Semantic Multiplicity Creates Strategic Flexibility in Streaming AI Agents

Akshaj Agadi[1,*] **Edison Yang**[2,*] **Neel Marripalapu**[3,*] **Puneeth Chandrasekaran**[4]
Andrew Kim[2,†] **Abhineil Gotur**[5,†] **Ishaan Agarwal**[6]

[1]Texas A&M University [2]Lambert High School [3]The University of Texas at Dallas
[4]Independent Researcher [5]Exea Labs [6]Rice University
a0708@tamu.edu ia36@rice.edu

[*]Equal contribution (co-first authors). [†]Equal contribution.

Abstract

Economic models often treat a tool call, bid, or contract as a single choice. Generative AI agents instead construct these actions one symbol at a time. If a counterparty can respond before the message is complete, the action's representation can affect the agent's choices. Several messages may encode the same final action; we call this semantic multiplicity. Extra representations keep an action available after different partial messages, so the agent can choose how to finish after observing a response.

We show that one duplicate representation can raise the guaranteed payoff from zero to one, the full payoff range. The arrangement of menus also matters: two interfaces with the same actions and the same menus at every depth can have opposite optimal first moves. We derive the minimum number and length of representations needed to produce a given tree of choices and show how preserving these choices raises total surplus in procurement. In 20 synthetic procurement games, extra representations raise model payoffs only when the buyer responds between message fields: the interaction is 1.95 for Gemini and 1.40 for DeepSeek, buffered differences are near zero, and larger models replicate the pattern. Both families also solve 17 of 20 matched ancestry games. Finally, in a two-model runtime implementation, the supplier keeps an exposed 200-unit minimum fixed and prices at \$95 or \$105 depending on the buyer's response.

1 Introduction

Economic models often treat a tool call, bid, or contract as a single choice. A generative agent makes such a choice by constructing a message one symbol at a time. The message is the action's representation, and a decoder maps its completed form to the economic action it implements. As the agent writes, an application may expose the part already emitted, called its prefix, and let a counterparty respond before the completed proposal takes effect and the action becomes binding. The response can then influence how the agent finishes. We ask when different representations of the same final action give the agent different strategic opportunities.

Several existing features make this interaction possible. Model APIs stream tool arguments as they are generated [OpenAI, 2026, Anthropic, 2026]; serving systems start executing a tool once enough arguments are available [Xu et al., 2024]; and agent runtimes pause a proposal, collect external review, and resume with the feedback [LangChain, 2026]. Combining these features lets an application obtain a response to an unfinished proposal. Its timing and cancellation rules then determine whether the response arrives before completion and whether the final message must keep the exposed prefix.

40th Conference on Neural Information Processing Systems (NeurIPS 2026). Workshop: Economics for Machine Learning.

After exposure, the actions that can still be completed form the prefix's *continuation menu*. When each action has one representation, choosing a branch rules out every action whose representation follows another branch. Having several representations of the same action is *semantic multiplicity*. In this case, continuation menus after different prefixes can contain the same action, including when neither prefix extends the other. The agent can therefore preserve access to that action while choosing how to finish after a response. Canonicalization keeps one representation per action and removes the alternative paths to that action.

Following these menus as the message grows produces a tree whose *ancestry* records which menus follow which earlier menus. We study how this structure affects strategic choices and how many representations are needed to produce a given tree. Together, the accepted representations and the timing of exposure determine how much flexibility the agent retains.

Our contributions. We first show that a single duplicate representation can raise the guaranteed payoff from zero to one (Theorem 1); Section 2 explains the complete game. The arrangement of menus also matters: changing ancestry can reverse the optimal first move for the matched trees in Figure 2 (Theorem 2). For a specified tree, we then derive the minimum number and length of representations needed to produce it (Theorem 3) and show how preserving these choices can increase surplus in procurement (Proposition 3).

To test these predictions, we run two experiments on Gemini and DeepSeek. The procurement experiment varies both the representations (one per action or with an extra copy) and the timing of the buyer's response (between fields or after the whole message is complete). The second experiment tests whether models change their first choice on the matched trees in Figure 2. We control a source of presentation bias by balancing display order across procurement games and testing both display orders within every ancestry game [Pezeshkpour and Hruschka, 2024, Wei et al., 2024].

2 How representations preserve choices

The smallest useful example has three contracts and two bits. A supplier signs a, b, or c by submitting a two-bit message decoded as $00 \mapsto a$, $01 \mapsto b$, $10 \mapsto b$, and $11 \mapsto c$. After a first bit of 0, the reachable contracts are $\{a, b\}$; after 1, they are $\{b, c\}$.

The two representations keep b available after either first bit. A buyer who responds between the bits can affect how the supplier finishes its message while b remains available. Canonicalization removes one representation of b. If 01 is retained, both a and b remain available after prefix 0, so the supplier can still adapt within this branch. Buffering keeps both bits private until completion. Figure 1 shows the effects of representation and timing.

A complete game. A buyer who minimizes the supplier's payoff first selects a public state $s \in \{0, 1\}$. The supplier chooses its first bit; the buyer observes it and chooses a response $y \in \{0, 1\}$; the supplier then completes the message. The supplier earns one if the decoded contract matches the entry below, and zero otherwise. The buyer receives the negative of this payoff, so the game is

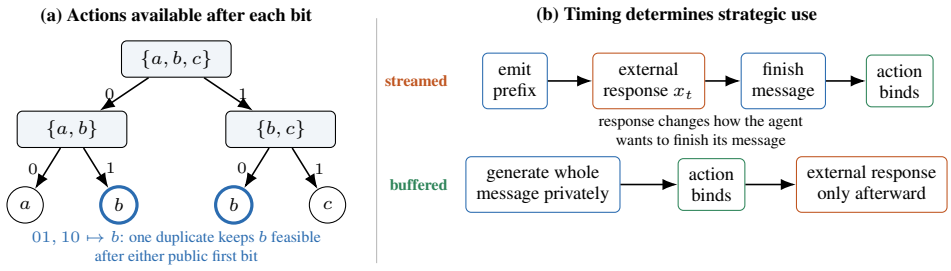


Figure 1: Several representations of one action, and when the buyer responds. In panel (a), one duplicate representation of b creates overlapping menus. In panel (b), streamed exposure permits an intervening response; buffering delays every response until completion.

zero-sum. The moves are *interleaved*: the buyer responds between the two bits.

	$y=0$	$y=1$
$s=0$	a	b
$s=1$	b	c

With both copies of b, the supplier chooses prefix 0 in state 0 and 1 in state 1, then matches either response. The game's value, its largest guaranteed payoff, is one. If $10 \mapsto b$ is deleted, the buyer selects $s=1$: $y=1$ defeats prefix 0, and $y=0$ defeats prefix 1. Deleting $01 \mapsto b$ gives the same failure in state 0. Either canonical codebook has value zero. The initial state lets the buyer select where the missing representation matters.

Theorem 1 (The value of one duplicate representation)**.** *There is a decoder using two bits to represent three actions, with one extra representation ($R=1$), and a zero-sum game with supplier payoffs in $[0,1]$. The supplier can guarantee one with this decoder. Keeping exactly one representation per action reduces its guaranteed payoff to zero, whichever copy is removed.*

Economic interpretation. A representation can preserve access to an action after a particular public history. In Kreps's theory, a menu is valued against uncertain future tastes [Kreps, 1979]. Here the representation determines which menu is available after the buyer's response.

3 Why tree structure matters

The first game shows how much a duplicate can matter. We now ask whether the menus alone describe the available flexibility. To compare interfaces, we use finite games on their continuation trees. At a public node, the agent chooses a child or another player responds, updating the public history. At a leaf, the agent completes any action in its menu. The mover, available choices, and payoffs depend only on the node and public history; we call these rules *representation-blind*.

Under the same rules, trees that match node for node and have the same menus support corresponding strategies and payoffs. Removing representations while retaining every action restricts the agent's choices and cannot increase its zero-sum value. Appendix A states and proves these supporting results as Propositions A.1 and A.2. Any costs or side effects of particular spellings must be included in the public history and payoffs.

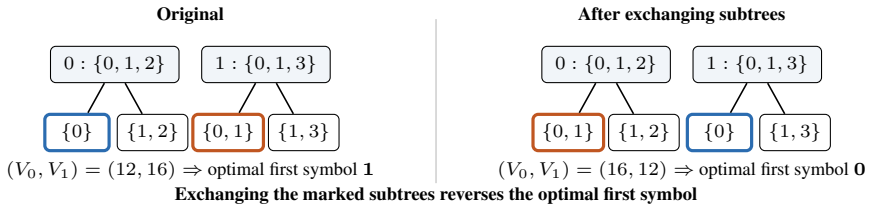
The matched trees in Figure 2 show why ancestry matters. Moving two subtrees changes which later menus the agent can reach from each first choice. The table and calculation below show how this reverses the optimal first symbol.

Theorem 2 (Ancestry can reverse the first choice)**.** *For every message length $H \geq 3$, two binary interfaces can have the same actions, total number of representations, representation count for each action, and menus at every depth, including repeats. The menu after each first symbol is also the same. There is a common zero-sum game in which the opponent responds between symbols and the rules are representation-blind, yet each interface has a unique optimal first symbol and the symbols are opposite.*

Construction and proof sketch. At $H=3$, exchange the marked subtrees in Figure 2. Their parent menus and the lists of menus at each depth are unchanged, while the displayed values give a strict reversal. For $H>3$, append all possible binary strings of length $H-3$ to the base messages. Label each action by the appended string and give these versions the base action's payoff regardless of later responses. The first choice still reverses, and the menus match at every added depth. Appendix A gives the full construction and payoff table; an independent verifier checks message lengths three through seven. □

A summary *ignores ancestry* if it is unchanged by subtree exchanges that preserve all the matching conditions in Theorem 2.

Corollary 1 (Menus at each depth do not determine the optimal choice)**.** *Two interfaces can agree on a summary that ignores ancestry and have different values for their first choices, with opposite unique optimal first symbols.*



Menu M	Value after $x = 0$	Value after $x = 1$
$\{0\}$	19	12
$\{0, 1\}$	19	16
$\{1, 2\}$	16	5
$\{1, 3\}$	6	10

The opponent responds after the first and second bits, choosing x then y. Each table entry is $W_x(M) = \min_y \max_{a \in M} u(a, x, y)$: the opponent chooses y, then the agent selects the best action in M. After each x, the agent takes the better child; the opponent chooses the x giving the lower payoff. For the original left branch, this gives

$$\min\{\max(19, 16), \max(12, 5)\} = 12.$$

Figure 2: A worked ancestry reversal. (V_0, V_1) gives the values after first symbols 0 and 1. Exchanging the marked subtrees leaves the four menu values fixed and reverses the optimal first choice.

Appendix A also gives a payoff table where the same two interfaces have different game values.

Economic interpretation. Strategic value depends jointly on which continuation menus occur and where they occur in public history. Two agents may expose the same menus with the same frequencies at every stage of a negotiation and nevertheless have different strategic values because each menu remains reachable after different histories.

4 How many representations are needed?

To formalize the representation requirements, let $\mathcal{A}$ be a finite nonempty set of economic actions. Each message contains H symbols from an alphabet Σ of size $q \geq 2$. The *codebook* $\mathcal{C} \subseteq \Sigma^H$ is the set of valid messages, and the decoder $D : \mathcal{C} \to \mathcal{A}$ maps each message to its action. Every action has at least one representation. Write $\mathcal{C}(a) = D^{-1}(a)$ for the representations of a and $r_a = |\mathcal{C}(a)|$ for their number. The total number of extra representations is

$$R(\mathcal{C}) = |\mathcal{C}| - |\mathcal{A}| = \sum_{a \in \mathcal{A}} (r_a - 1).$$

A decoder is *canonical* when each action has one representation, so $r_a = 1$ for every a.

The continuation menus in Figure 1 are $\{a, b\}$ after 0 and $\{b, c\}$ after 1. In general, write $p \preceq w$ when p is a prefix of message w. The continuation menu after p is

$$K_{\mathcal{C}}(p) = \{a \in \mathcal{A} : \exists w \in \mathcal{C}(a) \text{ with } p \preceq w\}.$$

Our baseline lets the agent choose its action as it constructs the message. The message is *append-only*: each new symbol extends the prefix already shown, and the completed message must contain it unchanged. The actions available after p are therefore exactly $K_{\mathcal{C}}(p)$. For two prefixes where neither extends the other, their continuation menus can contain the same action through different completions.

To record the choices during public interaction, follow the message one symbol at a time and write the continuation menu at each node. Stop a branch when the remaining symbols become private or the full message is complete. We call the resulting object a *continuation tree*, denoted $\mathcal{T}$. Its leaves L are the points where public interaction ends. At each leaf, the agent may complete any action in its menu privately.

Write K_v for the nonempty menu at node v. The tree has three basic properties. Its root menu is $\mathcal{A}$. Each child's menu is contained in its parent's, and the children together cover the parent menu. Each public node where play continues retains all valid choices of the next symbol. The last requirement is

called *observation completeness*. Appendix A gives the formal definition under the name *commitment skeleton*.

If the action is fixed before emission, prefixes can reveal it but later feedback cannot change it. When the agent chooses the action as it writes, play can take the form

$$s_1,\, x_1,\, s_2,\, x_2, \ldots, s_H, \quad D(s_1 \cdots s_H),$$

where x_t is the response after prefix $s_{1:t}$. The agent then chooses its next symbol using the response. Buffering the remainder ends public interaction at the current node; the counterparty sees the later symbols after the action has been completed.

Cancellation and restart. A costless restart after a response lets the agent choose any action in $\mathcal{A}$ again. Restart or compensation costs can be included as game moves and payoffs; the append-only baseline rules out restart. This requirement complements work on reversing completed actions [Zhai et al., 2026].

The alphabet size q counts branches the application treats as different. One symbol may span several model tokens. A common message length H is a normalization: messages can be padded after the action binds.

Definition 1 (Representing a continuation tree)**.** A decoder represents $\mathcal{T}$ if each node's outgoing edges can be assigned distinct symbols and the resulting prefix p_v at every node has menu $K_{\mathcal{C}}(p_v) = K_v$. At each nonleaf node, the tree must include all valid next symbols. Completion below a leaf may be private.

Given a desired continuation tree, we first ask whether any decoder can produce it. We then count the representations and symbols it needs.

Proposition 1 (When a tree can be represented)**.** *A finite continuation tree $\mathcal{T}$ can be represented over a q-symbol alphabet if and only if every node has at most q children, or equivalently $\Delta(\mathcal{T}) \le q$.*

Each child needs a different next symbol. Conversely, label the edges with distinct symbols and assign a completed message to each action in each leaf menu. Appendix A gives the proof.

Call a tree a *partition tree* when the children of each node have pairwise disjoint menus.

Proposition 2 (Trees with one representation per action)**.** *A decoder is canonical if and only if its full tree is a partition tree. Over a fixed q-symbol alphabet, a continuation tree can be represented by a canonical decoder if and only if it is a partition tree and $\Delta(\mathcal{T}) \le q$.*

When two representations of an action first diverge, the action appears in two child menus. A canonical decoder therefore separates the actions at each branch. If public interaction ends early, duplicates can first diverge during private completion. The public tree can then be a partition tree even when the full decoder contains duplicates.

To count the required copies, observe that each representation follows one path through the tree. Each action therefore needs a representation through every leaf where it must remain available. Let d_ℓ be the depth of leaf ℓ, and let $m_a(\mathcal{T}) = |\{\ell \in L : a \in K_\ell\}|$ count the leaves whose menus include a. The minimum number of representations of a is exactly $m_a(\mathcal{T})$.

In Figure 1, b belongs to both menus after the first bit. If public interaction ends there and completion becomes private, these two nodes are the leaves of the continuation tree. One representation of b cannot pass through both prefixes, so the second representation is necessary. We also need enough symbols below each leaf to write every action in its menu.

Theorem 3 (Minimum number and length of representations)**.** *For a target continuation tree $\mathcal{T}$ over a q-symbol alphabet with $\Delta(\mathcal{T}) \le q$,*

$$\begin{aligned} r_a^{\min}(\mathcal{T}) &= m_a(\mathcal{T}), \\ |\mathcal{C}|_{\min}(\mathcal{T}) &= \sum_{\ell \in L} |K_\ell|, \\ R_{\min}(\mathcal{T}) &= \sum_{\ell \in L} |K_\ell| - |\mathcal{A}|, \\ H_{\min}(\mathcal{T}) &= \max_{\ell \in L} \left[d_\ell + \left\lceil \log_q |K_\ell| \right\rceil \right]. \end{aligned}$$

One decoder attains both the minimum number of representations and the minimum common message length.

Proof sketch. Each completed message crosses exactly one leaf. An action available at $m_a(\mathcal{T})$ leaves therefore needs a representation through each. Summing over actions gives the minimum total count and redundancy. Below leaf ℓ, message length H allows only q^{H-d_ℓ} possible completions, so realizing K_ℓ requires $q^{H-d_\ell} \geq |K_\ell|$. Assign one distinct completion to each pair (ℓ, a) with $a \in K_\ell$ to attain both minima at once. Appendix A gives the full argument. □

An interface designer can use these bounds to check whether a proposed tree fits a message-length or representation budget before implementing its decoder. Figure 1 requires four representations, one extra copy ($R_{\min} = 1$), and two symbols per message ($H_{\min} = 2$). Appendix A also characterizes the trees that fit a given budget.

Surplus in the two-route example. Return to Figure 1 and suppose the two states are equally likely. The supplier observes the state before choosing the first bit, and the buyer then requests either a or b in state 0, or either b or c in state 1, each with probability one half. A match creates one unit of total surplus for buyer and supplier; a mismatch creates zero.

Interface	Probability of matching the request
Both representations of b	100%
Best canonical codebook	75%
Buffered message	50%

Both copies let the supplier satisfy every request. Keeping one copy loses half the requests in one of the two equally likely states. Buffering commits to one contract before the request, matching half the requests in either state.

For unequal state probabilities and match surpluses, let $\Pr(s = 1) = \pi$ and let a match in state s create total surplus v_s. Write $A_0 = (1 - \pi)v_0$ and $A_1 = \pi v_1$ for the states' contributions to expected surplus, and W_M, W_C, W_B for surplus with both copies, the best canonical codebook, and buffering. Then $W_M = A_0 + A_1$, $W_C = A_0 + A_1 - \frac{1}{2}\min_s A_s$, and $W_B = \frac{1}{2}(A_0 + A_1)$. The canonical codebook maximizing total surplus preserves both contracts in the state with larger A_s. If the supplier receives share ρ_s of match surplus, the buyer weights that state by $B_s = (1 - \rho_s)A_s$ and preserves the larger B_s. These different weights can make the buyer and a planner maximizing total surplus prefer different canonical codebooks.

The general surplus formula. The same reasoning applies to any continuation tree. A supplier observes a market state θ, then chooses a leaf menu. A request for action a arrives with probability $p_\theta(a)$. The supplier completes the requested action if it remains available. A match creates total surplus v_θ; a mismatch creates zero.

Write $w_\theta = \Pr(\theta)v_\theta$ for the state's weight in expected surplus. Define $p_\theta(K) = \sum_{a \in K} p_\theta(a)$ as the probability that the requested action belongs to menu K. Canonicalization assigns each action to one leaf where it is available. Let $L_a = \{\ell \in L : a \in K_\ell\}$ and $\mathcal{F}(\mathcal{T}) = \{f : \mathcal{A} \to L : f(a) \in L_a\}$ be the set of these assignments.

Proposition 3 (Surplus from completing contracts after a request)**.** *Expected total surplus is given below for duplicate representations, the best canonical codebook retaining every action, and buffering, respectively:*

$$W_M(\mathcal{T}) = \sum_\theta w_\theta \max_{\ell \in L} p_\theta(K_\ell),$$

$$W_C(\mathcal{T}) = \max_{f \in \mathcal{F}(\mathcal{T})} \sum_\theta w_\theta \max_{\ell \in L} p_\theta\big(f^{-1}(\ell)\big),$$

$$W_B = \sum_\theta w_\theta \max_{a \in \mathcal{A}} p_\theta(a),$$

and $W_M(\mathcal{T}) \geq W_C(\mathcal{T}) \geq W_B$. Moreover, $|\mathcal{F}(\mathcal{T})| = \prod_a m_a(\mathcal{T})$: the number of ways to assign actions to leaves is determined by the same counts $m_a(\mathcal{T})$ that give the representation requirements in Theorem 3.

The three expressions describe when choices become fixed. With duplicate representations, the supplier chooses the best leaf menu after observing θ. A canonical codebook first assigns each action to one leaf through f. Buffering commits the supplier to one action before the request. Appendix A proves the characterization. Thus, m_a gives the copies needed to keep a available, while the other actions sharing these menus determine the surplus.

5 Procurement experiment

Do model policies benefit from extra representations when the buyer responds between fields? We compare realized model payoff with the interface's exact ideal value $V(\mathcal{C})$. Instances, prompts, inference settings, and primary outcomes were specified before model evaluation; Appendix B records the settings.

The first experiment varies representations (one per action or with an extra copy) and timing (buyer response between fields or after completion) on 20 independent synthetic procurement games. Each game uses the same terminal contracts and payoff table across all four conditions. Its canonical codebook is obtained by deleting extra representations while keeping every action available. The first field selects a route through the codebook; the second selects an offer on that route. We select games in which exact backward induction gives opposite unique optimal routes under the two interleaved codebooks and an optimal payoff advantage of at least two units from multiplicity. From the procurement objective, two-field schema, and payoff schedule, the model must derive the menus, values of each route, and optimal route.

Under interleaving, the model first emits only the public routing field. For each possible buyer request, we ask the model to complete the message and score the chosen action. The outcome is the minimum payoff across requests. Under buffering, the complete route–offer message precedes the buyer move and again receives its worst payoff over requests. The primary study evaluates an enumerated set of adversarial buyer responses. The two-provider demonstration in Section 7 uses an LLM buyer. Each game contributes one presentation per condition. Route choice order alternates across the 20 games (ten in each order); action rows, response columns, codebook rows, and suffix choices are shuffled with the generator seed. The design uses 200 planned API calls per model. A failed route receives zero and ends that condition's evaluation. Appendix B lists the four conditions and gives the call accounting.

Write Y_{NI} and Y_{CI} for the noncanonical and canonical payoffs when the buyer responds between fields, and Y_{NB} and Y_{CB} for their buffered counterparts. The interaction I measures how much responding between fields changes the payoff advantage of extra representations. The realized flexibility ratio (RFR) divides I by the gap between the exact optimal values V_{NI} and V_{CI}:

$$I = (Y_{NI} - Y_{CI}) - (Y_{NB} - Y_{CB}), \qquad \mathrm{RFR} = I/(V_{NI} - V_{CI}).$$

Each game is one statistical cluster containing its four conditions. We report the mean interaction and the mean of the 20 within-game ratios $I_j/(V_{NI,j} - V_{CI,j})$. RFR expresses the interaction relative to the gap between optimal values and can exceed one when canonical play is imperfect. We report models separately using bootstrap 95% intervals that resample the 20 games, keeping all four conditions of each game together. The target population is instances from the specified filtered synthetic generator under the tested model settings. The primary models use the same instances and prompts. Each uses its provider's low effort setting, an 8,192-token limit, and structured responses. Noncompletion receives payoff zero, the minimum generated payoff. Supplementary replications keep the prompts and analysis; Appendix B separates them from the primary runs.

Results. Extra representations increase payoffs under interleaving by 1.90 units for Gemini and 1.40 for DeepSeek, compared with -0.05 and 0.00 under buffering. This timing difference yields mean interactions of 1.95 for Gemini (bootstrap 95% interval 1.15–2.80) and 1.40 for DeepSeek (0.35–2.35); both intervals exclude zero. Their RFRs are 0.683 and 0.550. Table 1 reports these contrasts, and Appendix B gives all four condition means and exact ideal values.

Gemini completes all 200 logical API calls. Five DeepSeek route calls exhaust the original allowance and receive zero: four noncanonical and one canonical. A supplementary sensitivity analysis therefore raised only the allowance to 65,536 tokens. All 200 calls then completed: interaction 1.95 (1.25–2.60), ratio 0.700 (0.450–0.900), buffered contrast 0.00, and increase in the interaction relative to the original budget: 0.55 (-0.70–1.90).

Table 1: Procurement results on the same 20 games. The first two rows are primary estimates; † denotes supplementary replications (DeepSeek V4 Pro; Gemini 3.1 Pro Preview). Interl. and Buff. give the payoff advantage of extra representations when the buyer responds between fields and after completion, respectively. Their difference is the interaction. RFR averages the interaction divided by each game's exact optimal-value gap. Brackets give bootstrap 95% intervals; for DeepSeek V4 Pro RFR 1.000 records attainment of the ideal interaction gap in all 20 games.

Model	Interl.	Buff.	Interaction [95% CI]	RFR [95% CI]
Gemini 3.6 Flash	+1.90	-0.05	+1.95 [1.15, 2.80]	0.683 [0.417, 0.967]
DeepSeek V4 Flash	+1.40	+0.00	+1.40 [0.35, 2.35]	0.550 [0.150, 0.900]
DeepSeek V4 Pro†	+2.65	+0.00	+2.65 [2.35, 2.95]	1.000
Gemini 3.1 Pro Preview†	+2.00	+0.00	+2.00 [1.40, 2.55]	0.750 [0.550, 0.900]

The larger-model replications show a similar pattern: both complete all 200 calls and have a buffered contrast of 0.00. DeepSeek V4 Pro yields interaction 2.65 (2.35–2.95) and RFR 1.000; the ideal interaction gap was attained in all 20 games. Gemini 3.1 Pro Preview yields 2.00 (1.40–2.55) and RFR 0.750 (0.550–0.900), with positive interaction in 15/20 games. The interaction persists for both larger models.

6 Ancestry experiment

The second experiment asks whether models choose the optimal first symbol on the matched trees in Figure 2. Each game pairs an original tree with a subtree-exchanged version, as in Theorem 2. The pair shares its payoff table, actions, total representation count, count for each action, menu after each first symbol, and menus at every depth, including repeats. Exact backward induction gives the pair opposite unique first symbols with a gap of at least two. A separate sample contains 20 games, ten each with messages of length five and six. Two counterbalanced displays per interface yield 80 API calls per model. Messages are extended as in the theorem's proof. The benchmark draws fresh payoffs for complete sequences of responses and keeps games in which every response round affects some payoff and, for each original action, changing the appended string can change its payoff. DeepSeek V4 Flash uses provider-defined high effort and a 96,000-token maximum; Gemini 3.6 Flash uses high thinking and the documented 65,536-token (64K) maximum [Google, 2026a]. The primary outcome is success on all four calls: the optimal first symbol for each interface under each display order. We call this a successful *quartet*, and count noncompletion as failure.

Results. Each model family solves 17/20 quartets (85%; Wilson 95% interval 64.0–94.8%). For message lengths five and six, quartet success is 10/10 and 7/10 for DeepSeek and 9/10 and 8/10 for Gemini. Five of the six quartet failures occur with six-symbol messages. In the five calls that completed with an incorrect answer, the optimal choice appeared second in the displayed options. Because the correct option changes position, a fixed preference for the first displayed option cannot pass both display orders. Appendix B reports outcomes per pair and per call, and output lengths.

Shared scoring and reproducibility. Exact backward induction supplies answer keys and ideal values for both experiments. The supplement provides code, instance tables, prompts, model responses, and calibration and inference-budget diagnostics [DeepSeek, 2026, Google, 2026b].

7 Implementation in an agent runtime

In a recorded interaction, the supplier exposes `{"minimum_units":200,`. The buyer can commit to an order quantity (`volume_commitment`) or request a fixed price (`price_lock`). Completing the same saved prefix after each response gives the following quotes:

Exposed field	Buyer response	Final price	Minimum quantity
`minimum_units:200`	Volume commitment	$95	200
`minimum_units:200`	Price lock	$105	200

The volume response comes from the recorded interaction; the price-lock response is a replay from the same prefix. The unfinished price changes while the 200-unit minimum stays fixed.

Our implementation uses a Gemini 3.1 Pro Preview supplier, a separately called DeepSeek V4 Pro buyer, and LangGraph. The supplier submits a quote with fields `unit_price_usd` and `minimum_units`; either JSON member order encodes the same quote [Bray, 2017]. LangGraph saves the first field and value before pausing. After the buyer responds, a new model call chooses the remaining field's value, and the runtime combines it with the saved field. Four timed replays verify prefix preservation and changes in completion based on the response. Appendix B gives the full comparison of representations, timing, and supplier payoffs.

Four conditions make the mechanism possible. E is *persistent exposure*: a prefix is public before completion and replacing it has a cost. R is a response to this prefix before completion. A is the ability to choose the remaining symbols based on the response. M is semantic multiplicity. Buffering or costless cancellation disables E; delaying the response until completion disables R; fixing the full message in advance disables A; canonicalization disables M.

The trace verifier checks three features: whether the runtime supplies E, R, A, M, whether the supplier changes its completion when the buyer's response changes, and whether representation and timing affect payoffs. Table 6 in Appendix B summarizes the available runtime features. Streams expose prefixes, standard human review pauses completed calls, and LangSmith accepts input during a run. The application determines how an unfinished call is retained [LangChain, 2026, Runkle and Trivedy, 2026].

A buyer acting on an exposed field can make withdrawal costly, and buffering delays execution [Rundgren et al., 2020, Brömme, 2026, Xu et al., 2024].

8 Related work and scope

An extensive-form game represents each public choice as a node once the tree is specified [Schelling, 1960, Bizzotto et al., 2026, Halpern et al., 2016]. We ask how the interface's accepted representations produce this tree and how many copies of each action it requires; Proposition A.1 shows which details can be renamed without changing strategies or payoffs.

Models of gradual contributions make partial actions depend on earlier moves [Admati and Perry, 1991]. In Compte and Jehiel [2004], a concession raises the other party's payoff from opting out, inducing gradual concessions through history-dependent outside options. Here the decoder determines which terminal actions remain reachable after an exposed prefix. Switching costs can make commitment endogenous [Caruana and Einav, 2008], and limited- or dynamic-commitment models restrict the player's feasible set [Bizzotto et al., 2026, Bade et al., 2009, Dutta and Ishii, 2016]. Observable-commitment work studies noisy complete actions [Bagwell, 1995, van Damme and Hurkens, 1997]; the signals here are incomplete prefixes. Models of revision and reversibility price deadlines and abandonment [Kamada and Kandori, 2020, Zhai et al., 2026].

Communication-complexity work counts the messages needed to implement or verify social choice [Nisan and Segal, 2006, Segal, 2007]. Theorem 3 does so for representations of a continuation tree. Information design structures messages for receivers [Crawford and Sobel, 1982, Kamenica and Gentzkow, 2011, Doval and Ely, 2020]. Here, additional copies expand the sender's continuation opportunities. Process encodings and computational games refine atomic choice [Nestmann and Pierce, 2000, Gorla, 2010, Halpern et al., 2016], without counting action copies or tracking menu ancestry. Constrained decoding enforces validity [Scholak et al., 2021, Dong et al., 2025, Banerjee et al., 2025]. Work on agents studies tool execution, disclosure, and economic interaction [Schick et al., 2023, Xu et al., 2024, Biswas et al., 2026, Wei et al., 2026, Fish et al., 2024]; benchmarks such as GLEE measure self-gain, efficiency, and fairness across economic environments [Shapira et al., 2024], whereas our paired design isolates representation and timing on fixed payoffs.

Scope. The experiments use synthetic games selected to contain the mechanism and a runtime configured to expose and preserve partial proposals. Deployment frequency and adoption costs (migration, latency, cancellation) remain open. Multiple representations can also preserve choices an observer believes were ruled out [Brömme, 2026].

References

Anat R. Admati and Motty Perry. Joint projects without commitment. *The Review of Economic Studies*, 58(2):259–276, 1991. doi: 10.2307/2297967.

Anthropic. Fine-grained tool streaming. Claude Platform Documentation, 2026. URL https://platform.claude.com/docs/en/agents-and-tools/tool-use/fine-grained-tool-streaming. Accessed 2026-08-18.

Sophie Bade, Guillaume Haeringer, and Ludovic Renou. Bilateral commitment. *Journal of Economic Theory*, 144(4):1817–1831, 2009. doi: 10.1016/j.jet.2009.03.004.

Kyle Bagwell. Commitment and observability in games. *Games and Economic Behavior*, 8(2): 271–280, 1995. doi: 10.1016/S0899-8256(05)80001-6.

Debangshu Banerjee, Tarun Suresh, Shubham Ugare, Sasa Misailovic, and Gagandeep Singh. CRANE: Reasoning with constrained LLM generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 2836–2857. PMLR, 2025. URL https://proceedings.mlr.press/v267/banerjee25a.html.

Anish Biswas, Kanishk Goel, Srivarshinee S, Jayashree Mohan, Alind Khare, Anjaly Parayil, Ramachandran Ramjee, and Chetan Bansal. Sutradhara: An intelligent orchestrator–engine co-design for tool-based agentic inference. arXiv preprint arXiv:2601.12967, 2026. URL https://arxiv.org/abs/2601.12967.

Jacopo Bizzotto, Toomas Hinnosaar, and Adrien Vigier. The limits of limited commitment. *American Economic Journal: Microeconomics*, 18(1):254–280, 2026. doi: 10.1257/mic.20240310.

Tim Bray. The JavaScript Object Notation (JSON) data interchange format. RFC 8259, RFC Editor, 2017. URL https://www.rfc-editor.org/rfc/rfc8259.html.

Arslan Brömme. Canonicalization failures as a recurring vulnerability class: Representation divergence in cryptographic systems and its avoidance. arXiv preprint arXiv:2608.06508, 2026. URL https://arxiv.org/abs/2608.06508.

Guillermo Caruana and Liran Einav. A theory of endogenous commitment. *The Review of Economic Studies*, 75(1):99–116, 2008. doi: 10.1111/j.1467-937X.2007.00470.x.

Olivier Compte and Philippe Jehiel. Gradualism in bargaining and contribution games. *The Review of Economic Studies*, 71(4):975–1000, 2004. doi: 10.1111/0034-6527.00311.

Vincent P. Crawford and Joel Sobel. Strategic information transmission. *Econometrica*, 50(6): 1431–1451, 1982. doi: 10.2307/1913390.

DeepSeek. DeepSeek API: Thinking mode, 2026. URL https://api-docs.deepseek.com/guides/thinking_mode. Accessed 2026-08-17.

Yixin Dong, Charlie F. Ruan, Yaxing Cai, Ziyi Xu, Yilong Zhao, Ruihang Lai, and Tianqi Chen. XGrammar: Flexible and efficient structured generation engine for large language models. In *Proceedings of Machine Learning and Systems*, volume 7, 2025. URL https://proceedings.mlsys.org/paper_files/paper/2025/hash/5c20ca4b0b20b0bd2f1d839dc605e70f-Abstract-Conference.html.

Laura Doval and Jeffrey C. Ely. Sequential information design. *Econometrica*, 88(6):2575–2608, 2020. doi: 10.3982/ECTA17260.

Rohan Dutta and Ryosuke Ishii. Dynamic commitment games, efficiency and coordination. *Journal of Economic Theory*, 163:699–727, 2016. doi: 10.1016/j.jet.2016.03.004.

Sara Fish, Yannai A. Gonczarowski, and Ran I. Shorrer. Algorithmic collusion by large language models. arXiv preprint arXiv:2404.00806, 2024. URL https://arxiv.org/abs/2404.00806. Revised March 2026.

Google. Gemini 3.6 Flash: Model documentation, 2026a. URL https://ai.google.dev/gemini-api/docs/models/gemini-3.6-flash. Accessed 2026-08-17.

Google. Gemini API: Thinking and generation configuration, 2026b. URL https://ai.google.dev/gemini-api/docs/thinking. Accessed 2026-08-17.

Daniele Gorla. Towards a unified approach to encodability and separation results for process calculi. *Information and Computation*, 208(9):1031–1053, 2010. doi: 10.1016/j.ic.2010.05.002.

Joseph Y. Halpern, Rafael Pass, and Lior Seeman. Computational extensive-form games. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 681–698. ACM, 2016. doi: 10.1145/2940716.2940733.

Yuichiro Kamada and Michihiro Kandori. Revision games. *Econometrica*, 88(4):1599–1630, 2020. doi: 10.3982/ECTA15272.

Emir Kamenica and Matthew Gentzkow. Bayesian persuasion. *American Economic Review*, 101(6): 2590–2615, 2011. doi: 10.1257/aer.101.6.2590.

David M. Kreps. A representation theorem for "preference for flexibility". *Econometrica*, 47(3): 565–577, 1979. doi: 10.2307/1910406.

LangChain. Human-in-the-loop middleware. LangChain Documentation, 2026. URL https://docs.langchain.com/oss/python/langchain/human-in-the-loop. Accessed 2026-08-18.

Uwe Nestmann and Benjamin C. Pierce. Decoding choice encodings. *Information and Computation*, 163(1):1–59, 2000. doi: 10.1006/inco.2000.2868.

Noam Nisan and Ilya Segal. The communication requirements of efficient allocations and supporting prices. *Journal of Economic Theory*, 129(1):192–224, 2006. doi: 10.1016/j.jet.2004.10.007.

OpenAI. Function calling: Streaming function calls. OpenAI API Documentation, 2026. URL https://developers.openai.com/api/docs/guides/function-calling#streaming. Accessed 2026-08-16.

Pouya Pezeshkpour and Estevam Hruschka. Large language models sensitivity to the order of options in multiple-choice questions. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2006–2017. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-naacl.130.

Anders Rundgren, Bret Jordan, and Samuel Erdtman. JSON canonicalization scheme (JCS). RFC 8785, RFC Editor, 2020. URL https://www.rfc-editor.org/rfc/rfc8785.html.

Sydney Runkle and Vivek Trivedy. The runtime behind production deep agents. LangChain Blog, 2026. URL https://www.langchain.com/blog/runtime-behind-production-deep-agents. Accessed 2026-08-26.

Thomas C. Schelling. *The Strategy of Conflict*. Harvard University Press, Cambridge, MA, 1960.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pages 68539–68551, 2023. URL https://proceedings.neurips.cc/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9895–9901. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.779.

Ilya Segal. The communication requirements of social choice rules and supporting budget sets. *Journal of Economic Theory*, 136(1):341–378, 2007. doi: 10.1016/j.jet.2006.09.011.

Eilam Shapira, Omer Madmon, Itamar Reinman, Samuel Joseph Amouyal, Roi Reichart, and Moshe Tennenholtz. GLEE: A unified framework and benchmark for language-based economic environments. arXiv preprint arXiv:2410.05254, 2024. URL https://arxiv.org/abs/2410.05254. Version 3, revised March 2026.

Eric van Damme and Sjaak Hurkens. Games with imperfectly observable commitment. *Games and Economic Behavior*, 21(1–2):282–308, 1997. doi: 10.1006/game.1997.0524.

Jiaqi Wei, Xuehang Guo, Pengfei Yu, Xiang Zhang, Wanli Ouyang, Siqi Sun, Qingyun Wang, and Chenyu You. When to think, when to speak: Learning disclosure policies for LLM reasoning. arXiv preprint arXiv:2605.03314, 2026. URL https://arxiv.org/abs/2605.03314.

Sheng-Lun Wei, Cheng-Kuang Wu, Hen-Hsen Huang, and Hsin-Hsi Chen. Unveiling selection biases: Exploring order and token sensitivity in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5598–5621. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.333.

Yechen Xu, Xinhao Kong, Tingjun Chen, and Danyang Zhuo. Conveyor: Efficient tool-aware LLM serving with tool partial execution. arXiv preprint arXiv:2406.00059, 2024. URL https://arxiv.org/abs/2406.00059.

Zhiyuan Zhai, Ming Li, and Xin Wang. Revisable by design: A theory of streaming LLM agent execution. arXiv preprint arXiv:2604.23283, 2026. URL https://arxiv.org/abs/2604.23283.

A Complete proofs and formal definitions

This appendix restates the formal development and gives complete proofs for every numbered result in the main text, together with the auxiliary lemma, corollary, and separation whose complete proofs appear here. Results use the numbering of their main-text counterparts and are cross-referenced accordingly. Supporting results stated only here have appendix numbers.

Streamed action interfaces

Let $\mathcal{A}$ be a finite nonempty set of terminal economic actions and let Σ be a finite alphabet with $q = |\Sigma| \geq 2$. A fixed-horizon streamed action interface is a pair $(\mathcal{C}, D)$, where $\mathcal{C} \subseteq \Sigma^H$ is a nonempty codebook and $D : \mathcal{C} \to \mathcal{A}$ is surjective. The representation class and multiplicity of action a are

$$\mathcal{C}(a) = D^{-1}(a), \qquad r_a = |\mathcal{C}(a)|.$$

The interface's total representation redundancy is

$$R(\mathcal{C}) = |\mathcal{C}| - |\mathcal{A}| = \sum_{a \in \mathcal{A}} (r_a - 1).$$

For every prefix p of at least one codeword, define its continuation menu

$$K_{\mathcal{C}}(p) = \{a \in \mathcal{A} : \exists w \in \mathcal{C}(a) \text{ with } p \preceq w\}.$$

Thus $K_{\mathcal{C}}(p)$ is the feasible-action correspondence after p. At depth t, the reduced marginal cover is

$$\mathcal{M}_t(\mathcal{C}) = \{K_{\mathcal{C}}(p) : |p| = t,\ p \preceq w \text{ for some } w \in \mathcal{C}\},$$

where repeated menu labels are removed.

The full decoder tree contains all valid prefixes through depth H. A strategic truncation selects the prefixes at which representation histories remain publicly payoff-relevant. We impose the following completeness condition, which is needed for the necessity direction below: whenever a selected strategic node is not a strategic leaf, all of its valid one-symbol children are selected. We call such a truncation *observation-complete*. Its leaf set is a prefix-incomparable cut: after a leaf is reached, completion may proceed privately and no descendant of that leaf is another strategic node.

Abstract commitment skeletons and serializability

Definition (Commitment skeleton)**.** A finite commitment skeleton on $\mathcal{A}$ is a finite rooted tree $\mathcal{T} = (V, E, \{K_v\}_{v \in V})$ whose node labels are nonempty subsets of $\mathcal{A}$ and which satisfies:

(i) $K_{\text{root}} = \mathcal{A}$;
(ii) $K_u \subseteq K_v$ whenever u is a child of v;
(iii) $\bigcup_{u \in \text{Ch}(v)} K_u = K_v$ at every nonleaf v.

Its strategic frontier L is its leaf set. Its maximum outdegree is denoted $\Delta(\mathcal{T})$. The skeleton is a *partition skeleton* if sibling labels are pairwise disjoint at every nonleaf.

This abstract target specifies which actions must remain executable after each public history.

Definition (Representing a continuation tree; Definition 1 in the main text)**.** A streamed interface $(\mathcal{C}, D)$ realizes $\mathcal{T}$ over Σ if outgoing edges at each node can be injectively labeled by symbols in Σ, thereby assigning each v a root-to-v prefix p_v, such that

$$K_{\mathcal{C}}(p_v) = K_v \qquad \text{for every } v \in V.$$

At every nonleaf v, the prefixes assigned to its children must be exactly all valid one-symbol extensions of p_v (observation completeness). Prefixes below p_ℓ, $\ell \in L$, are private or otherwise strategically irrelevant and need not be nodes of $\mathcal{T}$.

Lemma (Leaf union)**.** *For every node v of a commitment skeleton,*

$$K_v = \bigcup_{\ell \in L:\ \ell \text{ descends from } v} K_\ell.$$

Proof. Use backward induction on the height of v. The statement is immediate at a leaf. At a nonleaf, substitute the induction hypothesis for every child into the child-cover condition. □

Proposition (When a tree can be represented; Proposition 1)**.** *A finite commitment skeleton $\mathcal{T}$ is realizable over a q-symbol alphabet if and only if $\Delta(\mathcal{T}) \leq q$. Equivalently, an observation-complete strategic truncation of a streamed q-ary decoder is exactly a commitment skeleton of maximum outdegree at most q.*

Proof. For necessity, label each selected prefix by its decoder-induced menu. A valid child prefix can only remove completions, so its menu is contained in the parent's. Observation completeness implies that the valid one-symbol children collectively contain every completion below the parent and hence cover the parent menu. There are at most q distinct one-symbol children. The root menu is $\mathcal{A}$ by surjectivity.

For sufficiency, injectively label each node's outgoing edges by symbols of Σ, and let p_v be the resulting path string. For a leaf ℓ of depth d_ℓ, choose one distinct completion below p_ℓ for every $a \in K_\ell$ and decode that completion to a. A finite common horizon with enough such completions exists; the exact least horizon is established below. Different leaf prefixes are incomparable, so completions chosen below different leaves cannot collide. By the leaf-union lemma, the set of decoded actions below p_v is exactly K_v at every target node. Thus the constructed interface realizes $\mathcal{T}$. □

Canonicality

Call a decoder canonical if $r_a = 1$ for every $a \in \mathcal{A}$.

Proposition (Trees with one representation per action (full tree); first half of Proposition 2)**.** *A fixed-horizon decoder is canonical if and only if its full decoder tree is a partition tree.*

Proof. Suppose first that the decoder is canonical. If an action belonged to two sibling menus, its unique codeword would have to extend two distinct one-symbol children of the same prefix, which is impossible. Since sibling menus always cover the parent, they form a partition.

Conversely, suppose the full decoder tree is a partition tree. If distinct codewords w, w' decoded to the same action a, let p be their longest common prefix. At the next symbol, the strings enter different children of p, placing a in two sibling menus and contradicting partition branching. Surjectivity then gives exactly one codeword per action. □

Proposition (Trees with one representation per action (public tree); second half of Proposition 2)**.** *Over a fixed alphabet of size q, a commitment skeleton admits a canonical realization if and only if it is a partition skeleton and $\Delta(\mathcal{T}) \leq q$.*

Proof. In a canonical realization, an action cannot occur in two sibling menus: its unique codeword traverses only one child. Hence the realized skeleton is a partition skeleton, and injective edge labels require $\Delta(\mathcal{T}) \leq q$.

Conversely, assume $\Delta(\mathcal{T}) \leq q$. Recursive partitioning implies that every action belongs to exactly one frontier label. For each a, assign one distinct completion below the unique leaf ℓ for which $a \in K_\ell$, and decode it to a. The construction in the proposition on representing a tree realizes all target labels and uses exactly one codeword per action. □

Remark. The preceding result is existential. A noncanonical decoder can induce a partition skeleton above the strategic frontier if duplicate representations diverge only below that frontier. A truncated partition skeleton can therefore arise from either a canonical or a noncanonical decoder.

Exact representation resources

Fix a commitment skeleton $\mathcal{T}$ with frontier L, and let d_ℓ denote the depth of leaf ℓ. Define the frontier multiplicity of action a by

$$m_a(\mathcal{T}) = |\{\ell \in L : a \in K_\ell\}|.$$

Theorem (Minimum number and length of representations; Theorem 3)**.** *Fix a q-symbol alphabet and assume $\Delta(\mathcal{T}) \le q$. Among all realizations of $\mathcal{T}$ over this alphabet,*

$$r_a^{\min}(\mathcal{T}) = m_a(\mathcal{T}),$$
$$|\mathcal{C}|_{\min}(\mathcal{T}) = \sum_{a\in\mathcal{A}} m_a(\mathcal{T}) = \sum_{\ell\in L} |K_\ell|,$$
$$R_{\min}(\mathcal{T}) = \sum_{\ell\in L} |K_\ell| - |\mathcal{A}|.$$

If all codewords must have a common horizon over a q-symbol alphabet, then the exact minimum horizon is

$$H_{\min}(\mathcal{T}) = \max_{\ell\in L} \left[d_\ell + \left\lceil \log_q |K_\ell| \right\rceil \right].$$

The codeword-count and horizon minima are jointly attainable.

Proof. A completed codeword crosses exactly one frontier leaf because L is a prefix-incomparable cut. If a must remain feasible at $m_a(\mathcal{T})$ different leaves, at least that many codewords must decode to a. This gives $r_a \ge m_a(\mathcal{T})$ and, after summing, the claimed lower bounds for codeword count and redundancy.

Below a leaf ℓ at depth d_ℓ, a common horizon H provides only q^{H-d_ℓ} distinct suffixes. Realizing every action in K_ℓ therefore requires

$$q^{H-d_\ell} \ge |K_\ell|,$$

which is equivalent to the stated leaf-specific horizon lower bound. Taking the maximum gives the global lower bound.

Set H equal to that maximum. For each pair (ℓ, a) with $a \in K_\ell$, choose a distinct suffix of length $H - d_\ell$ below p_ℓ and decode the resulting codeword to a. The leaf-specific bound supplies enough suffixes; prefix incomparability prevents conflicts between leaves. This uses exactly one codeword per incidence (ℓ, a), attaining both lower bounds simultaneously. □

Corollary (Budget-feasible capacity region)**.** *A target skeleton is implementable over a q-symbol alphabet within maximum redundancy $R_{\max}$ and maximum common horizon $H_{\max}$ if and only if*

$$\Delta(\mathcal{T}) \le q, \qquad R_{\min}(\mathcal{T}) \le R_{\max}, \qquad H_{\min}(\mathcal{T}) \le H_{\max}.$$

Proof. Necessity follows from serializability and the two resource lower bounds. Their joint construction proves sufficiency. □

Strategic information in the history-indexed tree

The baseline is append-only within one proposal: every payload that binds on the active path must extend each public prefix already emitted. Four conditions mark the activation boundary. Let E denote *persistent exposure* before binding: a proper prefix is public and cannot be replaced by a fresh proposal without a payoff-relevant cost; in the append-only baseline, any payload that binds must extend it. Let R be an external response to that prefix, A the agent's ability to condition a later symbol in the same proposal on the response, and M semantic multiplicity. Each condition has a distinct role: E makes public history constrain the active serialization; R inserts an external move; A leaves the suffix responsive; and M keeps an action feasible after incomparable prefixes. Together they define the domain in which duplicate-representation, response-conditioned continuation can generate a payoff-relevant effect.

If, after a response, the agent may discard a prefix p and initiate any fresh codeword at zero cost, its effective menu resets to $\mathcal{A}$ and multiplicity adds no feasible-action capacity. A positive restart or compensation cost can be represented by adding a cancellation action and the corresponding protocol transition and payoff. The present skeleton rules out restart.

We now state exactly which game data the representation quotient retains.

Definition A.1 (Game rules on a continuation tree)**.** Fix a commitment skeleton

$$\mathcal{T} = (V, E, (K_v)_{v\in V}).$$

A finite perfect-information protocol game on $\mathcal{T}$ is a tuple

$$\Gamma = (N, Z, z_0, \iota, (B_i)_{i \in N \cup \{S\}}, g, (u_i)_{i \in N \cup \{S\}}).$$

Here S is the agent; N is a finite set of external players; Z is a finite protocol-state space encoding public external history and timing; and $z_0 \in Z$. A nonterminal public state is $(v, z) \in V \times Z$. The controller $\iota(v, z) \in N \cup \{S\}$ has a finite action set $B_{\iota(v,z)}(v, z)$.

If S moves at a nonleaf v, its legal actions are exactly $\mathrm{Ch}(v)$; choosing child w changes the public state to $(w, g(v, z, w))$. If an external player chooses x, the next state is $(v, g(v, z, x))$. At a frontier leaf ℓ, S privately chooses any $a \in K_\ell$ and play terminates with payoff $u_i(a, z)$ to each player. The induced transition graph is acyclic. The game is *representation-blind* because ι, B_i, g, and u_i are defined on the abstract state (v, z) and never take a literal prefix, edge symbol, private suffix, or codeword spelling as an argument.

The state z is unrestricted beyond finiteness: it can retain the full ordered external history, mover identities, public randomization already realized, and any protocol clock affecting future opportunities. Thus representation-blindness removes spelling while retaining payoff-relevant history.

Let $\phi : \mathcal{T} \to \mathcal{T}'$ be a rooted tree isomorphism with $K_{\phi(v)} = K_v$ for all v. Transport a protocol kernel to $\mathcal{T}'$ by replacing (v, z) with $(\phi(v), z)$, mapping an agent choice w to $\phi(w)$, and leaving external actions, transitions, and payoffs unchanged.

Proposition A.1 (Matching trees give the same strategies and payoffs)**.** *For any representation-blind protocol kernel transported through ϕ, there is a bijection between behavioral-strategy profiles that preserves the distribution of terminal pairs (a, z) and every player's expected payoff. Consequently attainable payoff profiles, Nash equilibria, subgame-perfect equilibria, and their welfare levels correspond. In a two-player zero-sum game, the minimax values are equal.*

Proof. Map a public history ending at (v, z) to the history with identical external actions ending at $(\phi(v), z)$. At an agent move, map child w to $\phi(w)$; at an external move, preserve the action; and at a frontier leaf, preserve a because $K_{\phi(\ell)} = K_\ell$. Induction over the acyclic transition graph yields a bijection of complete histories preserving terminal (a, z) and all payoffs.

Apply the history map at each decision point to transport behavioral action probabilities. This bijects strategy profiles and preserves outcome distributions profile by profile. A profitable deviation on one side therefore transports to one on the other, including in every proper subgame. The equilibrium, payoff, and welfare claims follow. Taking max–min proves the zero-sum specialization. □

The quotient discards literal symbols, complete codewords, multiplicity below the strategic frontier, and private suffix depth. It retains the full action-labeled ancestry and z. Symbol-specific information, costs, or side effects must be added to z, g, or u_i; otherwise the game lies outside the result.

Canonicalization and the value of flexibility

Call $\mathcal{C}' \subseteq \mathcal{C}$ an *action-surjective pruning* when $D|_{\mathcal{C}'} : \mathcal{C}' \to \mathcal{A}$ remains surjective. The protocol is restricted to the retained prefix subtree, with timing and payoff data unchanged at retained histories.

Proposition A.2 (Value after removing representations)**.** *Fix a two-player zero-sum protocol game with representation-blind rules. Every action-surjective pruning satisfies $V(\mathcal{C}') \leq V(\mathcal{C})$.*

Proof. Every agent strategy on the pruned tree is available on the original tree. Any opponent strategy on retained histories can be extended arbitrarily to deleted histories, which the pruned strategy never reaches. Thus pruning only restricts the maximizing player's strategic set, proving the inequality. □

One redundant representation can carry the entire normalized economic value.

Theorem (The value of one duplicate representation; Theorem 1)**.** *There exists a binary horizon-two decoder over three actions with redundancy one and a representation-blind game with payoffs in $[0, 1]$ such that the decoder has value one; every action-surjective canonical pruning has value zero.*

Proof. Let

$$00 \mapsto a, \qquad 01 \mapsto b, \qquad 10 \mapsto b, \qquad 11 \mapsto c.$$

The first-bit menus are $\{a, b\}$ and $\{b, c\}$, action b has multiplicity two, and the other actions have multiplicity one, so $R = 1$.

The buyer who minimizes the supplier's payoff first chooses a publicly observed state $s \in \{0, 1\}$. The supplier emits the first bit, the buyer observes it and chooses $y \in \{0, 1\}$, and the supplier completes. The supplier payoff is one exactly for

$$(s, y, a) = (0, 0, a), \quad (0, 1, b), \quad (1, 0, b), \quad (1, 1, c),$$

and is zero otherwise. In state zero, the supplier chooses prefix 0 and then selects a after $y = 0$ or b after $y = 1$. In state one, it chooses prefix 1 and selects b after $y = 0$ or c after $y = 1$. It guarantees one.

Every canonical action-surjective pruning retains 00 and 11 and exactly one representation of b. If it deletes 10, the buyer selects $s = 1$: after either retained first bit it can choose a y leaving no payoff-one completion. If it deletes 01, the symmetric argument with $s = 0$ applies. These are the only canonical action-surjective prunings, so all have value zero. □

Adaptive matching and interface design

Fix a commitment skeleton $\mathcal{T}$ with frontier L. For each action let $L_a = \{\ell \in L : a \in K_\ell\}$ and define the set of canonical selectors

$$\mathcal{F}(\mathcal{T}) = \{f : \mathcal{A} \to L : f(a) \in L_a\}.$$

Nature draws a finite market state θ, observed by the supplier before it selects a frontier leaf. A request is then drawn, with conditional probability $p_\theta(a)$ that action a is efficient. A match creates surplus $v_\theta > 0$ and a mismatch creates zero. Write $w_\theta = \Pr(\theta) v_\theta$ and $p_\theta(K) = \sum_{a \in K} p_\theta(a)$.

Proposition (Surplus from completing contracts after a request; Proposition 3)**.** *The welfare under multiplicity, the welfare-best action-surjective canonical pruning, and buffering are*

$$W_M(\mathcal{T}) = \sum_\theta w_\theta \max_{\ell \in L} p_\theta(K_\ell),$$

$$W_C(\mathcal{T}) = \max_{f \in \mathcal{F}(\mathcal{T})} \sum_\theta w_\theta \max_{\ell \in L} p_\theta\big(f^{-1}(\ell)\big),$$

$$W_B = \sum_\theta w_\theta \max_{a \in \mathcal{A}} p_\theta(a).$$

They satisfy $W_M(\mathcal{T}) \geq W_C(\mathcal{T}) \geq W_B$. *Moreover,* $|\mathcal{F}(\mathcal{T})| = \prod_a m_a(\mathcal{T})$.

Proof. After observing θ, the supplier selects a frontier leaf before the request arrives. Conditional on leaf ℓ, it can match precisely the requests in K_ℓ, giving expected state-θ surplus $v_\theta p_\theta(K_\ell)$. Optimizing leaf choice and averaging over states gives $W_M(\mathcal{T})$.

Use the minimal realization constructed in the proof of the exact resource theorem: it contains one codeword for every frontier incidence (ℓ, a) with $a \in K_\ell$. An action-surjective canonical pruning retains exactly one such word for each action. It is therefore equivalent to selecting $f(a) \in L_a$, and the retained menu at leaf ℓ is $f^{-1}(\ell)$. Conversely, every $f \in \mathcal{F}(\mathcal{T})$ defines such a pruning. Optimizing the supplier's leaf state by state and then the ex ante pruning gives $W_C(\mathcal{T})$. Because the choice of $f(a)$ has $|L_a| = m_a(\mathcal{T})$ possibilities independently for each action, $|\mathcal{F}(\mathcal{T})| = \prod_a m_a(\mathcal{T})$.

Under buffering, the supplier observes θ and then selects one complete action before the request, which gives W_B. For any selector f, $f^{-1}(\ell) \subseteq K_\ell$, so $W_M(\mathcal{T})$ weakly exceeds the welfare of that canonical pruning and hence $W_C(\mathcal{T})$. For every action a, the leaf $f(a)$ contains all request mass on a under the pruning. Thus, state by state,

$$\max_{\ell \in L} p_\theta(f^{-1}(\ell)) \geq \max_{a \in \mathcal{A}} p_\theta(a),$$

which proves $W_C(\mathcal{T}) \geq W_B$. □

Corollary (Surplus in the two-route example; example preceding Proposition 3)**.** *Use the decoder*

$$00 \mapsto a, \qquad 01 \mapsto b, \qquad 10 \mapsto b, \qquad 11 \mapsto c.$$

Nature draws $s \in \{0,1\}$ before the first bit, with $\Pr(s=1)=\pi$, and a request $y \in \{0,1\}$ with $\Pr(y=0 \mid s)=\Pr(y=1 \mid s)=1/2$ before the second. The requested contracts are a, b in state 0 and b, c in state 1. Thus $p_0(a)=p_0(b)=1/2$ and $p_1(b)=p_1(c)=1/2$, with all other conditional request probabilities zero. A match creates surplus $v_s > 0$, of which the supplier receives share $\rho_s \in (0,1)$. With $A_0=(1-\pi)v_0$ and $A_1=\pi v_1$,

$$W_M = A_0 + A_1, \qquad W_C = A_0 + A_1 - \tfrac{1}{2}\min_s A_s, \qquad W_B = \tfrac{1}{2}(A_0 + A_1).$$

The canonicalization loss is borne by the lower-A_s state and divided between supplier and buyer in shares ρ_s and $1-\rho_s$.

Let $B_s=(1-\rho_s)A_s$. If flexibility additionally shifts a transfer $\tau \geq 0$ from buyer to supplier, the buyer prefers it to its best canonical pruning if and only if $\tau \leq \frac{1}{2}\min_s B_s$. If b is the buyer's net engineering benefit of streaming relative to buffering, its best canonical streamed design weakly dominates buffering if and only if $b \geq -\frac{1}{2}\max_s B_s$.

Proof. The two frontier menus are $\{a,b\}$ and $\{b,c\}$. Hence multiplicity matches both requests in either state. A canonical selector must assign b to one leaf: assigning it left gives $A_0 + \frac{1}{2}A_1$, and assigning it right gives $\frac{1}{2}A_0 + A_1$. Buffering matches one of two requests in each state. This proves the welfare formulas. The welfare-best pruning preserves the state with larger A_s; the buyer-best pruning preserves the state with larger B_s. These choices can diverge. Applying the same losses to supplier and buyer shares and rearranging after adding τ or b gives the remaining claims. □

Marginal-cover insufficiency

Theorem (Intertemporal separation)**.** *Terminal action sets, action multiplicities, total redundancy, and the reduced cover $\mathcal{M}_t$ at every depth do not jointly determine strategic value. In particular, two length-three binary interfaces can agree on all these objects and nevertheless have values 1 and 0 in the same representation-blind interleaved zero-sum game.*

Proof. Let $\mathcal{A}=\{0,1,2,3\}$ and define two interfaces by

w	001	010	011	100	101	110	111
$D_C(w)$	0	1	2	0	1	1	3
$D_{C'}(w)$	0	1	3	0	1	1	2.

Both multiplicity vectors are $(2,3,1,1)$ and both redundancies are 3. Their reduced covers are identical at every depth:

$$\begin{aligned}
\mathcal{M}_0 &= \{\{0,1,2,3\}\},\\
\mathcal{M}_1 &= \{\{0,1,2\},\{0,1,3\}\},\\
\mathcal{M}_2 &= \{\{0\},\{0,1\},\{1,2\},\{1,3\}\},\\
\mathcal{M}_3 &= \{\{0\},\{1\},\{2\},\{3\}\}.
\end{aligned}$$

The ancestry coupling differs. Under C,

depth-one menu	depth-two children
$\{0,1,2\}$	$\{0\},\{1,2\}$
$\{0,1,3\}$	$\{0,1\},\{1,3\},$

Under C' the children are

$\{0,1,2\}$	$\{0,1\},\{1,2\}$
$\{0,1,3\}$	$\{0\},\{1,3\}.$

The agent emits a first bit; the opponent observes it and chooses $x \in \{0,1\}$. The agent emits a second bit; the opponent observes it and chooses $y \in \{0,1\}$. The agent emits the final bit and thereby chooses a decoded action. Its payoff is

	$(x,y)=00$	01	10	11
$a=0$	0	0	0	1
$a=1$	0	0	1	0
$a=2$	0	0	0	0
$a=3$	1	1	0	0.

The opponent minimizes. For a depth-two menu M, define

$$W_x(M) = \min_{y\in\{0,1\}} \max_{a\in M} u(a, x, y).$$

The only values needed for backward induction are

	$\{0\}$	$\{0,1\}$	$\{1,2\}$	$\{1,3\}$
W_0	0	0	0	1
W_1	0	1	0	0.

Under C, selecting the depth-one menu $\{0,1,3\}$ gives

$$\min_x \max\{W_x(\{0,1\}), W_x(\{1,3\})\} = 1,$$

so the agent guarantees 1. Under C', the two depth-one branches have respective values

$$\min_x \max\{W_x(\{0,1\}), W_x(\{1,2\})\} = 0,$$
$$\min_x \max\{W_x(\{0\}), W_x(\{1,3\})\} = 0.$$

Thus the opponent forces 0 after either first bit. Hence $V(C) = 1$ and $V(C') = 0$ despite equality of every claimed marginal statistic. □

The value separation above allows a literal first symbol to carry a different menu across interfaces. The next result rules out that simpler explanation and strengthens equality of reduced covers to equality of menu multisets.

Theorem (Ancestry can reverse the first choice; Theorem 2)**.** *For every horizon $H \geq 3$, there are two binary streamed interfaces and one representation-blind interleaved zero-sum game such that:*

(i) their terminal action sets, codeword counts, action multiplicities, and multisets of continuation menus at every depth are identical;
(ii) after each literal first symbol, the two interfaces expose the same continuation menu; and
(iii) each interface has a unique optimal first symbol, and those symbols are opposite.

Proof. At horizon three, use

w	000	001	010	011	100	101	110	111
$D_C(w)$	–	0	1	2	0	1	1	3
$D_{C^\dagger}(w)$	0	1	1	2	–	0	1	3.

Both interfaces have multiplicities $(2, 3, 1, 1)$. After literal root symbol 0, both menus equal $\{0, 1, 2\}$; after literal symbol 1, both equal $\{0, 1, 3\}$. Their depth-two menu multisets both equal

$$\big[\{0\}, \{0,1\}, \{1,2\}, \{1,3\}\big],$$

$C^\dagger$ exchanges the subtrees labeled $\{0\}$ and $\{0,1\}$ between the two root histories. All other depth-wise menu multisets coincide by direct enumeration.

Let the opponent move after depths one and two, and use the common payoff table

	$(x,y) = 00$	01	10	11
0	19	20	12	16
1	16	5	16	3
2	4	20	6	5
3	12	6	9	10.

Exact backward induction gives root-choice values

$$(G_C(0), G_C(1)) = (12, 16), \qquad (G_{C^\dagger}(0), G_{C^\dagger}(1)) = (16, 12).$$

The literal root menus and the unordered root-value pair are therefore matched, yet the unique optimal first symbol reverses.

For $H = 3 + k$, append every binary suffix $z \in \{0,1\}^k$ to each base codeword and replace base action a by the terminal action (a, z). Each (a, z) inherits the multiplicity of a. At depths through three, every base menu is replaced by the union of the corresponding action blocks. At a later depth, a prefix menu consists of the suffix extensions of one base action. Its multiplicity depends only on the matched base-action multiplicity. Thus menu multisets agree at every depth and literal root menus remain equal. Give (a, z) the displayed base payoff for the first two opponent replies, independently of z and later replies. Backward induction then preserves the two root-value vectors, proving the claim at every finite horizon. □

Call an interface summary *depth-anonymized* if it is invariant to re-parenting subtrees whenever terminal actions, codeword counts, action multiplicities, literal root menus, and the multiset of action-labeled menus at every depth are preserved.

Corollary (Menus at each depth do not determine the optimal choice; the corollary of Section 3)**.** *No depth-anonymized interface summary is strategically sufficient for every representation-blind interleaved game: equality of the summary can coexist with different root-action value vectors and opposite unique optimal first symbols. A separate common payoff table on the same interface pair also separates scalar game values.*

Proof. The preceding theorem's interfaces preserve every quantity in the definition. Its payoff table gives action-value vectors $(12, 16)$ and $(16, 12)$, proving opposite unique first symbols. Both optimized values are 16.

For scalar separation, use the binary payoff table from the intertemporal separation proof on the same literal-root-matched pair $C, C^\dagger$. With $W_x(M) = \min_y \max_{a \in M} u(a, x, y)$, the rows for menus $\{0\}, \{0,1\}, \{1,2\}, \{1,3\}$ are $(0,0,0,1)$ at $x = 0$ and $(0,1,0,0)$ at $x = 1$. In C, branch 0 has value $\min\{\max(0,0), \max(0,0)\} = 0$ and branch 1 has value $\min\{\max(0,1), \max(1,0)\} = 1$. In $C^\dagger$, branch 0 has value $\min\{\max(0,0), \max(1,0)\} = 0$, and branch 1 has value $\min\{\max(0,1), \max(0,0)\} = 0$. Hence their vectors are $(0,1)$ and $(0,0)$, and their scalar game values are 1 and 0. The two payoff witnesses establish the two insufficiency claims separately. □

Executable checks. The supplementary code contains four independent verifiers: one for the one-copy value of flexibility, one for the adaptive-matching characterization, one for the horizon-three separation, and one for the all-horizons ancestry family through horizon seven. They enumerate policies, prefix menus, or backward induction as appropriate and are covered by the standard-library test suite.

B Experimental condition levels and settings

Table 2: The four procurement conditions.

Condition	Model policy evaluated
Noncanonical, interleaved	Choose a route, observe a request, complete with the redundant codebook.
Canonical, interleaved	Same sequence with one representation per action.
Noncanonical, buffered	Complete the redundant message before the request.
Canonical, buffered	Complete the canonical message before the request.

Table 3: Mean payoffs in the four conditions over the same 20 procurement games. NI/CI use extra/one representation per action with a response between fields; NB/CB are the corresponding buffered conditions. Optimal values are computed separately in every game before averaging; each model uses the same answer keys. Noncompletion scores zero.

Model or exact policy	NI	CI	NB	CB
Exact optimal policy	5.45	2.80	2.40	2.40
Gemini 3.6 Flash	4.50	2.60	2.35	2.40
DeepSeek V4 Flash	3.75	2.35	2.40	2.40
DeepSeek V4 Flash (65K sensitivity)	4.75	2.80	2.40	2.40
DeepSeek V4 Pro	5.45	2.80	2.40	2.40
Gemini 3.1 Pro Preview	4.80	2.80	2.40	2.40

Table 4: Inference settings for the reported procurement runs. Each uses the same 20 games, prompts, structured choices, and analysis. Effort scales are provider-specific. The larger-budget Flash run is supplementary.

Endpoint	Status	Effort	Token maximum
Gemini 3.6 Flash	Primary	Low	8,192
DeepSeek V4 Flash	Primary	Low	8,192
DeepSeek V4 Flash	Budget sensitivity	Low	65,536
DeepSeek V4 Pro	Supplementary replication	Low	65,536
Gemini 3.1 Pro Preview	Supplementary replication	Low	8,192

Call accounting and intervals. Two buffered calls and, for each of the two interleaved codebooks, one route call followed by three request-specific completion calls give $20[2+2(1+3)] = 200$ planned logical API calls per model. A failed route call ends its condition without the three completions.

The 8,192-token Flash runs are the primary procurement analyses. After those results, the DeepSeek allowance sensitivity and both replications with larger models were specified separately and executed with unchanged instances, prompts, and analysis. Each study's addendum records its specification date and instance and prompt hashes. The four-call timestamped runtime replay followed the original implementation study with eight traces and holds each buyer response fixed.

Gemini Flash, the DeepSeek budget sensitivity, DeepSeek Pro, and Gemini Pro have 200 finalized logical calls each. Primary DeepSeek Flash has 185 because five failed route calls skip 15 continuations. Transport-only errors are retried independently of policy outcomes: counts are respectively 0, 1, 0, and 148 for those four complete runs, and 1 for primary DeepSeek Flash. The confirmatory procurement generator uses seed 20260819. Bootstrap means use 20,000 draws of 20 games with replacement, with seed 20260818 for the interaction and 20260819 for RFR. Saved JSON records include payoffs, model versions, responses, and transport errors. Transport errors are excluded from behavioral outcomes; analysis uses structured choices and leaves chain-of-thought content unexamined.

Additional ancestry outcomes. For both models, 37/40 interface pairs are successful: both interfaces yield the optimal first symbol within one display order. Each model selects the optimal symbol in 77/80 individual calls. A quartet contains both interfaces in both display orders, giving four calls

per game. DeepSeek's mean output length by message length ranges from 22.6k to 59.0k tokens; Gemini's ranges from 20.0k to 35.7k tokens. These are output-use summaries for the same runs.

Runtime payoffs and alternative suppliers. The implementation compares two payoff schedules, labeled `demand_risk` and `input_cost_risk`, under the four representation–timing conditions. The optimal first field is `minimum_units` in the first schedule and `unit_price_usd` in the second. All eight recorded supplier payoffs equal the values computed by exact backward induction for their respective games (Table 5).

Table 5: Supplier payoffs in the two runtime schedules (thousands of USD). Observed payoffs use the Gemini 3.1 Pro Preview supplier and DeepSeek V4 Pro buyer; exact values are computed by backward induction.

	`demand_risk`		`input_cost_risk`	
Condition	Observed	Exact	Observed	Exact
Noncanonical, interleaved	8	8	8	8
Canonical, interleaved	1	1	8	8
Noncanonical, buffered	1	1	1	1
Canonical, buffered	1	1	1	1

For `demand_risk`, the interaction is $(8{,}000 - 1{,}000) - (1{,}000 - 1{,}000) = 7{,}000$ dollars under the constructed payoff schedule. Earlier runs using DeepSeek V4 Flash and DeepSeek V4 Pro as supplier attained the exact game value in seven and six of the eight conditions, respectively. The successful eight-condition run uses Gemini as supplier and DeepSeek as buyer. The separate four-call replay fixes the buyer's response from saved prefixes to show how the completion changes.

Table 6: Runtime support for the four conditions. E requires a public prefix that must be kept or replaced at a cost. "Cfg." means the feature depends on the application's configuration. The final row includes a record of when each step occurs.

Pattern	E	R	A	M	Runtime behavior
Buffered tool call (final event only)	no	no	no	cfg.	Binding waits for validated completion.
OpenAI/Claude argument streaming [OpenAI, 2026, Anthropic, 2026]	cfg.	cfg.	cfg.	cfg.	The client determines whether a partial call is retained or discarded.
Conveyor partial execution [Xu et al., 2024]	yes	yes	no	cfg.	Execution overlaps decoding with a fixed continuation.
LangSmith streamed run with interrupt input [Runkle and Trivedy, 2026]	cfg.	yes	cfg.	cfg.	Input during a run preserves progress; the graph decides how to retain an unfinished call.
LangGraph JSON quote (ours)	yes	yes	yes	yes	The first member persists; the response selects the remaining member.

NeurIPS Paper Checklist

1. **Claims**
   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?
   Answer: [Yes]
   Justification: The abstract and Section 1 state the theory and empirical claims. Sections 5–7 report the experimental and implementation results, and Section 8 states the scope.
2. **Limitations**
   Question: Does the paper discuss the limitations of the work performed by the authors?
   Answer: [Yes]
   Justification: Sections 2–4 state the theoretical domain; Sections 5–7 delimit the synthetic and implementation evidence, sample uncertainty, provider-specific compute, and policy dependence. Section 8 consolidates the activation and adoption boundaries.
3. **Theory assumptions and proofs**
   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?
   Answer: [Yes]
   Justification: Sections 2–4 state the assumptions and give proofs or proof sketches for every numbered result; Appendix A gives the complete proofs, including the auxiliary lemma and corollary and the all-horizons ancestry construction.
4. **Experimental result reproducibility**
   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?
   Answer: [Yes]
   Justification: Sections 5–7 record generation, scoring, counterbalancing, inference settings, estimands, failure handling, clustering, fixed seeds, and the naturalistic runtime gate; the supplement contains code, raw records, frozen addenda, and exact commands.
5. **Open access to data and code**
   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?
   Answer: [Yes]
   Justification: The supplementary archive includes the generators, exact solver, analysis scripts, tests, frozen instances, raw response records, and reproduction instructions.
6. **Experimental setting/details**
   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?
   Answer: [Yes]
   Justification: Sections 5–7 and the addenda report suites, JSON schema, runtime, model settings, budgets, scoring, and failure rules. No model training is performed; calibration and confirmatory suites are disjoint.
7. **Experiment statistical significance**
   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?
   Answer: [Yes]
   Justification: Sections 5 and 6 report 20,000-draw fixed-seed cluster-bootstrap 95% intervals over the 20 independent procurement clusters and Wilson 95% intervals over the 20 independent matched ancestry clusters. Section 7 reports the complete predeclared trace set and exact finite-game inclusion gate for the naturalistic implementation.
8. **Experiments compute resources**
   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?
   Answer: [No]
   Justification: Model inference used hosted APIs whose worker hardware and memory are not disclosed. Sections 5–7 and the supplement report call counts, token budgets, observed token use, and latency; the exact solver and analysis require only standard CPU execution.

9. **Code of ethics**
   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?
   Answer: [Yes]
   Justification: The study uses synthetic games and automated API calls, collects no personal data, and involves no human participants.
10. **Broader impacts**
    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?
    Answer: [Yes]
    Justification: Sections 7 and 8 discuss the defensive value and latency costs of canonicalization and buffering, together with the risk that representation flexibility can preserve opaque revision opportunities.
11. **Safeguards**
    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?
    Answer: [N/A]
    Justification: The work releases no trained model, scraped dataset, personal data, or other high-risk asset; all benchmark instances are synthetic finite games.
12. **Licenses for existing assets**
    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?
    Answer: [N/A]
    Justification: No third-party dataset, model weights, or software package is redistributed. Hosted language models are experimental subjects, and their endpoint names and inference settings are disclosed in Sections 5–7 and Appendix B.
13. **New assets**
    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?
    Answer: [Yes]
    Justification: The new synthetic benchmark, verifier, and analysis artifacts are documented in Sections 5–7 and in the supplementary README, including schemas, generation commands, scoring, tests, and limitations.
14. **Crowdsourcing and research with human subjects**
    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?
    Answer: [N/A]
    Justification: The study involves neither crowdsourcing nor human subjects.
15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**
    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?
    Answer: [N/A]
    Justification: The study involves neither intervention with nor collection of data from human participants.
16. **Declaration of LLM usage**
    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.
    Answer: [Yes]
    Justification: Sections 5 and 6 describe the language models as experimental decision makers, their structured prompts and inference modes, and how their responses are scored.