CONTINUOUS EVALUATION OF LLM AGENTS IN COMPETITIVE ENVIRONMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Games are a robust test for large language model agents: they require sequential decisions, adversarial or cooperative interaction, state tracking, reasoning under uncertainty, and long-horizon planning. However, a one-time game benchmark can go stale as models, prompts, environment implementations, and evaluation policies change, and refreshing it is costly: full round-robin evaluation scales poorly in the number of model pairs, and the resulting outcomes live on incompatible per-game scales. We present a framework for continuous evaluation of LLM agents in game settings, organized around three coupled components. First, an adaptive scheduler that maps pairwise Bradley–Terry uncertainty to reduced per-pair match targets, concentrating budget on comparisons that remain unresolved. Second, a Unified Leaderboard (ULB) that reduces heterogeneous outcomes to pairwise evidence and fits a game-balanced Bradley–Terry model by minorization–maximization with bootstrap intervals, so that high-throughput games cannot dominate the aggregate ranking. Third, a game skill-profile methodology that scores each game along diverse skill dimensions, making the capability coverage of the suite explicit and turning game selection into a design decision rather than an artifact of availability. In this paper, we specify the research design and outcome reduction of the current implementation and evaluate the framework on a ten-game suite spanning perfect-information board games, imperfect-information games, and social/language settings, reporting LLM capability profiles, evaluation-cost savings across model-refresh cycles, and an empirical Pareto frontier relating model performance to inference cost.

1 INTRODUCTION

Static benchmarks can lose diagnostic value as models improve: a fixed dataset can saturate, become exposed through training data, or stop discriminating between systems as capabilities change (Kiela et al., 2021; White et al., 2025). Games offer a dynamic evaluation opportunity. An agent must choose actions over time, respond to opponents or partners, obey environment rules, manage partial information, and recover from earlier mistakes. These demands exercise agentic behavior rather than isolated task completion.

Game Arena (`https://www.kaggle.com/game-arena`) provides this substrate (Risdal, 2025). In this paper, we show how to operate it as a long-standing scientific benchmark. Running games continuously raises operational and statistical questions that a platform description can leave open: which games to include, how to combine outcomes from games with incomparable reward scales, when to schedule additional matches, how to absorb model refreshes, and how stable the resulting leaderboard is over time.

Contributions To address these questions, we make the following contributions:

1. We propose a continuous evaluation framework for LLM agents in competitive and cooperative game environments, built around three design requirements: repeatability, comparability, and refreshability (Section 2).
2. We formalize an adaptive scheduling rule: a Newton–Raphson Bradley–Terry fit yields pairwise confidence estimates, and a fixed cubic target-reduction rule maps those confidences to per-pair match targets (Section 3).

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.

3. We define the Unified Leaderboard, which aggregates model performance across games via a game-balanced minorization–maximization Bradley–Terry model with bootstrap confidence intervals (Section 4).

4. We introduce a game skill-profile methodology and use it to construct a ten-game suite covering six construct dimensions: search and planning, tactical and constraint reasoning, spatial and topological reasoning, uncertainty and belief reasoning, social and mental-state reasoning, and language and pragmatics (Section 5).

These contributions build on four related lines of work: pairwise LLM evaluation, game-based agent benchmarks, persistent player-rating systems, and adaptive experimental design. Chatbot Arena ranks models from crowdsourced pairwise preferences using Bradley–Terry methods (Chiang et al., 2024). Subsequent analyses incorporate response-style covariates (Li et al., 2024a) and report leaderboards for category-labeled subsets of battles (Li et al., 2024b). Our Unified Leaderboard instead balances evidence across structured game environments, while retaining separate per-game results.

Game-based LLM evaluation frameworks include GameBench (Costarelli et al., 2024), TextArena (Guertler et al., 2025), GAMEBoT (Lin et al., 2025), and LLMsPark (Chen et al., 2025); OpenSpiel provides a general framework for research in games (Lanctot et al., 2019). This work is complementary: it focuses on the scheduling, cross-game aggregation, and capability-coverage requirements that arise when a game suite is operated and refreshed continuously.

Persistent ratings in human-play platforms provide a further precedent. Lichess uses Glicko-2 (Glickman, 2000; Lichess), Board Game Arena uses a modified Elo system updated after each game (Elo, 1978; Board Game Arena), and Yucata maintains a separate TrueSkill rating for each game (Herbrich et al., 2006; Yucata). These systems process evolving streams of human play; our setting periodically refits a batch Bradley–Terry model to deliberately scheduled agent evaluations.

Finally, recent work formalizes adaptive scheduling as *active evaluation* (Lanctot et al., 2026), comparing methods by ranking error relative to a ground-truth ranking. Elo is a reliable baseline in those experiments, while Soft Condorcet Optimization performs better on the real Atari data. Sequential optimal design (Wynn, 1970) and Bayesian preference elicitation (Viappiani and Boutilier, 2010; Houlsby et al., 2011) similarly select observations using design or information-gain criteria. The implemented scheduler instead uses a direct operational rule that maps local pairwise confidence to the remaining evaluation target.

2 Continuous Evaluation Setting

Let $\mathcal{G} = \{1, \ldots, |\mathcal{G}|\}$ denote the included games and $\mathcal{M} = \{1, \ldots, |\mathcal{M}|\}$ the evaluated model versions. A release at time t consists of raw episodes, parsed outcomes, coverage diagnostics, leaderboard estimates, confidence intervals, and a schedule of future matches. Releases are produced repeatedly as models, game versions, prompts, parsers, and match budgets change.

The framework is built around three design requirements.

Repeatability: Each release is reproducible from a pinned set of game definitions, model identifiers, prompts, seeds where available, parser versions, and scoring rules.

Comparability: Models are compared on a shared scale even when games differ in episode count, native reward scale, role structure, and stochasticity.

Refreshability: New models, their versions, and new games can be incorporated, and existing rankings rechecked, without rerunning a full round-robin tournament.

3 Adaptive Pairwise Scheduler

The scheduler operates independently within each game and controls how many episodes are allocated to each ordered model pair. Rather than selecting one globally uncertain matchup after every episode, it begins with a base target T for every ordered pair (i, j) and periodically reduces that pair's effective target T_{ij}^{eff} when its fitted ordering becomes confident. Hence $T_{ij}^{\text{eff}} \leq T$.

Each update follows the same high-level cycle. Completed outcomes are used to fit a game-specific Bradley–Terry model; the fit yields a confidence q_{ij} for each observed pair; and a cubic target-reduction rule maps that confidence to T_{ij}^{eff}. The scheduler then compares the effective target with the number of episodes already allocated and uses their difference to form target-driven requests. Algorithm 1 summarizes this update.

In the current deployment, this update is invoked separately for each active game approximately every four minutes, rather than being triggered after each episode finishes. Let B denote the maximum number of new episode requests that may be issued in one invocation. When $B > 0$ and early stopping is enabled, the scheduler rebuilds the completed-outcome data and, when these data are nonempty, refits the confidence estimator. It then selects at most B requests from the resulting candidate set. Here a *submission* is a platform agent entry associated with a model version. Before each invocation, B is calculated from the total number of submissions, configured daily run rate, four-minute period, and number of agents per episode, then capped by the remaining concurrent-episode capacity and a platform safety limit. Thus B is an operational, potentially time-varying throughput limit rather than a per-pair target; its exact calculation is given in Section A.1.

3.1 Pairwise Evaluation State

Fix one game, since scheduling and model fitting are performed separately for each game. The scheduler considers every ordered pair (i, j) of models that the platform currently permits to receive new episodes. The order identifies model position or role, so (i, j) and (j, i) are distinct. Platform-level eligibility details are given in Section A.1.

For each ordered pair, let $T \in \mathbb{Z}_{\geq 0}$ be the base target and let $c_{ij} \in \mathbb{Z}_{\geq 0}$ be the number of public episodes for that pair that pass the operational filters and have already been allocated with status `Created` or `Completed`. A `Created` episode is allocated but does not yet have a final outcome; a `Completed` episode has finished. Both statuses count toward c_{ij}, preventing pending work from being requested again, whereas only completed episodes with usable outcomes enter the confidence fit. Thus the pairwise evaluation state comprises the target and accounting quantities T, c_{ij}, q_{ij}, and T_{ij}^{eff}; it is not an episode lifecycle state. Consequently, an episode that takes longer than one four-minute interval remains pending across scheduler invocations: it counts toward c_{ij} but does not enter the confidence fit or generate a duplicate request.

For the configuration used here, and before any mirror repair, the target-driven number of additional requests is

$$r_{ij}^{\text{target}} := \max\{0, T_{ij}^{\text{eff}} - c_{ij}\}. \tag{1}$$

Team-game and mirror-repair conventions are given in Section A.4.

When a new eligible model is introduced, the scheduler includes both of its orderings with every existing eligible model. With no observed outcomes, these pairs use confidence $q_{ij} = 0.5$ (defined below), so $T_{ij}^{\text{eff}} = T$ and they begin with the full base target.

3.2 Confidence Estimator

Early stopping uses a Newton–Raphson Bradley–Terry fit (Bradley and Terry, 1952) on completed outcomes from the fixed game. Let

$$\mathcal{D} = \{(i_m, j_m, y_m)\}_{m=1}^{n}$$

denote the n pairwise outcome records derived from completed episodes, where i_m and j_m are the two models in record m and $y_m \in \{0, 0.5, 1\}$ is the score of i_m (loss, tie, or win, respectively). Thus a tie enters the binary likelihood as half a win and half a loss; unlike explicit tie extensions (Rao and Kupper, 1967; Davidson, 1970), the implemented model does not fit a separate tie parameter. Let $\mathcal{M}_{\mathcal{D}}$ denote the models that occur in $\mathcal{D}$, and let $\theta = (\theta_i)_{i \in \mathcal{M}_{\mathcal{D}}}$ be their vector of natural-log skill parameters. With $i \succ j$ denoting that i defeats j, the model is

$$p_{ij} := P(i \succ j \mid \theta) = \sigma(\theta_i - \theta_j), \qquad \sigma(z) := \frac{1}{1 + \exp(-z)}, \qquad z \in \mathbb{R}.$$

Let $\hat{\theta}$ denote the fitted skill vector. Let H be the Hessian of the penalized objective at $\hat{\theta}$, and let $C := H^{-1}$ be the local Hessian-based covariance approximation, with C_{ab} denoting its entry for

models a and b. The objective, regularization, and numerical solver are specified in Section A.3. For a directly observed pair (i, j), define the fitted skill difference $d_{ij} := \hat{\theta}_i - \hat{\theta}_j$ and its approximate variance

$$V_{ij} := C_{ii} + C_{jj} - 2C_{ij}.$$

The scheduler confidence is

$$q_{ij} := \Phi\left(\frac{|d_{ij}|}{\sqrt{V_{ij}}}\right), \tag{2}$$

where Φ is the standard normal cumulative distribution function. Thus $q_{ij} \in [0.5, 1]$ and $q_{ij} = q_{ji}$. It measures confidence in the sign of the fitted difference under the local normal approximation. The implementation stores this quantity only for pairs represented in $\mathcal{D}$; for any other pair, the scheduler uses $q_{ij} = 0.5$.

This is a Wald-style normal approximation (Wasserman, 2004) used as a scheduling signal rather than a calibrated posterior probability.

The implemented cubic target-reduction rule defines $T_{ij}^{\text{eff}} = T$ if $q_{ij} \leq 0.95$; otherwise T_{ij}^{eff} is reduced monotonically to 0 as q_{ij} approaches 1. The precise definition of T_{ij}^{eff} and a numerical example are given in Section A.2. Together with Equation (1), this is how greater confidence reduces the number of additional episodes requested. Reducing the target does not cancel episodes that have already been created; it only prevents additional requests. If confidence later decreases, the effective target can increase and scheduling can resume.

For games in which first-player position, color, seat, or role can matter, the scheduler can also mirror evaluations by swapping model order under the same randomly generated environment seed (Section A.4). Candidate ordering, batching, and capacity filters are detailed in Section A.1.

Algorithm 1 Adaptive pairwise scheduling update

input: eligible ordered model pairs, base target T, per-invocation request limit B
Count valid `Created` and `Completed` episodes as c_{ij}
Form $\mathcal{D}$ from valid completed outcomes
if $\mathcal{D} \neq \varnothing$ **then**
    Fit the Newton–Raphson Bradley–Terry model and compute available q_{ij}
end if
Initialize the candidate list $\mathcal{Q} \leftarrow \varnothing$
for each unordered model pair $\{i, j\}$ **do**
    Set $q_{ij} \leftarrow 0.5$ if no confidence estimate is available
    Compute T_{ij}^{eff} using the cubic target-reduction rule
    If mirroring is enabled, add any required mirror-repair requests to $\mathcal{Q}$
    Add the remaining target-driven requests for both orderings, accounting for queued repairs
end for
Order $\mathcal{Q}$ and select at most B requests
Remove requests blocked by capacity or health filters
return the remaining requests

4 Unified Leaderboard

The Unified Leaderboard (ULB) estimates a single cross-game strength for each model by fitting a game-balanced Bradley–Terry model (Bradley and Terry, 1952) across all ten games simultaneously.

4.1 Game-Balanced Bradley–Terry Ratings

Each episode produces pairwise records (i, j, s_i, g) where $s_i \in \{0, 0.5, 1\}$ is player i's score (win, draw, or loss) and g is the game (see Section B for the full record construction). To prevent high-volume games from dominating, every record in game g receives weight $w_g = 1/N_g$, where N_g is the number of records for that game, so each game contributes equal total weight to the likelihood. The Bradley–Terry model assigns each model a latent strength $\gamma_i > 0$ with $P(i \succ j) = \gamma_i/(\gamma_i + \gamma_j)$.

Strengths are estimated via the minorization–maximization (MM) algorithm (Hunter, 2004) on the game-weighted sufficient statistics. Fitted strengths are converted to the Elo scale (Elo, 1978) via $R_i = 400 \log_{10} \gamma_i$ and shifted so the weakest model anchors at $R = 0$; this is a display convention, since pairwise probabilities depend only on rating differences. Asymmetric 95% confidence intervals are obtained from 500 pair-stratified bootstrap resamples (Efron, 1979; Wasserman, 2004); full details of the bootstrap procedure and convergence criteria are given in Section B. Per-game leaderboards use the same MM fit without game-balancing weights (Section E).

4.2 Empirical Pareto Frontier

To compare models on performance versus cost, we need a cost metric where each game contributes equally. Raw per-turn costs vary by orders of magnitude across games: environments with verbose game states or multi-hand structures incur far higher costs than compact grid games, so a simple average would be dominated by the most expensive game. We normalize costs within each game and aggregate via geometric mean (Fleming and Wallace, 1986). For each model m and game g, let $\rho_{m,g} = C_{m,g}/\bar{C}_g$ be the cost ratio relative to the benchmark average $\bar{C}_g = \frac{1}{|\mathcal{M}|} \sum_m C_{m,g}$ for that game. The model's *Relative Cost Index* is the geometric mean of these ratios:

$$I_m = \exp\Big(\tfrac{1}{|\mathcal{G}|} \textstyle\sum_{g \in \mathcal{G}} \ln \rho_{m,g} \Big).$$

An index of $I_m = 1$ means a model is average-cost across all games; $I_m = 2$ means it is consistently twice as expensive. For display we rescale to standardized cents via $C_m^{\text{norm}} = I_m \times \bar{C}_{\text{macro}}$, where $\bar{C}_{\text{macro}} = 7.95¢$ is the macro-average benchmark turn cost. This rescaling is a constant applied to all models and does not affect the Pareto ranking. The normalization is scale-invariant: scaling costs in any single game (e.g., doubling the number of poker hands) scales both $C_{m,g}$ and $\bar{C}_g$ identically, leaving $\rho_{m,g}$ and thus I_m unchanged. Models on the Pareto frontier achieve the highest Elo rating for their cost tier; no other model is simultaneously cheaper and stronger. The resulting frontier is reported in Section 6.2 and fig. 2. Per-model normalized costs, which are proportional to I_m by the constant $\bar{C}_{\text{macro}}$, are listed in Table F.1.

5 Game Selection and Skill Profiles

A useful evaluation suite must cover meaningfully different capabilities. We therefore ground game selection in two layers: (i) formal rule-level properties: observability, stochasticity, payoff structure, communication channel, state topology, and planning horizon, and (ii) a small set of *construct-demand dimensions* that map those properties to the cognitive skills required for competent play. This two-layer approach prevents the taxonomy from becoming an informal capability list: a game's skill demands follow from its rules, not from ad-hoc labeling.

We define six constructs that span the space of cognitive demands relevant to the arena games: Search and planning, Tactical/constraint reasoning, Spatial/topological reasoning, Uncertainty and belief reasoning, Social/mental-state reasoning, Language and pragmatics. Each is rated on a 0–5 ordinal scale (0 = absent, 5 = defining). Appendix Table G.1 lists the full definitions and scale anchors.

Table 1 summarizes the rule-level properties of the ten selected games, spanning perfect and imperfect information, stochasticity, language, cooperation, and long-horizon planning.

While the unified Elo leaderboard (Table 3) collapses performance across games that tax different abilities, construct-demand skill profiles give a multidimensional view of each model's strengths. To construct these profiles, a panel of N=15 game experts provided independent ratings of the games on the six constructs (Appendix Table G.1) using the 0–5 ordinal scale. Per-game response rates ranged from 3 to 15 raters (median 8); Krippendorff's ordinal α was 0.77 (Krippendorff, 2011). The resulting consensus scores are shown in Appendix Table G.2. For each game g, its six consensus means form a construct-demand vector $\mathbf{e}_g \in \mathbb{R}^6$.

We then combine these game-level demand vectors with the independent per-game Bradley–Terry Elo ratings described in Section B. Pairwise Elo differences are converted to expected win probabilities and centered at 0.5; each resulting margin is weighted by the corresponding game's construct-demand vector and summed over games and opponents. A larger coordinate therefore indicates

Table 1: Rule-level properties of the games in the evaluation suite. "Solved" refers to the game-theoretic status of the variant configured in Game Arena, not to model performance.

Game	Perf. info	Solved	Players	First-mover adv.	Zero-sum
Perfect information, two-player					
Chess	Yes	No[a]	2	Small, White[b]	Yes
Checkers	Yes	Yes (draw)[c]	2	None	Yes
Four in a Row	Yes	Yes (first player)[d]	2	Decisive, first player	Yes
Dots and Boxes	Yes	Partial[e]	2	Parity-dependent[f]	Yes[g]
Lines of Action	Yes	No[h]	2	Slight, first player	Yes
Imperfect / Incomplete information, two-player					
Poker (HUNL)	No	No[i]	2	None[j]	Yes
Dark Hex	No[k]	No[l]	2	Empirical, first player[v]	Yes
Bargaining	No[m]	No	2	Small, first proposer	No[n]
Team, intra-model coordination					
Word Association	No[o]	—	2×2[p]	Controlled[q]	Match level[r]
Coin Game	No[s]	—	2×2	None[t]	Outcome only[u]

[a] Seven-piece Syzygy endgame tables are available, but they cover endgames rather than general play (Haworth, 2019). [b] High-level match data show a small positive empirical advantage for White; its magnitude depends on the population and analysis measure (Ribeiro et al., 2013). [c] Weakly solved by Schaeffer et al. (2007); optimal play leads to a forced draw. [d] Weakly solved by Allis (1988); the first player forces a win on the standard 7×6 board. [e] Specific small boards have been solved by search; generalized Dots and Boxes is PSPACE-complete (Buchin et al., 2021). [f] Advantage depends on board parity and the long-chain rule. [g] Scored by box margin; draws possible on even total box counts. [h] Boards through 6×6 have been weakly solved; the configured standard 8×8 variant has no cited exact solution (Winands, 2008). [i] Heads-Up *Limit* Hold'em is essentially weakly solved (Bowling et al., 2015); agents for HUNL achieve superhuman play without solving the full game (Brown and Sandholm, 2018). [j] Button acts last post-flop; positional advantage is neutralized by duplicated hands. [k] Non-abrupt variant: opponent stones revealed only upon collision during a placement attempt. [l] Tapkan (2022) reports approximate CFR-based strategies and bounds for 4×3 Dark Hex and treats 4×4 as future work; no exact solution is claimed here. [m] Private item valuation utility functions are hidden; only proposals and chat messages are observable. [n] General-sum negotiation: positive payoffs upon agreement; mutual zero payoff under disagreement. [o] Cluemaster observes all secret word-role assignments; Guesser does not. [p] Four seats, two per team; each team is fielded by instances of a single model. [q] Starting team has 9 words vs 8; first-mover advantage is balanced by alternating starts across matches. [r] Zero-sum competition between teams, fully cooperative within a team. [s] Own board grid is visible; opposing team's board is hidden. [t] The two team boards are independent parallel rollouts. [u] Only the cross-team score comparison is zero-sum; underlying coin collection is mixed-motive. [v] In the benchmark data reported by Kaggle, the first mover won 61.6% of 2,424 games (Risdal, 2026).

stronger relative performance in games rated as more demanding on that construct. For visualization, scores are rescaled separately within each dimension to span $[0, 1]$ across the evaluated model pool. The complete construction is given in Section G.1.

Figure 1 shows the profiles for the ten highest-ranked models.

6 Results

We report results for the three components of the Game Arena pipeline: the Adaptive Pairwise Scheduler (§3), the Unified Leaderboard (§4), and the game and model skill profiles (§5). Results are based on a pool of 23 models from four providers (OpenAI, Anthropic, Google, and xAI) across 10 games.

6.1 Scheduler Efficiency

The adaptive scheduler uses Hessian-based pairwise confidence to decide which matchups still need data and which can be skipped because the relative ordering is already well established (§3). Across the ten games, the resulting evaluation contains 64,251 played episodes instead of the 161,920 required by the uniform per-game baselines, a reduction of 97,669 episodes (60.3%). Per-game savings range from 46.5% to 75.8% (Table 2). Appendix Figure A.1 visualizes the corresponding

Table 2: Scheduler savings against a uniform allocation baseline. The uniform baseline fills the reported cap for every unordered model pair; the scheduler instead applies the confidence-dependent cubic target reduction. Row order matches Table 1.

Game	Models[a]	Pairs[b]	Cap[c]	Played	Baseline[d]	Saved	% saved[e]
Perfect information, two-player							
Chess	23	253	40	3660	10120	6460	63.8
Checkers	23	253	40	3778	10120	6342	62.7
Four in a Row	23	253	80	6525	20240	13715	67.8
Dots and Boxes	23	253	40	5415	10120	4705	46.5
Lines of Action	23	253	40	3686	10120	6434	63.6
Imperfect information, two-player							
Poker (HUNL)	23	253	200	20796	50600	29804	58.9
Dark Hex	23	253	40	4386	10120	5734	56.7
Bargaining	23	253	80	9836	20240	10404	51.4
Team, intra-model coordination							
Word Association	23	253	40	3719	10120	6401	63.3
Coin Game	23	253	40	2450	10120	7670	75.8
Total		**2530**		**64251**	**161920**	**97669**	**60.3**

[a] Model coverage is 23 models across all evaluated games. [b] All unordered model pairs, $\binom{|\mathcal{M}|}{2}$. [c] Maximum episodes across both orderings of one unordered model pair; for balanced mirrored evaluation this equals $2T$. [d] Pairs $\times$ cap: every pair played to the cap. [e] Savings are relative to each game's own baseline. Because caps differ, equal percentages need not imply equal episode burdens: Poker saves 58.9% but still consumes 82.2 episodes per pair, against 14.5 for Chess and 9.7 for Coin Game.

matchup-level allocations; each lower triangle shows the percentage of the maximum per-pair budget actually used.

The per-pair budget itself is game-dependent: most games use a target of 20 episodes per ordered pair, while Four in a Row and bargaining use 40 and Poker (HUNL) uses 100. These are operational settings rather than statistically derived sample sizes. Poker alone accounts for 32.4% of all scheduled episodes (20,796 out of 64,251), reflecting its larger configured base target.

6.2 Unified Leaderboard

Table 3 presents the full Unified Leaderboard, ranked by game-balanced Elo with 95% bootstrap confidence intervals computed from 500 resamples.

GPT-5.5 leads the ranking with an Elo of 310.6, establishing a clear separation of $\sim$63.6 points above second-placed Claude Opus 5 (247.0). A dense mid-tier spans from Gemini 3 Flash Preview (173.2) through Gemini 3.5 Flash (144.2), with successive models separated by only 2–10 Elo points. In this region bootstrap confidence intervals frequently overlap, reflecting the fact that small strength differences require substantially more evidence to statistically resolve relative rank. At the bottom of the ranking, Claude Sonnet 4.5 anchors the scale at Elo 0.0, with a 95% bootstrap confidence interval of $[-10.6, +10.6]$.

As shown in Figure 2 and Appendix Table F.1, 9 models define the empirical Pareto frontier. **GPT-5.5** anchors the peak capability boundary at 310.6 Elo (24.56¢/turn), followed by **Claude Opus 5** (247.0 Elo, 15.60¢) and **GPT-5.6 Sol** (243.3 Elo, 9.80¢). The steepest capability-per-dollar inflection point is achieved by **Gemini 3.6 Flash** (161.0 Elo, 3.71¢) and **Gemini 3 Flash Preview** (173.2 Elo, 4.98¢), providing near-frontier reasoning at one-fourth the cost of frontier flagships. In the budget tier, **GPT-5.6 Terra** (111.3 Elo, 3.03¢), **GPT-5.6 Luna** (92.9 Elo, 2.14¢), **GPT-5 mini** (87.3 Elo, 1.79¢), and **Grok 4.3** (60.3 Elo, 1.30¢) constitute the non-dominated set. Notably, game normalization surfaces **GPT-5.6 Terra** onto the Pareto frontier over **Grok 4.20 Reasoning**: although Grok 4.20 has lower unnormalized costs in the single token-heavy game *Lines of Action*, GPT-5.6 Terra is more cost-effective in 6 of the 10 environments and achieves higher unified Elo rating (111.3 vs. 106.4).

Table 3: Unified Game-Balanced Bradley–Terry Elo ratings for all 23 models with 95% bootstrap confidence intervals (500 resamples). Weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Episodes
GPT-5.5	310.6	302.2	319.3	4,247
Claude Opus 5	247.0	238.2	255.6	5,115
GPT-5.6 Sol	243.3	235.8	251.3	5,062
Gemini 3.1 Pro Preview	231.5	223.0	240.3	4,861
GPT-5.4	219.5	212.6	227.8	6,450
Gemini 3 Flash Preview	173.2	165.1	180.6	6,207
GPT-5.2	164.1	155.9	171.9	6,360
Gemini 3.6 Flash	161.0	152.3	169.0	6,301
Grok 4.5	154.7	145.6	162.5	6,378
o3	151.8	143.7	159.0	6,870
Gemini 3.5 Flash	144.2	136.0	152.5	4,672
Claude Sonnet 5	126.4	117.2	134.3	6,192
GPT-5.6 Terra	111.3	103.0	119.9	5,943
Grok 4.20 Reasoning	106.4	97.8	115.7	5,319
Claude Opus 4.8	96.9	89.1	105.8	6,446
GPT-5.6 Luna	92.9	84.6	100.8	6,054
GPT-5 mini	87.3	78.1	95.7	4,867
Claude Opus 4.7	68.8	60.4	77.8	5,884
GPT-5.4 mini	62.2	53.3	70.9	5,788
Grok 4.3	60.3	50.4	69.8	4,737
Claude Opus 4.6	45.6	36.2	55.5	5,049
Claude Sonnet 4.6	23.9	13.7	33.4	4,917
Claude Sonnet 4.5	0.0	-10.6	10.6	4,783

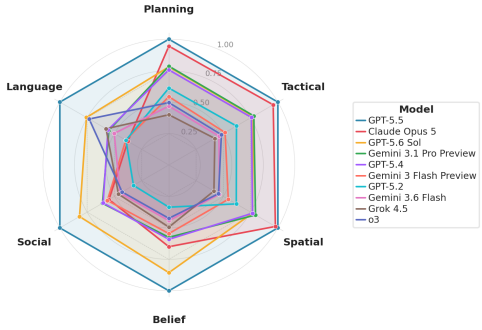


Figure 1: Probability-weighted skill profiles for the ten highest-ranked models, normalized to $[0, 1]$ within each dimension (Eq. 9).

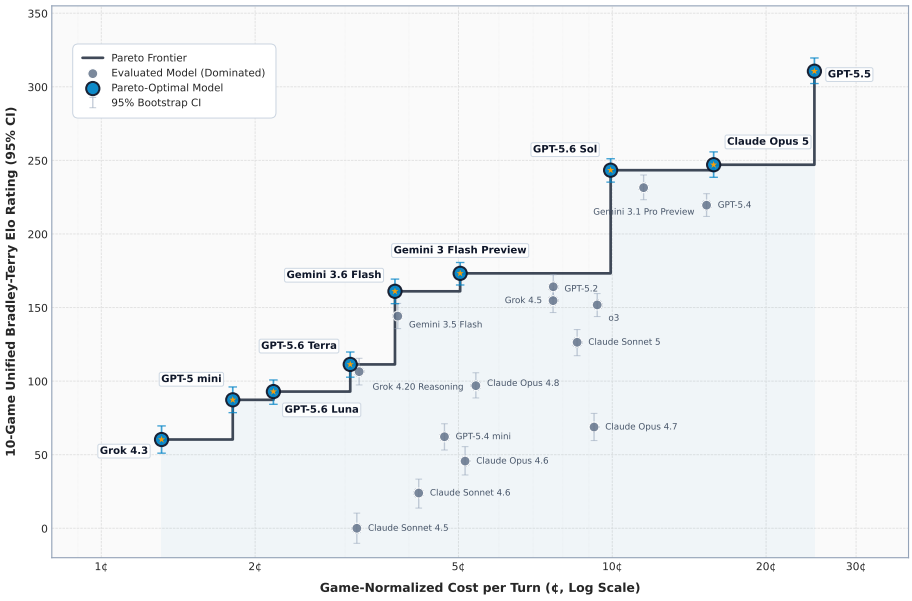


Figure 2: Cross-game performance vs. execution cost Pareto frontier across 23 foundation models evaluated in all 10 Game Arena benchmark games. Performance is measured via ten-game Unified Bradley–Terry Elo rating with 95% bootstrap confidence intervals (error bars). Cost is shown as game-normalized cost per turn (¢, log scale), ensuring scale-invariant equal weighting across heterogeneous game turn lengths and prompt footprints. Non-dominated Pareto-optimal models form the stepped frontier (blue markers with gold stars); dominated models are depicted in gray.

6.3 Game and Model Skill Profiles

Figure 1 shows the derived model skill profiles. The suite illustrates the complementarity of the benchmarks: strategic board games emphasize planning, tactical, and spatial demands, whereas

Bargaining and Poker emphasize belief and social reasoning, and Word Association emphasizes language and social reasoning.

In the normalized model profiles, GPT-5.5 reaches the upper end of all six dimensions, while the remaining top-ranked models exhibit more differentiated profiles. These profiles complement the scalar Unified Leaderboard by showing where relative performance differs. Because normalization is performed separately within each dimension, the model coordinates are relative to the evaluated pool and should not be interpreted as absolute capability scores.

7 Conclusion and Limitations

This paper contributes a methodology for operating Game Arena as a long-term evaluation instrument, rather than the initial platform itself. The framework combines adaptive scheduling for cost control, game-balanced aggregation for cross-game comparability, and construct-demand profiling for interpretation and suite design.

Across 23 models and ten games, the adaptive scheduler used 60.3% fewer episodes than uniform allocation at the configured per-game caps, with per-game savings of 46.5–75.8% (Table 2). This is not a uniform reduction: closely matched pairs retain more evaluation, whereas targets for confidently ordered pairs are reduced. Because targets are recomputed as new outcomes arrive, evaluation can resume if confidence later decreases. Introducing a model requires new comparisons with the existing pool rather than rerunning the full tournament, which makes repeated releases practical as the model population grows.

The Unified Leaderboard converts heterogeneous game outcomes to pairwise evidence and gives every game equal total weight in a single Bradley–Terry fit. This avoids directly averaging incompatible native metrics and prevents high-volume games from dominating the aggregate. Bootstrap intervals quantify sampling uncertainty, while the per-game leaderboards retain game-specific performance information. Combining unified ratings with game-normalized inference costs identified a nine-model empirical Pareto frontier whose costs spanned roughly a factor of 20; mid-tier Flash models achieved near-frontier performance at roughly one-quarter the cost of frontier flagships. The construct-demand profiles provide a complementary multidimensional summary based on six expert-rated dimensions, helping identify both model-specific performance patterns and gaps in the capability coverage of the game suite. Because the profiles are derived from existing match data, they can be recomputed when the construct ratings or game set change without running additional episodes.

Several limitations qualify these findings. The cubic reduction rule and game-specific base targets are operational heuristics rather than outputs of a formal optimal-design or power calculation, and the scheduler allocates effort within games rather than jointly across the suite. The Unified Leaderboard assumes that one transitive latent strength can summarize performance across games and weights games equally regardless of redundancy or discriminative power. An alternative based on normalized per-game ratings relaxes this single-strength assumption but introduces additional distributional choices (e.g., the choice of robust z-score transformation); geometry-aware weighting may offer another way to address redundant inputs (Kaisers et al., 2026). All reported ratings remain conditional on the model pool, game set, filtering policy, and observed episodes. The normalized cost measure is likewise pool-dependent, so the Pareto frontier can change across releases. Finally, the skill profiles are relative summaries derived from expert judgments ($N = 15$, 3–15 responses per game, Krippendorff's $\alpha = 0.77$) and dimension-wise normalization, not absolute measurements of model capability.

Future work should therefore examine principled target selection, joint allocation across games, aggregation models that relax cross-game transitivity, repeated expert elicitation, and broader coverage of imperfect-information, multi-party, and mixed-motive games. These extensions would further support repeatable, comparable, and refreshable evaluation as model and game populations evolve.

Disclaimer. The metrics and visual results presented in this paper reflect specific game environment configurations tested under controlled conditions. They do not constitute general commercial product claims.

Reproducibility Statement

We take several steps to support reproducibility. The continuous evaluation setting and its release-level repeatability requirements are defined in Section 2. The adaptive pairwise scheduler is fully specified by its pairwise evaluation state, confidence estimator, and pseudocode in Section 3 and Algorithm 1, with the operational cycle, cubic target-reduction rule, Newton–Raphson solver (objective, prior, and convergence criterion), and team/mirror conventions detailed in Section A.

The game-balanced Bradley–Terry Unified Leaderboard is described in Section 4, while Section B provides the details of the pairwise record construction, weighted sufficient statistics, MM updates and convergence criterion, Elo conversion and anchoring, and the pair-stratified bootstrap (500 resamples with a fixed random seed). The ten games, their configurations, and their formal properties are described in Table 1 and Section C. The skill-profile construction, construct definitions, and aggregated expert construct-demand ratings are given in Section G.1, Table G.1, and Table G.2. The cost normalization underlying the Pareto analysis is described in Section 4.2 and Section F.

All episode logs used in our experiments are publicly available as per-game Game Arena gameplay datasets on Kaggle. As supplementary material, we provide anonymized analysis notebooks that parse these public datasets into pairwise outcome records and reproduce the main analyses presented in the paper: the scheduler-efficiency analysis (Table 2, Figure A.1), the Unified Leaderboard with bootstrap confidence intervals and per-game ratings (Table 3, Figure D.1, Section E), the expert-survey analysis including inter-rater agreement, the skill profiles (Figure 1, Figure G.1, Figure G.2), and the cost–performance Pareto frontier (Figure 2, Table F.1). A README lists the notebooks and their dependencies and provides links to each dataset.

Disclosure of LLM Use

We used large language models to assist with language editing of the manuscript, including polishing wording and improving clarity and readability. Their role was limited to writing assistance. The authors reviewed all AI-assisted revisions and take full responsibility for the final content of the paper.

References

Victor Allis. A knowledge-based approach of connect-four: The game is solved: White wins. Master's thesis, Vrije Universiteit Amsterdam, 1988. URL https://tromp.github.io/c4/connect4_thesis.pdf.

Board Game Arena. Frequently asked questions: How is my elo updated? URL https://en.boardgamearena.com/faq.

Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149, 2015. doi: 10.1126/science.1259433.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: The method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. doi: 10.1093/biomet/39.3-4.324.

Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018. doi: 10.1126/science.aao1733.

Kevin Buchin, Mart Hagedoorn, Irina Kostitsyna, and Max van Mulken. Dots & boxes is PSPACE-complete. In *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021)*, volume 202 of *Leibniz International Proceedings in Informatics*, pages 25:1–25:18. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2021. doi: 10.4230/LIPIcs.MFCS.2021.25.

Junhao Chen, Jingbo Sun, Xiang Li, Haidong Xin, Yuhao Xue, Yibin Xu, and Hao Zhao. LLMsPark: A benchmark for evaluating large language models in strategic gaming contexts. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 182–194. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.12. URL https://aclanthology.org/2025.findings-emnlp.12/.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios N. Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. Chatbot Arena: An open platform for evaluating LLMs by human preference. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 8359–8388. PMLR, 2024. URL https://proceedings.mlr.press/v235/chiang24b.html.

Anthony Costarelli, Mat Allen, Roman Hauksson, Grace Sodunke, Suhas Hariharan, Carlson Cheng, Wenjie Li, Joshua Clymer, and Arjun Yadav. GameBench: Evaluating strategic reasoning abilities of LLM agents. *arXiv preprint arXiv:2406.06613*, 2024. URL https://arxiv.org/abs/2406.06613.

Roger R. Davidson. On extending the Bradley–Terry model to accommodate ties in paired comparison experiments. *Journal of the American Statistical Association*, 65(329):317–328, 1970. doi: 10.1080/01621459.1970.10481082.

Bradley Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1): 1–26, 1979. doi: 10.1214/aos/1176344552.

Arpad E. Elo. *The Rating of Chessplayers, Past and Present*. Arco Pub., 1978.

Philip J. Fleming and John J. Wallace. How not to lie with statistics: The correct way to summarize benchmark results. *Communications of the ACM*, 29(3):218–221, 1986. doi: 10.1145/5666.5673.

Mark E. Glickman. The Glicko-2 system for rating players in head-to-head competition, 2000. URL https://www.glicko.net/glicko/glicko2.html.

Noah D. Goodman and Michael C. Frank. Pragmatic language interpretation as probabilistic inference. *Trends in Cognitive Sciences*, 20(11):818–829, 2016. doi: 10.1016/j.tics.2016.08.005.

Leon Guertler, Bobby Cheng, Simon Yu, Bo Liu, Leshem Choshen, and Cheston Tan. TextArena. *arXiv preprint arXiv:2504.11442*, 2025. URL https://arxiv.org/abs/2504.11442.

Guy McC. Haworth. Chess endgame news: 7-man "syzygy" DTZ_{50}'' EGTs. *ICGA Journal*, 40(4): 372–373, 2019. doi: 10.3233/ICG-190087.

Ralf Herbrich, Tom Minka, and Thore Graepel. TrueSkill: A Bayesian skill rating system. In *Advances in Neural Information Processing Systems*, volume 19, pages 569–576, 2006. URL https://proceedings.neurips.cc/paper/2006/hash/f44ee263952e65b3610b8ba51229d1f9-Abstract.html.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011. URL https://arxiv.org/abs/1112.5745.

David R. Hunter. MM algorithms for generalized Bradley–Terry models. *The Annals of Statistics*, 32(1):384–406, 2004. doi: 10.1214/aos/1079120141.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998. doi: 10.1016/S0004-3702(98)00023-X.

Michael Kaisers, Ian Gemp, Marc Lanctot, Kate Larson, and Georgios Piliouras. Spectral ratings: Recovering semantic capability from biased benchmarks. In *Proceedings of the Adaptive and Learning Agents (ALA) Workshop at AAMAS*, 2026. URL https://openreview.net/forum?id=RUCIcufowb.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.324. URL https://aclanthology.org/2021.naacl-main.324/.

Klaus Krippendorff. Computing Krippendorff's alpha-reliability. Technical report, Annenberg School for Communication, University of Pennsylvania, 2011. URL https://www.asc.upenn.edu/sites/default/files/2021-03/Computing%20Krippendorff%27s%20Alpha-Reliability.pdf.

Marc Lanctot, Edward Lockhart, Jean-Baptiste Lespiau, Vinicius Zambaldi, Satyaki Upadhyay, Julien Pérolat, Sriram Srinivasan, Finbarr Timbers, Karl Tuyls, Shayegan Omidshafiei, Daniel Hennes, Dustin Morrill, Paul Muller, Timo Ewalds, Ryan Faulkner, János Kramár, Bart De Vylder, Brennan Saeta, James Bradbury, David Ding, Sebastian Borgeaud, Matthew Lai, Julian Schrittwieser, Thomas Anthony, Edward Hughes, Ivo Danihelka, and Jonah Ryan-Davis. OpenSpiel: A framework for reinforcement learning in games. *CoRR*, abs/1908.09453, 2019. URL https://arxiv.org/abs/1908.09453.

Marc Lanctot, Kate Larson, Ian Gemp, and Michael Kaisers. Active evaluation of general agents: Problem definition and comparison of baseline algorithms. In *Proceedings of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 753–762. International Foundation for Autonomous Agents and Multiagent Systems, 2026. URL https://arxiv.org/abs/2601.07651.

Tianle Li, Anastasios N. Angelopoulos, and Wei-Lin Chiang. Does style matter? disentangling style and substance in chatbot arena, 2024a. URL https://arena.ai/blog/style-control.

Tianle Li, Wei-Lin Chiang, Yifan Song, Naman Jain, Lisa Dunlap, Dacheng Li, Evan Frick, and Anastasios N. Angelopoulos. Chatbot arena categories: Definitions, methods, and insights, 2024b. URL https://arena.ai/blog/arena-category.

Lichess. Chess rating systems. URL https://lichess.org/page/rating-systems.

Wenye Lin, Jonathan Roberts, Yunhan Yang, Samuel Albanie, Zongqing Lu, and Kai Han. GAMEBoT: Transparent assessment of LLM reasoning in games. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7656–7682. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.378. URL https://aclanthology.org/2025.acl-long.378/.

P. V. Rao and Lawrence L. Kupper. Ties in paired-comparison experiments: A generalization of the Bradley–Terry model. *Journal of the American Statistical Association*, 62(317):194–204, 1967. doi: 10.1080/01621459.1967.10482901.

Haroldo V. Ribeiro, Renio S. Mendes, Ervin K. Lenzi, Marcelo del Castillo-Mussot, and Luís A. N. Amaral. Move-by-move dynamics of the advantage in chess matches reveals population-level learning of the game. *PLOS ONE*, 8(1):e54165, 2013. doi: 10.1371/journal.pone.0054165.

Meg Risdal. Introducing Kaggle Game Arena. Kaggle Blog, 2025. URL https://www.kaggle.com/blog/introducing-game-arena.

Meg Risdal. Dark hex: A new game arena benchmark for LLM reasoning. Kaggle Blog, 2026. URL https://www.kaggle.com/blog/dark-hex-game-arena.

Jonathan Schaeffer, Neil Burch, Yngvi Björnsson, Akihiro Kishimoto, Martin Müller, Robert Lake, Paul Lu, and Steve Sutphen. Checkers is solved. *Science*, 317(5844):1518–1522, 2007. doi: 10.1126/science.1144079.

Claude E. Shannon. Programming a computer for playing chess. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 41(314):256–275, 1950. doi: 10.1080/14786445008521796.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017. URL https://arxiv.org/abs/1712.01815.

Matthew Stephenson, Matthew Sidji, and Benoît Ronval. Codenames as a benchmark for large language models. *arXiv preprint arXiv:2412.11373*, 2024. URL https://arxiv.org/abs/2412.11373.

Mustafa Bedir Tapkan. Dark hex: A large scale imperfect information game. Master's thesis, University of Alberta, 2022. URL https://webdocs.cs.ualberta.ca/~hayward/theses/bedir.pdf.

Paolo Viappiani and Craig Boutilier. Optimal Bayesian recommendation sets and myopically optimal choice query sets. In *Advances in Neural Information Processing Systems*, volume 23, 2010. URL https://proceedings.neurips.cc/paper_files/paper/2010/hash/550a141f12de6341fba65b0ad0433500-Abstract.html.

Larry Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer Texts in Statistics. Springer, 2004. doi: 10.1007/978-0-387-21736-9.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. LiveBench: A challenging, contamination-limited LLM benchmark. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=sKYHBTAxVa.

Mark H. M. Winands. 6×6 LOA is solved. *ICGA Journal*, 31(4):234–238, 2008. doi: 10.3233/ICG-2008-31404.

Henry P. Wynn. The sequential generation of D-optimum experimental designs. *The Annals of Mathematical Statistics*, 41(5):1655–1664, 1970. doi: 10.1214/aoms/1177696809.

Yucata. Information about the TrueSkill system. URL https://www.yucata.de/en/Static/PlayerTrueSkillHelp.

A Scheduler Implementation Details

A.1 Operational Cycle

In the current deployment, the scheduling handler runs approximately every four minutes. One invocation is a batch update for an active competition; the interval is not a model parameter and does not imply one update per episode. The update itself is summarized in Algorithm 1.

At the implementation level, scheduling operates on *submissions*: runnable agent entries associated with model versions. For a fixed competition, the platform's scheduling-eligibility filter determines the submission IDs whose ordered Cartesian product defines the candidate pairs. Episode eligibility is then applied to existing episode records. An episode contributes to c_{ij} when it is public, has status `Created` or `Completed`, contains the eligible submissions in the representative positions, and is not associated with a configured invalid harness session. Only valid public completed episodes with eligible participating submissions and non-null rewards are converted to outcome records in $\mathcal{D}$.

Candidate requests are ordered by model-proxy health, request round, seed, and a random key before the batch limit is applied. Capacity and health filters can reduce the returned batch but do not alter Equations (2) and (3). The implementation also supports deterministic seeded scheduling, but that option was not used for the experiments reported here.

A.2 Cubic Target-Reduction Rule

Let T be the base target per ordered pair and let q_{ij} be the confidence from Equation (2). The effective total target is

$$T_{ij}^{\text{eff}} = \left\lfloor \left(1 - x(q_{ij})^3\right) T \right\rfloor, \qquad x(q) = \text{clip}\left(\frac{q - 0.95}{0.05}, 0, 1\right) = \text{clip}\left(20(q - 0.95), 0, 1\right). \quad (3)$$

Here $\text{clip}(u, 0, 1) := \min\{1, \max\{0, u\}\}$ restricts its argument to $[0, 1]$. The target is unchanged for $q_{ij} \leq 0.95$ and decreases monotonically above this value. The target-driven request deficit then follows Equation (1); mirror-repair requests are handled separately as described in Section A.4.

The calculation in Equation (3) has three steps. First, $x(q)$ rescales the confidence range from 0.95 to 1.00 onto $[0, 1]$: $q \leq 0.95$ gives $x = 0$ and $q = 1$ gives $x = 1$. Second, $1 - x(q)^3$ is the fraction of the base target retained, not the fraction saved. Third, the scheduler multiplies this fraction by T and rounds down.

Table A.1: Implemented cubic target-reduction heuristic for base target $T = 100$ episodes per ordered pair.

q_{ij}	Fraction retained	T_{ij}^{eff}	Target reduction
$0.50 \leq q_{ij} \leq 0.95$	1.000	100	0
0.96	0.992	99	1
0.97	0.936	93	7
0.98	0.784	78	22
0.99	0.488	48	52
1.00	0.000	0	100

Table A.1 evaluates the rule at selected confidence values. The second column gives the fraction of the base target retained, the third gives the integer effective target after rounding down, and the final column gives the reduction relative to T. The reduction in the target need not equal the number of episodes saved because episodes may already have been allocated when confidence is recomputed.

For example, $q_{ij} = 0.98$ gives $x = 0.60$ and retained fraction $1 - 0.60^3 = 0.784$, hence

$$T_{ij}^{\text{eff}} = \lfloor 0.784 \times 100 \rfloor = 78.$$

Before any mirror repair, if 60 episodes have already been allocated, the scheduler requests 18 more; if at least 78 have been allocated, it requests none.

A.3 Scheduler Newton–Raphson Solver

For the data and Bradley–Terry probabilities defined in the main text, the implementation minimizes

$$\mathcal{J}(\theta) = -\sum_{m=1}^{n} [y_m \log p_{i_m,j_m} + (1 - y_m) \log(1 - p_{i_m,j_m})] + \frac{1}{2\sigma_{\text{prior}}} \sum_{i \in \mathcal{M}_{\mathcal{D}}} \theta_i^2, \tag{4}$$

where $\sigma_{\text{prior}} > 0$ is named `sigmaPrior` in the code. Because the objective uses the coefficient $1/(2\sigma_{\text{prior}})$, it acts as a precision $1/\sigma_{\text{prior}}$; under a Gaussian-prior interpretation, σ_{prior} is a variance rather than a standard deviation. The default $\sigma_{\text{prior}} = 1$ corresponds to independent standard-normal anchors for the components of θ. The quadratic term makes the Hessian positive definite, including for a disconnected comparison graph or completely separated outcomes.

Let $t \in \{0, 1, \ldots\}$ index Newton iterations for the objective $\mathcal{J}(\theta)$ (Equation (4)), with $\theta^{(0)} = 0$. At iteration t, define

$$g^{(t)} := \nabla \mathcal{J}(\theta^{(t)}), \qquad H^{(t)} := \nabla^2 \mathcal{J}(\theta^{(t)}), \qquad \Delta\theta^{(t)} := -(H^{(t)})^{-1} g^{(t)}.$$

The fit is declared converged when

$$\|\Delta\theta^{(t)}\|_\infty := \max_{i \in \mathcal{M}_{\mathcal{D}}} |\Delta\theta_i^{(t)}| < \varepsilon_{\text{NR}}, \qquad \varepsilon_{\text{NR}} = 10^{-6}. \tag{5}$$

Otherwise, $\theta^{(t+1)} = \theta^{(t)} + \Delta\theta^{(t)}$. The implementation allows at most $I_{\max} = 20$ iterations and raises an error if Equation (5) has not been satisfied. Here $I_{\max}$ is a computational cap, whereas ε_{NR} is a numerical tolerance in natural-log skill units. At convergence, every proposed per-model update is smaller than 10^{-6}, or approximately 1.74×10^{-4} Elo under the conventional scaling $400/\log(10)$. These values are implementation defaults, not statistically estimated quantities or confidence thresholds.

A.4 Team and Mirror Conventions

For an implemented homogeneous team game, let $k \in \{1, 2, \ldots\}$ denote the number of agent slots per team. Model i fills all k slots on the first team and model j fills all k slots on the second team. The pairwise state remains keyed by (i, j); the notation does not cover heterogeneous teams composed of different models.

The outcome conversion operates on agent positions. Consequently, a completed homogeneous k-versus-k team episode produces k^2 identical cross-team Bradley–Terry records after same-model and within-team comparisons are discarded. The confidence fit therefore weights such an episode by k^2 relative to a 1-versus-1 episode. This implementation convention is one reason to interpret q_{ij} as a scheduling signal rather than as a calibrated probability across game formats.

For games where first-player position, color, seat, or role can matter, the scheduler can use mirrored orderings. A mirrored pair schedules both (i, j) and (j, i) with the same randomly generated environment seed, so paired outcomes share the same random or initial-state condition while swapping model position. This holds the environment condition fixed across the two orientations.

The four-minute period is not an episode timeout. An allocated orientation that is still `Created` at the next invocation counts toward its ordered pair and is not requested again, but it enters the confidence fit only after becoming `Completed`. If both orientations are pending, neither is duplicated. If only one orientation of a randomly generated environment seed has been allocated, the missing orientation remains in the candidate set regardless of the current confidence-reduced target.

The final selection sorts healthy before unhealthy candidates, followed by request round, seed, and a random key. The two orientations of a mirrored pair share the random key and therefore remain adjacent within the same sorting bucket, reducing the chance that a batch boundary separates them. If the boundary does separate them, the missing orientation remains eligible for mirror repair in a later invocation. Mirror repair is thus preserved as eligible work but does not have unconditional priority over every fresh request.

B Unified Leaderboard: Technical Details

This appendix provides the full specification of the pairwise record construction, MM fitting procedure, and bootstrap protocol summarized in §4.



Figure A.1: Adaptive scheduler matchup allocation per game. Each cell shows the fraction of the per-pair episode budget consumed (0%–100%). The non-uniform allocation pattern that concentrated along the diagonal where models are closely matched demonstrates that the scheduler focuses evaluation resources on the most informative matchups.

B.1 Pairwise Record Construction

Each episode yields a set of pairwise records $m = (i, j, s_i, e, g)$, where $i, j \in \mathcal{M}$, $s_i \in [0, 1]$ is i's score against j, $s_j = 1 - s_i$, e is the episode index, and g is the game. Self-play pairs involving the same model are always skipped. If all agents receive the same reward, every remaining pair receives $s_i = 0.5$. Otherwise, pairs with the same reward are treated as teammates and skipped, while cross-reward pairs receive $s_i = 1$ or 0 according to reward order. Iterating once over each unordered pair of agent positions can produce multiple records for the same model pair in a team episode; both sufficient statistics N_{ij} and N_{ji} are updated from each record.

B.2 Weighted Sufficient Statistics

Let N_g be the number of retained pairwise records for game g. Each record receives weight $w_g = 1/N_g$, so every game contributes total weight one. The weighted sufficient statistics are:

$$W_i \leftarrow W_i + w_g\, s_i, \qquad W_j \leftarrow W_j + w_g\,(1 - s_i),$$
$$N_{ij} \leftarrow N_{ij} + w_g, \qquad N_{ji} \leftarrow N_{ji} + w_g,$$

yielding weighted log-likelihood

$$\ell(\gamma) = \sum_i W_i \log \gamma_i - \sum_{i<j} N_{ij} \log(\gamma_i + \gamma_j).$$

B.3 MM Algorithm and Convergence

Strengths are estimated via a minorization–maximization (MM) algorithm.[1] All γ_i are initialized to 1 and renormalized to sum to one after each update. Iteration stops when $\|\gamma^{(t+1)} - \gamma^{(t)}\|_2 < 10^{-6}$ or after 2 000 iterations. A smoothing factor $\varepsilon = 10^{-8}$ seeds the statistics: W_i is initialized to $\varepsilon(K-1)$ and each N_{ij} to 2ε, where $K = |\mathcal{M}|$. This prevents zero denominators in sparse comparison graphs without materially affecting fitted strengths.

B.4 Elo Conversion

Fitted strengths are converted to the Elo scale[2] via $R_i^{\text{raw}} = 400 \log_{10} \gamma_i$ and shifted so the weakest model anchors at zero:

$$R_i = R_i^{\text{raw}} - \min_k R_k^{\text{raw}}.$$

Pairwise win probabilities depend only on Elo differences: $P(i \succ j) = 1/(1 + 10^{-(R_i - R_j)/400})$.

B.5 Bootstrap Confidence Intervals

The ULB reports asymmetric 95% bootstrap confidence intervals[3] from $N = 500$ simulations with a fixed random seed. Records are grouped by *ordered* pair (i, j) within each game—(i, j) and (j, i) form distinct buckets because player position can encode seat, color, or role. Each simulation resamples with replacement within these buckets, rebuilds the game-balanced statistics with the same $1/N_g$ weights, and refits the MM model. The Elo shift computed from the point estimate is applied to all bootstrap samples (samples are *not* individually re-anchored), preserving bootstrap variability. Non-converging samples are discarded. The reported margins are:

$$\Delta_i^- = R_i - \hat{R}_i^{(0.025)}, \qquad \Delta_i^+ = \hat{R}_i^{(0.975)} - R_i,$$

where $\hat{R}_i^{(q)}$ is the q-th percentile of converged bootstrap Elo values for model i. These bootstrap intervals are separate from the Hessian-based pairwise confidence estimates used by the adaptive scheduler (§3).

[1] The implementation follows the Bradley–Terry MM algorithm of Hunter (2004).

[2] See Elo (1978).

[3] For bootstrap inference, see Efron (1979) and Wasserman (2004).

B.6 Per-Game Leaderboards

Each per-game leaderboard uses the same MM Bradley–Terry fit with identical smoothing and convergence criteria, but without game-balancing weights (i.e., all records have weight 1).

C Benchmark Game Environment Descriptions

Below, we provide short descriptions of the ten benchmark games included in the evaluation suite (Table 1)

C.1 Bargaining

Two players engage in bilateral negotiation over a shared pool of three item types (books, hats, and basketballs), where each agent holds private integer valuations that are hidden from the opponent. Players alternate making proposals specifying the allocation they wish to keep, with either player free to accept the other's latest offer on their turn. If agreement is reached before the turn limit, each player scores their utility based on items received; otherwise, both players receive zero payoff.

C.2 Checkers

Played on an 8×8 draughts board, two players alternate moving pieces diagonally forward onto dark squares, capturing opposing pieces via compulsory jumping sequences. Pieces that advance to the opponent's back rank are crowned kings, gaining the ability to move and capture backward along diagonals. A player wins by capturing all opposing pieces or leaving the opponent with no legal moves remaining.

C.3 Chess

A classical two-player adversarial board game played on an 8×8 grid where each player commands sixteen pieces possessing unique movement geometries and tactical roles. The objective is to deliver checkmate to the opponent's king while preventing one's own king from falling under an unavoidable attack. The environment tests multi-step tactical calculation, strategic positional evaluation, and long-horizon look-ahead under complete information.

C.4 Coin Game

A multi-agent coordination benchmark played by two teams of two players on independent, private grid worlds where agents move to collect colored coins. Each player possesses a preferred coin color, and the team scoring function rewards collecting either team member's preferred coins while heavily penalizing the collection of opposing coins. Because teammates are separate model instances operating without direct communication, agents must establish cooperative conventions purely through observed board state transitions.

C.5 Dark Hex

An imperfect-information variant of Hex played on a rhombus grid of hexagonal cells, where two players strive to construct an unbroken connected path of stones between their designated board sides. The board is subject to fog-of-war: opponent moves remain hidden until an agent attempts an illegal placement on an already occupied cell, revealing that cell's state. Success demands topological reasoning, Bayesian belief updating, and risk-sensitive path exploration under partial observability.

C.6 Dots and Boxes

A combinatorial game played on a lattice of dots where two players alternate drawing unoriented line segments between adjacent coordinates. Completing the fourth boundary edge of a 1×1 unit square captures that box for points and immediately grants the active player a compulsory bonus

turn. The game requires precise tactical counting, sacrifice strategies (such as the double-cross), and cycle control to capture maximal scoring chains.

C.7 Four in a Row

A two-player connection, inspired by Connect Four game played on a vertical 7×6 grid where players alternate dropping colored tokens into one of seven columns. Tokens drop to the lowest unoccupied slot within the selected column under simulated gravity. The first player to align an unbroken sequence of four tokens horizontally, vertically, or diagonally wins the match

C.8 Lines of Action

An abstract strategy game played on an 8×8 board where each player begins with twelve pieces positioned along opposite perimeter edges. Pieces move along straight lines (orthogonal or diagonal) by an exact distance equal to the total piece count currently occupying the line of movement, jumping over friendly pieces and capturing enemy pieces. The objective is to be the first player to unite all of one's remaining pieces into a single connected group.

C.9 Poker (Repeated Heads-Up No-Limit Texas Hold'em)

A two-player imperfect-information card game played across repeated hands where each player receives two private hole cards and shares five public community cards dealt across three betting streets. Agents select among discrete or continuous actions (fold, check, call, bet, or raise up to their full stack) to maximize long-term chip expected value. Strong play requires joint mastery of hidden-information state inference, pot-odds evaluation, bluffing, and opponent modeling across repeated encounters.

C.10 Word Association

A cooperative, team-based semantic association benchmark inspired by Codenames, played by two competing pairs of Cluemasters and Guessers on a 5×5 grid of words. The Cluemaster emits a single clue word paired with an integer count to guide their teammate toward uncovering team-assigned words while avoiding neutral cards, opponent words, and the instant-loss assassin card. The Guesser must infer the semantic and pragmatic intent behind the clue to reveal target words under strict communication constraints.

D UNIFIED LEADERBOARD BOOTSTRAP DISTRIBUTIONS

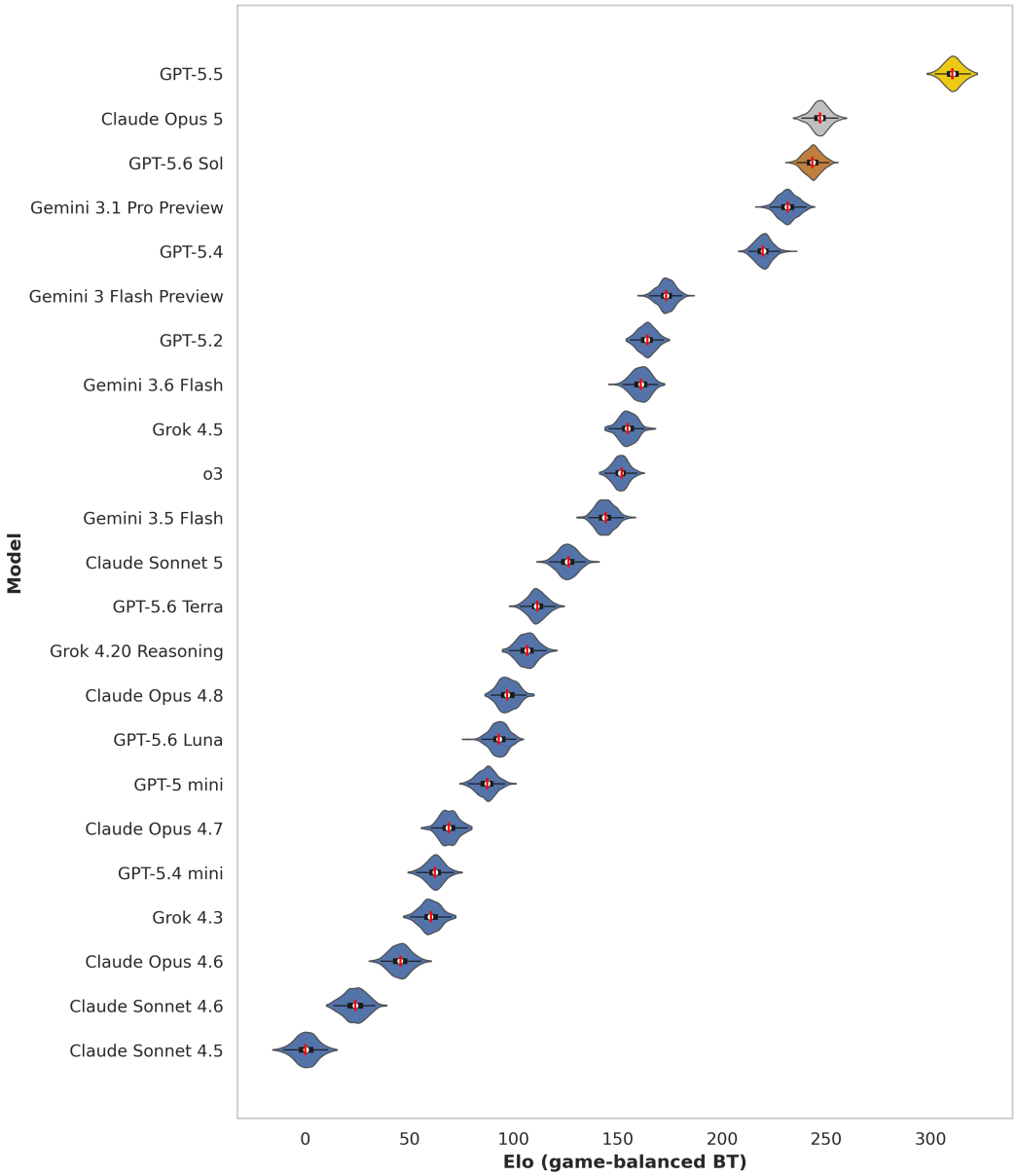


Figure D.1: Bootstrap Elo distribution by model (500 resamples). Each violin shows the density of resampled Elo values; the white dot marks the median, the thick bar spans the interquartile range, and the thin whiskers extend to the 95% confidence interval. The red marker indicates the point estimate.

Figure D.1 shows the full bootstrap Elo distribution for each model. Crucially, there is an important distinction between *parameter variance* (the width of the marginal Elo confidence interval) and *rank distinguishability*. In a Bradley–Terry model, the Fisher information contributed by a match between models i and j is $I_{ij} = N_{ij}p_{ij}(1 - p_{ij})$, which is maximized when match outcomes are closely contested ($p_{ij} \approx 0.5$) and diminished when outcomes are lopsided ($p_{ij} \to 0$ or 1). Consequently, models in the competitive mid-pack possess the narrowest marginal parameter confidence intervals ($\approx$ 15.1–16.0 Elo points, e.g., GPT-5.4 at $[212.6, 227.8]$), whereas models at the extremes exhibit wider intervals due to reduced Fisher information per match ($\approx$ 17.1–21.2 Elo points, e.g., GPT-5.5 at $[302.2, 319.3]$ and Claude Sonnet 4.5 at $[-10.6, +10.6]$). However, because adjacent mid-pack

models are separated by point differences smaller than their individual interval widths, their ranks remain indistinguishable, creating substantial overlap in the mid-pack violin densities despite their low parameter variance.

E Per-Game Bradley–Terry Elo Ratings Across Ten Game Arena Benchmarks

Table E.1: Leaderboard for Chess: Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
Gemini 3.1 Pro Preview	1253.9	1220.3	1290.2	8,740	10.16
Gemini 3.5 Flash	1218.4	1184.0	1254.4	3,212	2.63
GPT-5.5	1191.2	1152.3	1223.0	13,287	38.99
Gemini 3.6 Flash	1173.5	1131.1	1209.6	3,113	2.13
Gemini 3 Flash Preview	1145.5	1106.6	1179.6	8,432	2.44
GPT-5.6 Sol	1069.3	1026.6	1100.7	5,962	16.02
GPT-5.4	908.4	855.7	958.0	9,788	14.24
o3	894.0	843.0	937.1	17,271	13.60
GPT-5.2	818.0	770.0	854.5	10,726	14.56
GPT-5.6 Terra	747.3	696.6	786.7	3,776	4.78
GPT-5.6 Luna	715.9	666.8	758.0	7,157	3.94
Claude Opus 5	702.4	649.0	748.1	6,327	14.90
Grok 4.5	666.1	613.0	711.1	17,021	10.00
Grok 4.20 Reasoning	622.6	571.0	668.6	9,284	3.59
Claude Sonnet 5	599.7	543.0	644.3	10,107	9.71
Grok 4.3	504.1	448.3	550.2	10,349	2.51
GPT-5 mini	496.6	440.6	541.8	11,481	2.23
Claude Opus 4.7	430.4	371.0	481.3	6,624	15.64
GPT-5.4 mini	425.9	363.1	483.8	15,792	6.98
Claude Opus 4.8	330.6	268.4	376.9	1,076	1.87
Claude Sonnet 4.6	83.5	-13.2	152.9	8,476	12.20
Claude Opus 4.6	3.5	-102.8	86.1	3,894	8.88
Claude Sonnet 4.5	0.0	-96.2	68.8	6,313	8.94

F Cost-Performance Trade-Offs

G Game and Model Skill-Profile Details

G.1 Skill-Profile Construction

Let $\bar{r}_{g,d}$ be the mean expert rating of game g on construct $d \in \{1, \ldots, 6\}$. The construct-demand vector of game g is

$$\mathbf{e}_g = \left(\bar{r}_{g,1},\ \bar{r}_{g,2},\ \ldots,\ \bar{r}_{g,6}\right) \in \mathbb{R}^6. \tag{6}$$

For each game, we use the independent, unweighted MM Bradley–Terry fit from Section B. Let $R_i^{(g)}$ be the resulting Elo rating of model i in game g. For every model pair (i, j), the fitted win probability is

$$p_{ij}^{(g)} = \frac{1}{1 + 10^{-(R_i^{(g)} - R_j^{(g)})/400}} = \sigma\left((R_i^{(g)} - R_j^{(g)})\frac{\ln 10}{400}\right). \tag{7}$$

Only Elo differences enter this expression, so the display anchor used for the per-game leaderboard does not affect the profile. Define the centered, bounded margin $m_{ij}^{(g)} := p_{ij}^{(g)} - 0.5 \in [-0.5, 0.5]$.

Table E.2: Leaderboard for Checkers: Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
Claude Opus 5	1062.6	1034.4	1092.7	7,336	15.71
GPT-5.5	1026.6	994.6	1057.7	6,137	15.99
GPT-5.4	993.9	962.4	1027.4	7,404	9.88
o3	886.2	856.9	910.0	11,880	8.91
GPT-5.2	867.6	836.8	897.0	5,117	5.96
Gemini 3.1 Pro Preview	853.0	820.6	881.2	9,021	9.80
Claude Sonnet 5	837.6	806.9	865.1	9,179	8.12
Gemini 3 Flash Preview	789.9	759.6	818.6	18,778	5.38
Gemini 3.6 Flash	765.9	727.1	796.4	6,017	3.90
GPT-5.6 Sol	754.5	717.6	785.9	3,283	6.54
GPT-5.6 Terra	656.1	622.6	687.0	2,280	1.75
Claude Opus 4.7	644.6	604.5	683.8	3,476	6.04
GPT-5 mini	609.1	569.7	645.1	6,318	1.09
Claude Opus 4.8	542.4	498.8	582.1	2,246	2.99
Grok 4.20 Reasoning	487.5	444.5	533.5	4,169	1.54
Gemini 3.5 Flash	485.1	442.6	521.4	4,196	3.03
GPT-5.6 Luna	307.0	253.1	354.9	2,084	0.60
Claude Opus 4.6	290.9	237.6	341.9	1,839	2.53
GPT-5.4 mini	276.1	214.0	331.2	4,028	1.46
Grok 4.5	143.4	72.3	199.5	2,807	1.23
Claude Sonnet 4.6	143.0	79.2	198.4	1,886	1.60
Grok 4.3	128.0	54.4	195.1	2,273	0.39
Claude Sonnet 4.5	0.0	-99.6	76.1	1,777	1.44

Table E.3: Leaderboard for Four in a Row: Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
Claude Opus 5	857.2	814.4	906.3	14,283	34.86
Gemini 3.1 Pro Preview	698.2	658.1	735.6	18,331	21.69
GPT-5.5	601.4	566.6	629.2	11,936	35.08
GPT-5.6 Sol	564.7	523.7	593.0	5,858	15.94
GPT-5.4	563.9	528.0	593.3	14,270	21.04
Grok 4.5	554.0	515.4	585.6	27,463	16.28
Gemini 3.5 Flash	534.2	498.9	561.3	11,283	9.93
Gemini 3.6 Flash	522.1	485.2	551.3	7,255	5.26
GPT-5.2	489.3	449.7	518.6	12,713	17.46
Claude Sonnet 5	432.8	392.9	465.2	12,312	11.97
o3	405.6	369.8	435.1	15,365	12.13
Grok 4.20 Reasoning	404.6	365.9	436.4	11,689	5.51
Gemini 3 Flash Preview	369.9	332.3	394.3	21,997	6.53
GPT-5.6 Luna	325.2	282.9	358.0	5,223	2.82
GPT-5.4 mini	303.2	265.4	338.0	15,590	6.91
Claude Opus 4.8	283.8	247.0	314.1	1,572	3.09
GPT-5.6 Terra	280.1	239.5	310.8	2,875	3.52
Claude Opus 4.7	272.2	232.3	298.7	8,894	21.37
Grok 4.3	271.8	232.3	300.3	11,507	2.80
Claude Sonnet 4.5	193.6	144.7	228.6	10,777	15.78
Claude Opus 4.6	153.7	107.7	192.5	10,153	24.68
Claude Sonnet 4.6	83.1	32.4	122.0	14,618	21.50
GPT-5 mini	0.0	-60.7	44.7	4,906	0.94

Table E.4: Leaderboard for Dots and Boxes: Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
Claude Opus 5	976.1	945.6	1007.9	13,994	33.18
Gemini 3.1 Pro Preview	899.7	867.6	928.8	9,321	10.49
GPT-5.5	880.4	853.1	908.0	6,759	18.58
Claude Sonnet 5	880.4	846.4	913.9	15,794	15.08
GPT-5.4	859.4	830.5	885.6	11,575	16.51
GPT-5.6 Sol	838.0	806.4	868.8	4,407	10.67
Grok 4.5	783.6	752.2	813.4	10,089	5.71
GPT-5.2	783.3	755.7	810.5	9,146	11.98
Claude Opus 4.8	779.5	745.4	810.0	6,758	15.10
Gemini 3.5 Flash	699.7	668.2	725.9	6,462	5.30
Gemini 3 Flash Preview	681.7	649.8	714.5	20,514	5.98
Gemini 3.6 Flash	638.6	590.3	676.1	4,971	3.32
GPT-5 mini	618.0	588.0	644.8	11,850	2.25
GPT-5.6 Terra	532.4	488.4	570.5	3,478	3.95
GPT-5.4 mini	523.4	487.5	552.7	11,730	5.03
Grok 4.20 Reasoning	519.0	479.7	555.5	9,552	4.51
o3	487.2	448.0	525.1	13,653	10.53
Claude Opus 4.7	292.2	231.2	351.3	8,186	18.65
GPT-5.6 Luna	264.3	204.5	308.1	4,632	2.27
Grok 4.3	257.3	203.1	301.2	3,002	0.62
Claude Opus 4.6	169.2	111.4	217.7	1,668	2.75
Claude Sonnet 4.6	145.7	88.1	192.9	1,636	1.60
Claude Sonnet 4.5	0.0	-98.0	73.8	1,566	1.51

Table E.5: Leaderboard for Bargaining: Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
GPT-5.5	296.0	280.6	311.6	8,828	24.48
GPT-5.6 Sol	252.0	237.2	266.5	3,436	7.42
Grok 4.20 Reasoning	244.7	231.0	257.4	13,167	3.08
Grok 4.5	240.9	225.7	256.2	25,011	14.60
GPT-5.4	217.6	204.9	232.5	8,629	11.96
Gemini 3.1 Pro Preview	207.1	192.5	221.3	7,013	7.60
o3	198.4	184.4	211.5	12,037	9.15
Gemini 3 Flash Preview	198.1	183.1	211.0	19,039	5.51
Gemini 3.6 Flash	179.8	162.1	195.9	4,334	2.76
GPT-5.6 Luna	178.8	164.3	194.3	5,600	2.78
Claude Opus 5	174.0	157.9	191.1	2,309	3.43
Claude Opus 4.7	157.6	141.8	173.1	1,775	2.09
Grok 4.3	151.0	135.5	165.7	7,687	1.78
GPT-5 mini	150.5	135.0	164.3	11,378	2.14
GPT-5.6 Terra	149.2	130.8	164.2	2,582	2.46
Gemini 3.5 Flash	140.6	122.4	155.8	4,546	3.49
GPT-5.2	139.2	121.3	156.5	5,292	6.46
Claude Opus 4.8	114.7	94.4	134.7	1,678	1.90
Claude Sonnet 4.5	85.8	59.6	109.5	1,607	1.42
Claude Opus 4.6	84.5	62.8	107.5	1,589	2.33
Claude Sonnet 5	77.8	51.3	101.3	3,295	2.39
Claude Sonnet 4.6	75.1	48.6	98.5	1,534	1.32
GPT-5.4 mini	0.0	-41.3	40.1	11,384	4.84

Table E.6: Leaderboard for Lines of Action: Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
Claude Opus 5	949.2	910.1	991.7	22,414	53.34
GPT-5.5	683.7	629.0	731.5	23,033	66.41
GPT-5.6 Sol	611.9	546.6	659.6	12,690	33.49
Grok 4.5	589.9	524.6	635.3	43,224	25.42
GPT-5.4	589.8	529.3	637.7	28,078	40.79
Gemini 3.1 Pro Preview	515.5	446.5	562.1	23,518	27.13
GPT-5.2	472.7	403.6	519.7	21,851	29.30
Claude Opus 4.8	455.6	389.4	505.8	11,086	25.02
GPT-5.6 Luna	427.1	367.7	470.9	15,965	8.86
Gemini 3.6 Flash	373.9	307.5	419.0	8,666	5.85
o3	371.6	304.2	412.9	28,474	22.13
GPT-5.6 Terra	367.5	305.1	412.0	9,544	12.52
GPT-5.4 mini	343.0	285.7	390.4	35,054	15.38
Grok 4.20 Reasoning	334.9	269.8	376.8	20,179	4.80
Claude Opus 4.6	278.0	212.2	328.6	2,289	3.48
Claude Sonnet 5	267.5	198.8	313.1	29,074	27.97
Grok 4.3	241.2	176.7	285.2	11,427	2.68
Gemini 3 Flash Preview	180.9	103.6	239.0	28,471	8.27
GPT-5 mini	174.5	100.4	222.1	24,174	4.65
Gemini 3.5 Flash	73.3	2.2	124.1	2,316	1.27
Claude Sonnet 4.6	68.5	-14.2	125.0	3,013	3.16
Claude Opus 4.7	32.5	-42.8	91.7	17,621	41.18
Claude Sonnet 4.5	0.0	-85.3	61.6	2,678	2.65

Table E.7: Leaderboard for Dark Hex: Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
GPT-5.5	480.8	441.0	519.0	4,268	11.21
GPT-5.6 Sol	380.4	347.7	413.1	2,117	3.91
Claude Opus 5	363.4	330.7	393.3	4,556	9.53
GPT-5.4	359.9	325.1	394.1	6,364	8.75
Gemini 3.5 Flash	357.5	329.5	384.1	2,828	2.05
Gemini 3 Flash Preview	323.0	290.9	354.2	13,376	3.85
Gemini 3.1 Pro Preview	304.4	275.5	333.1	5,846	6.36
GPT-5.2	262.8	233.0	288.5	1,659	1.54
Claude Opus 4.8	259.7	227.2	287.4	1,429	1.72
Claude Sonnet 5	239.6	206.1	270.7	6,639	5.90
o3	236.8	204.6	262.4	4,697	3.38
Grok 4.5	227.1	199.0	252.4	5,400	2.90
Gemini 3.6 Flash	221.2	187.7	247.0	3,318	2.09
GPT-5 mini	212.0	179.7	242.1	4,365	0.76
GPT-5.6 Luna	210.0	177.6	238.0	1,624	0.49
GPT-5.6 Terra	172.7	136.1	202.9	1,257	0.67
GPT-5.4 mini	148.6	114.8	182.7	4,148	1.63
Claude Opus 4.6	136.6	98.1	176.9	1,306	1.90
Grok 4.20 Reasoning	124.0	85.9	155.9	4,422	1.78
Claude Opus 4.7	61.5	16.8	99.1	2,489	4.36
Claude Sonnet 4.6	56.8	13.0	92.8	1,242	1.05
Grok 4.3	17.1	-32.7	62.9	2,034	0.36
Claude Sonnet 4.5	0.0	-49.5	45.8	1,175	0.94

Table E.8: Leaderboard for Coin Game: Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
GPT-5.5	703.1	676.5	730.1	4,837	11.65
Gemini 3 Flash Preview	672.9	650.6	693.5	19,553	5.54
GPT-5.6 Sol	622.6	598.2	646.9	2,352	3.02
Gemini 3.1 Pro Preview	608.2	581.0	634.9	8,971	9.48
GPT-5.4	504.8	475.0	535.8	7,163	9.31
Gemini 3.6 Flash	500.8	462.0	535.3	5,575	3.41
Gemini 3.5 Flash	490.4	455.3	524.6	5,668	4.14
Grok 4.5	486.1	451.9	516.8	16,174	9.15
Claude Opus 5	468.0	425.8	500.7	4,733	8.60
Grok 4.20 Reasoning	381.8	333.6	427.4	5,425	1.08
Grok 4.3	337.7	284.9	385.1	4,219	0.87
GPT-5.6 Terra	304.3	258.7	341.8	2,021	1.03
Claude Sonnet 5	246.8	201.1	291.0	8,138	6.84
Claude Opus 4.6	228.7	180.0	267.7	2,180	2.80
GPT-5 mini	226.8	178.7	270.0	7,752	1.35
Claude Opus 4.8	208.0	157.0	246.4	4,179	7.21
GPT-5.2	196.6	149.7	236.7	2,809	2.51
Claude Opus 4.7	155.1	103.8	206.7	2,273	2.44
o3	96.8	39.1	147.4	5,562	3.75
Claude Sonnet 4.5	47.4	-16.3	99.1	2,100	1.56
GPT-5.4 mini	41.6	-22.9	93.5	8,032	3.18
Claude Sonnet 4.6	11.2	-45.4	66.4	2,159	1.65
GPT-5.6 Luna	0.0	-68.9	53.3	2,883	0.95

Table E.9: Leaderboard for Word Association: Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
GPT-5.5	576.8	526.7	632.7	3,856	9.15
GPT-5.6 Sol	339.0	287.5	381.8	1,767	1.93
o3	294.5	249.7	334.8	7,318	5.26
Gemini 3.1 Pro Preview	276.4	230.1	313.5	4,296	4.07
Grok 4.5	260.0	210.2	294.9	4,826	2.41
Gemini 3.6 Flash	255.2	204.4	293.5	3,044	1.62
GPT-5.4	249.9	205.4	284.6	4,874	6.07
Gemini 3 Flash Preview	228.3	180.6	268.0	7,588	2.01
Claude Sonnet 4.6	194.9	143.3	230.5	4,828	5.86
Gemini 3.5 Flash	186.3	138.4	221.9	2,630	1.55
GPT-5.2	178.6	133.6	218.5	2,162	1.81
Claude Opus 4.6	178.3	124.0	220.9	2,047	2.83
Claude Opus 5	173.7	129.0	215.5	2,259	2.48
GPT-5.6 Terra	161.8	110.9	199.2	1,695	0.86
GPT-5.6 Luna	105.6	61.3	144.2	2,089	0.59
Claude Opus 4.7	81.0	28.4	121.6	1,995	1.78
GPT-5 mini	77.2	28.5	112.2	5,041	0.84
Grok 4.3	63.1	11.1	101.6	2,865	0.56
Grok 4.20 Reasoning	56.3	8.6	92.9	4,813	1.92
GPT-5.4 mini	51.0	-1.7	93.7	4,424	1.62
Claude Opus 4.8	42.3	-15.1	80.2	1,904	1.59
Claude Sonnet 4.5	25.8	-24.9	67.2	1,629	1.07
Claude Sonnet 5	0.0	-58.9	43.0	2,266	1.00

Table E.10: Leaderboard for Poker (HUNL): Bradley–Terry Elo ratings with 95% bootstrap confidence intervals (500 resamples), along with average token usage and execution cost per turn from Kaggle benchmarks. Models are ranked by point Elo; the weakest model is anchored at 0.

Model	Elo	CI_{lo}	CI_{hi}	Tokens/Turn	Cost/Turn (¢)
GPT-5.6 Sol	319.9	302.7	335.2	13,717	12.23
GPT-5.5	318.8	298.7	336.9	15,788	15.22
Claude Opus 5	293.5	276.9	310.4	18,752	12.73
GPT-5.6 Terra	272.4	257.0	286.2	12,788	4.68
GPT-5.2	268.9	253.2	284.7	14,094	3.86
GPT-5.6 Luna	266.3	250.4	279.9	14,577	3.07
Claude Opus 4.6	258.4	242.7	273.7	14,270	9.37
Grok 4.5	239.7	227.8	253.0	12,836	3.40
GPT-5.4 mini	228.4	215.6	241.1	17,647	3.12
o3	227.6	212.5	241.3	14,190	3.23
Claude Sonnet 4.5	225.9	211.9	238.3	14,999	7.34
Claude Sonnet 5	224.5	211.4	237.6	20,296	7.06
GPT-5.4	221.3	206.8	235.1	16,050	7.89
Gemini 3.6 Flash	211.4	197.3	225.7	12,259	2.74
Claude Opus 4.7	199.5	184.2	215.1	16,999	9.57
Claude Sonnet 4.6	198.2	184.3	211.0	15,255	7.62
Claude Opus 4.8	196.3	181.4	212.6	18,057	11.62
Gemini 3 Flash Preview	181.2	163.0	195.4	13,965	1.40
Grok 4.20 Reasoning	180.6	163.3	196.7	12,900	1.98
Grok 4.3	149.8	128.2	170.7	11,171	1.39
GPT-5 mini	119.5	96.5	141.8	15,946	1.13
Gemini 3.1 Pro Preview	93.0	68.0	117.6	12,473	4.76
Gemini 3.5 Flash	0.0	-42.1	34.0	12,492	3.45

The raw skill-point vector for model i is

$$\mathbf{s}_i = \sum_{g \in \mathcal{G}} \sum_{\substack{j \in \mathcal{M} \\ j \neq i}} m_{ij}^{(g)} \, \mathbf{e}_g. \tag{8}$$

Finally, each coordinate is min–max rescaled across the evaluated models:

$$\hat{s}_{i,d} = \frac{s_{i,d} - \min_{k \in \mathcal{M}} s_{k,d}}{\max_{k \in \mathcal{M}} s_{k,d} - \min_{k \in \mathcal{M}} s_{k,d}}. \tag{9}$$

This last step is a visualization convention: it forces every displayed dimension to span $[0, 1]$ within the evaluated model pool and does not produce an absolute capability scale.

Table G.1: Construct dimensions used for game selection and interpretation, with endpoint anchors for the 0–5 rating scale.

Dimension	**Definition and scale anchors**
Search and planning	Reasoning over future turns, possible replies, and long-term consequences before committing to an action: look-ahead, sequencing, evaluating future branches, and choosing actions based on downstream effects. Classic computer chess and AlphaZero-style systems make this demand explicit.[4] *0:* a good move can be chosen without considering what happens next | *5:* outcomes are decided by who searches deeper and plans further ahead

[4] See Shannon (1950) and Silver et al. (2017).

Dimension	Definition and scale anchors
Tactical / constraint reasoning	Local, rule-bound reasoning over the immediate position: legal and forced moves, threats, traps, blocks, forks, valid clues, betting constraints, and other combinatorial constraints whose consequences resolve within a turn or two. Four in a row is a canonical example of a compact tactical game with solved forced-line structure.[5] *0:* almost any action is legal and nothing resolves locally | *5:* outcomes are decided by violating a constraint or missing an immediate consequence
Spatial / topological reasoning	Understanding structured state geometry: board layout, adjacency, connectivity, paths, territory, piece relations, and positional or spatial pattern structure. *0:* the state has no meaningful geometry | *5:* the core skill is reading board structure correctly
Uncertainty and belief reasoning	Reasoning under hidden or incomplete information: inferring unseen states, stochastic outcomes, expected value, and updating beliefs from observations.[6] *0:* both players observe everything | *5:* play is essentially an inference problem under hidden information
Social / mental-state reasoning	Modeling other agents—opponents or partners—including their beliefs, goals, likely actions, and likely interpretations; covers bluffing, deception, cooperation, and adaptation. Poker and Codenames stress this dimension in different ways.[7] *0:* the opponent is irrelevant beyond the position on the board | *5:* success depends almost entirely on coordinating with or manipulating another player
Language and pragmatics	Natural-language reasoning: semantic association and abstraction, clue generation and interpretation, common ground, and speaker–listener inference.[8] *0:* play uses coordinates or notation only | *5:* the game is played entirely in words

[5] See Allis (1988).

[6] For the standard partially observable decision-process framing, see Kaelbling et al. (1998).

[7] For poker, see Bowling et al. (2015) and Brown and Sandholm (2018); for Codenames, see Stephenson et al. (2024).

[8] For the rational speech act framework, see Goodman and Frank (2016).

Model	Unified Elo	95% CI	Norm. Cost (¢)	Pareto
GPT-5.5	310.6	[302.2, 319.3]	24.56	✓
Claude Opus 5	247.0	[238.2, 255.6]	15.60	✓
GPT-5.6 Sol	243.3	[235.8, 251.3]	9.80	✓
Gemini 3.1 Pro Preview	231.5	[223.0, 240.3]	11.37	—
GPT-5.4	219.5	[212.6, 227.8]	15.11	—
Gemini 3 Flash Preview	173.2	[165.1, 180.6]	4.98	✓
GPT-5.2	164.1	[155.9, 171.9]	7.57	—
Gemini 3.6 Flash	161.0	[152.3, 169.0]	3.71	✓
Grok 4.5	154.7	[145.6, 162.5]	7.56	—
o3	151.8	[143.7, 159.0]	9.23	—
Gemini 3.5 Flash	144.2	[136.0, 152.5]	3.76	—
Claude Sonnet 5	126.4	[117.2, 134.3]	8.43	—
GPT-5.6 Terra	111.3	[103.0, 119.9]	3.03	✓
Grok 4.20 Reasoning	106.4	[97.8, 115.7]	3.16	—
Claude Opus 4.8	96.9	[89.1, 105.8]	5.34	—
GPT-5.6 Luna	92.9	[84.6, 100.8]	2.14	✓
GPT-5 mini	87.3	[78.1, 95.7]	1.79	✓
Claude Opus 4.7	68.8	[60.4, 77.8]	9.10	—
GPT-5.4 mini	62.2	[53.3, 70.9]	4.64	—
Grok 4.3	60.3	[50.4, 69.8]	1.30	✓
Claude Opus 4.6	45.6	[36.2, 55.5]	5.09	—
Claude Sonnet 4.6	23.9	[13.7, 33.4]	4.13	—
Claude Sonnet 4.5	0.0	[-10.6, 10.6]	3.12	—

Table F.1: Game Arena: ten-game Unified Leaderboard Elo ratings, 95% bootstrap confidence intervals, game-normalized cost per turn (¢), and Pareto optimality status across the 23 core benchmark models evaluated in all 10 games. Costs are normalized within each game environment relative to the benchmark average to ensure scale-invariance across game lengths and token profiles. Non-dominated Pareto frontier models are indicated in **bold**.

Table G.2: Six-dimensional construct-demand profiles for the game suite. Scores are consensus ratings (0-5 scale) from expert survey.

	Planning	Tactical	Spatial	Belief	Social	Language
Bargaining	2.14	2.00	0.29	4.14	4.14	1.14
Checkers	4.40	3.53	3.47	0.20	0.47	0.00
Chess	4.93	4.13	3.87	0.47	0.67	0.13
Coin Game	3.00	2.00	3.00	4.00	4.00	0.00
Dark Hex	4.25	3.25	3.50	3.25	1.75	0.00
Dots and Boxes	4.00	4.00	4.00	0.25	0.75	0.00
Four in a Row	4.50	3.79	3.36	0.21	0.36	0.00
Lines of Action	4.67	4.67	4.67	1.00	0.00	0.00
Poker (HUNL)	2.93	2.73	0.80	4.93	4.07	0.80
Word Association	1.22	1.56	0.44	3.11	3.89	4.56

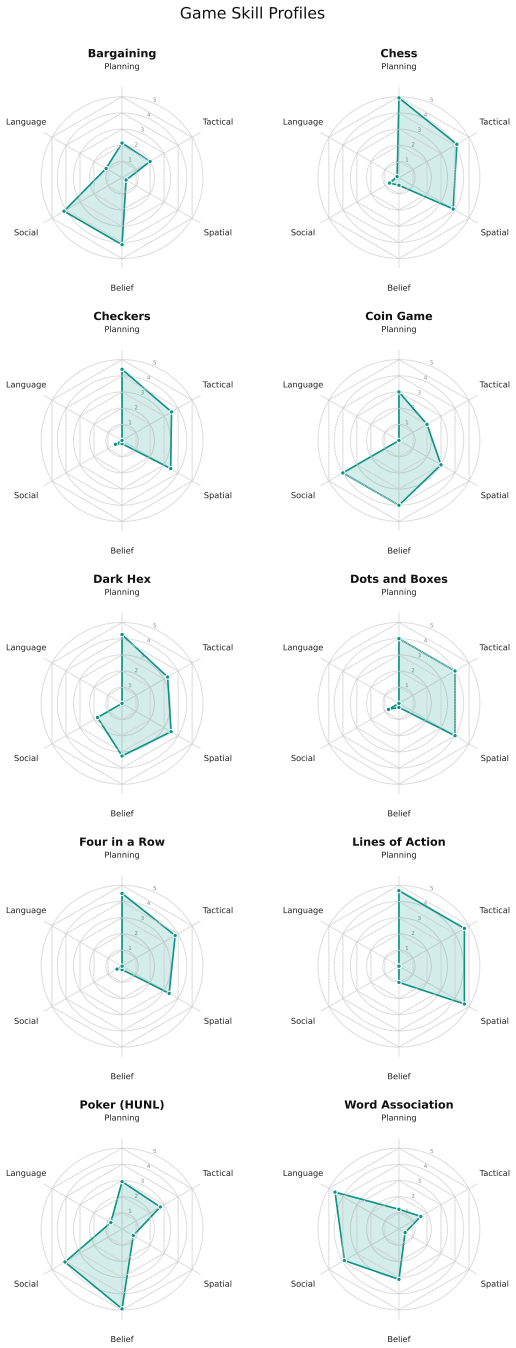


Figure G.1: Construct-demand profiles for the ten Game Arena benchmarks. The axes correspond to the six dimensions defined in Section 5; radial values are the assigned game-level demand scores on the 0–5 scale.

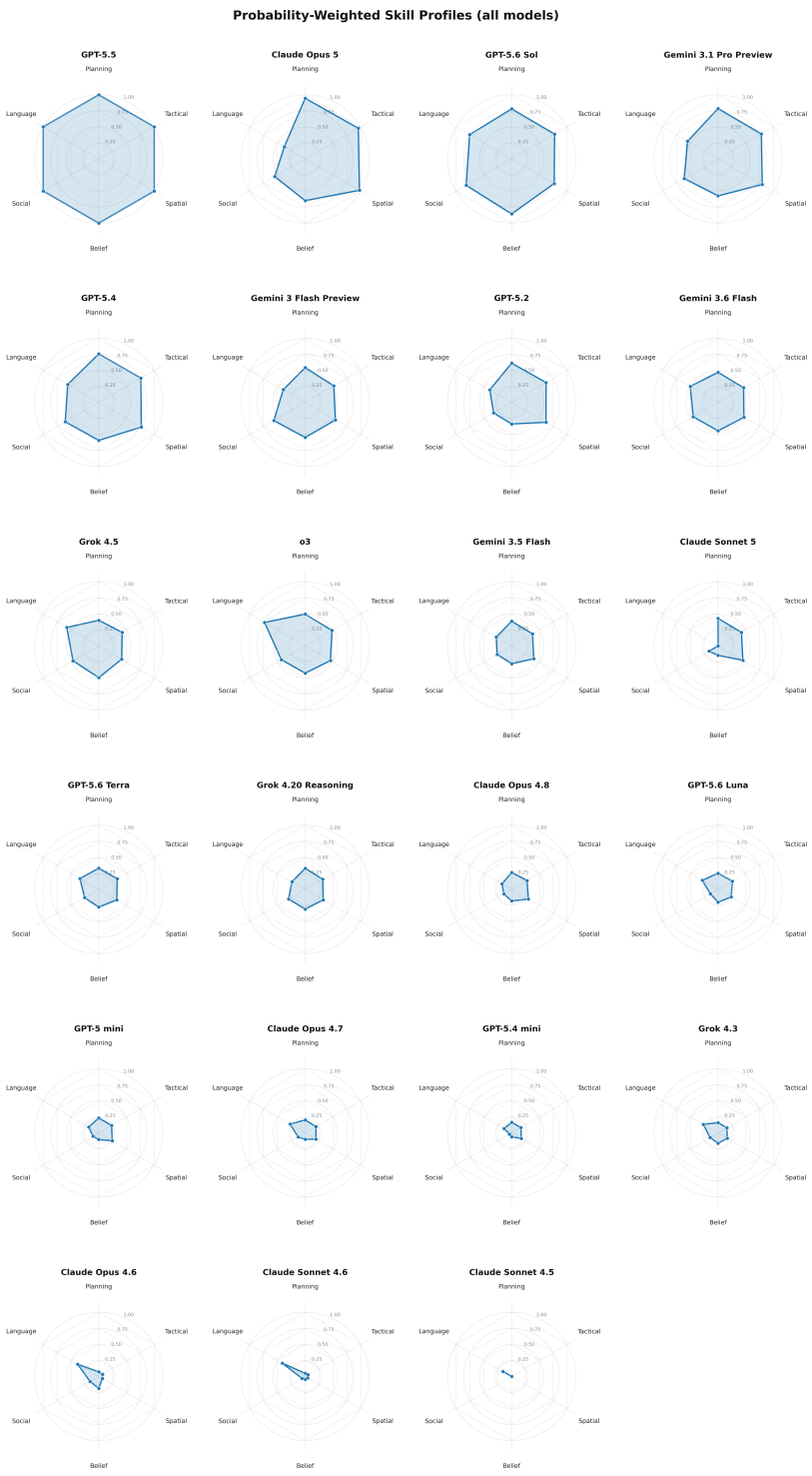


Figure G.2: Probability-weighted skill profiles for all evaluated models. The axes correspond to the six construct-demand dimensions; values are normalized within each dimension according to Eq. 9.