SCTL: STATE CONDITIONED TEMPORAL LEDGERS FOR LONG HORIZON LLM TEAMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon tasks give teams of large language model (LLM) agents many chances to drift from a cooperative plan. Existing methods fix each agent's share of the team's result before execution, although the work that remains changes as the task proceeds. We show that whether LLM agents stay on the plan depends on when each agent's share of the team's cost is charged, even when the total stays the same. We introduce the State-Conditioned Temporal Ledger (SCTL). At each checkpoint it estimates the cost that each group of agents would still bear from the observed state and splits the team's cost with the Shapley value. Each agent then sees how much it has paid and how much it still owes. Unlike classical time-consistent payment schedules, which are defined only along the agreed plan, SCTL makes each payment depend on the time and on the state. An agent that deviates keeps its unpaid balance as a penalty and pays it to the team when it returns. SCTL works on top of existing workflows without retraining. On nine long-horizon multi-agent LLM benchmarks and CoopEval, SCTL beats the strongest baseline fixed in advance in eight of ten settings, with a median long-horizon gain of 3.6%. On a controlled benchmark with exact values, it reduces the normalized gap in team cost by a median 37.1% relative to charging the same total in equal parts, and the advantage grows with the length of the task. Keeping the unpaid balance as a penalty cuts the cost gap of a delayed schedule from 0.611 to 0.166.

1 INTRODUCTION

Research on large language model (LLM) agents is shifting from single agents to teams that work on long tasks together, as in AgentVerse (Chen et al., 2024) and ChatDev (Qian et al., 2024). A team of LLM agents helps only if the agents keep coordinating over the whole task. Long tasks give agents many chances to drift from the cooperative plan, because they can abandon roles, duplicate work, or take conflicting actions as execution proceeds (Cemri et al., 2025). Each drift changes what the team sees and does next, so its effect carries over to later steps. Whether agents stay on the cooperative plan therefore affects the final task score (Zhu et al., 2025). Reaching the team's result costs effort, tokens, and time, and each agent carries part of this cost. When a benchmark reports only a score, we treat the part of the task that the team fails to complete as a cost of the same kind. How that part is set, and when it is charged, shapes whether the agent keeps to the plan.

Existing methods improve how agents communicate (Wang et al., 2025), train coordination policies (Park et al., 2025), or add instructions to the prompt (Nguyen et al., 2025). Shapley-based pricing (Hua et al., 2025) and contracts (Tewolde et al., 2026) assign each agent a share of the team's result before execution starts. However, an agent's incentive to follow the plan changes during a long task. As work gets done, the cost that an agent would pay after deviating from the plan falls, because less work remains. If its share was fixed at the start and is charged later, the part it still owes can exceed this cost. The agent then has a reason to deviate from the plan, even when the team loses from it. Human experiments on repeated games show that cooperation depends on how players weigh later payoffs against earlier ones (Kim, 2023).

Cooperative game theory studies how to keep an agreed split acceptable over time. Classical time-consistent payment schedules (Petrosyan, 1977) define each payment as a function of time along an agreed cooperative trajectory. The schedules assume a known game, a known horizon, and players who never leave the trajectory. In LLM teams none of these holds. Agents deviate, so the state leaves

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

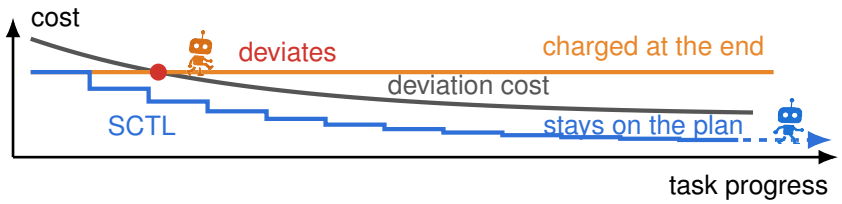

Figure 1: **Overview of SCTL (illustrative, no data).** If the whole share is charged at the end (orange), the amount still owed stays high while the deviation cost falls (gray), and after the red dot deviating is cheaper. SCTL recomputes the share at every checkpoint (blue), so staying on the cooperative plan stays cheaper.

the planned trajectory, the final step is often unknown, and no model of the task is available. We introduce the *State-Conditioned Temporal Ledger (SCTL)*, a payment mechanism for this setting. SCTL recomputes each agent's share of the team's cost at every checkpoint from the state that the team has reached, so each payment depends on the time and on the state, also off the plan. An agent that deviates keeps its unpaid balance as a penalty and pays it to the team after it returns to the plan, and the horizon is a forecast that moves with every checkpoint. An agent has a reason to deviate once this remaining balance exceeds the deviation cost, as sketched in Figure 1. When no agent deviates and the horizon is known, SCTL reduces to the classical schedule. We test SCTL on nine long-horizon multi-agent LLM benchmarks, including GovSim, Melting Pot, MultiAgentBench, and Collab-Overcooked, and on CoopEval, ten settings in total. A small controlled benchmark with exact values measures when agents deviate.

Our contributions are as follows.

- We introduce SCTL, a payment mechanism for LLM teams whose payments depend on the state as well as on time and which charges a penalty for deviating from the cooperative plan. SCTL works on top of existing workflows without retraining.
- We test when agents deviate with two controlled interventions. One changes only when the share is charged and hides the deviation cost from the agent. The other changes only the announced deviation cost and keeps the balances fixed.
- On these ten settings and seven LLMs, SCTL beats the strongest baseline in eight, and its advantage grows with the length of the task. On the controlled benchmark, the payment schedule changes when agents deviate, and the penalty keeps them on the plan. Rollout estimates explain most of the gain on the LLM benchmarks, and the per-agent ledger adds a smaller gain that grows when agents carry unequal shares.

2 Related work

Long-horizon coordination in LLM teams. LLM teams are commonly organized by roles, as in AgentVerse (Chen et al., 2024), ChatDev (Qian et al., 2024), and MetaGPT (Hong et al., 2024), while multi-agent debate improves single answers through rounds of argument (Du et al., 2024). Social-dilemma benchmarks ask whether agents preserve a shared resource, as in GovSim (Piatti et al., 2024) and Melting Pot (Agapiou et al., 2023). Cooperative benchmarks measure joint task completion in MultiAgentBench (Zhu et al., 2025), Collab-Overcooked (Sun et al., 2025), and ParaCook (Zhang et al., 2026). Mixed-motive benchmarks add private objectives in Cooperate to Compete (O'Neill et al., 2026), supply chains in CoffeeBench (Sugiura et al., 2026), open-ended survival in Alem (Tessera et al., 2026), and general-sum negotiation in Welfare Diplomacy (Mukobi et al., 2023). Failure analyses find that many errors arise from the interaction between agents rather than from single agents (Cemri et al., 2025). Methods that improve coordination change the prompt (Nguyen et al., 2025), the interaction protocol (Wang et al., 2025), or the policy through post-training (Park et al., 2025).

Incentives and payments for cooperating agents. LLM work prices cooperation among self-interested agents with Shapley values (Hua et al., 2025) and changes incentives through agreements. CoopEval compares mechanisms that sustain cooperation among LLM agents, such as contracts and mediation (Tewolde et al., 2026), and LLM agents also negotiate their own contracts (Wyse et al.,

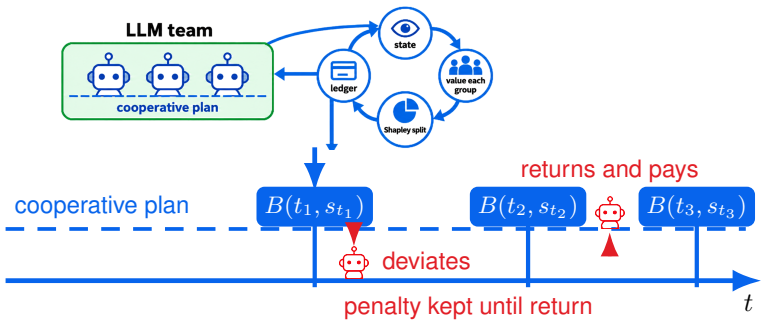

Figure 2: **Architecture of SCTL.** At every checkpoint SCTL reads the state, values each group of agents, splits the team's cost with the Shapley value, and updates each agent's ledger. The balance $B(t_k, s_{t_k})$ at checkpoint t_k depends on the time and on the state reached, and a deviating agent keeps its unpaid balance as a penalty until it returns.

2026). Contract, mediation, and negotiation methods fix the terms of cooperation before or during the task, but they do not say when each agent pays its part. Cooperative dynamic games study this timing question. Petrosyan (1977) introduced time consistency. Time consistency requires the unpaid part of an agreed allocation to equal what the same allocation rule would assign if the game restarted from any later state on the cooperative path. Payment schedules of this kind assume a known game model, a known cooperative path, and players who never leave it. Appendix A gives an extended review.

Lookahead and temporal credit. LLM agents often plan ahead at inference time. Reasoning via Planning (RAP) (Hao et al., 2023) uses the LLM as a model of the world and searches over reasoning steps with Monte Carlo tree search. Language Agent Tree Search (LATS) (Zhou et al., 2024) extends this search to agents that act in an environment. A second line moves credit in time rather than searching ahead. Potential-based shaping adds intermediate rewards without changing the optimal policy (Ng et al., 1999), and Return Decomposition for Delayed Rewards (RUDDER) redistributes a delayed return to the earlier steps that caused it (Arjona-Medina et al., 2019). Credit assignment methods act on a single agent's decisions or learning signal and do not split a team's cost among its members.

3 Method

SCTL is an inference-time layer for a team of LLM agents that work on a long task. SCTL has three layers. At regular checkpoints it estimates the value of each group of agents, an allocation rule turns these values into one share per agent, and a ledger turns each share into payments over time. Each payment depends on the time and on the state that the team has reached. An agent that deviates from the cooperative plan keeps its unpaid balance as a penalty and pays it to the team after it returns. SCTL does not change the agents' actions and only adds the ledger to each agent's prompt. Figure 2 shows one checkpoint and the ledger over time. Algorithm 1 in Section 3.4 gives the complete loop.

3.1 Interaction model

We model the interaction of the agents as a Markov decision process with several agents and partial observations (Oliehoek and Amato, 2016). The team is the set of agents $N = \{1, \dots, n\}$. Time runs in native steps $t = 0, 1, 2, \dots$, where a native step is one move of the underlying task, such as one round of a benchmark. At step t the task is in state $s_t \in \mathcal{S}$. Agent i receives the observation $o_{i,t} = \chi_i(s_t)$, which is the text of its prompt, and chooses an action $a_{i,t}$ from its action set $\mathcal{U}_i$. The joint action $a_t = (a_{1,t}, \dots, a_{n,t})$ moves the task to the next state $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$. At every step agent i bears a cost $c_i(s_t, a_t) \geq 0$, such as the tokens, tool calls, or effort it spends and its part of the loss when the task goes wrong. The environment states these costs in the prompt (Appendix G), and the policy of agent i is its LLM. We write $A_t \subseteq N$ for the agents that follow the plan at the current step t. An agent leaves A_t when it deviates or stops responding, and it rejoins A_t when it returns to the plan (Appendix M). On the LLM benchmarks the agents have no explicit option to deviate, so A_t changes only when an agent stops responding, and deviation shows up in the task outcomes.

Horizon. The final step of a long task is often unknown. SCTL therefore works with a forecast $\hat{T}_t$ of the final step, made at step t. When the task announces its final step T, the forecast is fixed at $\hat{T}_t = T$. Otherwise SCTL values the next H decision rounds of the team, where a decision round is one native step in which the agents act, so $\hat{T}_t = t + H$. The forecast moves forward with every checkpoint, as in receding-horizon control. Appendix J treats tasks without a final step. SCTL updates the ledger only at *checkpoints*, a set $\mathcal{T}$ of chosen native steps.

Cooperative behavior, trajectory, and payoff. Costs can be discounted at rate $\rho \geq 0$, as is standard in sequential decision problems (Sutton and Barto, 2018), and $\rho = 0$ means no discount. For a joint policy π, the expected discounted cost of agent i from step t is

$$J_i(\pi; t, s_t) = \mathbb{E}\left[\sum_{t'=t}^{\hat{T}_t - 1} (1+\rho)^{-(t'-t)} c_i(s_{t'}, a_{t'}) \;\middle|\; s_t, \pi\right], \tag{1}$$

and the team's cost is $J(\pi; t, s_t) = \sum_{i \in A_t} J_i(\pi; t, s_t)$. The *cooperative behavior* is the joint policy π^C that minimizes the team's cost, $\pi^C \in \arg\min_\pi J(\pi; t, s_t)$. The *cooperative cost* is this minimum, $V^C(t, s_t) = J(\pi^C; t, s_t)$, the lowest total cost that the team can reach from s_t. The *cooperative trajectory* $s_0^*, s_1^*, \ldots$ is the sequence of states that the team visits when every agent follows π^C from s_0. We also call π^C the *cooperative plan*. An agent *deviates* at step t when its action differs from the action $\pi_i^C(s_t)$ that the plan prescribes for it. After a deviation the state s_t can differ from s_t^*.

Fallback policy and coalitions. An agent that deviates follows a fixed *fallback policy* π_i^F, the policy that it uses when it coordinates with no one. On the LLM benchmarks it is the benchmark's own behavior of a single agent, kept fixed during the run, or an action that does nothing at that step (Appendix G). A *coalition* is any nonempty group $S \subseteq A_t$ of agents that keep following the cooperative plan while the agents outside S act on their own.

3.2 Values and shares

Why agents need a share. In an LLM team the agents draw on shared resources, such as a token budget or a pool of tool calls, so one agent's effort often lowers the cost of all others. Games in which a group shares costs are a standard class of cooperative games (Chander and Tulkens, 1997; Germain et al., 2003). A cost share states which part of the cooperative cost each agent carries.

Value of a coalition. To split the team's cost, SCTL needs the cost that each coalition would still pay from the state reached. The members of the coalition keep to the plan together, and every other agent follows its own policy. Shapley-based credit methods for LLM agents use the same values of groups (Hua et al., 2025; Yang et al., 2026). The value of S at step t in state s_t is

$$v_t^{\mathrm{cost}}(S \mid s_t) = \mathbb{E}\left[\sum_{i \in S} \sum_{t'=t}^{\hat{T}_t - 1} (1+\rho)^{-(t'-t)} c_i(s_{t'}, a_{t'}) \;\middle|\; s_t, \pi_S^C, \pi_{A_t \setminus S}^{\mathrm{out}}\right], \qquad v_t^{\mathrm{cost}}(\varnothing \mid s_t) = 0, \tag{2}$$

where π_S^C is the policy with which the members of S minimize their total cost and $\pi_{A_t \setminus S}^{\mathrm{out}}$ are the policies of the agents outside S. At the final step nothing remains to be paid, so $v_{\hat{T}_t}^{\mathrm{cost}}(S \mid s) = 0$ for every coalition S and state s. The value of the whole team is the cooperative cost, $v_t^{\mathrm{cost}}(A_t \mid s_t) = V^C(t, s_t)$. On the LLM benchmarks the outside agents follow their fallback policies, and on the controlled benchmark they play a feedback Nash equilibrium (Appendix N.10). The value depends on the state s_t, so it changes when the team leaves the cooperative plan.

The ledger therefore works in two orientations. On the controlled benchmark it counts costs, where lower is better. On the LLM benchmarks the costs c_i are not observed, so SCTL uses the benchmark loss ℓ_b in their place and counts the loss that cooperation avoids, where higher is better. The expected loss $L_t(S \mid s_t)$ plays the role of $v_t^{\mathrm{cost}}(S \mid s_t)$, and $w_t(S \mid s_t)$ is the loss that the coalition avoids,

$$L_t(S \mid s_t) = \mathbb{E}\Big[\ell_b \;\Big|\; s_t, \pi_S^C, \pi_{A_t \setminus S}^F\Big], \qquad w_t(S \mid s_t) = L_t(\varnothing \mid s_t) - L_t(S \mid s_t), \tag{3}$$

where the expectation is over the rest of the run from step t to the forecast final step. At that step the loss is fixed for every coalition, so $w_{\hat{T}_t}(S \mid s) = 0$. Both quantities are computed at each checkpoint before the ledger is shown to the acting agents (Appendix G).

Shares. Let v denote the value function at the state reached, $v_t^{\text{cost}}(\cdot \mid s_t)$ or $w_t(\cdot \mid s_t)$, and let A be the active agents. An allocation rule assigns each active agent a share $\Phi_i(v)$ of the team's value, and the shares add up to the team's value, $\sum_{i \in A} \Phi_i(v) = v(A)$. For costs a share is individually acceptable when no agent carries more than it would pay alone, $\Phi_i(v) \le v(\{i\})$, and for avoided loss when no agent receives less credit than it would secure alone. SCTL uses the Shapley value (Shapley, 1953),

$$\Phi_i(v) = \sum_{S \subseteq A \setminus \{i\}} \frac{|S|!\,(|A| - |S| - 1)!}{|A|!} \big[v(S \cup \{i\}) - v(S)\big]. \tag{4}$$

The bracket is the change in the coalition's value when agent i joins it, and the weight is the probability of that joining order when the team forms in a uniformly random order. Section 4.3 compares it with the proportional and equal-savings rules (Appendix C.3). Teams of up to three agents use Equation (4) exactly, and larger teams estimate it from four fixed random orders (Castro et al., 2009).

3.3 Payments with a penalty for deviation

SCTL does not charge an agent its whole share at once. At every checkpoint it recomputes the shares of the agents on the cooperative plan from the state that the team has reached. Let $\varphi_i(t, s_t) = (1+\rho)^{-t} \Phi_i\big(v_t^{\text{cost}}(\cdot \mid s_t)\big)$ be the share of agent i at step t in state s_t, discounted to step 0, and let $\Xi_i = \varphi_i(0, s_0)$ be its initial share. The ledger tracks what an agent still owes, what it is charged now, and what it has paid so far. The three quantities are the remaining balance $B_i(t, s_t)$, the charge $\beta_i(t)$ at step t, and the amount $P_i(t)$ charged before step t,

$$B_i(t, s_t) = \varphi_i(t, s_t) + O_i(t), \quad \beta_i(t) = B_i(t, s_t) - B_i(t+1, s_{t+1}), \quad P_i(t) = \sum_{t' < t} \beta_i(t'), \tag{5}$$

where $O_i(t)$ is the unpaid part of the penalty defined below, and $O_i(t) = 0$ for an agent that has never deviated. For such an agent the charged amount and the remaining balance add up to the initial share,

$$P_i(t) + B_i(t, s_t) = \Xi_i, \tag{6}$$

so the ledger only redistributes the share over time and adds no task cost. The ledger has two regimes. On the cooperative trajectory, $s_t = s_t^*$, the balance $B_i(t, s_t^*) = \varphi_i(t, s_t^*)$ is a function of time alone and equals the classical time-consistent schedule (Petrosyan, 1977). After a deviation, $s_t \neq s_t^*$, the same rule is evaluated at the state that the team has reached, and a deviating agent also carries its penalty $O_i(t)$. A charge is negative, a rebate, when the recomputed share rises, and schedules without rebates are studied by Petrosyan et al. (2017). At each checkpoint the prompt shows Ξ_i, $P_i(t)$, $B_i(t, s_t)$, and the latest charge. On the LLM benchmarks the ledger applies the same rule to the avoided loss w_t instead of v_t^{cost}. The prompt then shows the remaining credit and the credit realized so far (Appendix M).

Penalty for deviation. When agent i deviates at step t, it leaves the plan, and SCTL recomputes the shares of the agents that stay, who then carry the cost of the plan. The unpaid balance of agent i is not written off. SCTL keeps it as the penalty $O_i = B_i(t, s_t)$. If the agent returns to the plan at a later step, SCTL recomputes the shares on the larger team and adds the penalty to the new share of the agent, as in Equation (5). The agent then pays its old balance to the team on top of its new share, through its later charges, so its balance still reaches zero at the final step. At every state the team reaches, each balance is the recomputed share plus any retained penalty, so the time consistency condition extends to the states after a deviation.

How SCTL differs from classical payment schedules. Classical schedules (Petrosyan, 1977; Yeung and Petrosyan, 2012) assume a known game, a known horizon, and players who never leave the cooperative trajectory. SCTL drops these assumptions. Its payment is defined at every state, its forecast horizon moves with every checkpoint, and it estimates values by rollouts when no model of the task exists. SCTL does not assume that agents respond to the ledger, and Section 4 measures their response.

3.4 Estimating values and the complete loop

On the controlled benchmark SCTL computes every value exactly. On the LLM benchmarks it estimates each coalition value from $R = 2$ matched rollouts per coalition over a forecast window of

Algorithm 1 SCTL closed-loop coordination protocol

Input: task with initial state s_0, team N, plan π^C, rule Φ, checkpoints $\mathcal{T}$, rollouts R, window H, switches I_{pen}, I_{id}, I_{rec}

1: $P_i \leftarrow 0$, $O_i \leftarrow 0$, $B_i \leftarrow 0$ for all i, and `fallback`←false
2: **for** $t = 0, 1, \ldots$ until the task ends **do**
3: observe the state s_t and the active agents A_t
4: **if** $t \in \mathcal{T}$ and not `fallback` **then**
Forecasting the horizon (Section 3.1)
5: $\hat{T}_t \leftarrow T$ if the task announces its final step T, else $\hat{T}_t \leftarrow t + H$
Valuing coalitions (Section 3.2)
6: estimate $v_t(S \mid s_t)$ for all $S \subseteq A_t$ from R rollouts up to $\hat{T}_t$ ▷ Eq. (2)
Splitting the cost (Section 3.2)
7: $\varphi_i \leftarrow (1+\rho)^{-t}\Phi_i(v_t(\cdot \mid s_t))$ for all $i \in A_t$ ▷ Eq. (4)
Updating the ledger (Section 3.3)
8: **for all** i that left the plan since the last checkpoint **do**
9: **if** $I_{\text{pen}} = 1$ **then**
10: keep the penalty $O_i \leftarrow B_i$
11: **else**
12: write off B_i
13: $\beta_i \leftarrow B_i - (\varphi_i + O_i)$ and $B_i \leftarrow \varphi_i + O_i$ for all $i \in A_t$ ▷ Eq. (5)
14: $P_i \leftarrow P_i + \beta_i$, and show Ξ_i, P_i, B_i, β_i in the prompt of each $i \in A_t$
Acting
15: agents act with the ledger in context
16: **if** $I_{\text{id}} = 1$ and an agent chooses EXIT **then** `fallback`←true
17: execute the actions and observe s_{t+1}
18: **if** $I_{\text{rec}} = 1$ and an agent stops **then** recompute the shares of the rest
19: **return** outcome, ledger trace, deviation and return events

$H = 6$ decision rounds (Appendix G). Agents outside the coalition follow their fallback policies, and rollout transcripts never enter the acting context. Rollouts account for most of the tokens that SCTL uses (Table 37). Algorithm 1 combines these steps into the complete loop and follows the order of this section. Line 5 forecasts the final step $\hat{T}_t$ (Section 3.1), line 6 estimates the coalition values of Equation (2), and line 7 splits the cost with the Shapley value. Lines 8–13 keep the penalty and update the ledger of Equation (5), and the agents then act with the ledger in context (line 15). The switches I_{pen}, I_{id}, and I_{rec} turn on the penalty, the deviation protocol of the controlled benchmark, and the recomputation after an agent stops responding.

4 EXPERIMENTS

The experiments test three claims. SCTL raises the scores of long-horizon multi-agent LLM benchmarks without changing the actions available to the agents. Crossing the participation boundary predicts when agents deviate, and the penalty keeps them on the plan. The ablations identify which parts of SCTL cause these effects. Tables 1–3 report the main, supporting, and ablation results.

4.1 EXPERIMENTAL SETUP

LLM benchmarks. We evaluate SCTL on nine long-horizon multi-agent LLM benchmarks, GovSim (Piatti et al., 2024), Melting Pot (Agapiou et al., 2023), MultiAgentBench (MAB) (Zhu et al., 2025), Collab-Overcooked (C-OC) (Sun et al., 2025), ParaCook (Zhang et al., 2026), Cooperate to Compete (C2C) (O'Neill et al., 2026), CoffeeBench (Sugiura et al., 2026), Alem (Tessera et al., 2026), and Welfare Diplomacy (Mukobi et al., 2023), and on the CoopEval suite (Tewolde et al., 2026). We use seven models, GPT-6 Astra, GPT-5.6 Sol, Claude Fable 5.1, Claude Opus 5, Gemini 3.8 Flash, Qwen3-8B, and Qwen3-14B, with 20 instances per model and task. SCTL keeps each benchmark's own observations and actions and only adds the ledger to the prompt. Native+SCTL adds the ledger to the multi-agent workflow that comes with the benchmark. GovSim, C2C, and Alem report predeclared adapter indices and the other benchmarks their native metrics (Appendix G, with checkpoints and workflows).

Baselines. The baselines are existing methods for cooperation among LLM agents. Shapley-Coop (Hua et al., 2025) prices cooperation with Shapley values on seven tasks, self-negotiated contracts (Wyse et al., 2026) are used on CoffeeBench and Welfare Diplomacy, and contracts and mediation (Tewolde et al., 2026) on CoopEval. We also run the multi-agent workflow that comes with each benchmark, such as Universalization in GovSim (Piatti et al., 2024), and a prompt that asks every agent to consider the future consequences of its actions (CFC) (Nguyen et al., 2025). Before the main runs we fix the strongest of these methods for each benchmark and call it the fixed comparator.

It is the cooperation mechanism in GovSim, C2C, Alem, MAB, C-OC, Welfare Diplomacy, and CoopEval, and the benchmark workflow in CoffeeBench, Melting Pot, and ParaCook.

Controlled benchmark. The LLM benchmarks do not reveal the step at which deviating becomes cheaper for an agent than staying. We therefore add a controlled benchmark with exact coalition values, a dynamic game over a shared backlog of work in six variants G1–G6 (Appendix C). At each step an agent chooses STAY, which keeps executing the cooperative plan, or EXIT, which deviates from it for the rest of the task. K-Hidden hides the deviation cost from the agent, and Matched-B changes only the announced deviation cost while the balances stay fixed (Appendix N.10). The outcome G_{cost} is the team's extra cost over the cooperative plan divided by the extra cost of independent play, so 0 is the cooperative plan and 1 is independent play.

Ablation variants. Besides the baselines, we compare SCTL with variants of itself that change one part at a time. Frozen-SCTL keeps the ledger of the first checkpoint, UniformSchedule charges the same initial share in equal parts, and Delayed charges it all at the last step. SameInfoSummary shows the same values as plain text, ComputeMatched runs the same rollouts without showing their results, and NoMechanism shows no ledger. Appendix N.11 defines these and the other variants. Appendix F gives the decoding settings, and confidence intervals (CIs) are bootstrap intervals over task instances.

4.2 Main performance comparison

On the ten settings of Table 1A, the nine long-horizon benchmarks and CoopEval, SCTL has a higher mean score than the fixed comparator in 8 of 10 settings, and all eight paired intervals exclude zero. Across the nine long-horizon tasks, the median gain is 3.6%, with each gain signed so that positive values favor SCTL. CoffeeBench and CoopEval favor their fixed comparators, with paired intervals excluding zero. Within each of the four model families, SCTL beats the fixed comparator in 8 of 10 settings, with median gains of 3.4% to 3.8% (Appendix Tables 18 and 20). Native+SCTL improves $8/9$ long-horizon point estimates (Appendix Table 27). On the controlled benchmark in Table 1B, SCTL has the lowest G_{cost} and reduces the median per-game gap by 37.1% [32.8,41.9] relative to UniformSchedule.

4.3 Participation boundary and robustness

What we measure. The experiments on when agents deviate switch the penalty off, because with the penalty deviating never lowers what the agent owes (Appendix K). A deviation is then permanent, and all agents follow their fallback policies. The participation boundary is the first step at which the balance that an agent still owes exceeds its deviation cost, the cost it would pay from then on after deviating. It is defined only for methods that show the agent a balance. Under SCTL the balance stays below the deviation cost whenever the recomputed share is individually acceptable, a condition that cooperative differential games call agreeability (Jørgensen et al., 2003). Under Delayed the balance stays at the initial share while the deviation cost falls, so the schedule itself crosses the boundary. The instances are selected so that the SCTL path never crosses it. We fit a model of the first deviation at each checkpoint and report odds ratios (ORs) with instance-level bootstrap intervals and Holm-corrected p-values (Holm, 1979) (Appendix H).

At a checkpoint before any deviation, crossing the displayed participation boundary is associated with higher odds of a first deviation, $1.38\times$ in Matched-B and $1.52\times$ in K-Hidden (Table 2A). Both odds ratios are measured per checkpoint and do not translate into risk ratios. A placebo that changes the announced cost before any crossing gives an odds ratio of 1.02 (Table 2A). Replacing the Shapley value with the proportional or equal-savings rule changes scores on the LLM benchmarks by a median of 0.50% and 0.33%, and no interval on these benchmarks excludes zero (Appendix N.6). When one agent stops responding during the run, only the GovSim recovery effect remains significant after correction (Appendix H). Table 2B shows the gains by task group.

The deviation penalty, longer tasks, and another game. With permanent deviation and the Delayed schedule, keeping the unpaid balance payable after EXIT lowers G_{cost} from 0.611 to 0.166, and the crossing odds ratio falls to $1.05\times$ (Appendix E.5). When an agent can deviate and later return, carrying its unpaid balance into its new ledger lowers the share of steps off the plan from 0.241 to 0.165 and G_{cost} by 0.073 [0.044, 0.104] (Appendix N.2). The advantage of SCTL grows with the length of the task. On GovSim it rises from 1.9 to 6.2 points as the horizon grows from half to twice

Table 1: **Main performance matrix.** Panel A reports the ten LLM benchmark settings (metrics in Appendix G), and Panel B the controlled benchmark. Every score averages seven LLMs with 20 instances per model and task, and Appendix Table 18 gives each model. Boldface marks the best value in each column, and Appendix Tables 28 and 27 give paired intervals.

Panel A. LLM benchmarks										
Method	GovSim↑	C2C↑	Coffee↑	Alem↑	WDipl.↑	MPot↑	MAB↑	C-OC↑	ParaC.↓	CoopE.↑
Baselines										
Benchmark workflow	57.4	32.7	**18.9**	69.5	0.596	60.6	75.4	69.4	294.5	0.751
Cooperation mechanism	58.7	33.4	18.5	70.2	0.608	60.4	77.0	70.0	299.0	**0.804**
CFC-Prompt	55.8	31.8	17.1	68.2	0.574	58.9	75.4	68.1	300.1	0.714
SCTL										
SCTL (ours)	61.2	35.1	18.6	72.8	0.629	63.1	79.4	72.5	286.4	0.788
Native+SCTL	**62.8**	**35.9**	18.8	**73.5**	**0.636**	**64.0**	**80.2**	**73.4**	**281.7**	–
Ablation variants										
Frozen-SCTL	59.6	34.2	18.7	71.7	0.616	63.2	78.6	71.7	292.0	0.774
UniformSchedule	57.1	32.4	17.8	68.9	0.592	59.7	76.8	69.0	304.9	0.725
Delayed	54.4	30.5	16.4	65.8	0.558	57.3	73.2	64.8	329.6	0.646
ComputeMatched	52.1	30.5	16.4	64.1	0.546	53.4	70.3	62.3	330.7	–
SameInfoSummary	59.5	34.6	18.7	71.6	0.621	63.3	78.8	71.4	289.1	0.790
NoMechanism	52.3	30.4	16.5	64.0	0.548	53.6	70.2	62.5	330.2	0.693

Panel B. Controlled benchmark, G_{cost} ↓						
Method	G1	G2	G3	G4	G5	G6
Baselines						
CFC-Prompt	0.435	0.470	0.500	0.465	0.550	0.510
SCTL						
SCTL (ours)	**0.181**	**0.221**	**0.247**	**0.213**	**0.273**	**0.258**
Ablation variants						
UniformSchedule	0.305	0.347	0.382	0.351	0.440	0.405
Delayed	0.565	0.585	0.620	0.590	0.675	0.630
NoMechanism	0.505	0.535	0.575	0.530	0.610	0.575

Metric and baseline key. GovSim reports normalized mean survival time, C2C the private-objective progress index, and Alem the fixed-bound return index. WDipl. is Welfare Diplomacy, MPot is Melting Pot, ParaC. is ParaCook, and CoopE. is CoopEval. Cooperation mechanism is Shapley-Coop (Hua et al., 2025), except self-negotiated contracts (Wyse et al., 2026) in CoffeeBench and Welfare Diplomacy and contracts and mediation (Tewolde et al., 2026) in CoopEval. Section 4.1 lists the fixed comparator of each benchmark. ComputeMatched does not apply to CoopEval, and Native+SCTL covers the nine long-horizon tasks.

Table 2: **Participation boundary and robustness.** Panel A gives the odds of a first deviation once the displayed balance exceeds the deviation cost, and Panel B the median gain of SCTL in each task group, where positive values favor SCTL. LLM benchmarks I and II are the first five columns and the four long-horizon tasks among the last five columns of Table 1A.

Panel A. Deviations at the participation boundary			
Analysis	Odds ratio	95% CI	p
Matched-B boundary crossing → first deviation	1.38×	[1.16, 1.76]	0.001
K-Hidden boundary crossing → first deviation	1.52×	[1.30, 1.89]	< 0.001
Placebo before any crossing	1.02×	[0.91, 1.20]	0.74

Panel B. Median gain by task group						
	vs. NoMechanism		vs. Uniform		vs. fixed comparator	
Evaluation group	Effect	95% CI	Effect	95% CI	Effect	95% CI
Controlled G_{cost}	+57.9%	[54.2, 62.4]	+37.1%	[32.8, 41.9]	–	
LLM benchmarks I	+14.8%	[14.0, 15.7]	+6.3%	[5.4, 7.2]	+3.7%	[2.9, 4.4]
LLM benchmarks II	+14.6%	[13.8, 15.3]	+5.4%	[4.6, 6.3]	+3.4%	[2.6, 4.3]
CoopEval	+13.7%	[12.7, 14.8]	+8.7%	[7.2, 10.1]	−2.0%	[−3.3, −0.8]

its base length (Appendix N.1). On a game with a threshold public good, SCTL reduces G_{cost} by 24.2% relative to UniformSchedule (Appendix N.3).

4.4 Ablations

Table 3 reports the ablations. Most of the gain on the LLM benchmarks comes from showing the agents what the rollouts estimate, since ComputeMatched runs the same rollouts without showing them and stays within 0.6% of NoMechanism (Table 1A). Against StructuredTeamValue, which shows the team's total value in the same ledger fields without splitting it across agents, SCTL scores higher on $7/9$ tasks with $3/9$ intervals above zero. EqualSplitLedger, which splits the team's value

equally, stays within 0.33% of SCTL. At the highest asymmetry of a controlled stress test, the Shapley split lowers G_{cost} by 14.3% relative to an equal split (Appendix N.4). A 2×2 design separates the ledger format ($+1.19\%$) from the per-agent split ($+0.62\%$), with no reliable interaction (Appendix N.5). ActionLookahead spends the same rollout budget on a search over actions and scores below SCTL on $7/9$ tasks, with $4/9$ intervals above zero (Appendix Table 31). A quarter of the rollout budget keeps 75% of the SCTL margin on GovSim and MultiAgentBench (Appendix N.7).

Table 3: **Ablations.** Panel A changes one part of SCTL at a time, and Panel B changes only how the ledger is presented on the controlled benchmark. Dashes mark variants that exist only on the LLM benchmarks. Frozen-SCTL equals SCTL in the deterministic games G1–G6.

Panel A. Changing one part of SCTL

Variant	Changed component	Controlled $G_{\text{cost}}\downarrow$	Holdout $G_{\text{cost}}\downarrow$	GovSim↑
SCTL	reference	0.232	0.311	61.2
Frozen-SCTL	feedback from the observed state	0.232	0.376	59.6
TeamValueOnly	team value in one sentence	–	–	60.2
StructuredTeamValue	team value in ledger format	–	–	60.6
EqualSplitLedger	equal rather than Shapley allocation	–	–	61.0

Panel B. Changing how the ledger is presented, pooled controlled $G_{\text{cost}}\downarrow$

Presentation	SCTL	Uniform	Delayed	Ordering
K-Hidden (primary)	0.232	0.372	0.611	SCTL < Uniform < Delayed
K-Visible	0.158	0.212	0.261	SCTL < Uniform < Delayed
No remaining-balance line	0.181	0.235	0.301	SCTL < Uniform < Delayed
Free-form response	0.170	0.223	0.286	SCTL < Uniform < Delayed

Changes in how the ledger is presented move the cost levels, but the order SCTL < Uniform < Delayed in G_{cost} holds in every row of Table 3B. On the structural holdout, a separate set of games with random transitions and interacting actions (Appendix D.1), SCTL beats Frozen-SCTL by $+0.065$ [0.035,0.100].

4.5 Matched case study

A fixed payment schedule can make leaving the cooperative plan attractive in the middle of a long task. GovSim shows this, because the agents share a fishery over 24 rounds and deviating means taking more than the agreed catch. Figure 3 follows one matched trace, selected by a rule fixed before any outcome was inspected (Appendix N.8). Both runs use the same model and instance, and the focal agent starts with a credit of 0.342. SCTL updates the credit from the state that the team reaches, so the realized credit grows faster, 0.104 against 0.086 by step 6. With UniformSchedule the team has left the cooperative plan by step 12, while with SCTL it stays on the plan at every checkpoint. The SCTL team ends with a survival index of 62.4 against 57.8. Appendix N.9 gives a matched case on the controlled benchmark.

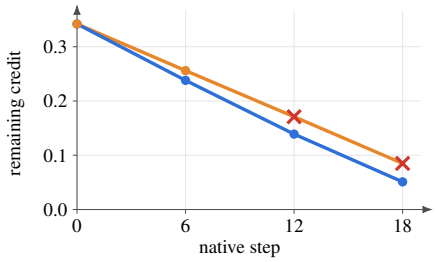

Figure 3: **Matched trace on GovSim.** Remaining credit of the focal agent under SCTL (blue) and UniformSchedule (orange). Dots mark checkpoints on the cooperative plan, and red crosses mark checkpoints after the team has left it.

5 Conclusion

SCTL is a payment mechanism for LLM teams whose payments depend on the time and on the state reached, with a penalty for deviating from the cooperative plan. On long-horizon multi-agent LLM benchmarks it beats the strongest baseline in eight of ten settings, and on the controlled benchmark it reduces the normalized cost gap by a median 37.1% relative to charging the same total in equal parts.

6 Limitations

SCTL needs extra rollouts at each checkpoint, and returns to the plan are tested only on the controlled benchmark (Appendix O).

AI USE STATEMENT

We used large language models to aid and polish the writing of this paper. The authors are responsible for all content of the paper, including the method, the experiments, the results, and the citations.

REFERENCES

John P. Agapiou, Alexander Sasha Vezhnevets, Edgar A. Duéñez-Guzmán, Jayd Matyas, Yiran Mao, Peter Sunehag, Raphael Köster, Udari Madhushani, Kavya Kopparapu, Ramona Comanescu, D. J. Strouse, Michael B. Johanson, Sukhdeep Singh, Julia Haas, Igor Mordatch, Dean Mobbs, and Joel Z. Leibo. Melting Pot 2.0. *arXiv preprint arXiv:2211.13746*, 2023.

Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the Shapley value based on sampling. *Computers & Operations Research*, 36(5):1726–1730, 2009. doi: 10.1016/j.cor.2008.04.004.

Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? In *Advances in Neural Information Processing Systems*, volume 38, 2025. Datasets and Benchmarks Track.

Parkash Chander and Henry Tulkens. The core of an economy with multilateral environmental externalities. *International Journal of Game Theory*, 26(3):379–401, 1997.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *International Conference on Learning Representations*, 2024.

Phillip J. K. Christoffersen, Andreas A. Haupt, and Dylan Hadfield-Menell. Get it in writing: Formal contracts mitigate social dilemmas in multi-agent RL. In *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems*, pages 448–456, 2023.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pages 2974–2982, 2018. doi: 10.1609/aaai.v32i1.11794.

Marc Germain, Philippe Toint, Henry Tulkens, and Aart de Zeeuw. Transfers to sustain dynamic core-theoretic cooperation in international stock pollutant control. *Journal of Economic Dynamics and Control*, 28(1):79–99, 2003. doi: 10.1016/S0165-1889(02)00107-0.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173, 2023.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations (ICLR)*, 2024.

Yun Hua, Haosheng Chen, Shiqin Wang, Wenhao Li, Xiangfeng Wang, and Jun Luo. Shapley-Coop: Credit assignment for emergent cooperation in self-interested LLM agents. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2965.

Steffen Jørgensen, Guiomar Martín-Herrán, and Georges Zaccour. Agreeability and time consistency in linear-state differential games. *Journal of Optimization Theory and Applications*, 119(1):49–63, 2003. doi: 10.1023/B:JOTA.0000005040.78280.a6.

Jeongbin Kim. The effects of time preferences on cooperation: Experimental evidence from infinitely repeated games. *American Economic Journal: Microeconomics*, 15(1):618–637, 2023. doi: 10.1257/mic.20200195.

David Levhari and Leonard J. Mirman. The great fish war: An example using a dynamic Cournot–Nash solution. *The Bell Journal of Economics*, 11(1):322–334, 1980.

Manuel Mosquera, Juan Sebastian Pinzón, Yesid Fonseca, Manuel Ríos, Nicanor Quijano, Luis Felipe Giraldo, and Rubén Manrique. Can LLM-augmented autonomous agents cooperate? An evaluation of their cooperative capabilities through Melting Pot. *IEEE Transactions on Artificial Intelligence*, 2025.

Gabriel Mukobi, Hannah Erlebach, Niklas Lauffer, Lewis Hammond, Alan Chan, and Jesse Clifton. Welfare Diplomacy: Benchmarking language model cooperation. *arXiv preprint arXiv:2310.08901*, 2023.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 278–287, 1999.

Dung Nguyen, Hung Le, Kien Do, Sunil Gupta, Svetha Venkatesh, and Truyen Tran. Navigating social dilemmas with LLM-based agents via consideration of future consequences. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence (IJCAI-25)*, pages 223–231, 2025. doi: 10.24963/ijcai.2025/26.

Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. SpringerBriefs in Intelligent Systems. Springer, 2016.

Abigail O'Neill, Alan Zhu, Mihran Miroyan, Narges Norouzi, and Joseph E. Gonzalez. Cooperate to compete: Strategic coordination in multi-agent conquest. *arXiv preprint arXiv:2604.25088*, 2026.

Elena Parilina and Georges Zaccour. Node-consistent core for games played over event trees. *Automatica*, 53:304–311, 2015.

Elena M. Parilina and Georges Zaccour. Payment schemes for sustaining cooperation in dynamic games. *Journal of Economic Dynamics and Control*, 139:104440, 2022. doi: 10.1016/j.jedc.2022.104440.

Chanwoo Park, Seungju Han, Xingzhi Guo, Asuman E. Ozdaglar, Kaiqing Zhang, and Joo-Kyung Kim. MAPoRL: Multi-agent post-co-training for collaborative large language models with reinforcement learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30215–30248, 2025. doi: 10.18653/v1/2025.acl-long.1459.

Leon A. Petrosyan. Stability of solutions in differential games with many participants. *Vestnik Leningrad University*, 19:46–52, 1977.

Leon A. Petrosyan, Artem A. Sedakov, Hao Sun, and Genjiu Xu. Convergence of strong time-consistent payment schemes in dynamic games. *Applied Mathematics and Computation*, 315:96–112, 2017. doi: 10.1016/j.amc.2017.07.030.

Giorgio Piatti, Zhijing Jin, Max Kleiman-Weiner, Bernhard Schölkopf, Mrinmaya Sachan, and Rada Mihalcea. Cooperate or collapse: Emergence of sustainable cooperation in a society of LLM agents. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186, 2024. doi: 10.18653/v1/2024.acl-long.810.

Bhavyesh Sajja, Max Kleiman-Weiner, Roger Zimmermann, and Zhi-Xuan Tan. Evaluating rational contracting in natural language. *arXiv preprint arXiv:2608.10475*, 2026.

Lloyd S. Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2:307–317, 1953.

Issa Sugiura, Daichi Hattori, Kazuo Araragi, Keita Ogawa, Shota Onose, Taro Makino, Teppei Usuki, and Takashi Ishida. CoffeeBench: Benchmarking long-horizon LLM agents in heterogeneous multi-agent economies. *arXiv preprint arXiv:2606.16613*, 2026.

Haochen Sun, Shuwen Zhang, Lujie Niu, Lei Ren, Hao Xu, Hao Fu, Fangkun Zhao, Caixia Yuan, and Xiaojie Wang. Collab-Overcooked: Benchmarking and evaluating large language models as collaborative agents. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 4922–4951, 2025. doi: 10.18653/v1/2025.emnlp-main.249.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

Kale-ab Abebe Tessera, Andras Szecsenyi, Cameron Barker, Alexander Rutherford, Davide Paglieri, Aidan Scannell, Henry Gouk, Elliot J. Crowley, Tim Rocktäschel, and Amos Storkey. Benchmarking open-ended multi-agent coordination in language agents. *arXiv preprint arXiv:2606.08340*, 2026.

Emanuel Tewolde, Xiao Zhang, David Guzman Piedrahita, Vincent Conitzer, and Zhijing Jin. CoopEval: Benchmarking cooperation-sustaining mechanisms and LLM agents in social dilemmas. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026.

Haochun Wang, Sendong Zhao, Jingbo Wang, Zewen Qiang, Bing Qin, and Ting Liu. Beyond frameworks: Unpacking collaboration strategies in multi-agent systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 21361–21375, 2025. doi: 10.18653/v1/2025.acl-long.1037.

Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. Shapley Q-Value: A local reward approach to solve global reward games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7285–7292, 2020. doi: 10.1609/aaai.v34i05.6220.

Tim Wyse, Kaitlin Bustos, Yulia Volkova, and Max Kleiman-Weiner. Commitment to cooperation with self-negotiated contracts. *arXiv preprint arXiv:2607.22750*, 2026.

Jiachen Yang, Ang Li, Mehrdad Farajtabar, Peter Sunehag, Edward Hughes, and Hongyuan Zha. Learning to incentivize other learning agents. In *Advances in Neural Information Processing Systems*, volume 33, pages 15208–15219, 2020.

Yingxuan Yang, Bo Huang, Siyuan Qi, Chao Feng, Haoyi Hu, Yuxuan Zhu, Jinbo Hu, Haoran Zhao, Ziyi He, Xiao Liu, ZongYu Wang, Muning Wen, Lin Qiu, Xuezhi Cao, Xunliang Cai, Yong Yu, and Weinan Zhang. Attribution-based analysis and optimization of modular agentic workflows. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 7247–7264, 2026. doi: 10.18653/v1/2026.findings-acl.359.

David W. K. Yeung and Leon A. Petrosyan. *Subgame Consistent Economic Optimization*. Birkhäuser, 2012.

David W. K. Yeung and Leon A. Petrosyan. *Subgame Consistent Cooperation: A Comprehensive Treatise*. Springer, 2016.

Shiqi Zhang, Xinbei Ma, Yunqing Xu, Zouying Cao, Pengrui Lu, Haobo Yuan, Tiancheng Shen, Zhuosheng Zhang, Hai Zhao, and Ming-Hsuan Yang. ParaCook: On time-efficient planning for multi-agent systems. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 27683–27701, 2026. doi: 10.18653/v1/2026.findings-acl.1378.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 62138–62160, 2024.

Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, and Jiaxuan You. MultiAgentBench: Evaluating the collaboration and competition of LLM agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8580–8622, 2025. doi: 10.18653/v1/2025.acl-long.421.

A Extended related work

This appendix reviews the work that SCTL builds on or is compared with. It follows the order of the main paper and covers LLM teams and their benchmarks, methods that change coordination, credit assignment, agreements and incentives, dynamic cooperative games, and methods that use lookahead or temporal credit. The last subsection states which parts of SCTL come from classical theory and which are new.

A.1 Multi-agent LLM systems and long-horizon benchmarks

LLM teams are usually organized by roles. AgentVerse recruits expert agents and adjusts the group as the task proceeds (Chen et al., 2024), ChatDev connects role agents in chat chains for software development (Qian et al., 2024), and MetaGPT encodes standard operating procedures in role prompts (Hong et al., 2024). A growing set of benchmarks tests whether such teams keep coordinating over many steps. MultiAgentBench measures collaboration and competition across several coordination structures (Zhu et al., 2025), Collab-Overcooked measures collaboration in a cooking environment (Sun et al., 2025), and ParaCook measures how efficiently agents plan parallel cooking tasks (Zhang et al., 2026).

Social-dilemma benchmarks test whether agents keep a shared resource or a cooperative outcome. GovSim studies common-pool resource management (Piatti et al., 2024). Melting Pot provides multi-agent environments that include Commons Harvest (Agapiou et al., 2023), and it has been used to evaluate LLM agents (Mosquera et al., 2025). Welfare Diplomacy turns Diplomacy into a general-sum game (Mukobi et al., 2023), and CoopEval compares mechanisms that sustain cooperation (Tewolde et al., 2026). Newer environments add private objectives and non-binding agreements (O'Neill et al., 2026), firms with different roles in an economy (Sugiura et al., 2026), and open-ended survival tasks (Tessera et al., 2026). Failure analyses of multi-agent systems find that many errors come from the interaction between agents, such as role violations and misaligned goals, rather than from single agents (Cemri et al., 2025).

A.2 Methods that change coordination

Several methods improve coordination by changing what agents do or say. Wang et al. compare collaboration strategies such as governance, participation, and interaction order, and show that these choices affect both accuracy and efficiency (Wang et al., 2025). MAPoRL trains several LLMs together with multi-agent reinforcement learning so that they learn to collaborate (Park et al., 2025). Prompting for the consideration of future consequences asks agents to weigh later outcomes before they act in social dilemmas (Nguyen et al., 2025). These methods set the coordination rules before execution. SCTL keeps the native rules and actions and adds a ledger that is recomputed from the observed state. Our CFC-Prompt control tests whether a plain instruction to consider the future is enough.

A.3 Credit assignment and coalition values

The Shapley value divides the value of a cooperative game among players by their average marginal contribution (Shapley, 1953). Exact computation needs every coalition, so larger games use sampling estimates (Castro et al., 2009), as SCTL does for more than three active agents. In multi-agent reinforcement learning, COMA assigns credit with a counterfactual baseline that averages over one agent's actions while the others stay fixed (Foerster et al., 2018), and Shapley Q-value turns a global reward into local rewards by marginal contribution (Wang et al., 2020). For LLM agents, Shapley-Coop uses marginal-contribution reasoning and pricing to coordinate self-interested agents (Hua et al., 2025), and ShapleyFlow evaluates coalitions of workflow upgrades and splits their gains with Shapley values (Yang et al., 2026). To value a coalition when the other agents react to it, our controlled games follow the construction in which the coalition acts jointly and every other agent acts alone (Chander and Tulkens, 1997). These works decide how much each agent contributes or receives. SCTL applies the same idea at every checkpoint and adds the question of when each share is paid.

A.4 Agreements, contracts, and incentives

A second line of work changes incentives through agreements. CoopEval benchmarks mechanisms that sustain cooperation among LLM agents in social dilemmas, including repetition, reputation, mediation, and contracts (Tewolde et al., 2026). Sajja et al. evaluate whether language models contract rationally in natural language (Sajja et al., 2026), and Wyse et al. study commitment to cooperation through self-negotiated contracts (Wyse et al., 2026). In reinforcement learning, agents can learn to reward other agents (Yang et al., 2020) or agree to formal contracts that change payoffs (Christoffersen et al., 2023). These mechanisms create or enforce an agreement. In the LLM benchmarks, SCTL makes no binding transfer and gives the acting model only an informational ledger.

A.5 Dynamic cooperative games and time consistency

Dynamic games study agents whose actions change a shared state over time. In the classical fish war, independent fishing leads to more intensive harvesting than cooperation does (Levhari and Mirman, 1980). Cooperative versions of such games ask how an agreed allocation should be paid out over time. Petrosyan introduced time consistency, which requires the unpaid part of the allocation to match what the same solution concept gives in every later subgame along the cooperative path, together with the imputation distribution procedure that achieves it (Petrosyan, 1977). Yeung and Petrosyan extend the idea to subgame consistency (Yeung and Petrosyan, 2012) and give a full treatment of subgame-consistent cooperation (Yeung and Petrosyan, 2016). Germain et al. design transfers that keep a dynamic core allocation stable in a stock pollution game (Germain et al., 2003). Parilina and Zaccour study the consistency of the core in games played over event trees (Parilina and Zaccour, 2015) and study payment schemes that sustain cooperation (Parilina and Zaccour, 2022). Petrosyan et al. study the convergence of strong time-consistent payment schemes (Petrosyan et al., 2017). Jørgensen et al. separate time consistency from agreeability, the requirement that cooperation stays individually acceptable at every intermediate state (Jørgensen et al., 2003). Experiments with humans show that cooperation in repeated games depends on time preferences (Kim, 2023), which supports studying payment timing as a behavioral variable.

A.6 Lookahead and temporal credit

Inference-time search spends extra model calls on candidate futures. Reasoning via Planning uses the LLM as a world model and plans with Monte Carlo tree search (Hao et al., 2023), and Language Agent Tree Search combines tree search with reasoning and acting (Zhou et al., 2024). Our ActionLookahead control spends the SCTL rollout budget in this way, by searching over native actions. Reinforcement learning redistributes credit over time with potential-based reward shaping, which keeps the optimal policy unchanged (Ng et al., 1999), and with return decomposition for delayed rewards (Arjona-Medina et al., 2019). These methods change what an agent learns or which action it picks. SCTL changes only the information the agent sees about its share, and it uses no learned reward.

A.7 What SCTL takes from classical theory and what is new

SCTL takes two things from cooperative game theory. It uses allocation rules such as the Shapley value to split the team's cost, and it keeps the accounting identity of the imputation distribution procedure, in which the charges and the remaining balance add up to the initial share. Classical time-consistent schedules define each payment along a known cooperative trajectory, for a known game and horizon, with players who never leave the trajectory. SCTL is built for LLM teams, where none of these assumptions holds. Its payment is a function of the time and of the state that the team has reached, so it is defined off the cooperative trajectory as well. It keeps the unpaid balance of an agent that deviates as a penalty and charges it after the agent returns, a payment that classical schedules do not define. It uses a forecast of the final step that moves with every checkpoint, estimates coalition values by rollouts when no model exists, shows the ledger in the prompt of each agent, and recomputes the shares after an agent stops responding. When no agent deviates and the horizon is known, SCTL reduces to the classical schedule. In the main paper we use the term temporal ledger for the object shown to the LLM, because LLM benchmarks receive an informational signal and no money is transferred.

B Notation

In the appendix we write $B_i(t)$, $K_i(t)$, $M_i(t)$, $\varphi_i(t)$, and $v_t^{\mathrm{cost}}(S)$ without the state argument when the state is the one reached on the realized path, so $B_i(t)$ stands for $B_i(t, s_t)$. The LLM benchmark ledger is indexed by the checkpoint order k, with τ_k the native step of the kth checkpoint.

Table 4: **Notation used in the paper, in order of first use.**

Symbol	Type	Meaning
N	set	agents in the team
t, $s_t \in \mathcal{S}$	step, state	native step of the task and task state at that step
$o_{i,t} = \chi_i(s_t)$	text	observation of agent i, which is the text of its prompt
$a_{i,t} \in \mathcal{U}_i$, a_t	actions	action of agent i from its action set, and the joint action
$\mathcal{P}$	kernel	transition probability of the task state
c_i	scalar	cost that agent i bears at one step
A_t, A_k	sets	agents that follow the plan at step t and at checkpoint k
T, $\hat{T}_t$, H	steps	announced final step, forecast of the final step at step t, and window length
$\mathcal{T}$	set	native steps at which the ledger is updated (checkpoints)
τ_k, s_k	step, state	native step of the kth checkpoint and the state there
ρ	scalar	discount rate
J_i, J, V^C	scalars	expected discounted cost of agent i and of the team, and the cooperative cost, the lowest team cost
π^C, π^C_S	policies	cooperative plan, and the policy with which the members of S minimize their total cost
s^*_t	state	state that the cooperative plan reaches at step t when no agent deviates
π^F_i	policy	fallback policy of agent i
S	set	coalition, a group of agents that keep to the plan
$v^{\text{cost}}_t(S \mid s_t)$	scalar	value of coalition S at step t in state s_t, the cost that its members still bear
$\pi^{\text{out}}_{A_t \setminus S}$	policies	policies of the agents outside coalition S
$\pi^{\text{FN}}_{A_t \setminus S \mid S}$	policies	feedback Nash policies of the agents outside S on the controlled benchmark
ℓ_b, $\zeta_{\tau_k:}$	loss, run	benchmark loss and the rest of a run from checkpoint k
$L_t(S \mid s_t)$, $w_t(S \mid s_t)$	scalars	expected benchmark loss when S cooperates, and the loss this avoids
$\Phi_i(v)$	map	cost share of agent i under an allocation rule
$\varphi_i(t, s_t)$	scalar	share of agent i at step t in state s_t, discounted to step 0
Ξ_i	scalar	initial share $\varphi_i(0, s_0)$
$B_i(t, s_t)$, $\beta_i(t)$, $P_i(t)$	scalars	remaining balance, charge at step t, and amount charged before step t
O_i	scalar	penalty, the unpaid balance kept when agent i deviates and paid to the team after it returns
$B^U_i(t)$, $B^D_i(t)$	scalars	balance under UniformSchedule and under Delayed
R	integer	rollouts per coalition value
x_t, u^i_t	scalars	backlog and effort of agent i on the controlled benchmark
γ_i, ω_i, e_i, λ	scalars	effort weight, backlog weight, arriving work, and decay rate
G_{cost}	scalar	normalized gap in realized team cost, not clipped to $[0, 1]$
$K_i(t, s_t)$	scalar	deviation cost, what agent i pays from step t if it deviates
$M_i(t, s_t)$	scalar	participation margin $K_i(t, s_t) - B_i(t, s_t)$
t^{cross}_i, t^{cross}	steps	first step with a negative margin, for agent i and for the team
$B^\theta_i(t, s_t)$, θ	scalars	balance under delay θ, 0 for SCTL and 1 for Delayed (Appendix K)
$W^\theta_i(t)$	scalar	lowest remaining cost that a forward-looking agent can secure under schedule θ
$C_i(k)$, $\delta_i(k)$, $Q_i(k)$	scalars	remaining credit, credit update, and credit realized before checkpoint k on the LLM benchmarks
d_k, Υ_i	scalars	benchmark discount through checkpoint k and initial credit account $C_i(0)$
H_{cp}	integer	number of scheduled checkpoints on an LLM benchmark
D_{rt}, Z_{rt}	binary	first-deviation indicator and boundary-crossing indicator for run r at checkpoint t
$\mathcal{D}_t$, A^{ret}_t	sets	agents that stop responding at step t, and agents that return at step t

C Controlled environments and allocation rules

C.1 Dynamic game solved by the environment

The controlled benchmark models a shared queue of work, where each agent pays for its own effort while the benefit reaches the whole team, as the shared resource in GovSim does for LLM agents.

Agent i reduces the shared backlog $x_t \geq 0$ with effort $u_t^i \geq 0$ and pays for its effort and for the backlog that remains,

$$c_i(u_t^i, x_t) = \tfrac{\gamma_i}{2}(u_t^i)^2 + \tfrac{\omega_i}{2}x_t^2, \qquad x_{t+1} = (1-\lambda)x_t + \sum_{j \in N}(e_j - u_t^j), \tag{7}$$

where γ_i and ω_i weight effort and backlog, e_j is the work that arrives for agent j at each step, and $\lambda \in (0, 1)$ is the rate at which the backlog decays. The backlog equation has the classic form of a dynamic game over a shared stock (Levhari and Mirman, 1980). The six games G1–G6 differ in the shape of the backlog cost, the team size, and bounds on the effort, and each has a known number of steps T and a discount rate $\rho = 0.05$. We use Equation (7) as the primary controlled environment. For every agent subset S, the environment solves the subset-continuation problem in which S acts jointly and agents in $N \setminus S$ act independently (Chander and Tulkens, 1997). The resulting groups play a feedback Nash equilibrium, in which each group selects its control as a function of the current state. The independent-agent configuration is also the fallback policy triggered by an endogenous EXIT in the identification experiment. Thus, $K_i(t)$ and the EXIT consequence use the same continuation semantics. In the controlled recovery arm, the active set A_t coordinates while every agent in $N \setminus A_t$ follows its independent feedback policy. The state equation continues to sum over the original N. Active agents use the recomputed joint-plan controls, and agents outside the active team use their independent controls. If only one active agent remains, that agent follows the independent feedback policy until the fixed horizon.

For G1, the state cost is quadratic as in Equation (7). G2 and G3 keep the same effort term and dynamics but use the linear state cost $\omega_i x_t$. Their optimal task controls are state-independent conditional on the remaining horizon. G2 and G3 provide transfer tests, while G1 provides the state-feedback example. G3 has five agents with distinct cost and workload parameters. These three environments use the interior solver, and every joint, independent-agent, and subset-continuation pass satisfies $\max_{i,t} u_t^i/e_i \leq 0.99$.

G4–G6 are bounded-control stress families in the same workload class. They impose $0 \leq u_t^i \leq e_i$ and use a deterministic active-set fixed-point solver. At each time step and represented state, each coalition group solves its convex box-constrained best response while the other groups are held fixed. Best-response updates continue until the maximum control change and first-order projected residual are both below 10^{-8}, with a limit of 500 sweeps. A task instance is admitted only when every represented coalition continuation converges under the same deterministic initialization and selection rule. Multiple fixed points are not ruled out by the theory, so the implementation records the selected fixed point and its residual. G4–G6 are used as bounded-control stress tests of the same protocol.

Table 5: **Primitive parameters of the three base controlled games.** Vector entries follow agent order. G2 and G3 use the linear state cost described above. Together with the time-0 present-value convention and Equations (7)–(19), these values are sufficient to recompute the deterministic base-game quantities below.

Quantity	G1	G2	G3
n	3	3	5
T	24	24	24
γ	$(1.49, 2.22, 2.15)$	$(1.22, 2.27, 1.61)$	$(3.10, 1.77, 3.75, 1.89, 3.01)$
ω	$(.017, .016, .024)$	$(.136, .111, .116)$	$(.141, .137, .141, .118, .117)$
e	$(1.73, 1.26, 1.00)$	$(1.34, .81, 1.00)$	$(1.03, 2.13, 1.76, 1.64, 1.31)$
λ	0.15	0.18	0.17
x_0	0.90	1.10	1.50
ρ	0.05	0.05	0.05
$v_0^{\text{cost}}(N)$	47.7924	37.2673	195.5965
$t_i^{\text{cross}}(1)$	$(7, 7, 6)$	$(4, 4, 4)$	$(4, 5, 4, 5, 5)$
min SCTL relative slack	0.0776	0.0490	0.0388
max all-pass u_t^i/e_i	0.9042	0.9770	0.9548

Table 6: **Parameter centers for bounded-control stress families G4–G6.** The action set is $0 \le u_t^i \le e_i$, and these centers define the fixed generator around which the stress-test bank is constructed.

Quantity	G4	G5	G6
n	4	4	6
T	28	28	32
γ	$(1.12, 1.86, 2.73, 3.36)$	$(1.31, 2.57, 1.78, 3.09)$	$(2.58, 1.54, 3.37, 2.09, 2.91, 1.83)$
ω	$(.015, .021, .013, .028)$	$(.108, .149, .121, .162)$	$(.128, .114, .151, .119, .145, .132)$
e	$(1.56, 1.21, .96, 1.82)$	$(1.24, .93, 1.47, .84)$	$(1.11, 1.53, .97, 1.68, 1.27, 1.39)$
λ	0.16	0.19	0.17
x_0	1.25	1.35	1.80
ρ	0.05	0.05	0.05

We report the base-game Shapley shares as

$$\Xi^{G1} = (13.6908, 12.2636, 21.8381), \qquad \Xi^{G2} = (14.6896, 10.9270, 11.6507),$$

$$\Xi^{G3} = (43.1874, 40.6744, 43.4160, 33.6964, 34.6222).$$

G2 has a three-way earliest-crossing tie and G3 has a two-way tie, so agent identity is never a confirmatory endpoint.

C.2 Published subset values

Table 7 gives every nonempty subset continuation value for G1 and G2. Table 8 gives the full G3 list. The stated environment parameters determine these design values.

Table 7: **All nonempty subset values for G1 and G2.** Subset labels use one-based agent indices.

S	G1	G2	S	G1	G2
$\{1\}$	20.7690	18.9642	$\{1,2\}$	37.3259	31.6307
$\{2\}$	18.2119	14.7029	$\{1,3\}$	45.4903	32.1161
$\{3\}$	29.1965	15.6649	$\{2,3\}$	45.1931	28.8522
$\{1,2,3\}$	47.7924	37.2673			

Table 8: **All nonempty subset continuation values for G3.**

S	$v_0^{\mathrm{cost}}(S)$	S	$v_0^{\mathrm{cost}}(S)$	S	$v_0^{\mathrm{cost}}(S)$
1	56.4240	2	55.3495	3	56.2902
4	47.4607	5	46.7268	12	109.6932
13	111.3044	14	102.0784	15	101.8245
23	109.6857	24	100.6685	25	100.3952
34	102.0383	35	101.7829	45	92.7592
123	157.1769	124	147.2286	125	148.3805
134	150.3549	135	151.5416	145	141.5477
234	147.5323	235	148.6808	245	139.0749
345	141.7855	1234	182.3486	1235	185.8843
1245	174.7736	1345	180.3436	2345	175.5709
12345	195.5965				

C.3 Allocation formulas and admissibility

The proportional rule gives $\Phi_i(v) = v(\{i\})\, v(A) / \sum_{j\in A} v(\{j\})$, and the equal-savings rule gives $\Phi_i(v) = v(\{i\}) - \Delta/|A|$ with $\Delta = \sum_{j\in A} v(\{j\}) - v(A)$.

For a controlled cost game v, the Shapley cost share (Shapley, 1953) is

$$\Phi_i^{\mathrm{Sh}}(v) = \sum_{S \subseteq A\setminus\{i\}} \frac{|S|!(|A| - |S| - 1)!}{|A|!} \big[v(S \cup \{i\}) - v(S)\big], \tag{8}$$

where A is the current active agent set. The proportional rule is admitted only when $\sum_{j\in A} v(\{j\}) > 0$ and assigns

$$\Phi_i^{\mathrm{Prop}}(v) = \frac{v(\{i\})}{\sum_{j\in A} v(\{j\})} v(A). \tag{9}$$

The equal-savings (ES) rule divides $\Delta_A = \sum_{j\in A} v(\{j\}) - v(A)$ equally,

$$\Phi_i^{\mathrm{ES}}(v) = v(\{i\}) - \frac{\Delta_A}{|A|}. \tag{10}$$

For an LLM benchmark savings game w, the same formulas are applied in benefit orientation. Shapley allocates marginal loss reduction, proportional allocation weights positive singleton savings, and the equal-savings analogue distributes the grand-coalition synergy $w(A) - \sum_i w(\{i\})$ equally. The resulting quantities are coordination-credit shares and are never interpreted as additive agent costs.

For the admission statistic, define $\mathcal{T}_{\mathrm{slack}} = \{0, \ldots, T-4\}$. The truncation isolates the non-degenerate continuation region. At $T-1$ the one-step singleton and cooperative continuations coincide structurally, forcing relative slack to zero, and by $T-3$ the shorter-horizon base games already enter this terminal-collapse regime. The index $T-4$ is therefore fixed by the finite-horizon construction before any LLM outcome is observed. The final three steps remain in the behavioral at-risk set and are excluded only from the normalized-slack admission statistic. Controlled rule-instance pairs in the behavioral-identification bank must satisfy share efficiency, SCTL continuation individual rationality, group feasibility for every represented subset, the control-bound margin, non-negative SCTL charges in charge-only conditions, and the normalized SCTL-safety condition below. The subset-continuation construction can violate group feasibility for some parameter draws. We record each such failure as a rejection reason.

Table 9: **Allocation-rule checks on the three base games.** Group slack is $\min_{\emptyset\neq S\subsetneq N}[v_0^{\mathrm{cost}}(S) - \sum_{i\in S}\Xi_i]$. Min charge is $\min_{i,0\le t<T}\beta_i(t)$. Relative SCTL slack is $\min_{i,t\in\mathcal{T}_{\mathrm{slack}}}(K_i(t) - B_i(t))/K_i(t)$. The final three steps are excluded only from this admission statistic because finite-horizon deviation cost mechanically collapses near the endpoint.

Game	Rule	$t_i^{\mathrm{cross}}(1)$	group slack	min charge	rel. SCTL slack
G1	Shapley	$(7,7,6)$	5.9483	0.2895	0.0776
G1	Proportional	$(6,6,6)$	5.4453	0.2908	0.0921
G1	Equal-savings	$(7,8,5)$	6.7950	0.2882	0.0727
G2	Shapley	$(4,4,4)$	3.7759	0.2337	0.0490
G2	Proportional	$(4,4,4)$	3.5958	0.2342	0.0521
G2	Equal-savings	$(3,4,4)$	4.0216	0.2326	0.0461
G3	Shapley	$(4,5,4,5,5)$	12.1046	0.8944	0.0388
G3	Proportional	$(5,5,5,5,5)$	11.8762	0.8946	0.0433
G3	Equal-savings	$(4,5,4,5,5)$	13.3309	0.8909	0.0402

The published subset values permit independent recomputation of the allocation shares and initial group slack.

Table 10: **Allocation-rule comparison.** Controlled rows use the unfiltered performance bank. LLM benchmark rows report each task's native metric. The final column gives continuation-value evaluations before caching in controlled tasks. CoffeeBench has one controllable agent, so all efficient allocation rules coincide there.

Task	Shapley	Proportional	Equal-savings	Value evaluations
G1	0.181	0.186	0.183	7/4/4
G2	0.221	0.217	0.226	7/4/4
G3	0.247	0.253	0.244	31/6/6
G4	0.213	0.217	0.211	15/5/5
G5	0.273	0.270	0.279	15/5/5
G6	0.258	0.263	0.256	63/7/7
GovSim	61.2	60.8	61.0	rollout-dependent
C2C	35.1	35.3	34.8	rollout-dependent
CoffeeBench	18.6	18.6	18.6	singleton-equivalent
Alem	72.8	72.3	72.6	rollout-dependent
Welfare Diplomacy	0.629	0.631	0.626	rollout-dependent
Melting Pot	63.1	62.7	63.2	rollout-dependent
MultiAgentBench	79.4	79.0	79.2	rollout-dependent
Collab-Overcooked	72.5	72.7	72.2	rollout-dependent
ParaCook	286.4	285.6	287.9	rollout-dependent
CoopEval	0.788	0.784	0.791	scripted game-table lookup

C.4 Deterministic admissibility checks

Every controlled proposal first passes solver, feasibility, allocation-efficiency, and group-feasibility checks. The behavioral-identification bank then applies the SCTL-safety rule of Appendix N.10. Along the SCTL path without noise, $B_i(t) \leq K_i(t)$ for every agent at every before the last step state $t < T$, and normalized slack is at least 0.03 on $\mathcal{T}_{\text{slack}} = \{0, \ldots, T-4\}$. This admission filter fixes the safe reference schedule before model execution. UniformSchedule, Delayed, and intermediate schedules receive no analogous crossing filter, and all checkpoints, including the final three, remain eligible behavioral risk rows.

The task-performance bank is generated independently with a second fixed seed and applies the same solver and allocation checks without the SCTL-safety rule. Table 1 Panel B and every controlled task-performance comparison use this unfiltered bank. Every stored crossing path is checked by direct evaluation of $B_i^{\theta}(t) > K_i(t)$ at every before the last step reference-path step, independently of any behavioral deviation.

D Task-instance bank

Two banks of instances with separate random seeds keep the deviation experiments apart from the evaluation of task performance. The identification bank, used for the deviation experiments, applies the SCTL-safety rule, because these experiments move the participation boundary with the schedule and measure where agents deviate. The performance bank applies only solver, feasibility, and allocation checks. Each bank keeps 20 instances per game, so each bank has 120 instances before they are combined with schedules and models.

Generator specification. Table 11 records the generator specification used for the primary G1–G6 banks. Each positive coordinate is perturbed independently around the family center before the solver and feasibility checks are applied. The values n, T, and ρ remain fixed, and the identification and performance streams use separate fixed seeds.

Table 11: **Generator specification for the primary G1–G6 banks.** Multipliers apply coordinate-wise to the base values in Tables 5–6.

Primitive	Proposal multiplier	Identification seed	Performance seed
γ_i, ω_i, e_i	$U[0.85, 1.15]$	14011	29017
λ, x_0	$U[0.90, 1.10]$	shared	shared
n, T, ρ	fixed at family center	shared	shared

The proposal rule spans the variation represented by the retained banks. Each proposed and retained instance records the primitive parameters, proposal index, bank seed, solver status, feasibility status, SCTL-safety status, and retained/rejected disposition. These records support Equation (18) on the performance bank and the rejection accounting in Table 12.

Table 12: **Behavioral-identification bank acceptance and rejection accounting.** Counts are fixed before LLM outcomes are inspected. The SCTL-safety rejection column makes the identification filter explicit.

	G1	G2	G3	G4	G5	G6
Proposals	21	46	39	31	36	44
Accepted	20	20	20	20	20	20
Rejected: solver / feasibility	0	8	7	5	7	10
Rejected: SCTL safety	1	18	12	6	9	14

The performance bank retains the first 20 proposals from its independent stream in each family that pass the solver, feasibility, and allocation checks, and it does not apply the SCTL-safety rejection. The controlled performance mean in Table 1 is therefore a property of the unfiltered bank, while the confirmatory first-deviation association models, reported through odds ratios, use only the identification bank. Inference resamples instances as the top-level unit, so repeated model-condition cells on one parameterization are not treated as independent tasks.

Because the performance bank is unfiltered, SCTL safety serves only as a diagnostic on this bank. For each instance and schedule we record the first step before the end at which $B_i^\theta(t) > K_i(t)$ holds for some agent on the full path without noise. This count does not depend on whether an LLM deviates earlier. SCTL crosses in 1/20, 9/20, 8/20, 5/20, 6/20, and 8/20 retained performance instances for G1–G6, respectively (37/120 overall). UniformSchedule crosses in 58/120 instances, with family counts 5/20, 12/20, 11/20, 8/20, 10/20, and 12/20. Under the same full-horizon definition, Delayed crosses in 120/120 instances, with 20/20 in each family. These counts differ from Table 33, which splits runs by whether the observed first deviation happened before or after the crossing.

Table 13: **Behavioral-identification-bank retained crossing histogram.** Bins use the Delayed participation crossing $t^{\text{cross}}(1)$.

Game	$t^{\text{cross}} \le 3$	$t^{\text{cross}} = 4$–5	$t^{\text{cross}} = 6$–7	$t^{\text{cross}} \ge 8$
G1	0	2	13	5
G2	3	13	4	0
G3	2	14	4	0
G4	1	5	9	5
G5	3	7	7	3
G6	2	6	8	4

Episodes sample retained instances uniformly. Inference resamples task instances as the top-level unit, so repeated LLM calls on one parameterization are not treated as independent task instances.

D.1 Stochastic-coupled structural holdout

The primary G1–G6 bank preserves one workload-control family so the schedule manipulation can be checked against exact continuation values. We therefore add a secondary structural holdout that is

generated independently and excluded from the primary bank, generator tuning, and confirmatory families. The holdout has four agents and horizon $T = 18$. Its scalar task state follows

$$x_{t+1} = 0.80x_t - \sum_{i=1}^{4} h_i u_t^i + \varepsilon_t^{\text{env}}, \qquad \varepsilon_t^{\text{env}} \in \{-0.04, 0, 0.04\}, \tag{11}$$

where $h_i > 0$ is the control effectiveness of agent i, so positive control reduces the unresolved state, and $\varepsilon_t^{\text{env}}$ is an independent discrete environment shock with probabilities $(0.25, 0.50, 0.25)$. Agent i has stage cost

$$c_i^{\text{hold}}(u_t, x_t) = \frac{\gamma_i}{2}(u_t^i)^2 + \nu_i x_t^2 + \kappa_i \sum_{j \neq i} (u_t^i - u_t^j)^2. \tag{12}$$

Here c_i^{hold} is the holdout-specific stage cost, ν_i is defined as the full coefficient on x_t^2 (unlike the half-coefficient convention used for ω_i in Equation (7)), and κ_i weights pairwise control mismatch. The pairwise term changes the strategic structure because an agent's cost depends directly on other agents' controls. Expected joint, singleton, and subset continuation values are computed exactly by finite-horizon quadratic recursion with the discrete shock expectation evaluated at every step. The pre-specified center is $\gamma = (1.40, 1.80, 2.20, 1.60)$, $\nu = (.150, .120, .180, .140)$, $h = (.120, .150, .100, .130)$, $\kappa = (.030, .040, .025, .035)$, and $x_0 = 1.30$. The generator perturbs these quantities within $\pm 15\%$ and retains proposals whose nominal initial cooperative controls stay below 0.24, whose time-zero Shapley allocation has positive singleton slack, and whose normalization is positive. It does not impose the SCTL participation-safety filter used by the G1–G6 identification bank. The holdout retains 20 of 24 proposals, and the paired episode records use the same deterministic generator and seed rules.

Table 14: **Secondary structural holdout on its separately generated 20-instance holdout bank, with stochastic transitions and pairwise action coupling.** Lower G_{cost} is better. The pooled K-Hidden odds ratio uses the same step-level crossing indicator as the primary identification analysis. The odds ratio is estimated only by the pooled crossing model, which has no single-condition G_{cost}.

Condition	$G_{\text{cost}} \downarrow$	first-deviation odds ratio
NoMechanism	0.657	–
UniformSchedule	0.515	–
Delayed	0.730	–
Frozen-SCTL	0.376	–
SCTL	0.311	–
Pooled K-Hidden crossing model	–	1.44× [1.18, 1.92]

The paired SCTL advantage over Frozen-SCTL is $+0.065$ [0.035,0.100] in oriented G_{cost}, directly testing realized-state recomputation when stochastic transitions move the execution path away from its initial prediction.

E Experimental conditions and ablations

E.1 Named experimental arms

The following names are used consistently across prompts, tables, and analyses.

- **NoMechanism.** In the controlled deviation experiments, the model receives the task state, active team, fallback rule, and STAY/EXIT schema but no Ξ_i, $P_i(t)$, $B_i(t)$, or $K_i(t)$ ledger field. The EXIT consequence is the same independent fallback used in the other identification arms. LLM benchmark tasks retain their native action interface.
- **UniformSchedule.** The same assigned total Ξ_i is charged uniformly over the finite horizon. Its own $B_i(t) > K_i(t)$ path is evaluated directly.
- **CFC-Prompt.** The condition uses the same task and budget with the future-consequence prompt adapted from CFC. Schedule quantities are added only when the original task exposes them.
- **ComputeMatched.** This LLM benchmark control uses the NoMechanism interface and receives the same continuation-rollout call budget as the matched coordination arm. The extra

rollouts are discarded and do not create allocation or ledger fields. CoopEval has no corresponding row because its mechanism-comparison wrapper uses no continuation estimator.

- **SameInfoSummary.** This representation control receives the same first-checkpoint assigned account, net realized credit, remaining coordination credit, and latest update as SCTL, rendered as a concise natural-language summary. It receives no structured ledger labels or table-like formatting, and it uses the same continuation-rollout estimates, checkpoint timing, and native task interface.
- **TeamValueOnly.** This matched-budget LLM benchmark control uses the same checkpoint policy and continuation-rollout estimator as SCTL but shows only the scalar team continuation value $w_k(A_k) = L_k(\varnothing) - L_k(A_k)$. It exposes no per-agent allocation, first-checkpoint account, realized credit, remaining credit, or agent-specific update. The scalar is inserted as one labeled natural-language sentence at the same checkpoint location used by the SCTL suffix.
- **StructuredTeamValue.** This format-matched control carries the same scalar team continuation account through assigned, realized, remaining, and latest-update fields with the same checkpoint timing as SCTL. It does not allocate the account across agents. When $|A_k| = 1$, allocation is degenerate, so StructuredTeamValue uses the same singleton account fields as SCTL and the two prompts coincide exactly. This removes an otherwise irrelevant label-only formatting difference in singleton adapters.
- **EqualSplitLedger.** This control keeps SCTL's per-agent assigned/realized/remaining fields and state updates but replaces the Shapley allocation with the equal split $w_k(A_k)/|A_k|$.
- Delayed**.** This condition uses the delayed family in Equation (16). The Delayed condition uses $\theta = 1$.
- **SCTL.** In controlled identification this condition uses $\theta = 0$. In LLM benchmarks it recomputes the savings game and coordination-credit ledger at every scheduled checkpoint while preserving the native action interface.
- **K-Visible.** The environment-computed $K_i(t)$ is shown next to the temporal ledger for the SCTL or UniformSchedule presentation comparison. K-Visible is the manipulation check.
- **NoRecovery.** After an injected loss of an agent, the survivors continue under the benchmark's default fallback without recomputing the joint allocation.
- **Reallocate.** After the same injected loss of an agent, the environment first advances to s_{t+1} with surviving team A_{t+1}. Settled transfers through step t remain fixed, the remaining balances of departed agents are written off, and the survivors recompute the continuation plan, subset values, allocation, and temporal schedule from s_{t+1}. Each survivor receives $\Xi_i^{(t+1)} = P_i(t+1) + (1+\rho)^{-(t+1)}\Phi_i(v_{t+1}^{\text{cost}})$ before the next participation check.

E.2 Schedule and control conditions

The primary schedule-identification matrix contains SCTL, Delayed, and three fixed intermediate values of θ, $\theta \in \{0.25, 0.50, 0.75\}$. Intermediate delays are analyzed with the exact step-level indicator $B_i^{\theta}(t) > K_i(t)$. The five protocol ablations use the fully delayed schedule, with 840 episodes per variant. K-Visible, no-remaining-balance, and free-form presentation are also evaluated for SCTL and UniformSchedule on the same 840-episode-per-schedule bank, producing matched three-schedule comparisons. These rows are descriptive presentation controls and do not enter the primary multi-schedule regression. The pooled primary OR of $1.52\times$ remains the confirmatory identification result.

E.3 Matched-B fallback-shift experiment

This experiment changes the announced independent exit cost while holding the cooperative balance path fixed. For an original fallback continuation cost $K_i(t)$, arm q uses $K_i^{(q)}(t) = qK_i(t)$. The acting prompt displays the current scalar $K_i^{(q)}(t)$ as the announced cost of deviating at that checkpoint. The multiplier q is an internal arm label and is not shown to the model. The intervention does not alter the cooperative dynamics, joint controls, allocation Ξ_i, transfer schedule, transfer history, or $B_i(t)$ path before the decision. Direct announced-cost and cost-by-time effects are addressed through placebo and time-sensitivity analyses.

The multiplier is selected deterministically before any LLM call. For every retained task instance under the fixed fully-delayed identification schedule used by this matched-B experiment, the environment evaluates the grid

$$\mathcal{Q}_{\text{mult}} = \{0.70, 0.75, \ldots, 1.25, 1.30\}.$$

Let t_q^{cross} be the team participation crossing under $K_i^{(q)}(t)$. The base arm uses $q_0 = 1$. Among values below one, q_- is the value closest to one for which $t_{q_-}^{\text{cross}} \leq t_{q_0}^{\text{cross}} - 2$. Among values above one, q_+ is the value closest to one for which $t_{q_+}^{\text{cross}} \geq t_{q_0}^{\text{cross}} + 2$ or no analyzed crossing occurs. Instances without both shifts are excluded from this experiment only and remain in the main controlled bank. This rule changes the crossing without selecting on an LLM outcome.

Each admitted instance is evaluated as a matched triplet $(q_-, 1, q_+)$ for every acting model. This matched-B experiment uses exactly one pre-specified charge schedule, the fully-delayed identification schedule. The main multi-schedule hazard experiment remains separate. Episode seeds, task state, schedule, allocation, and prompt fields are identical across the triplet except for the announced exit-cost scalar. If one agent deviates, later steps from that member are no longer at risk. The primary matched analysis therefore uses only common at-risk strata that exist for all three members of the triplet at a given step. A secondary full-risk-set analysis uses all available at-risk rows, multiplier-arm intercepts, a shared normalized-time spline, and model fixed effects. Under the fully delayed schedule, $B_i(t)/\Xi_i = 1$ before terminal settlement, so the normalized remaining balance $\max_i B_i(t)/\Xi_i$ is constant and does not enter this sensitivity fit. The secondary full-risk-set analysis gives OR $1.35\times$ [1.13, 1.66].

The Matched-B template adds one runtime field, `announced_exit_cost`, which displays $K_i^{(q)}(t)$ at the current checkpoint. The internal multiplier used to construct this scalar is not displayed.

E.4 LLM BENCHMARK TRACE PARTICIPATION EXTRACTION

GovSim and Collab-Overcooked retain their native action interfaces. The trace analysis therefore uses plan adherence without changing what any acting model sees. At each observed benchmark state, a frozen evaluation-only planner generates a common reference plan $\mathcal{R}_t^{\text{ref}}$ after the acting-model call. The same planner, inputs, and decoding are used for SCTL, UniformSchedule, Delayed, ComputeMatched, SameInfoSummary, NoMechanism, PlaceboLedger, and ShuffledLedger. Its output is never exposed to an acting model in any arm. The plan therefore serves only as a trace ruler and cannot provide an additional planning signal to SCTL.

The wrapper maps $\mathcal{R}_t^{\text{ref}}$ to an allowed native-action set $\mathcal{A}_{it}^{\text{plan}}$ for each active agent. If the planner emits a specific native action, the set contains that action. If it emits a role or subgoal, the benchmark adapter supplies the deterministic set of native actions consistent with that role or subgoal. This mapping is frozen before outcome analysis and stored with the trace. If no unambiguous mapping exists at a step, that step is marked non-scorable and omitted from the denominator.

For each scorable step,

$$I_t^{\text{plan}} = \mathbf{1}\Big[a_{it} \in \mathcal{A}_{it}^{\text{plan}} \; \forall i \in A_t\Big].$$

Here a_{it} is agent i's native action at step t. The primary trace outcomes are the adherence fraction $F_{\text{adh}} = \sum_t I_t^{\text{plan}} / T_{\text{obs}}$ and normalized time to first coordination deviation $T_{\text{persist}} = T_{\text{dev}}/T_{\text{obs}}$, where $T_{\text{dev}} = \inf\{t : I_t^{\text{plan}} = 0\}$ and T_{obs} is the number of scorable steps. An episode with no observed deviation is right-censored at T_{obs}. As a robustness check, a persistent deviation requires two consecutive scorable steps with $I_t^{\text{plan}} = 0$. A secondary recovery measure records whether I_t^{plan} returns to one within the next two scorable steps. On GovSim, the recovery rates are 42% for SCTL and 31% for UniformSchedule, a difference in percentage points (pp) of $+11$ pp [$+4$, $+18$]. On Collab-Overcooked, they are 39% and 30%, a difference of $+9$ pp [$+2$, $+16$].

The primary trace contrast compares SCTL with UniformSchedule within the same benchmark instance and acting model. SameInfoSummary is the primary representation contrast. ComputeMatched, NoMechanism, Delayed, PlaceboLedger, and ShuffledLedger are secondary descriptive controls. Confidence intervals resample benchmark instances. The trace-score association uses benchmark and model fixed effects and remains descriptive because adherence is a post-treatment variable. Causal mediation is outside this analysis.

PlaceboLedger control. To separate informative timing from the mere presence of four ledger fields, a secondary PlaceboLedger control uses the same prompt schema and a state-independent normalized decay curve fixed before evaluation. The curve is anchored to the initial allocated credit and to a pre-specified terminal fraction shared across instances. The curve never uses the realized or future SCTL endpoint. The placebo values are therefore available causally before the episode and are independent of benchmark state, native actions, and continuation estimates. The continuation-rollout budget is still executed and discarded, so compute matches the SCTL arm. On GovSim, PlaceboLedger gives survival 53.0, compared with 52.3 for NoMechanism and 61.2 for SCTL. On C-OC, Progress Completeness is 63.2, compared with 62.5 and 72.5. PlaceboLedger stays close to NoMechanism on the native metrics, while remaining below SCTL.

Same-information representation control. SameInfoSummary receives exactly the numerical ledger state shown by SCTL at each checkpoint, namely the first-checkpoint assigned account, net realized credit, remaining credit, and the latest realized update. The same quantities appear in the same order inside two short natural-language sentences, while SCTL uses labeled ledger fields. The acting model receives no additional plan text. This contrast isolates representation. Temporal ordering and state-conditioned updating are unchanged. SCTL has the higher point estimate on $7/10$ settings, while the paired interval excludes zero on $3/10$. Table 27 reports the per-task intervals.

E.5 Ablation set

The five protocol ablations hold the fully delayed schedule and the unfiltered bank fixed. K-Visible adds the fallback scalar to the K-Hidden prompt. The four other variants each change one part of the protocol.

- **No write-off after deviation**. The remaining balance remains payable after deviation. Under the retained non-negative charge conditions, EXIT then adds the independent deviation cost without canceling the remaining balance, making immediate deviation dominated by staying. This arm applies the penalty of Section 3.3 to permanent deviation. Its reported G_{cost} uses physical environment cost and excludes the settlement penalty. The crossing association is $1.05\times$, incidence of any deviation is 23.0%, and the physical environment cost is $G_{\text{cost}} = 0.166$.
- **Reverse transfer direction**. Scheduled charges become credits with unchanged magnitudes. Under the cost-side indicator this gives $Z_{rt} \equiv 0$ in the retained controlled games. incidence of any deviation is 27.0% and the physical environment cost is $G_{\text{cost}} = 0.184$. The intervention changes transfer semantics, so its role is directional stress testing while the primary schedule comparisons keep payoff semantics fixed.
- **No remaining-balance line**. $B_i(t)$ is omitted but remains recoverable from $\Xi_i - P_i(t)$. This variant tests whether the explicit balance line matters when the balance can still be computed.
- **Free-form decision protocol**. The structured STAY/EXIT response format is replaced by the benchmark-compatible free-form decision interface.

Table 15: **Matched presentation controls on the controlled performance bank.** Each cell is pooled $G_{\text{cost}} \downarrow$ across G1–G6 and all seven model slots. Presentation changes move absolute performance strongly, but SCTL remains below Uniform and Delayed in every matched row.

Presentation	SCTL	UniformSchedule	Delayed
K-Hidden (primary)	0.232	0.372	0.611
K-Visible	0.158	0.212	0.261
No remaining-balance line	0.181	0.235	0.301
Free-form response	0.170	0.223	0.286

Table 15 reports the format-matched SCTL/Uniform/Delayed comparisons. The No-write-off and Reverse-transfer interventions are not presentation controls because they change the economics of deviation. Table 3 summarizes the matched controls in the main paper. No post-loss reallocation belongs to the separate recovery study in Table 35.

The model panel is fixed before outcome analysis. Main comparisons and ablations use all seven configured model slots. The main matrix uses single-model teams, meaning every agent in one

episode is assigned the same model slot even though role prompts and observations may differ. The evaluation matrix covers all seven slots on all controlled games and LLM benchmark settings. Aggregate tables use a fixed macro-average across model slots, and model-specific SCTL-condition outcomes are reported separately as descriptive model variation. Appendix F.1 adds a smaller mixed-model-team stress test.

F MODELS AND INFERENCE SETTINGS

The evaluation configuration fixes seven acting-model slots before outcome analysis, including systems accessed through an application programming interface (API) and open-weight systems. Table 16 records the human-readable labels and configured aliases or revisions. Open-weight slots are pinned to repository revisions. Broader open-weight scale coverage remains outside the present design.

Table 16: **Model panel for the controlled and LLM benchmark matrices.** API rows show configured labels and aliases, while Qwen rows are pinned to the listed Hugging Face revisions. Main-matrix episodes use single-model teams. Appendix F.1 reports a separate mixed-model stress test.

Model	Provider / type	Configured alias / revision
GPT-6 Astra	OpenAI API	`gpt-6-astra`
GPT-5.6 Sol	OpenAI API	`gpt-5.6-sol`
Claude Fable 5.1	Anthropic API	`claude-fable-5-1`
Claude Opus 5	Anthropic API	`claude-opus-5`
Gemini 3.8 Flash	Google API	`gemini-3.8-flash`
Qwen3-8B	open-weight	`Qwen/Qwen3-8B@e8bbd825`
Qwen3-14B	open-weight	`Qwen/Qwen3-14B@ec3b5103`

Table 17: **Frozen harness-level inference settings.** These are fixed protocol settings. Unsupported provider-specific fields are omitted.

Setting	Controlled identification	LLM benchmark / recovery
Temperature	0.0 where supported	0.0 where supported
Top-p	1.0 where supported	1.0 where supported
OpenAI reasoning effort	`low`	`low`
Anthropic adaptive effort	`low`	`low`
Gemini 3.8 Flash thinking	`LOW`	`LOW`
Qwen sampling	`do_sample=false`	`do_sample=false`
Visible output cap	256 tokens	1024 tokens
Malformed output	one deterministic retry	one deterministic retry
Continuation rollouts	none	2 per required state
Rollout receding window	none	6 decisions
Episode-call concurrency	32 maximum	
Continuation-call concurrency	16 maximum	

The protocol assigns the same provider-compatible decoding configuration to all mechanism arms for a given model slot. The OpenAI-labeled slots configure `reasoning.effort=low`, the Anthropic-labeled slots configure adaptive thinking with effort `low`, the Gemini-labeled slot configures `thinking_level=LOW`, and the Qwen slots use deterministic decoding with thinking mode disabled. The controlled response parser accepts only `STAY` or `EXIT`, with one correction-only retry for malformed output. The protocol assigns independent streams to acting-model, environment, and continuation-rollout randomness. ComputeMatched isolates continuation rollouts from the acting prompt, the acting random number generator (RNG), environment state, and acting cache. Continuation-rollout seeds are deterministically derived from benchmark, task instance, model slot, condition, checkpoint state, coalition, and rollout index. The derivation is mirrored in `experiment_config.yaml`.

The complete Qwen revisions are `e8bbd8252970581ea5b08b6a5b3e668adaf3161a` for Qwen3-8B and `ec3b5103878832857b9e0f6ab32eb37ee5e28e9a` for Qwen3-14B. The statistical model treats model identity as a fixed effect. Primary tables report the pre-specified pooled estimate across the seven models. Table 18 reports SCTL-condition outcomes by model as descriptive model variation.

Table 18: **Model-specific SCTL outcomes.** Controlled performance uses the unfiltered performance bank. LLM benchmark columns retain each benchmark's native metric. The final row reproduces the macro-average used in Table 1.

Model	Controlled ↓	GovSim ↑	CoopEval ↑	Melt. Pot ↑	MAB ↑	C-OC ↑	ParaCook ↓
GPT-6 Astra	0.214	64.0	0.828	65.9	81.5	75.3	272.7
GPT-5.6 Sol	0.220	62.8	0.820	64.7	81.1	74.1	276.0
Claude Fable 5.1	0.215	63.1	0.844	65.1	82.4	74.6	268.1
Claude Opus 5	0.238	60.6	0.772	62.5	78.6	71.8	291.0
Gemini 3.8 Flash	0.229	61.8	0.795	63.4	79.8	72.9	283.7
Qwen3-8B	0.258	57.6	0.719	59.4	75.7	68.9	309.7
Qwen3-14B	0.250	58.5	0.737	60.6	76.7	69.8	303.7
Macro-average	0.232	61.2	0.788	63.1	79.4	72.5	286.4

F.1 MIXED-MODEL-TEAM STRESS TEST

The main matrix varies model identity across episodes but keeps all agents within an episode on the same model. To test whether SCTL depends on that single-model assumption, we add a pre-specified mixed-model stress test on GovSim, MultiAgentBench, Collab-Overcooked, and Welfare Diplomacy. The Frontier/API mix cycles GPT-6 Astra, Claude Fable 5.1, and Gemini 3.8 Flash across active agent positions. The Cross-scale mix cycles GPT-5.6 Sol, Qwen3-14B, and Qwen3-8B. For each task and composition we use ten fixed instances. The model-to-role assignment is cyclically rotated by instance so no model is permanently tied to one role. If a task has more active positions than models in the composition, the ordered cycle repeats. The continuation estimator uses the same model assignment as the acting trajectory for each coalition state.

Table 19 pools the two compositions after preserving within-instance pairing. Its scope is restricted to the two pre-specified model compositions and four tasks.

Table 19: **Mixed-model-team stress test pooled across two pre-specified model compositions.** Means are in benchmark-native units. Paired intervals compare SCTL with SameInfoSummary on the same task, composition, and instance.

Task	Native pre-spec.	Frozen-SCTL	SameInfo	SCTL	SCTL−SameInfo [95% CI]
GovSim ↑	56.8	58.2	58.8	59.7	+0.9 [0.5,1.3]
MultiAgentBench ↑	74.9	76.4	77.0	77.7	+0.66 [-0.01,1.29]
Collab-Overcooked ↑	68.8	69.8	70.4	70.5	+0.10 [-0.41,0.62]
Welfare Diplomacy ↑	0.590	0.602	0.608	0.614	+0.006 [0.003,0.009]

SCTL exceeds Frozen-SCTL and SameInfoSummary on all four pooled point estimates. With task instance as the top-level bootstrap unit, the SameInfo interval excludes zero on two of four tasks. These intervals are not directly comparable to the main single-model-panel intervals because each mixed-team contrast pools two fixed compositions with cyclic role rotation before instance-level resampling. Because SameInfo differs only in representation, the stress test supports robustness to mixed-model teams while leaving the incremental value of structured formatting modest and task-dependent. The stress test contains 320 design-specified episodes. Using the same per-task call and coalition-batch footprints as Tables 23–24, the estimated footprint adds $27,280$ acting calls, while 12,360 continuation batches and 24,720 rollout calls follow deterministically from the frozen estimator schedule. These are tracked separately from the main-suite totals.

F.2 Descriptive provider-family benchmark stratification

The seven-slot panel comes from four model families, with two OpenAI-labeled slots, two Anthropic-labeled slots, one Gemini-labeled slot, and two Qwen open-weight slots. We stratify the evaluation episodes by these labels and recompute the SCTL-versus-fixed-comparator effect within each task. This is a descriptive check of whether the sign of the effect is stable across model families. Family sizes are 2/2/1/2, so no family-level inferential claim is made. Each family preserves the same eight-positive/two-negative task pattern as the aggregate result, with CoffeeBench and CoopEval remaining the two negative tasks.

Table 20: **Descriptive model-family stratification of the seven-slot panel.** Panel A summarizes oriented relative gains versus the fixed comparator across all ten LLM benchmark settings. Panel B gives the corresponding family-by-task matrix. Family sizes are 2/2/1/2 and no family-level inference is performed.

Panel A. Family summary

Provider family	Models	SCTL better / 10	Median oriented gain
OpenAI	2	8/10	3.4%
Anthropic	2	8/10	3.4%
Gemini	1	8/10	3.5%
Qwen open-weight	2	8/10	3.8%

Panel B. Oriented relative effect by LLM benchmark setting (%)

Family	GovSim	C2C	Coffee	Alem	Welfare	Melt.	MAB	C-OC	ParaCook	CoopEval
OpenAI	+4.5	+5.3	−1.4	+3.5	+3.3	+4.0	+2.9	+3.4	+2.6	−1.8
Anthropic	+4.1	+4.9	−1.7	+3.6	+3.3	+4.2	+3.0	+3.5	+2.8	−2.1
Gemini	+4.4	+5.4	−1.8	+3.8	+3.6	+4.0	+3.0	+3.4	+2.9	−2.2
Qwen open-weight	+4.2	+5.0	−1.5	+4.0	+3.7	+4.2	+3.4	+3.9	+3.0	−1.9

The family-by-task matrix preserves the same eight-positive/two-negative task pattern as the aggregate result. The values are positive for GovSim, C2C, Alem, Welfare Diplomacy, Melting Pot, MultiAgentBench, Collab-Overcooked, and ParaCook in every configured family, and negative for CoffeeBench and CoopEval in every family. Weighting the four rows by their 2/2/1/2 model counts reproduces the aggregate task effects in Table 28 to one-decimal rounding. The single-slot Gemini row and small family sizes preclude a provider-family generalization claim.

F.3 Fixed-comparator composition stress test

To test whether the ledger can compose with the fixed comparator itself, we add SCTL to the pre-specified comparator mapping used in Table 1. The comparator identity is fixed before this composition and is not reselected from the composition outcomes. When the comparator is the same reproduced native workflow used by Native+SCTL, the composition is the same configuration and reuses that confirmatory arm. This applies to CoffeeBench, Melting Pot, and ParaCook. Duplicate reruns of the same configuration are excluded from the analysis because they do not define distinct arms. For mechanism-prompt comparators, Comparator+SCTL is a distinct composition arm. This stress test measures empirical compatibility with the comparator layer. Prompt-based mechanism comparators may interact with the ledger, so the composition is interpreted empirically without assuming additive effects.

Table 21: **SCTL composed directly with the fixed comparator.** Positive oriented effects favor Comparator+SCTL. ParaCook is lower-is-better and is oriented accordingly.

Task	Comparator	Comparator+SCTL	Oriented effect [95% CI]
GovSim ↑	58.7	61.1	$+2.4$ [1.7,3.4]
C2C ↑	33.4	34.6	$+1.2$ [-0.05,2.59]
CoffeeBench ↑	18.9	18.8	-0.1 [-0.3,0.1]
Alem ↑	70.2	71.7	$+1.5$ [1.0,2.4]
Welfare Diplomacy ↑	0.608	0.620	$+0.012$ [-0.001,0.027]
Melting Pot ↑	60.6	64.0	$+3.4$ [2.8,4.2]
MultiAgentBench ↑	77.0	78.2	$+1.2$ [0.6,2.0]
Collab-Overcooked ↑	70.0	71.4	$+1.4$ [0.7,2.3]
ParaCook ↓	294.5	281.7	$+12.8$ [10.9,15.2]
CoopEval ↑	0.804	0.801	-0.003 [-0.008,0.005]

The composed arm improves $8/10$ point estimates and $6/10$ paired percentile intervals exclude zero in its favor. The two non-improving tasks are CoffeeBench and CoopEval. Among tasks whose fixed comparator is a mechanism prompt, Comparator+SCTL is often below standalone SCTL even when it improves on the mechanism alone, indicating interaction between coordination layers rather than strict additivity.

G LLM BENCHMARKS AND BASELINE MAPPING

The LLM benchmark wrapper separates the quantity used to build the temporal ledger from the fixed metric used for final task evaluation. Most final metrics are benchmark-native. GovSim, C2C, and Alem use predeclared adapter indices so that the reported values have a well-defined continuous scale across the seven-model panel. ParaCook's published planning-time metric is penalized Order Completion Time (pOCT), with lower values preferred. For benchmark b, the adapter defines a loss ℓ_b that is computed from the rest of the run, is non-negative, and is fixed before the experiments. Lower is better. For a benchmark whose metric is higher-is-better, the loss is the remaining gap to a documented upper bound, which we call the shortfall. For a lower-is-better metric, the loss is the metric itself. Inside the rollouts that estimate coalition values, the agents in S follow the benchmark's own cooperative workflow π_S^C, and the SCTL ledger is not shown. This prevents the ledger from influencing the values from which it is built. The rollouts add no planner text, transcripts, or tools to the context of the acting agents. A coalition estimate at checkpoint state s_k starts R matched rollouts from that state, in which the agents outside S follow the benchmark's fixed fallback or an idle action. The main estimator uses $R = 2$ rollouts and a window of six decision rounds. When the task continues beyond the window, the adapter adds a fixed estimate of the shortfall at the end of the task. The estimator computes

$$\widehat{L}_k(S) = \frac{1}{R}\sum_{z=1}^{R} \ell_b(\zeta^{(z)}_{\tau_k:}), \qquad \widehat{w}_k(S) = \widehat{L}_k(\varnothing) - \widehat{L}_k(S).$$

Thus $w_k(\varnothing) = 0$, and the quantity that SCTL splits is the estimated reduction in loss. SCTL therefore never charges a team-level loss to the members of a coalition as a cost. For the agent, the reading is simple. The remaining credit is the part of the estimated loss reduction that is still assigned to it, and the realized credit is the part already settled. This lets the agent compare a locally attractive action with the value of staying coordinated. The LLM benchmark experiments test whether this signal is useful. The ledger fields do not create a binding incentive. The loss transform, fallback, rollout window, stopping rule, checkpoint policy, and Shapley approximation are fixed before any schedule outcome is inspected. The final evaluation always uses the benchmark's fixed score.

Table 22: **LLM benchmark continuation semantics and episode-length unit used by each adapter.** The temporal ledger uses the continuation loss in the fourth column, while final evaluation uses the fixed evaluation metric in the second column.

Benchmark	Evaluation metric	Native horizon	Continuation loss ℓ_b	Outside-S behavior
GovSim	mean survival time (%) ↑	24 rounds	remaining survival-time shortfall over the rollout horizon	benchmark independent/default policy
C2C	objective progress (%) ↑	40 turns	remaining private-objective shortfall plus treaty-break penalty	private-negotiation fallback policy
CoffeeBench	Net income ↑	90 days	income shortfall relative to a fixed profitability target	benchmark reference-firm policy
Alem	fixed-bound return index ↑	200 env. steps	return shortfall relative to the fixed adapter ceiling	frozen idle or default policy
Welfare Diplomacy	welfare points ↑	20 seasons	welfare shortfall relative to rollout upper bound	benchmark diplomacy fallback policy
Melting Pot	apples/agent ↑	100 env. steps	horizon harvest capacity minus realized apples/agent	frozen idle or default policy
MultiAgentBench	Task Score ↑	18 benchmark stages	documented score ceiling minus rollout Task Score	benchmark fallback role policy
Collab-Overcooked	Progress ↑	25 macro-steps	100 minus rollout progress completeness	documented idle adapter
ParaCook	pOCT ↓	20 plan steps	rollout pOCT	benchmark fallback planning policy
CoopEval	normalized score ↑	10 rounds	$1-$ normalized public-goods score	frozen scripted continuation policy

Table 23: **Benchmark-level horizon summary in benchmark-native units, with coordination checkpoints and estimated model-call footprint.**

Benchmark	Native length	Coord. checkpoints	Acting calls / ep.	Input tokens / ep.
GovSim	24 rounds	3	58	41.8k
C2C	40 turns	3	96	67.5k
CoffeeBench	90 days	4	214	121.3k
Alem	200 env. steps	3	142	88.4k
Welfare Diplomacy	20 seasons	4	168	109.6k
Melting Pot	100 env. steps	3	72	53.1k
MultiAgentBench	18 benchmark stages	3	54	45.7k
Collab-Overcooked	25 macro-steps	3	61	46.9k
ParaCook	20 plan steps	3	52	39.4k
CoopEval	10 rounds	3	19	14.8k

Native length is reported in each environment's own progress unit, whereas acting calls count model invocations under the adapter schedule. Environment-only transitions, scripted or idle action transitions, and macro-actions can advance the native horizon without issuing a new model call. Table 23 therefore reports both quantities explicitly.

Table 24: **Continuation-batch derivation for one SCTL episode in each rollout-estimated LLM benchmark adapter.** $|A_k|$ is the controllable active-set size used by the benchmark adapter. For $|A_k| \leq 3$ the estimator evaluates all coalitions. For larger sets it uses four fixed permutations with shared empty and grand coalitions. The sum row reports only batches per episode.

Benchmark	$\lvert A_k \rvert$	Checkpoints	Batches/checkpoint	Batches/episode
GovSim	6	3	22	66
C2C	4	3	14	42
CoffeeBench	1	4	2	8
Alem	4	3	14	42
Welfare Diplomacy	7	4	26	104
Melting Pot	4	3	14	42
MultiAgentBench	3	3	8	24
Collab-Overcooked	2	3	4	12
ParaCook	3	3	8	24
Sum per one instance/model sweep	–	–	–	364

With 20 instances for each of seven acting models, Table 24 gives $364 \times 140 = 50{,}960$ cached continuation batches for one full SCTL arm. This value is used directly in Table 37.

Comparator identities are frozen in `experiment_config.yaml` before confirmatory task runs. For each setting, the reproduction protocol chooses one standalone comparator from the reproduced native and mechanism candidates using benchmark documentation and non-confirmatory reproduction checks. We refer to this choice as the strongest comparator fixed in advance, or fixed comparator. No confirmatory test-instance score is used to rank or select the candidates.

The acting LLM never receives rollout transcripts. It receives only the compact temporal ledger derived from the allocated savings share. Increasing R therefore increases estimator compute without adding rollout text to the acting-agent prompt. A continuation batch is one coalition-state estimate with $R = 2$ rollout replicates in the main configuration. Values are cached by benchmark state, active set, coalition, adapter version, and rollout seed. Matched arms reuse a cached estimate while their realized states remain identical. After actions diverge, each arm estimates from its own realized state. Equality of estimator rules does not imply identical later trajectories.

The first scheduled checkpoint is indexed $k = 0$ and initializes Υ_i. No ledger is shown before that checkpoint, and no separate episode-start estimator pass is charged. The terminal boundary closes the ledger without an additional rollout. Each long-horizon adapter exposes three scheduled coordination checkpoints per episode at approximately 25, 50, and 75 percent of the benchmark horizon, with an additional event-triggered checkpoint after a change in the active agents. CoffeeBench and Welfare Diplomacy also expose a late-horizon checkpoint because their negotiation and reallocation decisions remain consequential very near the terminal phase. Exact Shapley is used for active sets of at most three agents. Larger active sets use four fixed-seed permutation samples per checkpoint, following standard sampling-based Shapley approximation (Castro et al., 2009). Each sampled permutation evaluates the nested coalition sequence required for one Monte Carlo Shapley estimate, and repeated coalitions are cached. This rule makes the continuation-batch budget a deterministic function of active-set size and checkpoint count.

ComputeMatched receives the same continuation-rollout budget but exposes no estimator-derived fields. Continuation RNG, caches, and rollout environment copies are isolated from acting RNG and the live benchmark state. Every rollout result is discarded before the acting call. ComputeMatched is therefore a null control for spending the estimator budget without transmitting estimator information. Frozen-SCTL performs one open-loop planning pass from the initial checkpoint. That pass predicts the scheduled future checkpoint states under the frozen cooperative policy, evaluates the same required coalition set on those predicted states, and stores the resulting checkpoint-indexed ledger. The stored ledger is then replayed without conditioning on subsequently realized states. It therefore has the same number of planned ledger points as SCTL but no online state feedback. Delayed is different. It holds only the initial credit constant until the terminal checkpoint, while UniformSchedule linearly realizes that same initial credit. PlaceboLedger controls for ledger format and monotone-looking values. ShuffledLedger uses the checkpoint-indexed values from the open-loop Frozen-SCTL plan, all of which are available before the episode, and applies a seeded permutation to their temporal

order. SameInfoSummary supplies the same continuation estimates, checkpoint timing, and carried account state without the structured ledger representation. It targets representation, but the full control set is not factorial, so we do not interpret any single contrast as a pure component effect. Frozen-SCTL is the closest state-feedback control because it replays an open-loop checkpoint ledger while leaving the other SCTL components unchanged. Placebo and ShuffledLedger are additional two-task diagnostics of informative continuation content and temporal alignment. Because shuffled values can be inconsistent with the realized state, the Shuffled-to-Frozen difference is not interpreted as a pure ordering effect.

GovSim. GovSim supplies the long-horizon common-resource setting and Universalization as the pre-specified native baseline (Piatti et al., 2024). Final evaluation reports normalized mean survival time, defined as 100 times the mean fraction of the 24-round horizon survived. The metric is a continuous episode-length index and does not encode a binary survival indicator. The adapter exposes six controllable decision roles by splitting the benchmark community workflow into six ledger-addressable positions while preserving the underlying resource dynamics and native action interface. Continuation rollouts start from the current resource state. The mechanism baseline is an adapted Shapley-Coop pricing-and-negotiation prompt with the same task observations and action schema (Hua et al., 2025).

Cooperate to Compete (C2C). C2C is a four-player mixed-motive conquest environment with private objectives, private negotiations, and non-binding agreements (O'Neill et al., 2026). Because one acting model controls all four evaluated seats, final evaluation uses a predeclared team objective-progress index. We do not report it as a seat-level win rate. For each seat, the index is the fraction of its private objective criteria satisfied at termination, averaged across the four seats and scaled to 0–100. The coordination adapter keeps the benchmark-native turn structure and appends the ledger at the three scheduled checkpoints. The mechanism baseline is an adapted Shapley-Coop pricing-and-negotiation baseline that uses the same acting model, turn interface, and task observations (Hua et al., 2025).

CoffeeBench. CoffeeBench evaluates long-horizon economic coordination in a six-firm coffee supply chain over 90 simulated days (Sugiura et al., 2026). The focal metric is benchmark-native net income. The adapter controls only the evaluated roaster. The other five firms remain on the fixed reference policies defined by the benchmark. Consequently A_k contains one controllable decision maker in this adapter, the coalition game has only $\varnothing$ and the focal-roaster singleton, and Shapley reduces to the singleton continuation credit. CoffeeBench therefore isolates temporal state conditioning under singleton credit assignment. Multi-agent allocation is outside this adapter. The mechanism baseline uses a self-negotiated-contract prompt adapted to the same evaluated-roaster interface (Wyse et al., 2026).

Alem. Alem is an open-ended multi-agent coordination benchmark with Craftax-like survival dynamics, soft specialization, communication, and controllable coordination difficulty (Tessera et al., 2026). We use a fixed difficulty and report a predeclared return index equal to 100 times episode return divided by the fixed adapter ceiling used by the adapter. This index is not the benchmark's cross-level normalized return. The continuation loss is shortfall to that same fixed ceiling. The mechanism baseline is the Shapley-Coop style pricing baseline adapted to the homogeneous-team setting (Hua et al., 2025).

Welfare Diplomacy. Welfare Diplomacy is a general-sum, multi-season adaptation of Diplomacy designed to evaluate cooperation under mixed motives (Mukobi et al., 2023). The native score is welfare points accumulated by the end of the fixed-horizon game. The mechanism baseline is a self-negotiated-contract baseline adapted from Wyse et al. (2026) and run with the same season horizon.

Melting Pot. Commons Harvest is a spatial shared-resource task with local regeneration (Agapiou et al., 2023; Mosquera et al., 2025). The continuation loss uses the known per-step harvest capacity and remaining rollout horizon, avoiding a post-hoc score anchor. The pre-specified native comparator is the benchmark-default coordination policy used in the reproduced evaluation. The mechanism baseline is an adapted Shapley-Coop pricing prompt that preserves the Commons Harvest action interface (Hua et al., 2025).

MultiAgentBench. The native comparison uses the benchmark Task Score and the reproduction protocol pre-specifies the corresponding coordination reference among GRAPH COORDINATION,

GROUP DISCUSSION, and COGNITIVE PLANNING before confirmatory task runs (Zhu et al., 2025). The mechanism baseline is the adapted Shapley-Coop style pricing baseline on the same benchmark slice (Hua et al., 2025).

Collab-Overcooked and ParaCook. Collab-Overcooked uses the in-context collaborative agent that comes with the benchmark as its baseline, whose architecture includes memory, reflection, communication, and error handling (Sun et al., 2025). ParaCook uses the planning reference pre-specified by the reproduction protocol from input–output prompting (IO), chain-of-thought prompting (CoT), Plan Like a Graph (PLaG), and multi-step reasoning-and-acting prompting (MultiStepReAct) for the corresponding model and task setting (Zhang et al., 2026). The coordination layer does not change either benchmark's native action space. For Collab-Overcooked and ParaCook, the mechanism baseline is an adapted Shapley-Coop pricing prompt layered on the same reproduced native observation and action interface (Hua et al., 2025).

CoopEval. CoopEval is treated as a separate mechanism-comparison environment and is outside the long-horizon rollout-estimator family (Tewolde et al., 2026). Its repeated public-goods slice uses a game-defined payoff table with a frozen scripted continuation policy for ledger construction. No coalition-rollout estimator or ComputeMatched arm is used. The acting LLM policy remains stochastic, so the reported benchmark outcome retains sampling variation even though ledger construction uses the fixed game table. Repetition, reputation, mediation, and contracting baselines are reproduced on the same slice. The mechanism reference used in Table 1 is fixed by the reproduction protocol before confirmatory task runs. Three evenly spaced round checkpoints define SCTL and Frozen-SCTL, even though no rollout estimator is used in this wrapper.

Table 25: **SCTL versus Frozen-SCTL.** Effects are oriented so that positive values favor SCTL.

Task	Frozen-SCTL	SCTL	Oriented effect [95% CI]
GovSim	59.6	61.2	$+1.6$ [1.1,2.3]
C2C	34.2	35.1	$+0.9$ [0.6,1.2]
CoffeeBench	18.7	18.6	-0.1 [-0.2,0.1]
Alem	71.7	72.8	$+1.1$ [0.8,1.6]
Welfare Diplomacy	0.616	0.629	$+0.013$ [0.010,0.016]
Melting Pot	63.2	63.1	-0.1 [-0.5,0.4]
MAB	78.6	79.4	$+0.8$ [0.5,1.1]
C-OC	71.7	72.5	$+0.8$ [0.3,1.4]
ParaCook	292.0	286.4	$+5.6$ [3.9,7.4]
CoopEval	0.774	0.788	$+0.014$ [0.011,0.017]

Table 26: **Composition with the reproduced native workflow.** Effects are oriented so positive values favor Native+SCTL.

Task	Native	Native+SCTL	Oriented effect [95% CI]
GovSim	57.4	62.8	$+5.4$ [4.8,6.1]
C2C	32.7	35.9	$+3.2$ [2.9,3.6]
CoffeeBench	18.9	18.8	-0.1 [-0.3,0.1]
Alem	69.5	73.5	$+4.0$ [3.6,4.5]
Welfare Diplomacy	0.596	0.636	$+0.040$ [0.036,0.045]
Melting Pot	60.6	64.0	$+3.4$ [2.8,4.2]
MultiAgentBench	75.4	80.2	$+4.8$ [4.4,5.3]
Collab-Overcooked	69.4	73.4	$+4.0$ [3.4,4.8]
ParaCook	294.5	281.7	$+12.8$ [10.9,15.2]

Table 27: **Paired representation and composition contrasts.** All effects are oriented so positive values favor the SCTL-containing condition. Native+SCTL is not run on CoopEval.

Task	SCTL − SameInfoSummary [95% CI]	Native+SCTL − Native [95% CI]
GovSim	+1.7 [0.4,3.1]	+5.4 [4.8,6.1]
C2C	+0.5 [-0.1,1.0]	+3.2 [2.9,3.6]
CoffeeBench	−0.1 [-0.5,0.3]	−0.1 [-0.3,0.1]
Alem	+1.2 [0.6,1.8]	+4.0 [3.6,4.5]
Welfare Diplomacy	+0.008 [0.002,0.014]	+0.040 [0.036,0.045]
Melting Pot	−0.2 [-1.3,1.0]	+3.4 [2.8,4.2]
MAB	+0.6 [-0.3,1.6]	+4.8 [4.4,5.3]
C-OC	+1.1 [-0.1,2.2]	+4.0 [3.4,4.8]
ParaCook	+2.7 [-2.8,8.3]	+12.8 [10.9,15.2]
CoopEval	−0.002 [-0.012,0.008]	–

Table 28: **Per-task comparison with the fixed comparator used in Table 1.** Effects are relative and oriented so positive values favor SCTL. Holm p values are from two-sided exact sign-flip tests on instance-averaged paired effects across the ten comparator contrasts. Comparator identity is fixed before confirmatory task runs and does not use confirmatory test-set outcomes.

Task	comparator	SCTL	oriented effect [95% CI]	Holm p
GovSim	adapted Shapley-Coop	61.2	+4.3% [2.5,6.0]	0.0011
C2C	adapted Shapley-Coop	35.1	+5.1% [3.4,7.0]	< 0.001
CoffeeBench	native reference policy	18.6	−1.6% [-2.8,-0.3]	0.031
Alem	adapted Shapley-Coop	72.8	+3.7% [2.4,5.2]	< 0.001
Welfare Diplomacy	self-negotiated contracts	0.629	+3.5% [2.4,4.5]	< 0.001
Melting Pot	native coordination policy	63.1	+4.1% [2.6,5.7]	< 0.001
MultiAgentBench	adapted Shapley-Coop	79.4	+3.1% [2.1,4.2]	< 0.001
Collab-Overcooked	adapted Shapley-Coop	72.5	+3.6% [2.2,4.9]	< 0.001
ParaCook	native planning baseline	286.4	+2.8% [1.4,4.1]	0.0038
CoopEval	Contract/Mediation mechanism	0.788	−2.0% [-3.3,-0.8]	0.0117

Table 29: **Checkpoint-density sensitivity on fixed ten-instance-per-model subsets.** Each cell reports SCTL followed by its oriented advantage over UniformSchedule.

Task	2 checkpoints	3 checkpoints	6 checkpoints
GovSim	60.6 (+2.4)	61.8 (+3.4)	62.0 (+3.3)
C2C	35.4 (+2.1)	36.3 (+3.1)	36.2 (+2.9)
Alem	72.1 (+2.4)	73.3 (+3.7)	73.4 (+3.5)

G.1 Continuation-estimator sensitivity

Let H_{roll} denote the receding rollout horizon in benchmark-native decisions. The LLM benchmark mechanism depends on rollout-estimated continuation values, so compute matching alone does not test estimator error. The sensitivity protocol evaluates SCTL and UniformSchedule on fixed GovSim and MultiAgentBench subsets around the main $R = 2$, $H_{\text{roll}} = 6$ configuration. We vary rollout replicates to $R = 1$ and $R = 4$, and separately extend the receding window to $H_{\text{roll}} = 8$ while keeping $R = 2$. The main-configuration column reuses the same ten-instance subset from the primary runs. We also perturb cached main-configuration estimates with seeded mean-one multiplicative noise at coefficient of variation (CV) 0.20. The same perturbation seed is shared across matched schedule arms.

Table 30: **Continuation-estimator sensitivity on fixed ten-instance-per-model subsets.** Each cell reports the SCTL task score followed by the SCTL-minus-UniformSchedule difference. The main configuration is $R = 2, H_{\text{roll}} = 6$.

Task	R=1, H=6	R=2, H=6	R=4, H=6	R=2, H=8	CV 0.20
GovSim survival-time index	61.1 (+2.9)	61.8 (+3.4)	62.0 (+3.5)	61.9 (+3.4)	60.7 (+2.3)
MAB Task Score	79.2 (+2.8)	79.9 (+3.4)	80.0 (+3.5)	79.8 (+3.2)	78.8 (+2.1)

The main two-rollout, six-decision configuration is already near the higher-budget plateau on these subsets. Doubling rollout replicates or extending the window gives only small changes, while halving replicates and estimator perturbation reduce the margin more visibly. These tests assess estimator-budget sensitivity. LLM benchmark ground-truth validation is unavailable.

Matched-budget rollout-information controls. On all nine rollout-estimated LLM benchmark tasks we spend the same main estimator budget in two alternative ways. **ActionLookahead** evaluates benchmark-native candidate actions with the same $R = 2$, $H_{\text{roll}} = 6$ rollout budget and selects the action with the best estimated continuation outcome. It receives no temporal ledger. **TeamValueOnly** uses the same checkpoint policy and rollout estimator as SCTL but exposes only the scalar team continuation value $w_k(A_k) = L_k(\varnothing) - L_k(A_k)$. It omits agent-level allocation, the first-checkpoint account, realized credit, remaining credit, and the latest per-agent update. The scalar is inserted as one labeled natural-language sentence at the same checkpoint location used by SCTL. The first control asks whether direct search is a better use of the compute. The second directly tests whether a team-level rollout signal explains the gain without cooperative allocation and temporal accounting.

Table 31: **Matched-budget rollout-information controls on the nine rollout-estimated LLM benchmark tasks.** TV is TeamValueOnly and AL is ActionLookahead. Effects are oriented so positive values favor SCTL.

Task	TV	AL	SCTL	SCTL−TV [95% CI]	SCTL−AL [95% CI]
GovSim	60.2	59.8	61.2	+1.0 [0.2,1.9]	+1.4 [0.4,2.5]
C2C	34.7	34.5	35.1	+0.4 [-0.2,1.2]	+0.6 [-0.1,1.5]
CoffeeBench	18.7	18.8	18.6	−0.1 [-0.4,0.3]	−0.2 [-0.6,0.2]
Alem	72.2	72.0	72.8	+0.6 [0.1,1.3]	+0.8 [0.2,1.5]
Welfare Diplomacy	0.625	0.624	0.629	+0.004 [-0.001,0.010]	+0.005 [-0.001,0.012]
Melting Pot	63.2	63.4	63.1	−0.1 [-0.8,0.7]	−0.3 [-1.0,0.6]
MultiAgentBench	78.7	78.5	79.4	+0.7 [0.2,1.3]	+0.9 [0.4,1.6]
Collab-Overcooked	71.6	71.2	72.5	+0.9 [0.3,1.6]	+1.3 [0.7,2.0]
ParaCook	288.2	289.0	286.4	+1.8 [-0.6,5.1]	+2.6 [-0.5,6.5]

TeamValueOnly recovers most of the LLM benchmark improvement from NoMechanism on several tasks, but it also changes presentation. StructuredTeamValue therefore keeps the same team-level account in matched assigned/realized/remaining fields and updates it at the same checkpoints. EqualSplitLedger keeps the per-agent ledger but replaces Shapley with an equal split. Table 32 reports these matched controls. SCTL is above StructuredTeamValue on $7/9$ point estimates and $3/9$ paired intervals. EqualSplitLedger differs from SCTL by at most 0.33% in relative task score, indicating that the exact allocation rule contributes little in this LLM benchmark suite.

Table 32: **Format- and allocation-matched LLM benchmark controls.** STV is StructuredTeamValue and ESL is EqualSplitLedger. The final column gives the oriented SCTL–STV paired effect, with intervals from the paired bootstrap over task instances. Positive values favor SCTL. In CoffeeBench only one agent is active, so STV and SCTL coincide and their difference is zero.

Task	TeamValueOnly	STV	ESL	SCTL	SCTL−STV [95% CI]
GovSim	60.2	60.6	61.0	61.2	+0.6 [0.1,1.2]
C2C	34.7	34.8	35.0	35.1	+0.3 [−0.2,0.8]
CoffeeBench	18.7	18.6	18.6	18.6	0.00 [0.00,0.00]
Alem	72.2	72.3	72.6	72.8	+0.5 [0.1,1.0]
Welfare Diplomacy	0.625	0.626	0.628	0.629	+0.003 [−0.001,0.007]
Melting Pot	63.2	63.2	63.0	63.1	−0.1 [−0.6,0.4]
MultiAgentBench	78.7	78.9	79.2	79.4	+0.5 [−0.3,1.3]
Collab-Overcooked	71.6	72.0	72.3	72.5	+0.5 [0.1,1.0]
ParaCook	288.2	287.4	286.8	286.4	+1.0 [−1.1,3.2]

H STATISTICAL EVALUATION DETAILS

H.1 PRIMARY FIRST-DEVIATION HAZARD MODEL

Index an episode by $r = (\iota, f, a, m)$, where ι is the task instance, f is the controlled-game index, a is the schedule arm, and m is the acting-model index. Let T_f be the horizon of controlled game f and θ_a the delay parameter attached to arm a. Let $D_{rt} = 1$ when the first deviation in the team occurs at checkpoint t, conditional on no earlier deviation, and define

$$Z_{rt} = \mathbf{1}\Big[\exists i \in A_t : B_i^{\theta_a}(t) > K_i(t)\Big].$$

The pre-specified discrete-time model is

$$\operatorname{logit} \Pr(D_{rt} = 1) = \alpha_a^{(0)} + \xi_f + r_{\text{time}}(t/T_f) + \mu_m + \upsilon_\iota + \eta Z_{rt}, \tag{13}$$

where $\alpha_a^{(0)}$ is a schedule-arm intercept, ξ_f is a controlled-game fixed effect, $r_{\text{time}}(t/T_f)$ is a shared cubic spline in normalized horizon position with knots at 0.25, 0.50, and 0.75, μ_m is a fixed model effect over the seven-model set, and υ_ι is a task-instance random intercept. The crossing coefficient η gives the reported odds ratio $\exp(\eta)$. Game effects absorb level differences across G1–G6 and normalized time prevents the different horizons from sharing an unscaled checkpoint axis. The pre-specified directional alternative is $\eta > 0$ in the K-Hidden arm. The reported confirmatory Wald p-value is nevertheless two-sided. Intermediate-delay conditions enter through their exact Z_{rt} paths, so a crossing that later clears is coded correctly.

We resample task instances with replacement and refit Equation (13) on 5,000 bootstrap samples to obtain percentile confidence intervals. Confirmatory p-values come from two-sided Wald tests on the fitted coefficient and are Holm-adjusted within the pre-specified testing group. A secondary sensitivity model replaces the shared spline with game-by-condition normalized-time splines and gives $1.37\times$ [1.12,1.80]. A condition-specific spline without the game interaction gives $1.41\times$ [1.14,1.88], with Wald $p = 0.006$. As direct temporal placebos, shifting every crossing path two checkpoints earlier gives $1.04\times$ [0.91,1.26], while shifting it two checkpoints later gives $0.99\times$ [0.86,1.20]. Neither shifted indicator excludes one. To separate gradual late-horizon deterioration from a threshold association, we also fit the normalized continuous margin $\widetilde{M}_{rt} = \min_i(K_i(t) - B_i^{\theta_a}(t))/\Xi_i$ together with Z_{rt}. A $0.10\Xi_i$ deterioration in the margin has OR $1.11\times$ [1.05, 1.20], and the crossing indicator retains OR $1.24\times$ [1.08, 1.51] conditional on the continuous term. The narrow timing diagnostic compares steps $\{t^{\text{cross}}, t^{\text{cross}} + 1\}$ with earlier at-risk steps and gives OR $1.44\times$ [1.17, 1.79]. These continuous-margin and narrow-window fits are sensitivity analyses and do not define additional confirmatory families.

Main-performance event logging. The performance record stores the task instance and first deviation time for each episode. For deterministic controlled games, physical G_{cost} is obtained from Equation (18). The prescribed cooperative controls are evaluated through the first deviation, the fixed independent fallback is evaluated thereafter, and the resulting physical team cost is normalized by the instance-specific joint-plan and independent-continuation costs. an episode without deviation follows the cooperative plan to the horizon and has $G_{\text{cost}} = 0$. Table 33 is descriptive. Confirmatory first-deviation inference still comes from the separate identification and Matched-B protocols.

Table 33: **Joint EXIT and physical-cost summary for the five main controlled arms.** The identity $E[G_{\text{cost}}] = P(\text{EXIT})E[G_{\text{cost}} \mid \text{EXIT}]$ holds up to rounding. "Before/after" uses each schedule's displayed participation crossing and is descriptive. CFC-Prompt and NoMechanism display no ledger and have no crossing.

Condition	EXIT	median $t_{\text{exit}}/(T-1)$	$E[G \mid \text{EXIT}]$	$E[G]$	before cross	at/after cross
SCTL	47.9%	0.348	0.485	0.232	29.8%	18.1%
UniformSchedule	73.1%	0.304	0.509	0.372	44.0%	29.0%
Delayed	97.9%	0.217	0.624	0.611	57.6%	40.2%
CFC-Prompt	86.1%	0.222	0.567	0.488	–	–
NoMechanism	91.2%	0.217	0.609	0.555	–	–

Within the joint Delayed main-performance log, the descriptive checkpoint hazard OR for post-crossing versus pre-crossing at-risk steps is 1.59. This descriptive OR is close to, but distinct from,

the separately generated confirmatory K-Hidden OR of 1.52, and it is not added to the confirmatory family.

H.2 MATCHED-B FALLBACK-SHIFT MODEL

For the experiment in Appendix E.3, let $D_{\iota qmt}$ be the first-deviation indicator for task instance ι, fallback multiplier arm q, model index m, and step t, conditional on no earlier deviation. The primary analysis uses common at-risk matched strata $\sigma = (\iota, t, m)$. The matched-B experiment uses one fixed fully delayed charge schedule, so within a stratum the state, complete B vector, transfer history, step, model, and schedule are identical. Only the fallback rule and therefore $V_{\iota qmt}$ differ. We fit the conditional logistic model

$$\operatorname{logit} \Pr(D_{\iota qmt} = 1 \mid \sigma) = \alpha_\sigma^{\mathrm{str}} + \psi_- \mathbf{1}[q = q_-] + \psi_+ \mathbf{1}[q = q_+] + \eta_K V_{\iota qmt}, \tag{14}$$

where $\alpha_\sigma^{\mathrm{str}}$ is eliminated by conditioning on the matched stratum. The fully delayed schedule is fixed across the matched comparison and therefore does not require a separate stratum index. The multiplier-arm effects ψ_- and ψ_+ absorb a direct prompt effect of the announced exit-cost level, and η_K is the crossing coefficient. The pre-specified directional alternative is $\eta_K > 0$. The reported Wald p-value is two-sided. The coefficient compares crossing status after controlling for the multiplier arm, while $B_i(t)$ remains constant within each matched comparison. The secondary full-risk-set analysis uses all available at-risk rows with multiplier-arm intercepts, a shared normalized-time spline, and model fixed effects. Because $\theta = 1$ implies $\max_i B_i(t)/\Xi_i = 1$ before terminal settlement, this normalized remaining balance is collinear with the intercept and does not enter the fit. The full-risk-set analysis gives OR $1.35\times$ [1.13, 1.66].

The schedule-induced test in Equation (13) and the matched-B test in Equation (14) form the mechanism-identification family and receive Holm correction across the two confirmatory p-values. As placebos for the announced cost shift, we estimate both its main arm effect and an arm-by-time interaction on common at-risk rows before any triplet member reaches a participation crossing.

For the identification bank, 108 of 120 instances satisfy both deterministic crossing-shift requirements and 12 are excluded from this experiment only. Across seven acting models and three multiplier arms, this yields 2,268 matched-B episodes. The admitted bank has median $q_- = 0.90$ and $q_+ = 1.10$, with deterministic median crossing shifts of -2.0 and $+2.5$ steps. The adjusted estimate is $\exp(\eta_K) = 1.38$ [1.16, 1.76], with Holm-adjusted two-sided Wald $p = 0.001$. Before any triplet member crosses, the pooled announced-cost-arm odds ratio is 1.02 [0.91, 1.20], with two-sided Wald $p = 0.74$, and the arm-by-time interaction is 1.01 [0.95, 1.10], with two-sided Wald $p = 0.64$.

H.3 REASONING-ENABLED K-HIDDEN DIAGNOSTIC

A secondary diagnostic tests what information the model cites when K-Hidden allows a short rationale before the final STAY/EXIT token. The probe uses G1 and G4, SCTL and Delayed, all 20 retained instances, and all seven models, for 560 episodes. The prompt exposes the same fields as K-Hidden and still withholds $K_i(t)$. The visible-output cap is 256 tokens, and the parser reads only the final line `Decision: STAY` or `Decision: EXIT`.

The analysis repeats Equation (13) and codes nonexclusive rationale categories. The reasoning-enabled diagnostic gives a crossing odds ratio of $1.34\times$ [1.09, 1.78]. Remaining assigned cost or transfer history appears in 68% of rationales, the announced deviation cost or rule in 44%, and an attempted numerical reconstruction of hidden deviation cost in 16%. These categories overlap. The rationale distribution places more mass on visible ledger and fallback-rule information than on numerical reconstruction of the hidden scalar.

H.4 LLM BENCHMARK COORDINATION-PERSISTENCE ANALYSIS

For GovSim and Collab-Overcooked, Table 34 compares eight conditions, namely SCTL, Uniform-Schedule, SameInfoSummary, ComputeMatched, NoMechanism, Delayed, PlaceboLedger, and ShuffledLedger. Paired differences are computed within benchmark instance and model. SCTL–Uniform is the primary schedule contrast, while SCTL–SameInfoSummary is the primary representation contrast. Delayed is included because its nearly constant pre-terminal ledger can become stale while still occupying the same structured prompt fields. Confidence intervals use 5,000 instance-bootstrap

replicates. For T_{persist}, the primary descriptive summary is the restricted mean normalized persistence through the benchmark horizon, which handles right-censoring without assuming proportional hazards. The persistent-deviation definition from Appendix E.4 is a sensitivity analysis.

To connect behavior with task outcome without claiming mediation, we additionally fit

$$Y_r = \alpha_0 + \eta_{\text{adh}} F_{\text{adh},r} + \chi_{\text{task}(r)} + \mu_{\text{model}(r)} + \varepsilon_r, \tag{15}$$

where Y_r is the benchmark-native episode score after orienting it so larger is better, α_0 is an intercept, and η_{adh} is the adherence coefficient. The terms $\chi_{\text{task}(r)}$ and $\mu_{\text{model}(r)}$ are benchmark and model fixed effects, and ε_r is the residual. Equation (15) is a descriptive trace-level association. Causal indirect-effect claims are outside this analysis.

Temporal-order control. The ShuffledLedger control uses the full checkpoint-indexed ledger produced by the open-loop Frozen-SCTL planning pass, which is available before acting begins. A seeded permutation is fixed before the episode and replayed at scheduled checkpoints. This preserves the open-loop value multiset and prompt length without requiring access to future realized SCTL states. Under ShuffledLedger, GovSim survival is 57.0 and C-OC Progress Completeness is 69.0, compared with 61.2 and 72.5 under SCTL. Coordination adherence under ShuffledLedger is 0.64 on GovSim and 0.65 on C-OC. Because the precomputed open-loop value multiset and prompt structure are fixed, while downstream task states are allowed to diverge after different actions, the degradation tests sensitivity to temporal ordering without assuming identical post-treatment trajectories. The result remains a secondary diagnostic.

Table 34: **Coordination-persistence trace diagnostics.** F_{adh} is plan-consistent-step fraction and T_{persist} is restricted mean normalized persistence to first deviation. NM is NoMechanism, D is Delayed, U is UniformSchedule, SI is SameInfoSummary, CM is ComputeMatched, P is PlaceboLedger, and Sh is the causally precomputed ShuffledLedger. The paired SCTL–Uniform contrast is primary.

Benchmark	Metric	NM	CM	D	P	Sh	U	SI	SCTL	SCTL−U [95% CI]
GovSim	$F_{\text{adh}} \uparrow$	0.57	0.58	0.55	0.59	0.64	0.67	0.70	0.74	+0.07 [0.03,0.11]
GovSim	$T_{\text{persist}} \uparrow$	0.51	0.52	0.49	0.54	0.59	0.62	0.66	0.69	+0.07 [0.02,0.12]
C-OC	$F_{\text{adh}} \uparrow$	0.59	0.59	0.56	0.61	0.65	0.67	0.70	0.73	+0.06 [0.01,0.11]
C-OC	$T_{\text{persist}} \uparrow$	0.54	0.54	0.50	0.56	0.60	0.62	0.65	0.68	+0.06 [0.01,0.11]

The pooled descriptive association in Equation (15) is $+1.7$ [0.8, 2.6] benchmark-score points for a 0.10 increase in F_{adh}. The persistent-deviation sensitivity check retains the same broad ordering, with SCTL highest and Delayed lowest on both benchmarks. These trace quantities describe action persistence. Causal decomposition of benchmark score is outside this diagnostic analysis.

H.5 Confirmatory families and multiplicity

The design has four confirmatory families, with Holm correction at family level 0.05 within each family and no gatekeeping between families.

1. **Mechanism identification.** Test $\eta > 0$ in Equation (13) and $\eta_K > 0$ in the matched-B Equation (14), with Holm correction across these two tests. The K-Visible arm is a manipulation check and intermediate-θ movement is secondary. The LLM benchmark coordination-persistence analysis is also secondary because the LLM benchmark tasks do not expose a direct participation decision.
2. **Schedule task performance.** Compare SCTL against NoMechanism and UniformSchedule using G_{cost} for controlled tasks and the pre-specified task metric for each LLM benchmark setting. The family contains 32 paired contrasts, formed by two baselines across six controlled games and ten LLM benchmark settings. Two-sided exact sign-flip tests on instance-averaged paired effects are Holm-adjusted across the full family (Holm, 1979). For the 12 controlled contrasts, the adjusted values lie on the $n = 20$ exact sign-flip/Holm grid. For G1–G6 they are, versus NoMechanism, 6.10×10^{-5}, 6.10×10^{-5}, 6.10×10^{-5}, 6.10×10^{-5}, 6.10×10^{-5}, and 6.10×10^{-5}, and versus UniformSchedule, 6.10×10^{-5}, 6.10×10^{-5}, 6.10×10^{-5}, 7.63×10^{-5}, 7.63×10^{-5}, and 7.63×10^{-5}. All 32 adjusted values remain below 0.001, and the largest remains $p_{\text{Holm}} = 1.51 \times 10^{-4}$ for CoffeeBench SCTL versus UniformSchedule.

3. **Fixed-comparator performance.** Compare SCTL with the fixed comparator on all ten LLM benchmark settings. Two-sided exact sign-flip tests are computed on instance-averaged paired effects and Holm-adjusted across these ten contrasts. The eight positive contrasts remain significant after adjustment, while CoffeeBench and CoopEval significantly favor their comparators.
4. **Membership-loss stress test.** On LLM benchmark tasks with at least two controllable active agents, compare Reallocate with NoRecovery after a matched injected loss of an agent using the pre-specified task outcome. CoffeeBench is excluded because removing its only controllable agent leaves no surviving team for reallocation. Controlled recovery is reported only as a structural recomputation diagnostic because the post-loss environment response is deterministic conditional on the task bank. LLM benchmark task-level inference is interpreted after Holm correction. Broad recovery is not treated as confirmed when only one task remains significant.

Agent identity is descriptive because multiple controlled games have tied participation crossings.

Table 35: **Post-loss recovery outcomes on long-horizon tasks with at least two controllable active agents.** LLM benchmark rows use a separately seeded 10-instance-per-model recovery bank distinct from the main 20-instance-per-model evaluation bank, and the controlled pooled row uses all 120 G1–G6 instances of the unfiltered performance bank. Effects are oriented so that positive values favor Reallocate.

Task	NoRecovery	Reallocate	Oriented effect
Controlled pooled G_{cost} ↓	0.421	0.350	$+0.071$
GovSim ↑	51.6	55.1	$+3.5$
C2C ↑	30.4	32.0	$+1.6$
Alem ↑	65.0	67.8	$+2.8$
Welfare Diplomacy ↑	0.552	0.570	$+0.018$
Melting Pot ↑	55.0	57.4	$+2.4$
MultiAgentBench ↑	72.0	74.3	$+2.3$
Collab-Overcooked ↑	65.0	67.2	$+2.2$
ParaCook ↓	318.0	309.6	$+8.4$

Table 36: **Recovery contrasts after matched injected loss of an agent.** LLM benchmark rows use the separately seeded 10-instance-per-model recovery bank, and the controlled pooled row uses all 120 G1–G6 instances of the unfiltered performance bank. LLM benchmark confidence intervals use percentile bootstrap resampling of task instances. LLM benchmark p-values use two-sided exact sign-flip tests on instance-averaged paired effects and are Holm-adjusted across the eight applicable tasks. Only GovSim remains below 0.05 after adjustment. The controlled row is a deterministic recomputation diagnostic and has no behavioral p-value.

Task	Oriented effect	95% bootstrap CI	Holm p
Controlled pooled	$+0.071$	[0.053, 0.088]	–
GovSim	$+3.5$	[2.75, 4.35]	0.016
C2C	$+1.6$	[0.25, 2.99]	0.246
Alem	$+2.8$	[0.50, 5.42]	0.246
Welfare Diplomacy	$+0.018$	[0.002, 0.035]	0.246
Melting Pot	$+2.4$	[0.60, 4.24]	0.246
MultiAgentBench	$+2.3$	[0.15, 4.53]	0.246
Collab-Overcooked	$+2.2$	[0.53, 3.85]	0.246
ParaCook	$+8.4$	[2.08, 15.45]	0.246

H.6 Power and pilots

Pilot episodes are excluded from the confirmatory sample. The schedule-manipulation power-planning pass uses 2,000 simulated datasets, family-wise level 0.05, 80% target power, the exact retained-instance crossing paths, and the shared-time-spline specification in Equation (13). It uses 4,200 controlled identification episodes from the five fixed schedule arms, a planning baseline first-deviation probability of 0.112 per at-risk checkpoint, and a detectable odds ratio near $1.30\times$. Task-performance power is planned separately on the unfiltered performance bank. The matched-B

experiment is calibrated separately after the deterministic fallback-shift admission pass fixes the number of matched triplets. Its simulation uses Equation (14) and the two-test Holm mechanism family. These additional episodes are not included in the 4,200-episode schedule-identification count. The pilot-only variance pass supplies calibration inputs and remains excluded from confirmatory inference.

H.7 Compute accounting

Episode counts and continuation-estimator schedules follow the configured design. LLM benchmark acting-call entries use the fixed per-episode footprints in Table 23. Controlled STAY/EXIT acting-call totals account for truncation after the first deviation. The nine rollout-estimated long-horizon tasks use 20 instances per model and condition, so one full LLM benchmark arm contains $9 \times 20 \times 7 = 1{,}260$ episodes. Controlled performance uses five main conditions plus two additional allocation rules, giving $120 \times 7 \times 7 = 5{,}880$ episodes. The five protocol ablations add $120 \times 7 \times 5 = 4{,}200$ episodes. The structural holdout contributes $20 \times 7 \times 5 = 700$ controlled episodes. A continuation batch is one cached coalition-state estimate. Exact Shapley uses all $2^{|A_k|}$ coalitions for $|A_k| \leq 3$. Larger teams use four fixed permutations. The generic distinct-coalition upper bound is $2 + 4(|A_k| - 1)$ per checkpoint after sharing the empty and grand coalitions. For the frozen permutation seeds used here, the deduplicated coalition sets attain the displayed counts in Table 24, whose deterministic sum gives 50,960 batches for one full SCTL arm. For four-agent adapters this is 14 versus 16 coalitions under exact Shapley, so the permutation rule provides a uniform estimator policy across larger teams. At $|A_k| = 4$ the reduction from exact Shapley is only two coalitions. ComputeMatched and SameInfoSummary use that full budget. Frozen-SCTL evaluates the same checkpoint-indexed coalition set in its single open-loop initial planning pass and therefore also uses the full planned batch count. UniformSchedule and Delayed use only the first-checkpoint allocation, while CFC-Prompt and NoMechanism use no continuation estimator. This decomposition gives 133,280 continuation batches across the six non-SCTL control arms grouped below. The LLM benchmark allocation comparison additionally evaluates Proportional and Equal-savings with the full estimator budget. These two arms are counted together with StructuredTeamValue and EqualSplitLedger in the LLM benchmark structure/allocation row. LLM benchmark recovery uses a separately seeded 10-instance-per-model bank, distinct from the main 20-instance-per-model evaluation bank. Checkpoint and estimator sensitivities use separate 10-instance-per-model subsets, and their compute row counts only additional cells after reusing matching main-configuration cells. The recovery-plus-composition row combines 1,680 controlled recovery episodes from all 20 instances in each of G1–G6, 1,120 LLM benchmark recovery episodes across the eight applicable tasks on the separately seeded recovery bank, and 1,260 Native+SCTL composition episodes. The controlled and LLM benchmark recovery banks are distinct. CoopEval has no ComputeMatched arm because its ledger is constructed from a fixed game table. Its accounting row includes seven main SCTL/control arms, the four reproduced public-goods mechanism baselines, and the Proportional and Equal-savings allocation arms reported in Table 10. Comparator+SCTL adds distinct arms only where the mapped comparator is not the native workflow, namely GovSim, C2C, Alem, Welfare Diplomacy, MultiAgentBench, Collab-Overcooked, and CoopEval. CoffeeBench, Melting Pot, and ParaCook reuse the corresponding Native+SCTL arm.

Table 37: **Compute accounting for the reported experiment suite.** A continuation batch is one cached coalition-state estimate. The main configuration uses two rollout trajectories per batch. Sensitivity rows use their stated alternative budgets. LLM benchmark acting-call counts use the accounting footprints in Table 23.

Study block	Episodes	Acting calls	Cont. batches	Rollout calls
Controlled identification	4,200	249,180	0	0
Matched-B fallback shift	2,268	72,660	0	0
Reasoning K-Hidden probe	560	27,860	0	0
Controlled performance + allocation	5,880	334,980	0	0
Controlled protocol ablations	4,200	339,240	0	0
Matched-format SCTL/Uniform controls	5,040	424,440	0	0
Structural holdout	700	28,980	0	0
SCTL long-horizon arm	1,260	128,380	50,960	101,920
Comparator+SCTL composition	980	83,720	40,600	81,200
ComputeMatched long-horizon arm	1,260	128,380	50,960	101,920
ActionLookahead long-horizon arm	1,260	128,380	50,960	101,920
TeamValueOnly long-horizon arm	1,260	128,380	50,960	101,920
LLM benchmark structure/allocation controls	5,040	513,520	203,840	407,680
Other long-horizon controls	7,560	770,280	133,280	266,560
Native and mechanism baselines	2,520	256,760	0	0
Shuffled + placebo ledgers	560	33,320	21,840	43,680
Estimator + checkpoint sensitivity	1,960	145,600	60,200	128,800
CoopEval main + mechanism suite	1,820	34,580	0	0
Recovery + Native+SCTL composition	4,060	294,000	105,980	211,960

Table 38: **Token-budget accounting corresponding to Table 37.** The accounting model uses a mean rollout input of $9.0k$ tokens and a mean rollout-output allowance of 0.9k tokens per six-decision receding trajectory. Controlled one-token STAY/EXIT blocks use a 1.2-token surfaced-output allowance per acting call, while LLM benchmark native-action blocks use their benchmark-specific allowances. Provider-hidden reasoning tokens are excluded. Dollar cost and wall-clock time are not reported.

Study block	Acting input	Acting output	Rollout input	Rollout output
Controlled identification	149.9M	0.299M	0	0
Matched-B fallback shift	44.9M	0.087M	0	0
Reasoning K-Hidden probe	19.6M	4.03M	0	0
Controlled performance + allocation	227.9M	0.402M	0	0
Controlled protocol ablations	210.3M	4.32M	0	0
Matched-format SCTL/Uniform controls	263.1M	8.71M	0	0
Structural holdout	18.8M	0.035M	0	0
SCTL long-horizon arm	85.9M	12.8M	917.3M	91.7M
Comparator+SCTL composition	58.1M	8.4M	730.8M	73.1M
ComputeMatched long-horizon arm	85.9M	12.8M	917.3M	91.7M
ActionLookahead long-horizon arm	85.9M	12.8M	917.3M	91.7M
TeamValueOnly long-horizon arm	85.9M	12.8M	917.3M	91.7M
LLM benchmark structure/allocation controls	343.7M	51.2M	3.67B	366.9M
Other long-horizon controls	515.5M	76.8M	2.40B	239.9M
Native and mechanism baselines	171.8M	25.6M	0	0
Shuffled + placebo ledgers	24.8M	3.3M	393.1M	39.3M
Estimator + checkpoint sensitivity	104.4M	14.6M	1.21B	121.0M
CoopEval main + mechanism suite	26.9M	5.0M	0	0
Recovery + Native+SCTL composition	$200.6M$	$22.8M$	1.91B	190.8M

The main LLM benchmark configuration uses two rollout replicates per cached continuation batch and a six-decision receding rollout window. Acting-model input totals exclude estimator trajectories, which are reported separately. Episode counts and coalition-estimator counts follow the fixed protocol. LLM benchmark acting-call totals use the benchmark footprints in Table 23, while controlled acting-call totals account for truncation after the first deviation. The recovery-composition row uses 1,680 controlled recovery episodes on the unfiltered controlled performance bank, 1,120 LLM benchmark recovery episodes, and 1,260 Native+SCTL composition episodes. For LLM benchmark recovery, NoRecovery retains 24,920 continuation batches, while Reallocate adds one event-triggered post-loss checkpoint per episode. The active-set sizes after the injected loss imply 5,180 additional batches, so the two recovery arms contribute 55,020 batches and the combined row contributes 105,980 batches

after adding Native+SCTL. Acting input combines the controlled-performance per-call planning allowance with the benchmark-specific LLM benchmark allowances in Table 23. The eight-decision estimator-sensitivity cell scales the rollout footprint proportionally to window length.

I CASE-STUDY PROTOCOL AND STORED TRACE

The case study uses held-out seed `G4-case-017`, fixed before the displayed outcome is inspected. It illustrates the mechanism sequence and remains outside confirmatory inference. The stored trace designates the agent attaining the first Delayed participation-boundary crossing as the focal agent. All normalized ledger values in the figure and table refer to that same agent. The SCTL and Delayed traces use the same model, primitive task parameters, cooperative plan, allocation, and decoding configuration. The displayed decision trace ends at the first endogenous EXIT in a condition. Outcome evaluation then continues the environment to the benchmark horizon under the fixed fallback after EXIT, matching Algorithm 1. If no EXIT occurs, the cooperative plan continues to the horizon. No membership-loss recovery is active in this case.

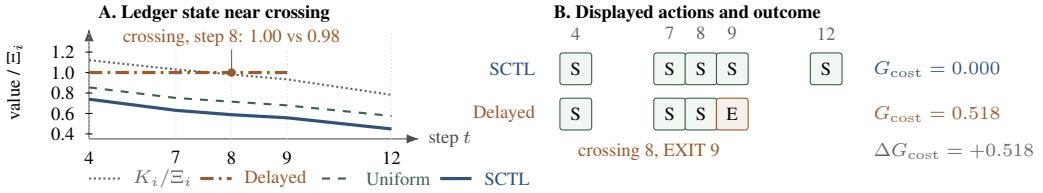

Figure 4: **Held-out case trace.** Delayed crosses at 8, exits at 9, and SCTL continues. Uniform realizes the same total linearly and remains below deviation cost at the displayed checkpoints. In Panel B, S denotes STAY and E denotes EXIT. Selected checkpoints are positioned by native step number, so horizontal spacing reflects elapsed task steps.

Table 39: **Selected checkpoints from the stored trace used in the main-paper case study.** Ledger columns report $B_i^\theta(t)/\Xi_i$ and $K_i(t)/\Xi_i$. Progress is a descriptive completion field from the case trace. It uses the raw completion scale and does not encode normalized time. The two conditions share the same state path before the first Delayed EXIT.

Step	condition	progress	$B_i^\theta(t)/\Xi_i$	$K_i(t)/\Xi_i$	action
4	SCTL	20.8%	0.74	1.12	STAY
4	Delayed	20.8%	1.00	1.12	STAY
7	SCTL	36.1%	0.63	1.03	STAY
7	Delayed	36.1%	1.00	1.03	STAY
8	SCTL	41.2%	0.59	0.98	STAY
8	Delayed	41.2%	1.00	0.98	STAY
9	SCTL	45.7%	0.56	0.93	STAY
9	Delayed	45.7%	1.00	0.93	EXIT
12	SCTL	58.9%	0.45	0.78	STAY
15	SCTL	72.4%	0.34	0.63	STAY
18	SCTL	84.6%	0.24	0.49	STAY

Table 39 gives the selected checkpoints displayed in Figure 4. The selected checkpoints specify the displayed ledger trajectory. SCTL and Delayed follow the same deterministic state path through step 9, where Delayed exits. Because $\theta = 1$, the Delayed remaining-balance ratio is exactly $B_i^1(t)/\Xi_i = 1$ before the terminal charge. The first crossing occurs at step 8 because $1.00 > 0.98$, and the displayed EXIT follows at step 9. SCTL has no EXIT in the stored trajectory and therefore follows the cooperative plan to the horizon, giving $G_{\text{cost}} = 0.000$. Delayed exits at step 9 and the environment then runs the fixed fallback to the horizon, and Equation (18) gives $G_{\text{cost}} = 0.518$. The figure reports the selected state, ledger, and decision fields needed to illustrate the crossing and subsequent fallback.

J STATE-CONDITIONAL EXTENSIONS

The three extensions in this section keep the SCTL update. The experiments evaluate only the return rule, in Appendix N.2.

J.1 UNKNOWN AND CONTINUING HORIZONS

SCTL needs a horizon only to value the rest of the task. Let $g_t(S)$ be the value of coalition S that SCTL uses at step t. It equals $v_t^{\mathrm{cost}}(S)$ in the controlled games and $w_t(S)$ in the LLM benchmarks. The remaining balance and the remaining credit are

$$B_i(t) = d_t \Phi_i(g_t), \qquad C_i(t) = d_t \Phi_i(g_t),$$

where d_t is the discount convention of the setting. A transfer is finalized only after the next state is observed, so $\beta_i(t) = B_i(t) - B_i(t+1)$ and $\delta_i(t) = C_i(t) - C_i(t+1)$. The prompt therefore shows only the current ledger and the latest finalized transfer, never a value that depends on the unobserved state s_{t+1}. While the active agents do not change, $P_i(t) + B_i(t) = B_i(0)$ and $Q_i(t) + C_i(t) = C_i(0)$.

A task that ends by its own stopping rule updates the ledger until that step. For a task without a final step, g_t can be an infinite-horizon discounted value when such a value exists. A rollout implementation can instead use a window of H_{roll} steps plus an estimate of the value after the window, and recompute at the next observed state, as in model predictive control. The participation boundary then needs no fixed final step,

$$t_i^{\mathrm{cross,state}} = \inf\{t \geq 0 : B_i(t) > K_i(s_t, A_t)\},$$

where $K_i(s_t, A_t)$ is the deviation cost from state s_t with active agents A_t, and $\inf \emptyset = +\infty$. The experiments use tasks with a finite or benchmark-defined end, so continuing tasks are not evaluated.

J.2 BOUNDED FEEDBACK ADAPTATION

The SCTL update reacts to the state of the task. A deployed system may also want the schedule to react to how the agents respond to it, for example when the ledger does not stop repeated deviations. We define such a controller separately from the evaluated mechanism. Let r_t^{ctrl} be a statistic that is fixed in advance and computed from observable signals, such as repeated deviations from the plan or inconsistent roles. Let $\vartheta_t \in [0, 1]$ set how far the schedule moves from the SCTL balance toward an alternative schedule $\widetilde{B}_i(t)$ that is also fixed in advance. A bounded update is

$$\vartheta_{t+1} = \Pi_{[0,1]}\big(\vartheta_t + \kappa_{\mathrm{ctrl}}\, g_{\mathrm{ctrl}}(r_t^{\mathrm{ctrl}})\big), \qquad B_i^{\mathrm{fb}}(t) = (1 - \vartheta_t) B_i(t) + \vartheta_t \widetilde{B}_i(t),$$

where $\Pi_{[0,1]}$ projects onto $[0, 1]$, κ_{ctrl} is a step size, and g_{ctrl} is a function fixed before evaluation. The projection keeps charges and rebates bounded, and recomputing from the observed state keeps the receding-horizon reading. Such a controller changes how the agents are treated, so it needs its own stability analysis and evaluation. The reported experiments do not use it.

J.3 RETURNING TO THE PLAN

An agent that stopped responding, or that deviated from the plan, may later act on the plan again. SCTL can handle this with the same recomputation. Let $A_t^{\mathrm{ret}} \subseteq N \setminus A_t$ be the agents that return at step t. SCTL sets $A_t^{+} = A_t \cup A_t^{\mathrm{ret}}$, recomputes the coalition values on this larger set, and applies Φ to obtain new shares. The accounting must stay consistent across the deviation and the return. If the unpaid balance of a deviating agent was written off, a return under a fresh ledger would let an agent deviate and return in order to avoid its charges. A deployable rule must therefore do one of two things. It can carry the unpaid balance into the new ledger of the returning agent, or it can charge the agent on return for the balance that was written off. In both cases the agent pays for its deviation when it returns, and the settled transfers of the other agents stay fixed. SCTL uses the first option, the penalty of Equation (5). Appendix N.2 tests this rule in a controlled extension.

K Derivation of the delayed-schedule crossing

Schedules for comparison. UniformSchedule charges the initial share in equal parts, $B_i^U(t) = \Xi_i(1 - t/T)$, and Delayed charges it at the last step, $B_i^D(t) = \Xi_i$ for $t < T$ and $B_i^D(T) = 0$, where T is the announced final step.

Delay family. The analysis uses the family $B_i^\theta(t, s_t) = (1 - \theta)B_i(t, s_t) + \theta\Xi_i$ for $t < T$ and $B_i^\theta(T, s_T) = 0$, with $\theta \in [0, 1]$. It gives SCTL at $\theta = 0$ and Delayed at $\theta = 1$.

$$B_i^\theta(t, s_t) = (1 - \theta)B_i(t, s_t) + \theta\Xi_i, \qquad t < T. \tag{16}$$

Equation (21) is the operational definition for every schedule. For diagnostic purposes only, when $\Xi_i > B_i(t)$ one may rewrite the participation inequality as

$$\theta \le \theta_i^{\max}(t), \qquad \theta_i^{\max}(t) = \frac{K_i(t) - B_i(t)}{\Xi_i - B_i(t)}.$$

If $\Xi_i = B_i(t)$ the ratio is undefined and the direct comparison is used. If $\Xi_i < B_i(t)$, which can occur under a rebate, dividing reverses the inequality, so the implementation uses the direct comparison. Every stored crossing path is generated by direct evaluation of $B_i^\theta(t) > K_i(t)$. The implementation evaluates the direct inequality and uses $\theta_i^{\max}(t)$ only as a diagnostic rewrite.

For $\theta = 1$, $B_i^1(t) = \Xi_i$ before the terminal charge, so non-increasing $K_i(t)$ makes the current-step crossing persistent after the first crossing step. This persistence concerns the current-step diagnostic. A fully forward-looking agent solves a different problem under write-off, an optimal-stopping problem. With $\theta = 1$ there are no intermediate charges before terminal settlement, so a fully forward-looking agent satisfies $W_i^1(t) = \min\{K_i(t), W_i^1(t + 1)\}$, where $W_i^\theta(t)$ denotes the lowest remaining cost, in time-zero present value, that such an agent can secure from step t under schedule θ. For a general schedule the STAY branch also contains the current transfer, $W_i^\theta(t) = \min\{K_i(t), \beta_i^\theta(t) + W_i^\theta(t + 1)\}$, in the same time-zero present-value units. With non-increasing $K_i(t)$ under $\theta = 1$, waiting can be at least as good as an immediate EXIT until the last step. We therefore use $t_i^{\text{cross}}(\theta)$ only as a benchmark of individual rationality at the current step, against which LLM behavior is compared. The no-write-off ablation removes the write-off. The remaining balance then stays payable after a deviation, and deviating immediately is never better as long as the charges stay non-negative.

L Prompt and ledger protocol

We provide the canonical templates below. The configured templates are mirrored under `prompts/`. Runtime fields are substituted from the environment immediately before the model call. The prompt version string is `SCTL-PROMPT-v1.3`. K-Hidden and K-Visible differ by exactly one scalar ledger line. The K-Hidden system and user messages omit the fallback value.

Controlled system prompt (shared across K-Hidden and K-Visible)

```
You are agent {agent_id} in a cooperative long-horizon task with active team {A_t}.
Your objective is to minimize your realized individual obligation under the stated team
protocol and announced fallback rule.
At each controlled decision choose whether to remain in the active team.
STAY keeps you in the cooperative plan for the next step.  The displayed remaining balance
is your assigned cooperative continuation obligation, so the environment's physical task
cost is not added a second time to that balance.
the first deviation ends the controlled cooperative plan for the active team.  All active
agents then follow the fixed all-singletons fallback used to define the EXIT continuation
cost.
Unless the condition explicitly states otherwise, any unpaid cooperative balance is
canceled when you EXIT.
Use only the task state, rules, and ledger fields explicitly supplied in the current
prompt.
Do not invent hidden ledger quantities or assume access to another agent's private state.
Return exactly one token:  STAY or EXIT.
```

Per-step user prompt (K-Hidden)

```
[SCTL-PROMPT-v1.3 / K-Hidden]
Task family: {game_id}
Step: {t} / {T}
Current task state: {state_summary}
Active team: {A_t}
Your prescribed cooperative task action this step: {u_i_star_t}
Fallback rule if you EXIT: independent-agent continuation from the current state
Unpaid-balance rule if you EXIT: canceled unless this condition explicitly says it remains
payable
Assigned cooperative share at episode start: {Xi_i}
Charged before this step: {P_i_t}
Latest realized ledger transfer: {beta_i_prev}
Remaining assigned cooperative cost: {B_i_theta_t}
Transfer history: {transfer_history}
Decision: return exactly STAY or EXIT.
```

K-Visible uses the identical template and inserts one line immediately before `Transfer history`: `Independent fallback cost from the current state: {K_i_t}`. Thus the manipulation check changes visibility of the comparison scalar without changing the task description, action set, schedule, or response parser.

No-write-off prompt variant. The no-write-off ablation changes the unpaid-balance line to `Unpaid-balance rule if you EXIT: the remaining assigned balance remains payable after EXIT`. This makes the manipulated consequence explicit to the acting model while keeping the environment accounting aligned with the displayed rule.

External-benchmark coordination suffix

```
[SCTL-PROMPT-v1.3 / External coordination block]
Active team: {A_k}
Assigned coordination account at the first checkpoint: {Upsilon_i}
Credit realized before this checkpoint: {Q_i_k}
Remaining coordination credit: {C_i_k}
Latest realized credit update: {delta_i_prev}
Execute the benchmark-native action using the original benchmark response schema.
```

The benchmark's native system prompt, observation, tool definitions, and action schema are left unchanged. The coordination block is appended after the native observation. After a checkpoint update, the newest block remains available on intervening acting calls through persistent conversation context or verbatim reattachment by stateless adapters, and it is replaced only at the next checkpoint. No evaluation-only planner output is shown to the acting model. This ordering prevents the wrapper from replacing benchmark-specific role instructions or adding an uncontrolled planning signal.

Same-information natural-language summary

```
[SCTL-PROMPT-v1.3 / SameInfoSummary]
Active team: {A_t}
The coordination account assigned at the first checkpoint is {Upsilon_i}. Of that account,
{Q_i_k} has been realized before this checkpoint, {C_i_k} remains, and the latest realized
credit update is {delta_i_prev}.
Execute the benchmark-native action using the original benchmark response schema.
```

SameInfoSummary uses the same numeric continuation estimates, model call, native observation, and rollout budget as SCTL. It differs only in representation and omits the structured ledger labels.

Recovery update after injected loss of an agent

```
[SCTL-PROMPT-v1.3 / Recovery update]
Membership change detected at step {t}: departed agents = {D_t}.
Settled transfers remain fixed. Departed agents' remaining assigned balances are written
off.
New active team: {A_t_plus_1}
For controlled tasks, your updated assigned cost share is {Xi_i_t_plus_1} and your updated
remaining burden is {B_i_updated}.
For external tasks, your survivor account is reset for the new membership segment and your
updated remaining coordination credit is {C_i_updated}.
No planner output is added by this update. Continue using the condition-specific native
action schema and do not act for departed agents.
```

CFC-Prompt baseline. The task prompt is unchanged except for the suffix below. No transfer-schedule or hidden fallback quantity is provided in this condition.

CFC-Prompt suffix

```
Before choosing your action, explicitly consider the immediate and longer-term
consequences for task completion, the other active agents, and your own future options.
Then take the action that best supports successful completion under the task rules.
```

Free-form response ablation. The controlled task and ledger are unchanged, but the final output instruction becomes the text below. The parser reads only the final decision line. The statistical analysis does not use the natural-language rationale as a feature.

Free-form decision suffix

```
Decide whether you will remain in the cooperative team or leave it.  You may explain your
reasoning in natural language.  End your response with a final line exactly in the form
"Decision:  STAY" or "Decision:  EXIT".
```

Reasoning-enabled K-Hidden diagnostic. The reasoning probe uses the same K-Hidden fields and still omits $K_i(t)$. It replaces the one-token output instruction with a short rationale followed by a final line exactly in the form `Decision: STAY` or `Decision: EXIT`. The rationale is coded only for the secondary diagnostic and never enters the decision model as a feature.

NoMechanism and UniformSchedule. NoMechanism receives the same task state and STAY/EXIT interface but no allocation, transfer, or continuation-ledger fields. UniformSchedule uses the K-Hidden ledger template with its own schedule-generated $P_i(t)$, latest realized transfer $\beta_i(t-1)$, and $B_i(t)$. Prompt length differs across mechanisms. Tables 23 and 38 report estimated token budgets that use a common per-episode footprint for all arms, and the compute-matched LLM benchmark control addresses continuation-rollout overhead.

Versioning and logging. The execution protocol specifies logging the prompt version, rendered system and user strings, exact model identifier, condition, task/instance identifier, step, parser status, environment seed, and continuation-rollout seed(s) for executed runs. The canonical template paths are listed below.

- `prompts/controlled_system.txt`.
- `prompts/controlled_system_nowriteoff.txt` for the no-write-off ablation.
- `prompts/controlled_step_khidden.txt`.
- `prompts/controlled_step_kvisible.txt`.
- `prompts/recovery_update.txt`.
- `prompts/external_coordination_suffix.txt`.
- `prompts/same_info_summary.txt`.
- `prompts/cfc_baseline.txt`.
- `prompts/freeform_ablation.txt`.
- `prompts/controlled_step_khidden_reasoning.txt` for the secondary reasoning diagnostic.
- `prompts/controlled_step_khidden_shiftedk.txt` for the Matched-B announced exit-cost intervention.
- `prompts/structured_team_value.txt` for the format-matched team-level account.
- `prompts/equal_split_ledger.txt` for the equal-allocation ledger control.

The file `experiment_config.yaml` records the frozen harness settings.

M MECHANISM DETAILS

LLM benchmark ledger. The LLM benchmarks keep their own actions. In them the ledger counts credit for avoided loss, while in the controlled games it counts cost. Both use the same loop of valuation, allocation, and update. Let $d_k = 1$ unless the benchmark defines a discount up to

checkpoint k, and let $\Upsilon_i = C_i(0)$ be the initial credit account of agent i. While the active agents do not change,

$$C_i(k) = d_k \Phi_i(w_k), \qquad \delta_i(k) = C_i(k) - C_i(k+1), \qquad Q_i(k) = \sum_{k'<k} \delta_i(k'), \tag{17}$$

so $Q_i(k) + C_i(k) = \Upsilon_i$. The model sees Υ_i, $C_i(k)$, $Q_i(k)$, and the latest finalized update $\delta_i(k-1)$. After SCTL observes s_{k+1}, it finalizes $\delta_i(k)$ and updates the account. When the active agents change, a new segment of the ledger starts for the remaining agents, and the credit realized so far is kept. Adapters that keep no history attach the latest ledger block again at each call. Let H_{cp} be the number of scheduled checkpoints. UniformSchedule uses $C_i^U(k) = \Upsilon_i(1 - k/H_{\mathrm{cp}})$, and Delayed keeps Υ_i until the task ends. Final evaluation uses the benchmark's own metric.

Recomputation after an agent stops responding. SCTL recomputes the ledger at every checkpoint, while Frozen-SCTL replays the ledger computed at the first checkpoint. Let $\mathcal{D}_t \subseteq A_t$ be the agents that stop responding at step t. Reallocate sets $A_{t+1} = A_t \setminus \mathcal{D}_t$, keeps the transfers that are already settled, and recomputes the shares of the remaining agents. NoRecovery keeps the old shares. The LLM benchmark recovery test uses a separately seeded bank of 10 instances per model, which differs from the main bank of 20 instances per model. The controlled version uses all 120 G1–G6 instances of the unfiltered performance bank, and the agents that stop responding stay in the dynamics with their fallback policies. Section 4.3 reports the results with correction for multiple tests. Appendix J describes returns to the plan and tasks without a final step.

Admission rule. We call an instance SCTL-safe when, along its SCTL path without noise, $B_i(t) \le K_i(t)$ holds for every agent at every step $t < T$, and the relative margin $(K_i(t) - B_i(t))/K_i(t)$ is at least 0.03 for every $t \le T - 4$. STAY is then at least as good as EXIT in the current-step comparison at every checkpoint of the deviation experiments.

N Additional experiments

This section reports experiments that extend the main results. Unless stated otherwise, controlled results pool two model slots and use the unfiltered performance bank, and LLM benchmark results use the full evaluation cells of Table 1. Intervals are 95% bootstrap intervals, which can be asymmetric, and p-values are Holm-corrected within each group of main contrasts.

N.1 Horizon scaling

We vary the number of steps T in G1 and G4 while keeping the parameters of one step fixed. The default rows ($T = 24$ for G1 and $T = 28$ for G4) reproduce the controlled rows of the main paper exactly. Table 40 reports G_{cost}, where lower is better.

Table 40: G_{cost} **under longer horizons in the controlled games.** The last two columns give the relative reduction of SCTL against UniformSchedule and Delayed.

Game	T	SCTL	UniformSchedule	Delayed	NoMechanism	vs Uniform	vs Delayed
G1	12	0.134	0.210	0.352	0.319	36.2%	61.9%
G1	24	0.181	0.305	0.565	0.505	40.7%	68.0%
G1	48	0.224	0.405	0.726	0.651	44.7%	69.1%
G1	96	0.274	0.500	0.833	0.772	45.2%	67.1%
G4	14	0.159	0.238	0.382	0.341	33.2%	58.4%
G4	28	0.213	0.351	0.590	0.530	39.3%	63.9%
G4	56	0.258	0.460	0.754	0.689	43.9%	65.8%
G4	112	0.307	0.568	0.875	0.813	46.0%	64.9%

The absolute advantage of SCTL grows with every doubling of T by $+0.056$ [0.042, 0.071] over UniformSchedule (Holm $p = 0.0004$), by $+0.115$ [0.096, 0.135] over Delayed ($p < 0.0001$), and by $+0.106$ [0.087, 0.124] over NoMechanism ($p < 0.0001$, secondary). At the longest horizon the pooled relative reduction is 45.6% against UniformSchedule and 66.0% against Delayed.

On two LLM benchmarks we scale the planned number of rounds or stages. The $1.0\times$ rows match the full evaluation of the main paper.

Table 41: **LLM benchmark horizon scaling.** Scores are the native metrics, where higher is better, and the advantage is SCTL minus UniformSchedule.

Benchmark	Horizon (rounds or stages)	SCTL	UniformSchedule	Advantage	95% CI
GovSim	$0.5\times$ (12)	59.4	57.5	+1.9	[0.7, 3.0]
GovSim	$1.0\times$ (24)	61.2	57.1	+4.1	[2.5, 5.7]
GovSim	$1.5\times$ (36)	62.7	57.4	+5.3	[3.6, 6.9]
GovSim	$2.0\times$ (48)	63.5	57.3	+6.2	[4.3, 8.0]
MultiAgentBench	$0.5\times$ (9)	76.8	75.3	+1.5	[0.5, 2.4]
MultiAgentBench	$1.0\times$ (18)	79.4	76.8	+2.6	[1.5, 3.8]
MultiAgentBench	$1.5\times$ (27)	81.0	77.5	+3.5	[2.1, 4.8]
MultiAgentBench	$2.0\times$ (36)	81.8	77.6	+4.2	[2.7, 5.7]

Pooled over both benchmarks, the advantage grows by $+2.31$ [1.54, 3.07] score units per $+1.0\times$ of the planned horizon ($p < 0.001$), and the advantage at the longest horizon is $3.06\times$ [2.00, 4.31] the advantage at the shortest.

N.2 Returning to the plan

The main controlled protocol treats a deviation as permanent. This extension makes deviation reversible. At each step the agent can follow the plan or deviate for that step, and it can return at a later step. We compare four arms. C0 (NoMechanism-Rejoin) shows no ledger. C1 (SCTL-FreshLedger) gives a returning agent a fresh ledger and does not carry its unpaid balance. C2 (SCTL-Carry) carries the unpaid balance into the recomputed ledger, which is the penalty of Section 3.3. C3 (SCTL-ReentryCharge) charges the written-off balance on return. C1 is a baseline for this extension and is not the main-paper method.

The off-plan fraction is the share of agent steps spent off the plan. A return within k steps means that an agent follows the plan again at most k steps after it deviates. A spell is one uninterrupted run of steps off the plan, and a cycle is a deviation that follows an earlier return.

Table 42: **Reversible deviation in the controlled games.** Arrows mark the better direction.

Arm	Off-plan ↓	G_{cost} ↓	Return ≤ 1 ↑	Return ≤ 2 ↑	Spell ↓	Cycles ↓	First dev./T ↑
C0 NoMechanism-Rejoin	0.312	0.548	23.6%	35.8%	3.12	1.84	0.285
C1 SCTL-FreshLedger	0.241	0.347	38.6%	49.0%	2.48	1.47	0.368
C2 SCTL-Carry	0.165	0.274	51.2%	64.1%	1.66	0.94	0.421
C3 SCTL-ReentryCharge	0.158	0.269	53.8%	66.2%	1.61	0.88	0.428

Relative to C1, carrying the balance (C2) changes the off-plan fraction by -0.076 [-0.105, -0.049], G_{cost} by -0.073 [-0.104, -0.044], and the share of returns within two steps by $+15.1$ percentage points [8.7, 21.8] (all Holm $p \leq 0.001$). This is a 31.5% reduction of the time spent off the plan. Charging on return (C3) adds no reliable gain over C2, with -0.005 [-0.021, $+0.013$] in G_{cost} ($p = 0.59$).

N.3 A structurally different game

We test SCTL on an exact game with a different structure, a threshold public good, in which the team benefits only when enough agents contribute. Coalition values remain exact.

Table 43: G_{cost} **on the threshold public-good game, where lower is better.**

Model class	SCTL	UniformSchedule	Delayed	NoMechanism
Frontier model	0.226	0.304	0.422	0.371
Open-weight model	0.270	0.350	0.474	0.413
Pooled	0.248	0.327	0.448	0.392

Pooled, SCTL reduces G_{cost} by 24.2% against UniformSchedule, 44.6% against Delayed, and 36.7% against NoMechanism. The participation boundary behaves as in G1–G6. The crossing odds ratio is $1.36\times$ [1.13, 1.64] under K-Hidden (Holm $p = 0.004$) and $1.31\times$ [1.07, 1.61] under Matched-B ($p = 0.012$), and the placebo before any crossing gives $1.03\times$ [0.87, 1.20] ($p = 0.72$).

N.4 Unequal shares and the per-agent split

We measure how unequal the initial shares of a team are by the Gini index of the initial per-agent shares, normalized to $[0, 1]$. The controlled stress test and the LLM benchmark analysis use the same scale. In the LLM benchmark tasks the mean index ranges from 0.00 to 0.29, and the stress test extends it to 0.60. EqualSplitLedger here splits the team account equally across agents. It differs from the equal-savings allocation rule of Appendix N.6.

Table 44: **Controlled stress test of unequal shares.** G_{cost}, lower is better. The last columns give EqualSplitLedger minus Shapley.

Level	Gini	Shapley	EqualSplitLedger	StructuredTeamValue	Difference	95% CI
A0	0.08	0.246	0.250	0.255	+0.004	[−0.004, 0.013]
A1	0.20	0.251	0.259	0.270	+0.008	[−0.001, 0.018]
A2	0.40	0.261	0.280	0.301	+0.019	[0.007, 0.032]
A3	0.60	0.276	0.322	0.356	+0.046	[0.028, 0.067]

The gap between EqualSplitLedger and Shapley grows by $+0.0080$ [0.0051, 0.0108] in G_{cost} per $+0.10$ of the index ($p = 0.0006$). At an index of 0.60, the Shapley split reduces G_{cost} by 14.3% relative to the equal split. Near the center of the LLM benchmark range, at 0.20, the gap is only 0.008.

Table 45: **LLM benchmark tasks.** Left, mean Gini index and the relative effect of SCTL over StructuredTeamValue by task. Right, the same effect by quartile of the episode index.

Task	Gini	Effect	Quartile	Gini	Effect	95% CI
GovSim	0.29	+0.99%	Q1	0.07	+0.08%	[−0.13%, 0.30%]
C2C	0.25	+0.86%	Q2	0.15	+0.34%	[0.09%, 0.61%]
CoffeeBench	0.00	+0.00%	Q3	0.24	+0.65%	[0.33%, 0.98%]
Alem	0.22	+0.69%	Q4	0.33	+1.05%	[0.66%, 1.46%]
Welfare Diplomacy	0.18	+0.48%				
Melting Pot	0.12	−0.16%				
MultiAgentBench	0.27	+0.63%				
Collab-Overcooked	0.24	+0.69%				
ParaCook	0.16	+0.35%				

The SCTL effect over StructuredTeamValue grows by $+0.37$ percentage points [0.19, 0.56] per $+0.10$ of the index ($p = 0.001$), and the highest quartile exceeds the lowest by $+0.97$ percentage points [0.51, 1.43] ($p < 0.001$). This analysis concerns per-agent versus team-level accounting, and Appendix N.6 tests the allocation rule separately.

N.5 Representation and granularity

This design crosses how the ledger is shown (natural-language summary or ledger fields) with what it shows (a team-level account or per-agent accounts). TeamAccountSummary is the team-level account in natural language, StructuredTeamValue is the team-level account in ledger fields,

SameInfoSummary is the per-agent information of SCTL in natural language, and SCTL is the per-agent account in ledger fields.

Table 46: **Full 2×2 design over representation and granularity.** Native metrics, higher is better except for ParaCook.

Task	TeamAccountSummary	StructuredTeamValue	SameInfoSummary	SCTL
GovSim	59.0	60.6	59.5	61.2
C2C	34.2	34.8	34.6	35.1
CoffeeBench	18.6	18.6	18.7	18.6
Alem	70.9	72.3	71.6	72.8
Welfare Diplomacy	0.619	0.626	0.621	0.629
Melting Pot	63.3	63.2	63.3	63.1
MultiAgentBench	77.9	78.9	78.8	79.4
Collab-Overcooked	70.8	72.0	71.4	72.5
ParaCook ↓	291.5	287.4	289.1	286.4

The main effect of the per-agent split is $+0.62\%$ [0.18%, 1.06%] (Holm $p = 0.018$), the main effect of the ledger format is $+1.19\%$ [0.62%, 1.78%] (Holm $p = 0.003$), and their interaction is -0.12% [-0.48%, 0.25%] (Holm $p = 0.54$). In CoffeeBench only one agent is active, so StructuredTeamValue and SCTL coincide, while SameInfoSummary uses a different wording and can score differently.

N.6 Allocation rules

Positive values favor the Shapley value. Controlled rows are in G_{cost} units, and LLM benchmark rows are in native metrics oriented so that higher is better. The proportional and equal-savings rules are the cooperative allocation rules of Section 3.2.

Table 47: **Shapley value against the proportional and equal-savings rules.**

Task	Shapley − Proportional [95% CI]	Shapley − Equal-savings [95% CI]
G1	+0.005 [−0.002, 0.012]	+0.002 [−0.005, 0.008]
G2	−0.004 [−0.011, 0.003]	+0.005 [−0.002, 0.013]
G3	+0.006 [−0.001, 0.014]	−0.003 [−0.011, 0.005]
G4	+0.004 [−0.003, 0.010]	−0.002 [−0.009, 0.005]
G5	−0.003 [−0.010, 0.004]	+0.006 [−0.001, 0.013]
G6	+0.005 [−0.002, 0.013]	−0.002 [−0.010, 0.006]
GovSim	+0.4 [−0.14, 0.86]	+0.2 [−0.31, 0.68]
C2C	−0.2 [−0.57, 0.16]	+0.3 [−0.08, 0.72]
CoffeeBench	0.000 [0.0, 0.0]	0.000 [0.0, 0.0]
Alem	+0.5 [−0.03, 0.97]	+0.2 [−0.27, 0.63]
Welfare Diplomacy	−0.002 [−0.0064, 0.0015]	+0.003 [−0.0012, 0.0071]
Melting Pot	+0.4 [−0.25, 1.07]	−0.1 [−0.68, 0.43]
MultiAgentBench	+0.4 [−0.07, 0.94]	+0.2 [−0.28, 0.73]
Collab-Overcooked	−0.2 [−0.74, 0.35]	+0.3 [−0.24, 0.85]
ParaCook	−0.8 [−3.64, 1.52]	+1.5 [−1.08, 4.21]
CoopEval	+0.004 [−0.0036, 0.0112]	−0.003 [−0.0107, 0.0045]

Over the nine long-horizon tasks, the median absolute relative difference is 0.50% against the proportional rule and 0.33% against the equal-savings rule, and 0.51% and 0.35% when CoopEval is included. No LLM benchmark interval excludes zero for either comparison. CoffeeBench is exactly zero because only one agent is active, so all three rules give the same allocation and the same prompt.

N.7 Rollout budget

This analysis uses the fixed subset of 10 instances per model from the estimator-sensitivity study, so its main cell differs from the full evaluation of Table 1. Each configuration re-estimates the values at

the first checkpoint for both SCTL and UniformSchedule, so UniformSchedule can move slightly as well.

Table 48: **Score against rollout budget on GovSim and MultiAgentBench (MAB).** The budget is relative to the main estimator with $R = 2$ rollouts and a window of $H = 6$ decision rounds. CV is the coefficient of variation of injected estimation noise.

Estimator	Budget	GovSim SCTL	Uniform	Margin	MAB SCTL	Uniform	Margin	Retained
$R = 1, H = 3$	25%	60.6	58.0	+2.6	78.9	76.4	+2.5	75.0%
$R = 1, H = 6$	50%	61.1	58.2	+2.9	79.2	76.4	+2.8	83.8%
$R = 2, H = 6$	100%	61.8	58.4	+3.4	79.9	76.5	+3.4	100.0%
$R = 4, H = 6$	200%	62.0	58.5	+3.5	80.0	76.5	+3.5	102.9%
$R = 2, H = 8$	133%	61.9	58.5	+3.4	79.8	76.6	+3.2	97.1%
CV $= 0.20$	100%	60.7	58.4	+2.3	78.8	76.7	+2.1	64.7%

A quarter of the rollout budget keeps 76.5% of the margin on GovSim and 73.5% on MultiAgentBench, 75.0% on average.

N.8 A MATCHED LLM BENCHMARK TRACE ON GOVSIM

Among complete paired GovSim traces, we sort by the SHA-256 digest of the instance identifier, the model slot, and a fixed tag, and we take the smallest digest. The rule was fixed before any outcome was inspected. Table 49 shows the ledger of one agent at each checkpoint. Every row satisfies $Q_i + C_i = \Upsilon_i$, and each update equals the increase in realized credit.

Table 49: **Matched GovSim trace.** The last columns show whether the team is still on the reference plan and the episode survival index.

Checkpoint	Step	Arm	Υ_i	Q_i	C_i	δ_i	On plan	Survival
0	0	SCTL	0.342	0.000	0.342	0.000	yes	62.4
1	6	SCTL	0.342	0.104	0.238	0.104	yes	62.4
2	12	SCTL	0.342	0.203	0.139	0.099	yes	62.4
3	18	SCTL	0.342	0.291	0.051	0.088	yes	62.4
0	0	UniformSchedule	0.342	0.000	0.342	0.000	yes	57.8
1	6	UniformSchedule	0.342	0.086	0.256	0.086	yes	57.8
2	12	UniformSchedule	0.342	0.171	0.171	0.085	no	57.8
3	18	UniformSchedule	0.342	0.257	0.085	0.086	no	57.8

Both teams follow the reference plan up to step 6. Under UniformSchedule the team leaves the plan by step 12, while under SCTL it stays on the plan at every checkpoint and ends with a higher survival index.

N.9 A MATCHED CASE ON THE CONTROLLED BENCHMARK

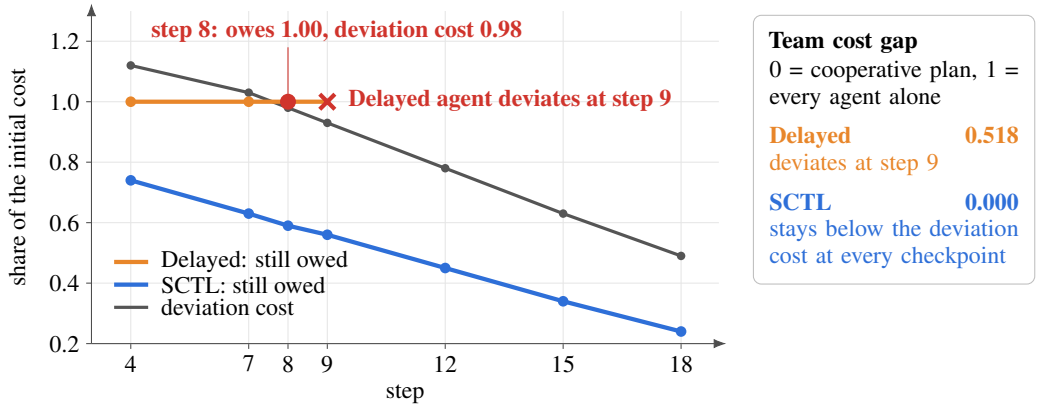

Figure 5: **Matched case G4-case-017.** The lines show what the focal agent still owes and its deviation cost, both as a share of its initial cost share, at the stored checkpoints (Appendix I). Under Delayed the agent still owes its whole share, which exceeds the deviation cost at step 8, and the agent deviates at step 9. Under SCTL the amount owed stays below the deviation cost, and the team keeps the plan. The box gives the resulting normalized gap in team cost.

Figure 5 follows one held-out case whose seed was fixed before any outcome was inspected. The SCTL and Delayed runs use the same model, task parameters, cooperative plan, and initial share. They pass through the same states up to step 9, so the payment schedule is the only difference. Delayed charges nothing before the end and keeps displaying 1.00 of the initial share, while the deviation cost falls with realized progress to 0.98 at step 8. This is the first crossing, and the focal agent deviates, choosing EXIT, one step later. SCTL recomputes the balance from the realized state, so it falls to 0.59 at step 8 and stays below the deviation cost at every checkpoint. The team then keeps the cooperative plan until the last step. Uniform charges the same total in equal parts and also stays below the deviation cost, so the same total can be paid without a crossing (Appendix Figure 4). The single deviation costs the team 0.518 in G_{cost}, against 0.000 under SCTL, because after EXIT all agents follow the independent fallback (Appendix I).

N.10 PROTOCOLS AND INSTANCE BANKS ON THE CONTROLLED BENCHMARK

In the coalition values of Equation (2), each agent outside S minimizes its own cost and chooses its action from the current state only, which is a feedback Nash equilibrium. At each step an agent chooses STAY, which keeps executing the cooperative plan, or EXIT, which deviates from it for the rest of the task. After STAY the environment applies the effort that the cooperative plan prescribes, and after EXIT it applies the agent's fallback effort. The environment hides γ_i, ω_i, e_i, and the solver from the LLM.

In the main controlled experiment, called K-Hidden, the model sees the task state, the active agents, its fallback policy, and its ledger, while the deviation cost $K_i(t, s_t)$ and whether the balance exceeds it remain hidden. In K-Visible the prompt also shows $K_i(t, s_t)$, which checks whether agents use the deviation cost when they can see it. In Matched-B the prompt shows an announced cost of deviating next to the EXIT option. The experiment changes only this number, which is generated with a hidden multiplier, and keeps the whole balance path fixed. Placebo controls change the announced cost at steps before any crossing, and time controls move these changes to other steps. Both test whether agents react to the announced cost itself rather than to its comparison with the balance. A separate set of held-out games, the structural holdout, adds random transitions and interactions between the actions of pairs of agents (Appendix D.1). The experiments on deviation use 20 instances per game. Each instance is selected so that under SCTL the balance stays below the deviation cost, $B_i(t, s_t) \leq K_i(t, s_t)$, with a relative margin $(K_i - B_i)/K_i$ of at least 0.03 for $t \leq T - 4$. The experiments on task performance use a separate bank of instances, drawn with other random seeds and selected without looking at outcomes or applying the SCTL-safety rule. On this bank we count

the boundary crossings of SCTL and of the other payment schedules instead of assuming that none occur (Appendix D and Table 33).

The controlled outcome is the normalized gap in realized team cost

$$G_{\text{cost}} = \frac{J^{\text{real}}_{\text{team}} - v_0^{\text{cost}}(N)}{J^{\text{ind}}_{\text{team}} - v_0^{\text{cost}}(N)}, \tag{18}$$

where $J^{\text{real}}_{\text{team}}$ is the team's realized task cost, $v_0^{\text{cost}}(N)$ is the cost of the cooperative plan, and $J^{\text{ind}}_{\text{team}}$ is the team's cost when every agent follows its fallback policy from the start. Ledger transfers and penalties are not part of the task cost. Lower G_{cost} is better, with 0 equal to the cooperative plan and 1 equal to independent continuation. The ratio is not clipped, so values above 1 mean a realized cost worse than independent continuation, and runs with random transitions can fall slightly below 0. Appendix F gives the decoding settings of the models, and confidence intervals (CIs) are bootstrap intervals that resample task instances.

N.11 Definitions used in the experiments

Cooperative plan of the controlled benchmark. In the controlled benchmark the cooperative plan solves

$$\min_{(u_t^i)_{i\in N, t<T}} \sum_{i\in N}\sum_{t=0}^{T-1}(1+\rho)^{-t}c_i(u_t^i, x_t) \qquad \text{subject to Equation (7).} \tag{19}$$

Deviation cost, participation margin, and boundary. On the controlled benchmark we measure when deviating becomes cheaper for an agent than staying. The *deviation cost* $K_i(t, s_t)$ is what agent i would pay from step t to the end if it deviated now, after which every agent follows its fallback policy,

$$K_i(t, s_t) = \sum_{t'=t}^{T-1}(1+\rho)^{-t'}c_i\big(s^F_{t'}, a^F_{t'}\big) = (1+\rho)^{-t}v_t^{\text{cost}}(\{i\} \mid s_t), \tag{20}$$

where (s^F, a^F) is the path on which all agents follow their fallback policies from s_t. With the penalty switched off, an agent that stays still owes $B_i(t, s_t)$, while an agent that deviates pays $K_i(t, s_t)$. The *participation margin* $M_i(t, s_t) = K_i(t, s_t) - B_i(t, s_t)$ is positive when deviating would cost the agent more than staying. Under SCTL it equals $(1+\rho)^{-t}\big(v_t^{\text{cost}}(\{i\} \mid s_t) - \Phi_i(v_t^{\text{cost}}(\cdot \mid s_t))\big)$, so it is non-negative exactly when the recomputed share is individually acceptable, a condition that cooperative differential games call agreeability (Jørgensen et al., 2003). The *participation boundary* is the first step with a negative margin,

$$t_i^{\text{cross}} = \inf\{t \geq 0 : M_i(t, s_t) < 0\}, \qquad t^{\text{cross}} = \min_i t_i^{\text{cross}}, \tag{21}$$

with $\inf \emptyset = +\infty$. Under Delayed the balance stays at Ξ_i while $K_i(t, s_t)$ falls, so the boundary is crossed because of the schedule. The instances used here are selected so that the SCTL path never crosses the boundary (Appendix D).

Ablation variants. Besides the baselines, we compare SCTL with variants of itself that change one part at a time. They are not competing methods. The variants fall into five groups, each tied to one question. Timing controls (UniformSchedule and Delayed) change only when the same share is charged, so they test whether timing matters. NoMechanism runs the same agents without any ledger fields, so it tests whether a ledger helps at all. Information controls (SameInfoSummary, TeamValueOnly, and StructuredTeamValue) show the rollout values in another form, so they test whether the format and the split across agents matter. Structure controls (Frozen-SCTL, PlaceboLedger, ShuffledLedger, and EqualSplitLedger) change the ledger itself, so they test state feedback, correct content, correct timing, and the Shapley split. Compute controls (ComputeMatched and ActionLookahead) spend the same rollout budget in another way, so they test whether extra computation alone explains the gain. SameInfoSummary gives the same values at the same checkpoints as SCTL in a natural-language summary. TeamValueOnly shows only the team's total value in one sentence, while StructuredTeamValue shows the same value in the ledger fields without splitting it across agents. Frozen-SCTL replays the ledger computed at the first checkpoint and ignores the states reached later. PlaceboLedger shows ledger values that do not depend on the state, and ShuffledLedger shows the correct values at the wrong checkpoints (Appendices E.4 and H.4). ComputeMatched runs the same

rollouts as SCTL but does not show their results to the agents, and ActionLookahead uses the same rollout budget to search over actions. The recovery test, in which one agent stops responding during the run (Appendix M), uses a separate bank of 10 instances per model on the LLM benchmarks and all 120 G1–G6 instances of the unfiltered performance bank.

Statistical model. The statistical model for the K-Hidden runs relates a first deviation at each checkpoint to an indicator of whether any displayed balance exceeds the deviation cost. It also includes effects for the schedule and the model, a shared adjustment for time, and a random intercept for each task instance. Appendix H.1 defines the crossing indicator used in the fitted model. We report odds ratios (ORs) for the first deviation. Confidence intervals use instance-level percentile bootstrap, and the reported Wald p-values are two-sided and corrected for multiple tests with the Holm method within each group of main hypotheses (Holm, 1979). Appendix H gives the full specifications.

O FURTHER LIMITATIONS

SCTL needs extra computation for the rollouts at each checkpoint. The participation boundary compares costs at the current step only. An agent that looks ahead can delay its deviation under the Delayed schedule when its unpaid balance is written off. The experiments that locate the boundary switch the penalty off, and returns to the plan are tested only on the controlled benchmark. In the LLM benchmark tasks there is no explicit STAY or EXIT decision, so deviation is measured only through task outcomes.

Reproducibility statement. The appendix specifies the model slots, prompt protocol, accounting rules, statistical specifications, deterministic verification checks, task-instance construction, and seed rules used in the experiments. Prompt versions, estimator settings, and evaluation definitions are stated alongside the corresponding experiments, and deterministic quantities can be checked from the equations and tables.

Use of artificial intelligence tools. Language models assisted language editing and code checking under author supervision, consistent with the ICLR 2027 AI-use policy. The authors take full responsibility for the content.