slot with the best joint score. Two variants are studied. **DSM-Welfare** offers many candidate slots at once and weights coordination, trading more disclosure for better outcomes. **DSM-Private** offers fewer slots and penalizes disclosure heavily, trading coordination for privacy. Together they bracket the tradeoff between information shared and scheduling quality.

4.4 Privacy Measurement: Valuations of Possible States

CalBench measures structural privacy using Valuations of Possible States (VPS), a framework from the distributed constraint optimization literature [10]. VPS quantifies how much an observer's belief about a target agent's calendar state moves away from a uniform prior as a result of communication. For each round r, target agent i, and observer agent j, the prior is $p_0 = 0.5$ (maximal uncertainty), and VPS is:

$$\mathrm{VPS}^{r}_{j\to i} \;=\; \sum_{k\,\in\,\mathcal{S}} \left|\mathrm{Bel}^{r,\mathrm{post}}_{j\to i}[k] - p_0\right|, \tag{7}$$

where $\mathrm{Bel}^{r,\mathrm{post}}_{j\to i}[k]$ is the observer's posterior belief that slot k of agent i is occupied. VPS is reported in slot-equivalent units: a maximal belief movement on one slot contributes one slot-equivalent of leakage. This unit is comparable across uniform- and varied-cost calendars because slot occupancy has the same privacy cost regardless of the errand's displacement cost.

The belief update rule is a strength-weighted nudge:

$$\mathrm{Bel}'[k] \;=\; (1-\alpha)\,\mathrm{Bel}[k] \;+\; \alpha\, e, \tag{8}$$

where $e \in [0, 1]$ is the evidence value (1.0 = slot is feasible/occupied; 0.0 = slot is infeasible/free) and $\alpha \in [0, 1]$ is the inference strength. Updates are accumulated over a round, and end-of-round leakage is computed from Equation (7).

VPS for Baselines. For non-LLM reference protocols, messages carry known semantics, so beliefs can be updated deterministically from message content. IMAP `COSTS` replies reveal exact per-slot feasibility: any slot with a finite reported cost is updated with $e = 1.0$, $\alpha = 1.0$; a slot with infeasible status receives $e = 0.0$, $\alpha = 1.0$. SD-MAP `PROPOSE` messages receive a softer update ($\alpha = 0.70$) because proposing a slot is strong but not logically equivalent to a hard feasibility report; `REPLY` messages receive the full-strength update. DSM `PROPOSALS` are treated as full evidence of initiator feasibility ($e = 1.0$, $\alpha = 1.0$), while `SCORES` are mapped to fractional evidence values $s/(D - 1)$.

Reflection-Calibrated VPS for LLM Agents. LLM agents communicate in unstructured natural language, so rule-based belief updating from message content is not reliable. Instead, CALBENCH uses a *reflection-calibrated* VPS measurement. After each round, a measurement-only prompt, not inserted into the game context, asks each agent how its beliefs about every other agent's calendar changed. The agent reports signed integer deltas in $\{-3, -2, -1, 0, 1, 2, 3\}$ per slot, where positive values indicate increased belief in occupation. The VPS estimate counts only belief movements in the truth-directed direction (toward the target's actual calendar state), treating opposite-direction movements as zero calibrated leakage. This makes the LLM estimate conservative: it captures only leakage that the acting model itself reports as moving beliefs toward the truth. A post-hoc external LLM-as-judge audit over delivered messages finds systematically higher leakage than self-reflection, with 4,639 of 7,870 nonzero judge belief rows moving in the

truth-directed direction (precision 0.589) versus 0.528 for self-reflection. This supports interpreting reflection-calibrated VPS as a *lower bound* on observable privacy leakage, while the external audit provides a robustness check that the relative ordering of models is stable.

4.5 Experimental Setup

The canonical evaluation suite consists of 90 tasks with 5 sequential meetings, 16 calendar slots, N=5 agents, and 3 rotating participants per meeting. The suite contains 45 uniform-cost and 45 varied-cost tasks, with per-agent calendar density sampled from $\{0.6, 0.8, 1.0\}$ and blocked-errand counts from $\{2, 4, 6\}$. Tasks are balanced across easy, medium, and hard difficulty buckets as defined by the CP-SAT feasibility score (Equation 1).

Seven models are evaluated: Claude Sonnet 4.6, Gemini 3.1 Pro, Gemini 3 Flash, GPT-5.4 Mini, Llama 4 Maverick, Qwen 3.6 Plus, and DeepSeek V4 Pro. Four models (Claude Sonnet 4.6, Gemini 3.1 Pro, Gemini 3 Flash, and GPT-5.4 Mini) are evaluated across three calibration slices (N=135 agent-game seats per cost setting), while Llama 4 Maverick, Qwen 3.6 Plus, and DeepSeek V4 Pro are evaluated on single entrant slices (N=45 seats per setting). All runs use mixed-model teams, private agent-to-agent DMs as the communication channel, and the same prompt scaffold.

4.6 Results

Main model results are summarized in Table 2, and baseline results are summarized in Table 3.

Setting	Model	N	Meetings	Coord.	Excess	Messages/mtg	Fairness	Excess cal. VPS
Uniform	Claude Sonnet 4.6	135	2.54	84.69%	0.26	3.46	0.585	13.68
	Gemini 3.1 Pro	135	2.58	85.93%	0.44	3.07	0.616	8.87
	Gemini 3 Flash	135	2.59	86.17%	0.25	3.36	0.474	8.90
	GPT-5.4 Mini	135	2.50	83.21%	0.40	3.08	0.541	5.38
	Llama 4 Maverick	45	2.31	77.04%	0.44	3.11	0.529	**5.30**
	Qwen 3.6 Plus	45	**2.80**	**93.33%**	0.40	**2.96**	0.489	10.49
	DeepSeek V4 Pro	45	2.69	89.63%	**0.18**	3.21	**0.396**	19.18
Varied	Claude Sonnet 4.6	135	2.53	84.20%	0.43	3.50	0.913	12.85
	Gemini 3.1 Pro	135	2.45	81.73%	0.36	3.40	0.748	8.10
	Gemini 3 Flash	135	2.36	78.52%	0.59	3.56	0.721	7.21
	GPT-5.4 Mini	135	2.30	76.79%	1.49	3.44	1.376	4.60
	Llama 4 Maverick	45	2.31	77.04%	**0.24**	**3.20**	**0.627**	**3.49**
	Qwen 3.6 Plus	45	2.38	79.26%	0.53	3.76	0.662	8.56
	DeepSeek V4 Pro	45	**2.53**	**84.44%**	0.38	3.24	0.964	15.27

Table 2: Results on the canonical mixed-agent CalBench task set, split by cost setting. N is the number of evaluated agent-game seats. *Meetings* is the mean number of successfully scheduled meetings per agent-game (max 3). *Messages/mtg* counts direct, groupchat, and broadcast messages per successful participant-meeting. *Excess* is positive scheduled-only excess displacement cost, $\max(0, c_{g,i}^{\text{real}} - c_{g,i}^{\text{oracle}})$, using the 1/2/3 varied-cost scale. *Fairness* is the absolute deviation of each agent's signed excess burden from the within-game mean. *Excess cal. VPS* is calibrated slot-equivalent VPS. Lower is better for excess, messages, fairness, and VPS.

Task Success and Coordination Cost. In the uniform-cost setting, all tested models achieve high task success, ranging from 77.04% (Llama 4 Maverick) to 93.33% (Qwen 3.6 Plus). Excess costs remain low, ranging from 0.18 (DeepSeek V4 Pro) to 0.44 (Gemini 3.1 Pro and Llama 4 Maverick). This suggests that when all errands carry the same cost, the coordination problem reduces largely to finding a feasible consistent slot, which most models handle reasonably well.

The varied-cost setting is harder. Task success falls for most models, and excess cost separates models more clearly. DeepSeek V4 Pro achieves the highest task success (84.44%) with low excess cost (0.38).

Setting	Baseline	N	Meetings	Coord.	Excess	Messages/mtg	Fairness	VPS
Uniform	IMAP	45	3.00	100.00%	0.26	2.00	0.530	12.40
	DSM-private	45	1.37	45.78%	0.98	4.54	0.809	0.00
	DSM-welfare	45	3.00	100.00%	0.27	2.69	0.530	25.05
	SD-MAP	45	1.87	62.22%	1.68	7.48	0.741	0.12
Varied	IMAP	45	3.00	100.00%	0.32	2.00	0.665	12.40
	DSM-private	45	1.47	48.89%	2.08	4.43	1.865	0.00
	DSM-welfare	45	2.97	99.11%	0.46	2.78	0.816	23.55
	SD-MAP	45	1.89	63.11%	3.64	7.30	1.376	0.08

Table 3: Deterministic baseline results on the canonical mixed-agent CalBench task set. Note that *VPS* is computed from the VPS for baselines, and is not directly comparable to the reflection-calibrated VPS reported for language-model agents.

Llama 4 Maverick achieves the lowest excess cost (0.24), despite lower task success (77.04%). Claude Sonnet 4.6 maintains high task success (84.20%) with 0.43 excess cost. In contrast, GPT-5.4 Mini achieves only 76.79% task success and incurs the highest excess cost (1.49), about 6.2 times higher than Llama 4 Maverick and 3.9 times higher than DeepSeek V4 Pro. This divergence shows that coordination quality is not a single quantity: a model can complete many meetings while still making costly scheduling decisions.

Completion alone therefore does not characterize schedule quality. In varied-cost games, GPT-5.4 Mini and Llama 4 Maverick have nearly identical task success (76.79% vs. 77.04%), but GPT-5.4 Mini incurs much higher excess cost (1.49 vs. 0.24) and fairness cost (1.376 vs. 0.627). In uniform-cost games, Qwen 3.6 Plus has the highest task success (93.33%) but not the lowest excess cost; DeepSeek V4 Pro schedules fewer meetings (89.63%) while achieving the lowest excess cost (0.18). CalBench's cost-sensitive metrics therefore capture failures that task success alone would miss.

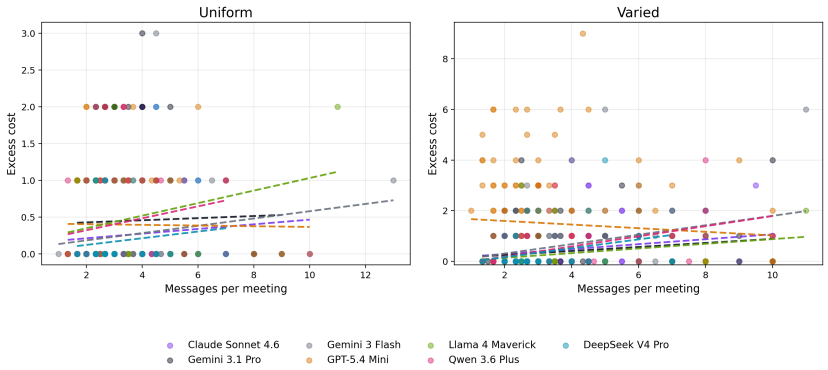

Figure 2: Communication efficiency. The x-axis reports messages per successful participant-meeting, and the y-axis reports positive scheduled-only excess displacement cost. Dashed lines show per-model OLS fits

Message Volume Is Not Coordination Quality. First, we find that agents overwhelmingly use the meeting participant groupchat modality over private DMs and public all-member groupchat. Across all seven models, 98% of all messages were sent through this medium. Second, a natural hypothesis is that more communication should improve coordination: more messages should allow agents to share more constraints and reduce excess cost. The data do not support this. Across both cost settings, raw message count per scheduled meeting is a poor predictor of excess cost.

In the uniform-cost setting, Qwen 3.6 Plus sends the fewest messages (2.96 per successful participant-meeting) but has relatively high excess cost (0.40). DeepSeek V4 Pro sends more messages (3.21) yet achieves the lowest excess cost (0.18). Claude Sonnet 4.6 sends the most messages (3.46) without achieving the lowest excess cost. In the varied-cost setting, Llama 4 Maverick is both the least verbose and lowest-cost model (3.20 messages; 0.24 excess cost), but this pattern does not hold generally. GPT-5.4 Mini sends 3.44 messages per successful participant-meeting and incurs the highest excess cost (1.49),

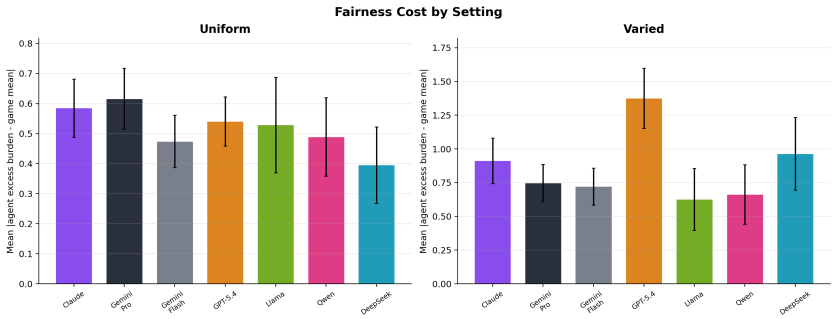

Figure 3: Fairness cost by model and cost setting. Lower values mean an agent's signed excess burden is closer to the within-game average. Error bars show 95% confidence intervals across task instances

while DeepSeek V4 Pro sends fewer messages (3.24) and incurs much lower excess cost (0.38). Qwen 3.6 Plus sends the most messages in varied-cost games (3.76) but has higher excess cost (0.53) than Gemini 3.1 Pro (0.36) and DeepSeek V4 Pro (0.38).

These patterns establish that *what* agents say, not *how much* they say, determines coordination quality. A model can be verbose without improving the schedule, or terse while still communicating the right feasibility and cost information.

Fairness and the Privacy-Fairness Tension. Fairness costs are small and tightly clustered in the uniform-cost setting, where all displacement costs are equal: model means range from 0.396 to 0.616. The varied-cost setting produces larger burden-allocation differences, with fairness costs ranging from 0.627 to 1.376. Llama 4 Maverick achieves the lowest mean fairness cost in the varied-cost setting (0.627), followed by Qwen 3.6 Plus (0.662) and Gemini 3 Flash (0.721). GPT-5.4 Mini has the highest fairness cost (1.376), indicating that its excess burden differs most from the within-game average. Descriptively, GPT-5.4 Mini's varied-cost fairness cost is 0.604 higher than the mean of the

other six models (0.773), and about 2.19 times higher than Llama 4 Maverick's fairness cost.

The mechanism behind this pattern suggests a model-specific privacy-fairness tension, rather than a universal tradeoff. In the varied-cost setting, GPT-5.4 Mini has the second-lowest VPS (4.60), suggesting relatively low calendar-state disclosure, but the highest fairness cost (1.376). By contrast, Llama 4 Maverick has both the lowest VPS (3.49) and the lowest fairness cost (0.627). Thus, low disclosure does not necessarily imply poor fairness. The issue appears to be what information is withheld: privacy-preserving behavior becomes harmful when agents suppress cost-relevant signals needed for equitable burden allocation.

A message-level audit shows that GPT-5.4 Mini proposes its own slots and reports its own availability at comparable rates to other models. The difference lies in cost and constraint language: GPT-5.4 Mini mentions its own slot costs or constraints in only 6.3% of outgoing varied-cost messages, compared to 29.2% for the other-model average ($p < 5 \times 10^{-6}$). This 4.6-fold reduction in qualitative cost signaling, words like *difficult*, *cheap*, or *I can move my errand*, may leave teammates with insufficient information to decide when GPT-5.4 Mini should absorb displacement burden. The agent's privacy preservation is genuine, but in this case it comes at the cost of coordination fairness: silence about costs is not merely private; it is also uninformative.

Privacy Leakage Across Models. VPS varies independently of coordination quality. In the uniform-cost setting, Llama 4 Maverick and GPT-5.4 Mini have the lowest VPS values (5.30 and 5.38, respectively), but they do not achieve the best coordination outcomes: Llama 4 Maverick has the lowest task success (77.04%) and tied-highest excess cost (0.44), while GPT-5.4 Mini has 83.21% task success and 0.40 excess cost. DeepSeek V4 Pro has the highest VPS in the uniform-cost set-

ting (19.18), but also the lowest excess cost (0.18) and lowest fairness cost (0.396).

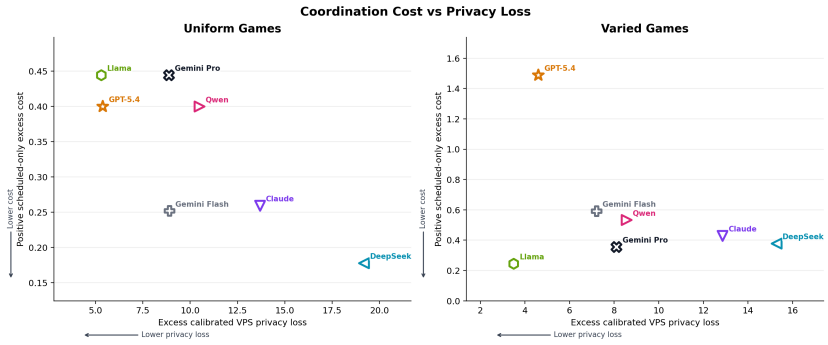

Figure 4: Privacy–cost plane. The closer to the origin the model is, the lower the excess cost and privacy cost it incurred.

The varied-cost setting shows the same separation. Llama 4 Maverick has the lowest VPS (3.49) and lowest excess cost (0.24), but lower task success (77.04%). GPT-5.4 Mini has the second-lowest VPS (4.60) but the highest excess cost (1.49) and fairness cost (1.376). DeepSeek V4 Pro has the highest VPS (15.27), but also the highest task success (84.44%) and low excess cost (0.38).

This independence of privacy from coordination performance is a core methodological contribution of CalBench. Models that leak more information are not automatically better coordinators, and models that leak less are not automatically worse.

Semantic Context Leakage. The benchmark also audits whether agents disclose the natural-language semantic contexts of their private events, for example mentioning that a blocked slot is a medical appointment. This is measured through keyword matching against the label bank and manual review of flagged messages. In our analysis, no model disclosed the natural-language semantic label of a private calendar item.

4.7 Summary

CalBench separates three behaviors that standard benchmarks conflate. First, completing the scheduling task is a weaker condition than coordinating well. Models can achieve full meeting completion while incurring substantial avoidable costs. Task success is necessary but not sufficient as an evaluation criterion. Second, communication volume does not reliably reduce coordination cost. Raw message count is a poor proxy for coordination quality; what matters is the precision and relevance of the information exchanged. Third, privacy can conflict with fairness when agents suppress cost-relevant signals that teammates need to make equitable allocation decisions. A model optimizing for low VPS may inadvertently produce the largest burden in fairness in the group.

These findings together suggest that effective cooperative coordination requires a form of *calibrated disclosure*: agents must share enough information to enable joint cost optimization and fair burden allocation, without leaking structural private information beyond what coordination requires. This is a demanding target. The non-LLM baselines show that even deterministic mechanisms face fundamental tradeoffs between disclosure and cost optimality. LLM agents, operating in natural language with richer expressive capacity, have the potential to approach this target more flexibly, but the current results show they reliably fall short in cost-sensitive settings.

The privacy-fairness tension in particular has implications beyond calendar scheduling. Any multi-agent system in which agents hold heterogeneous private costs and must allocate shared burden, such as supply chains, resource scheduling, or distributed planning, will face the same structural problem: an agent that suppresses cost signals preserves its own privacy but deprives the group of information needed for equitable allocation. Designing agents that share the right information at the

right level of precision, in natural language, under partial information, is an open research problem that CalBench is designed to make empirically tractable.

5 Conclusion

Two empirical settings, poker and calendar scheduling, instantiate complementary forms of multi-agent reasoning under partial information. Poker is adversarial: agents must reason about hidden states, opponent beliefs, strategic signals, and exploitability. Calendar scheduling is cooperative: agents must coordinate across private calendars, exchange information selectively, preserve privacy, and allocate disruption costs fairly. Together, these environments test whether language-model agents can act effectively when success depends not only on the external task state, but also on the private information, behavior, and incentives of other agents.

Across both domains, the central finding is that current agents exhibit local strategic competence but limited trajectory-level adaptation to others. In poker, agents infer plausible opponent ranges, recognize bluffing opportunities, and reason about how actions may affect opponent beliefs. However, these inferences do not consistently accumulate into adaptive opponent models or policy changes. Agents often treat opponent descriptions as fixed labels, reuse stale assumptions, or store observations without updating later action selection. In calendar scheduling, agents can exchange availability, negotiate candidate slots, reschedule commitments, and reach agreement. Yet task completion alone hides important failures: agents may incur avoidable displacement costs, communicate imprecisely, allocate burden unevenly, or preserve privacy by withholding cost-relevant information needed for fair coordination.

These failures suggest that multi-agent competence should be evaluated over trajectories, not only over final outcomes or isolated decisions. A general way to express this is to model a multi-agent interaction as a coupled trajectory. Let

$$x_t^i = (o_t^i,\, m_t^i,\, a_t^i) \tag{9}$$

denote agent i's interaction record at time t, where o_t^i is its private observation, m_t^i is its message, and a_t^i is its environment action. The joint interaction record at time t is then $\mathbf{x}_t = \{x_t^i\}_{i=1}^N$, and the full trajectory is

$$\tau = (s_0,\, \mathbf{x}_0,\, s_1,\, \mathbf{x}_1,\, \ldots,\, s_{T-1},\, \mathbf{x}_{T-1},\, s_T) \tag{10}$$

where s_t is the latent global state at time t and $s_{0:T} = (s_0, \ldots, s_T)$ denotes the full state sequence. Each agent observes only part of this state, but the outcome depends on the joint behavior of all agents. Let $\tau_i = (x_0^i, \ldots, x_{T-1}^i)$ denote agent i's local trajectory and τ_{-i} the trajectories of the other agents. A plausible multi-agent evaluation objective is then

$$R_i(\tau) = \sum_{\ell=1}^{L} \lambda_\ell\, f_\ell(\tau_i,\, \tau_{-i},\, s_{0:T}) \tag{11}$$

where each f_ℓ is a scalar-valued component evaluating one dimension of performance (i.e., task completion, utility, fairness, privacy cost, communication efficiency, or adaptation to other agents), and the weights $\lambda_\ell \geq 0$, with $\sum_\ell \lambda_\ell = 1$, specify how these dimensions are traded off in a given setting. Agent i is evaluated not only by what it did, but by how its behavior related to the observations, messages, actions, and outcomes of the other agents.

The formulation applies directly to both domains, though the reward components differ. In poker, f_ℓ can capture whether the agent adapted to opponent behavior, avoided exploitable patterns, and selected strategies robust to hidden information, not only final chip gain. In calendar scheduling, f_ℓ can capture whether agents minimized disruption, revealed private constraints appropriately, and allocated costs fairly, not only whether a meeting was scheduled.

This suggests a broader training principle: multi-agent reinforcement learning should optimize both outcome-level and interaction-level properties. Outcome-level rewards evaluate whether the task was completed and joint utility was high. Interaction-level rewards evaluate how that outcome was produced (i.e., whether agents communicated efficiently, preserved appropriate privacy, revised stale beliefs, distributed burden fairly, and remained robust to changes in others' behavior, etc.). These process signals matter for credit assignment because the same final outcome can arise from very different interaction paths: a meeting can be scheduled by needlessly disrupting one participant, or a poker hand won through pure luck. Final success alone cannot distinguish these cases, especially when resources are constrained.

Real-world agents will cooperate with humans, negotiate with other assistants, compete with strategic actors, and operate under incomplete information. These settings require agents to maintain representations of other agents, decide what to reveal, interpret what others reveal, revise beliefs over time, and select actions whose value depends on the behavior of others. If AI systems are to become reliable participants in human and machine societies, they must be evaluated as participants in coupled multi-agent trajectories.

References

[1] Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment, 2025.

[2] Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

[3] Noam Brown and Tuomas Sandholm. Superhuman AI for multi-player poker. *Science*, 365(6456):885–890, 2019.

[4] Tim R. Davidson, Adam Fourney, Saleema Amershi, Robert West, Eric Horvitz, and Ece Kamar. The collaboration gap, 2025.

[5] Jinhao Duan, Renming Zhang, James Diffenderfer, Bhavya Kailkhura, Lichao Sun, Elias Stengel-Eskin, Mohit Bansal, Tianlong Chen, and Kaidi Xu. GTBench: Uncovering the strategic reasoning limitations of LLMs via game-theoretic evaluations. In *Advances in Neural Information Processing Systems*, volume 37, pages 28219–28253, 2024.

[6] Ferdinando Fioretto, Enrico Pontelli, and William Yeoh. Distributed constraint optimization problems and applications: A survey. *Journal of Artificial Intelligence Research*, 61:623–698, 2018.

[7] Jose L. Garcia, Karolina Hajkova, Maria Marchenko, and Carlos Miguel Pati no. Reproducibility study of cooperation, competition, and maliciousness: Llm-stakeholders interactive negotiation, 2025.

[8] Arpandeep Khatua, Hao Zhu, Peter Tran, Arya Prabhudesai, Frederic Sadrieh, Johann K. Lieberwirth, Xinkai Yu, Yicheng Fu, Michael J. Ryan, Jiaxin Pei, and Diyi Yang. Cooperbench: Why coding agents cannot be your teammates yet, 2026.

[9] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan

Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents, 2025.

[10] Rajiv T. Maheswaran, Jonathan P. Pearce, Emma Bowring, Pradeep Varakantham, and Milind Tambe. Privacy loss in distributed constraint reasoning: A quantitative framework for analysis and its applications. *Autonomous Agents and Multi-Agent Systems*, 13:27–60, 2006.

[11] Rajiv T. Maheswaran, Milind Tambe, Emma Bowring, Jonathan P. Pearce, and Pradeep Varakantham. Taking DCOP to the real world: Efficient complete solutions for distributed multi-event scheduling. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 310–317. IEEE Computer Society, 2004.

[12] Saaduddin Mahmud, Eugene Bagdasarian, and Shlomo Zilberstein. CoLLAB: A framework for designing scalable benchmarks for agentic LLMs. In *Workshop on Scaling Environments for Agents*, 2025.

[13] Pragnesh Jay Modi, Wei-Min Shen, Milind Tambe, and Makoto Yokoo. ADOPT: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161(1–2):149–180, 2005.

[14] Pragnesh Jay Modi and Manuela Veloso. Multiagent meeting scheduling with rescheduling. In *Proceedings of the Fifth Workshop on Distributed Constraint Reasoning*, Toronto, Canada, 2004.

[15] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. DeepStack: Expert-level artificial in-

telligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

[16] Abhijnan Nath, Hannah VanderHoeven, and Nikhil Krishnaswamy. Craft: Grounded multi-agent coordination under partial information, 2026.

[17] Laurent Perron and Frédéric Didier. Cp-sat, 2025. Google OR-Tools, version 9.12.

[18] Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024.

[19] Huaixiu Steven Zheng, Swaroop Mishra, Hugh Zhang, Xinyun Chen, Minmin Chen, Azade Nova, Le Hou, Heng-Tze Cheng, Quoc V. Le, Ed H. Chi, and Denny Zhou. Natural plan: Benchmarking llms on natural language planning, 2024.

[20] Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. SOTOPIA: Interactive evaluation for social intelligence in language agents. In *The Twelfth International Conference on Learning Representations*, 2024.

[21] Richard Zhuang, Akshat Gupta, Richard Yang, Aniket Rahane, Zhengyu Li, and Gopala Anumanchipalli. PokerBench: Training large language models to become professional poker players. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(24):26175–26182, 2025.

[22] Martin Zinkevich, Michael Johanson, Michael H. Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. In *Advances in Neural Information Processing Systems*, volume 20, pages 1729–1736. Curran Associates, Inc., 2007.