RuleSmith: Multi-Agent LLMs for Automated Game Balancing

Ziyao Zeng[1] **Hao Wang**[2] **Chen Liu**[1] **Youheng Yao**[1] **Jingcheng Ni**[3]
Tianyu Liu[1] **Xiatao Sun**[1] **Fengyu Yang**[1] **Chenyu You**[4] **Xiaofeng Liu**[1]
Daniel Rakita[1] **Ronald R. Coifman**[1] **Yuval Kluger**[1] **Zhiwen Fan**[2]

[1]Yale University [2]Texas A&M University [3]Brown University [4]Stony Brook University
ziyao.zeng@yale.edu

Abstract

Game balancing is a longstanding challenge requiring repeated playtesting, expert intuition, and extensive manual tuning. We introduce *RuleSmith*, a framework that couples multi-agent LLM self-play with surrogate-based optimization over a parameterized rule space. As a proof of concept, we build *CivMini*, a simplified civilization-style game with two asymmetric factions governed by 12 tunable parameters. LLM agents interpret textual rulebooks and game states to propose actions that are validated by the game engine, yielding noisy estimates of faction win rates. Our primary implementation uses Gaussian-process Bayesian optimization with Expected-Improvement-based adaptive sampling to allocate more evaluation games to promising candidates. RuleSmith consistently finds near-balanced configurations: three independent optimization runs reach near-even empirical win rates in independent 200- and 300-game evaluations, and additional experiments with NTBEA and TPE show that the same self-play interface supports different search components. Cross-model evaluations and small human studies provide preliminary evidence of transfer across player populations. RuleSmith produces interpretable numerical rule adjustments without task-specific policy training. Project and code: https://adonis-galaxy.github.io/RuleSmith-website/.

1 Introduction

Balancing asymmetric strategy games is a core challenge in game design and multi-agent research. Factions differ in abilities, mobility, and economic structure, making it hard to tell whether a given ruleset produces fair or strategically rich gameplay. Traditional approaches rely on human experts cycling through manual tuning, heuristic adjustments, and subjective playtesting. These workflows are slow, expensive, and hard to scale. As games grow more complex, this process becomes not just time-consuming, but often intractable.

Many applications beyond games, including economic modeling, policy analysis, and cybersecurity, face a similar challenge: outcomes depend on parameterized rules, and assessing how rule changes propagate through multi-agent interactions is expensive. We believe the core idea of LLM self-play as a rule evaluator could extend to such settings, though our current work is a proof of concept in a controlled game environment; broader applications remain future work.

Large language models (LLMs) offer a promising new approach. LLMs can interpret rulebooks, reason over game states, and act as zero-shot simulators without task-specific policy training. Prior work shows that LLM agents can negotiate, plan over multiple steps, and exhibit complex strategic behavior [1–3]. LLM-assisted game design also uses language models to propose parameter edits

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

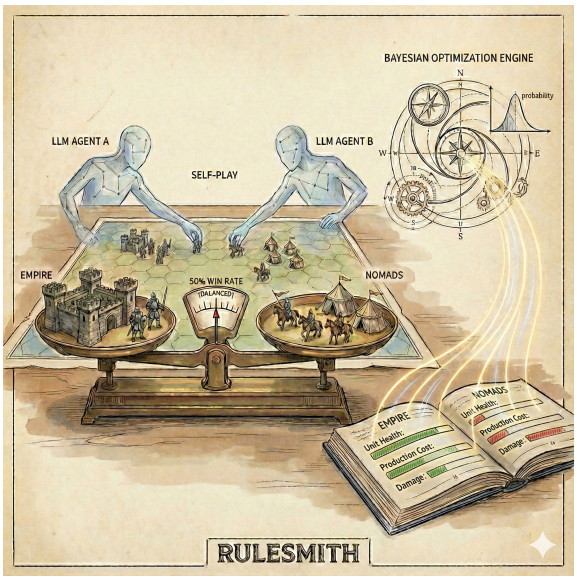


Figure 1: **Overview of RuleSmith.** RuleSmith treats asymmetric game balancing as optimization over game rollouts using multi-agent LLMs given a natural-language rulebook. It optimizes interpretable parameters such as unit HP, attack damage, resource gain, production cost, and scoring weights. For each candidate game parameter setting, two LLM agents play the game zero-shot, and their outcomes estimate the faction win rates and balance loss. A Bayesian optimizer then proposes new settings, while adaptive sampling allocates more games to promising candidates. This enables closed-loop balancing without task-specific policy training, reducing manual playtesting and parameter tuning. Figure generated by Nano Banana Pro.

from simulation feedback [4]. We study a complementary role: LLM agents supply the gameplay evaluations, while a separate optimizer searches the rules.

We introduce **RuleSmith**, a framework for automatically balancing asymmetric games through LLM self-play and Bayesian optimization. As a proof of concept, we built **CivMini**, a fully parameterized, grid-based, turn-based game inspired by civilization-style mechanics. Two asymmetric factions, Empire and Nomads, have distinct units, production systems, and combat stats. RuleSmith exposes tunable parameters governing HP, damage, resource gathering, production costs, and scoring weights. This shifts the object of optimization from agent policies to the game rules themselves.

To evaluate a candidate ruleset, RuleSmith runs two LLM agents and collects win rates from multiple self-play games. Since LLM gameplay evaluation is expensive and noisy, we pair Bayesian optimization with acquisition-based adaptive sampling: the number of games scales with each candidate's Expected Improvement, concentrating compute where balance improvement is most expected.

RuleSmith discovers configurations with near-even empirical faction win rates from imbalanced starting settings. Three independent optimization runs and larger held-out evaluations quantify the repeatability and uncertainty of these results. The optimized parameters describe how HP, resource efficiency, and production costs jointly affect outcomes under the chosen player population.

Contributions.

- **Zero-shot self-play from natural-language rulebooks.** LLM agents play CivMini from a rulebook and structured game states without task-specific training, with legality enforced by the engine.
- **Multi-agent pipeline for automated game balancing.** RuleSmith makes balance queryable through LLM self-play and connects this evaluator to interchangeable search components. Our primary implementation uses Gaussian-process Bayesian optimization and Expected-Improvement-based adaptive evaluation budgets.
- **Empirical validation across models and game configurations.** RuleSmith consistently reaches near-balance: three independent runs yield near-even empirical win rates, with confidence intervals from larger re-evaluations. We study stronger optimizer baselines, model and map variations, action validity, trajectory diversity, and transfer to small human-player samples.

2 Related Work

Game design automation. AI-based playtesting evaluates and refines board, trading-card, and video games using scripted or learned players [5–7]. Bayesian optimization has been used for game

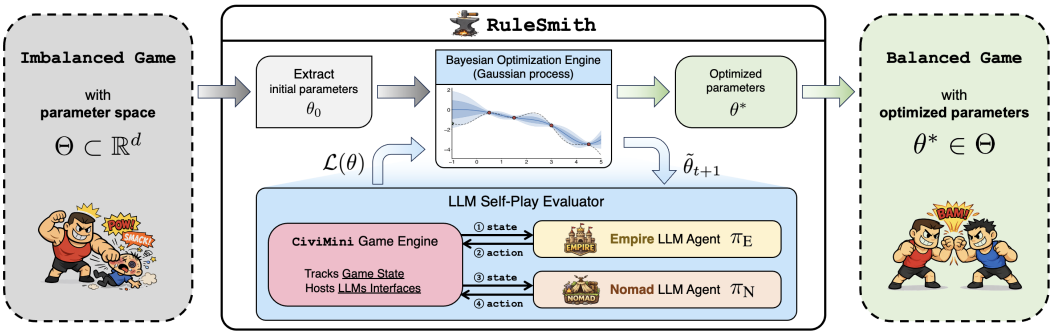


Figure 2: **Overview of the RuleSmith method.** We represent an asymmetric, turn-based strategy game (CivMini) as a parameterized rule space $\theta \in \Theta$, covering economy, combat, production, and scoring parameters. Given a candidate rule configuration θ_t, two role-specific LLM agents (*Empire* and *Nomads*) play N_t self-play games in the CivMini environment, producing a balance loss $\mathcal{L}(\theta) = |w_E(\theta) - 0.5| + |w_N(\theta) - 0.5| + 0.5\, w_D(\theta)$, where w_E, w_N, and w_D are the Empire win rate, Nomads win rate, and draw rate, respectively. A Bayesian optimizer maintains a surrogate model $g(\theta)$ over a continuous relaxation of the rule space and selects new candidates $\tilde{\theta}_{t+1}$ by maximizing an acquisition function. The number of games N_t is adaptively determined based on the Expected Improvement: promising candidates receive more games for accurate evaluation. Each continuous proposal is mapped to a valid, discrete ruleset via a deterministic discretization operator $D(\cdot)$ before evaluation. Cartoons in this figure were generated using ChatGPT-5.2.

testing [8]; multi-fidelity methods motivate allocating different evaluation budgets to expensive candidates [9]. Procedural content generation (PCG) creates levels, maps, and mechanics [10–12]; PCGML learns from examples [13], while discriminative and player-aware approaches guide content evaluation [14–16]. RuleSmith searches numerical rules of an asymmetric executable game, with LLM self-play supplying the evaluations.

LLMs as designers versus evaluators. Huang et al. [4] present AI Game Designer, which uses LLM-guided reasoning to adjust level parameters from the statistical outputs of simulation. In that architecture, the LLM proposes design edits. RuleSmith instead uses LLMs as competing players: their rollouts provide a noisy balance objective, and an external optimizer proposes rule configurations. The engine executes and validates actions, while the LLM players supply behavior conditioned on the rulebook. These roles are complementary. Agent-based game-parameter tuning also predates LLMs: NTBEA was developed for noisy, expensive game evaluation [17, 18]. Our contribution is the empirical side of this question: whether such a loop converges to balance at all, measured with multi-seed runs, optimizer baselines, large-sample re-evaluations, and significance tests. To our knowledge, this assignment of roles has not been studied for asymmetric rule spaces.

Multi-agent self-play. Self-play reinforcement learning primarily optimizes policies under fixed rules. AlphaGo and AlphaZero master board games [19–21], while OpenAI Five and AlphaStar address complex multiplayer environments [22, 23]. CFR, neural fictitious self-play, and PSRO study strategic play and equilibrium approximation [24–27]; opponent-learning and population-based approaches model competitive adaptation [28, 29]. We optimize the rules rather than train player policies, using instruction-following LLMs as the evaluation population.

LLM agents. LLM-agent surveys cover tool use, autonomous planning, and simulation [30–32]. ReAct and HuggingGPT coordinate reasoning and actions [33, 34], while CAMEL, generative agents, and Mindstorms organize interacting agents [35, 3, 36]. ALYMPICS studies strategic interaction in an LLM game-theory playground [37], and language models can themselves act as optimizers [38] or compete in debate [39, 40]. RuleSmith grounds interaction in engine-validated game actions and connects the resulting outcomes to rule optimization.

3 Method

RuleSmith takes five inputs: (i) an engine that executes actions and returns outcomes, (ii) a parameterized rule space with bounds and native granularities, (iii) a textual rulebook, (iv) an instruction-following LLM that proposes actions, and (v) a scalar balance objective. The engine executes the rules and validates actions; the LLM agents supply player decisions. The optimizer queries the same self-play interface regardless of its search algorithm. CivMini instantiates this interface in a controlled setting.

We consider balancing an asymmetric, parameterized turn-based strategy game so that two factions have approximately equal win probabilities under a specified LLM-player population. Let $\theta \in \Theta \subset \mathbb{R}^d$ denote a vector of game parameters (e.g., unit damage, production cost, score weights), and let π_{E} and π_{N} denote the policies of the LLM agents playing *Empire* and *Nomads*.

For a fixed parameterization θ, let $w_E(\theta)$, $w_N(\theta)$, and $w_D(\theta)$ denote the probabilities that a rollout ends in an Empire win, a Nomads win, or a draw, respectively. These probabilities satisfy

$$w_E(\theta) + w_N(\theta) + w_D(\theta) = 1. \tag{1}$$

In practice, we estimate them from N self-play games. For example, the empirical Empire win rate is

$$\widehat{w}_E(\theta) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\{\text{Empire wins game } i \mid \theta\}. \tag{2}$$

The quantities $\widehat{w}_N(\theta)$ and $\widehat{w}_D(\theta)$ are estimated analogously.

Our goal is to choose θ so that faction win probabilities are close to 0.5 under the specified player population, while avoiding frequent draws. We therefore use the balance loss

$$\mathcal{L}(\theta) = |w_E(\theta) - 0.5| + |w_N(\theta) - 0.5| + 0.5\, w_D(\theta). \tag{3}$$

The first two terms measure deviations from the target win probabilities. With no draws, their sum equals $|w_E - w_N|$. When draws occur, $w_E + w_N < 1$, so draws also affect the target-deviation terms; the explicit $0.5 w_D$ term further discourages unresolved games. This term is a safeguard that keeps the optimizer away from configurations in which neither side can win; the coefficient is a design choice, not a tuned optimum, and because the term is almost always zero, any moderate value would serve the same purpose. Draws are rare in the additional evaluations (0/500 in one and 2/300 in another), so this term contributes at most a few thousandths to the measured loss. City destruction or score-based resolution determines almost all outcomes.

In practice, we treat $\mathcal{L}$ as an expensive and noisy black-box function: each evaluation of $\mathcal{L}(\theta)$ requires multiple LLM-driven game simulations and is affected by stochasticity in model decisions and game dynamics. We therefore optimize (3) with Bayesian optimization over a continuous relaxation of Θ, combined with a discrete projection that maps suggested parameters to valid game configurations.

3.1 Parametric Asymmetric Game: CivMini

To study asymmetric game balancing in a controlled setting, we introduce *CivMini*, a minimalistic two-faction, turn-based asymmetric strategy game inspired by 4X games[1] such as *Civilization* [41]. CivMini is implemented as a deterministic, grid-based environment with a fully observable state and a small but expressive set of unit types and actions.

Map, factions, and units. The game is played on a 7×7 grid with two asymmetric factions: *Empire* and *Nomads*. Each starts with a city (Empire at $(1, 1)$, Nomads at $(5, 5)$) and units at fixed coordinates. Empire specializes its mobile units: `Farmers` gather resources and `Soldiers` fight. Nomads deploy a single combat unit type, `Cavalry`, with higher mobility (2 cells per turn versus 1 for Empire units). Cavalry cannot `GATHER`, but Nomads earn resources from kills, incentivizing aggression. Cities of both factions can also actively `PRODUCE_RESOURCE`. Farmers have fixed HP (5.0); Soldier and Cavalry HP are optimizable.

[1] 4X games: A subgenre of strategy games characterized by eXplore, eXpand, eXploit, and eXterminate, emphasizing large-scale empire management and long-horizon planning.

Actions and turn structure. Play proceeds in discrete turns, up to a fixed maximum number of turns $T_{\max}$. In the main experiments, $T_{\max}$ is fixed and is not part of the 12-parameter rule space; Appendix F.1 studies game-design variants with different map sizes and corresponding turn limits. In each turn, players alternate controlling all of their units. We use a multi-action mode in which *each unit* of the current player performs exactly one action per turn, allowing a single model response to coordinate the units while the engine resolves their actions sequentially. For each unit, the action space includes:

- `GATHER`: collect resources from the current tile (Farmers only),
- `MOVE`: move up to the unit's movement allowance in Manhattan distance to an unoccupied tile (Cavalry move up to 2 cells, Farmers and Soldiers up to 1; Cities cannot move),
- `BATTLE`: attack an adjacent enemy unit or city, dealing parameterized damage (Soldiers, Cavalry, and Cities can battle),
- `PRODUCE_RESOURCE`: cities generate resources,
- `PRODUCE_UNIT`: cities spend resources to spawn new units in adjacent empty cells,
- `PASS`: take no action.

One LLM call proposes the current player's actions jointly, reducing inference cost relative to calling the model for every unit. Actions then resolve in a fixed order (city first, followed by units in creation order), and each is rechecked against the updated state. Thus, a city cannot spend resources gathered later in the same player turn. An earlier move can also invalidate a later move, in which case the later action becomes `PASS`. A single-action mode is available, but all main experiments use joint action proposals.

The engine enforces hard constraints such as one action per unit per turn, one unit per tile (no stacking), adjacency requirements for combat, and immobile city tiles. City tiles are modeled as attackable entities with hit points, making it possible to win by conquest.

Victory and scoring. Games may end early if one faction destroys the opposing city. Otherwise, when the turn limit is reached, the game is resolved by a score-based tie-breaker. The final score for each faction is a weighted combination of (i) remaining resources, (ii) the number of battles won, and (iii) the number of surviving units, each controlled by a corresponding parameter in θ:

$$\begin{aligned} \text{score} = \theta_{\text{res}} \cdot \text{resources} + \theta_{\text{battle}} \cdot \text{battles_won} \\ + \theta_{\text{unit}} \cdot \text{surviving_units}. \end{aligned} \tag{4}$$

The balance score is defined in Eq. (3); it penalizes deviations from the target win rate and draws.

Parameterization. CivMini exposes $d = 12$ tunable parameters that jointly define the game rules:

- *Economy* (3 parameters): initial resources, Empire farmer gather amount, and Nomads kill resource gain;
- *Combat* (4 parameters): damage per attack and HP for Empire Soldiers and Nomad Cavalry;
- *Production* (2 parameters): unit production cost in resources for each faction;
- *Scoring* (3 parameters): weights for resources, battles won, and surviving units.

Table 1 gives the parameter ranges and grids. Mobility, initial units, and Farmer HP remain fixed to preserve faction identity; setting Farmer HP to 5 maintains vulnerability to raids. The remaining numerical strength, economy, cost, and score parameters are searched. This partition and its bounds are designer inputs, and optima at the boundaries can motivate expanding the ranges.

3.2 LLM Self-Play as an Evaluator

Given a parameterized game instance θ, we use two LLM agents to play the Empire and Nomads roles and estimate the resulting balance. At each decision point, the environment constructs a structured natural-language description of the game state, including the current turn index, each faction's resources and unit positions, a faction-specific strategy guide, and the list of legal actions. This description is concatenated with a role-specific system prompt and rules retrieved from a lightweight RAG system (TF-IDF + cosine similarity). Since we use multi-action mode, the agent is asked to produce a JSON object containing actions for *all* units simultaneously:

```
{"empire_farmer_0": {"action_type": "GATHER"},
"empire_city": {"action_type": "PRODUCE_UNIT",
"produce_unit_type": "soldier", "to": {"x": 1, "y": 2}}}
```

The engine parses and validates the response: a unit whose action is missing, unparseable, or outside its enumerated legal set receives a fallback legal action chosen according to a fixed priority (`BATTLE`, `GATHER`, `MOVE`, `PASS`) with random tie-breaking. Actions are checked again during sequential execution, and any action invalidated by an earlier action in the same turn becomes `PASS`. The opponent acts after the current player finishes.

Strategy guides supply state-dependent heuristic action recommendations, and JSON examples specify the output format. Zero-shot refers to the absence of task-specific policy training (Appendix J). For each candidate θ, we run N games with independent rollout randomness. Stochastic decoding (temperature 0.7, top-p 0.9) varies the actions and thus the trajectories under identical parameters. Appendix I quantifies the resulting diversity and execution-level conflicts. Engine validity does not imply optimal action selection or the absence of strategically biased play.

3.3 Bayesian Optimization over Rule Space

Enumerating the roughly 1.9×10^{10} discrete configurations would require prohibitively many LLM rollouts. We therefore employ Bayesian optimization [42] over a continuous relaxation of the rule space.

We maintain a Gaussian process (GP) surrogate model $g(\theta)$ of the balance loss $\mathcal{L}(\theta)$, implemented with scikit-optimize [43]. At iteration t, the optimizer selects a new candidate $\tilde{\theta}_t$ by maximizing an acquisition function over the history $\mathcal{D}_{t-1} = \{(\theta_i, f(\theta_i))\}_{i=1}^{t-1}$. Before evaluation, a deterministic projection D maps $\tilde{\theta}_t$ to a valid discrete configuration $\theta_t = D(\tilde{\theta}_t)$ by rounding and clamping each coordinate to the native grid in Table 1: integer steps except for the resource-score weight, which uses steps of 0.1. The surrogate is updated with the evaluated, discretized coordinates.

Acquisition-based adaptive sampling. To improve efficiency, inspired by multi-fidelity optimization [44, 45], we introduce an adaptive sampling strategy that allocates more evaluation budget to promising candidates. At each iteration, we dynamically adjust the number of self-play games N_t based on the Expected Improvement (EI) of the proposed candidate:

$$\mathrm{EI}(\theta) = (y^* - \mu(\theta))\Phi(z) + \sigma(\theta)\phi(z), \tag{5}$$

where y^* is the best observed value, $\mu(\theta)$ and $\sigma(\theta)$ are the GP posterior mean and standard deviation, $z = (y^* - \mu(\theta))/\sigma(\theta)$, and Φ and ϕ are the standard normal CDF and PDF. The selected candidate receives an integer number of games based on its EI relative to the running maximum. Let $e_t = \mathrm{EI}(\tilde{\theta}_t)$, $M_t = \max_{i \le t} e_i$, and $\epsilon > 0$ prevent division by zero. Then

$$N_t = \mathrm{clip}_{[N_{\min}, N_{\max}]}\left(\left\lfloor N_{\min} + (N_{\max} - N_{\min}) \min\left\{ \frac{e_t}{\max(M_t, \epsilon)}, 1 \right\} \right\rfloor\right). \tag{6}$$

We use $N_{\min} = 16$ and $N_{\max} = 64$; the released implementation uses $\epsilon = 10^{-3}$. The first 10 random initialization points receive $N_{\min}$ games because no fitted posterior is available. Zero EI also yields $N_{\min}$. EI is computed on the pre-evaluation posterior, so N_t is fixed before observing the candidate's outcomes. This is an allocation heuristic whose practical benefit is assessed empirically, rather than an optimal allocation guarantee.

4 Results

We first summarize what the theory does and does not cover (Section 4.1) and provide the implementation details (Section 4.2). We then report the main balancing results across agent pairings (Section 4.3), their repeatability and statistical uncertainty (Section 4.4), ablations of the search component and the player population (Section 4.5), and extended game configurations (Section 4.6).

4.1 Theoretical Results

Our theory concerns balance estimation and idealized GP-based search; it does not guarantee the implemented procedure. Proposition C.1 shows that exact balance is attainable under role symmetry

Table 1: **Optimizable parameters in CivMini.** We optimize 12 parameters governing economy, combat, production, and scoring. Precision indicates the discretization granularity applied before evaluation.

Parameter	Range	Prec.	Parameter	Range	Prec.
Economy			*Production*		
Initial resources	$[2, 10]$	int	Empire unit cost	$[2, 10]$	int
Empire farmer gather	$[1, 5]$	int	Nomads unit cost	$[2, 10]$	int
Nomads kill gain	$[1, 10]$	int			
Combat			*Scoring*		
Empire damage	$[1, 5]$	int	Resource weight	$[0.1, 0.5]$	0.1
Nomads damage	$[1, 5]$	int	Battle weight	$[1, 5]$	int
Empire soldier HP	$[4, 16]$	int	Unit weight	$[1, 5]$	int
Nomads cavalry HP	$[4, 16]$	int			

when the draw probability is zero, while Lemma C.2 shows that the empirical projected balance loss from self-play concentrates around the true projected loss at rate $O(N_t^{-1/2})$. Proposition C.3 provides an idealized GP-LCB regret bound under explicit calibration and information-gain assumptions. It is a statement about the balance loss and the GP model class: under these assumptions, some GP-based rule has vanishing average regret. It is not a guarantee for EI or for adaptive replication, which we evaluate empirically, including an LCB variant (Section 4.4). Statements and proofs are in Appendix C.

4.2 Implementation Details

Model and infrastructure. We use InternVL3.5 [46] in FP16 as the decision-making agent for both factions. Many current model families are released as vision-language models, and InternVL3.5 provides 2B and 8B checkpoints with the same architecture, tokenizer, and prompt format, allowing agent capacity to be varied without changing anything else. Inputs and outputs are text only; the visual encoder is never invoked and no result depends on visual input, although the interface would permit inputs such as rendered game maps in future work. We combine the 2B and 8B checkpoints into the four faction pairings in Table 2. Game optimizations and evaluations are parallelized across 8 NVIDIA A100 80GB GPUs. Optimization using InternVL3.5-8B for both sides over 100 iterations takes around 40 hours. Several experiments were run on NVIDIA RTX 6000 Ada GPUs. For generalization experiments (Appendix E), we additionally use Qwen3.5-9B as an optimization agent and Gemini 2.5 Flash and Gemini 2.5 Pro as evaluation agents.

Optimization setup. We optimize 12 parameters (Table 1) with a Matérn-$5/2$ GP and EI over 100 iterations, beginning with 10 random points; proposals are projected before evaluation and before the surrogate update. The remaining settings are conventional rather than tuned. Temperature 0.7 and top-p 0.9 are standard sampling values under which play varies across games while staying coherent. A fixed query retrieves three rule entries, keeping the prompt short. The 16-turn cap only bounds worst-case cost, since most games end earlier (Appendix I). Sixteen games per candidate correspond to two evaluation waves on 8 GPUs, and 64 caps the cost of a single candidate. Parameter bounds are coarse brackets (Table 1; for example, Soldier HP 4–16 around the released default of 8), not tuned intervals. Our budget did not permit ablating each of these choices; we prioritized the ablations most likely to change conclusions (model scale, map size, agent type, and the search component). Appendix J lists the settings in full.

Evaluation protocol. During optimization, for each candidate θ, we run $N_t \in [16, 64]$ independent self-play games. Games terminate when a city is destroyed (instant victory) or the turn limit is reached (score-based resolution). The balance loss $\mathcal{L}(\theta)$ is given in Eq. (3). The main experiments evaluate selected checkpoints on 100 held-out games. The additional statistical study evaluates three independent optimization runs on separate 200- and 300-game batches and the optimized configuration of Table 10 on 500 games. These held-out evaluations are separate from the candidate evaluations used by the optimizer: in-loop evaluations are intentionally noisy, which the adaptive game count manages, and all reported numbers come from held-out evaluations. The $50\% \pm 5\%$ criterion refers to observed win rates, not a confidence-certified population bound.

4.3 Main Empirical Results

Table 2 shows win rates across optimization and evaluation pairings. Diagonal entries meet the empirical near-balance criterion. Off-diagonal, upgrading one side shifts outcomes in its favor; the largest shift occurs when reversing the optimization asymmetry (optimizing with E_{8B} vs N_{2B} and evaluating with E_{2B} vs N_{8B}). Both unoptimized configurations we evaluated are decisively imbalanced: the authors' initial configuration (Appendix D.3) yields an Empire win rate of 100%, and the default configuration of the released code yields 11.3% for Empire over 300 games (Table 3). The optimized configuration in Table 10 yields 51%|49% over 100 games and 47.6%|52.4% over 500 games.

The tested configurations also show empirical transfer across LLM families. Table 4 shows that configurations optimized with InternVL3.5-8B or Qwen3.5-9B remain near-balanced when evaluated by the other family (balance score ≤ 0.04 across all four matchups). Further generalization results, including Gemini 2.5 Flash/Pro and human players, are in Appendix E.

4.4 Repeatability and Statistical Uncertainty

Three independent GP-based optimization runs vary optimization initialization and self-play randomness. At 200 games per final configuration, the mean empirical loss is 0.0267 (sample SD 0.0058); at 300 games, it is 0.0333 (SD 0.0200). All six point estimates fall within the empirical near-balance criterion. Table 3 gives the 300-game evaluations and a 500-game re-evaluation of the original configuration, with Wilson 95% intervals and two-sided exact binomial tests. No configuration from the GP (EI), NTBEA, or TPE runs shows a detectable departure from equal win probability at these sample sizes. This supports repeatability in the tested setting, although failing to reject $p_E = 0.5$ is not an equivalence test establishing $p_E \in [0.45, 0.55]$.

Table 2: **Optimized winning chances of Empire and Nomads.** Rows specify the agent pairing used to optimize rules; columns specify the evaluation pairing. Each cell reports *Empire wins* | *Nomads wins* over 100 held-out games using the best optimization checkpoint (empirical loss ≤ 0.1). *E*/*N* denote the factions; subscripts denote model sizes. Greener cells indicate more even outcomes; empirical near-balance ($50\% \pm 5\%$) is bolded.

Optimize \ Evaluate	E_{2B} **vs** N_{2B}	E_{2B} **vs** N_{8B}	E_{8B} **vs** N_{2B}	E_{8B} **vs** N_{8B}
E_{2B} **vs** N_{2B}	**48**|**52**	32|68	73|27	**55**|**45**
E_{2B} **vs** N_{8B}	81|19	**47**|**53**	91|9	75|25
E_{8B} **vs** N_{2B}	37|63	6|94	**52**|**48**	29|71
E_{8B} **vs** N_{8B}	**53**|**47**	24|76	81|19	**51**|**49**

Table 3: **Additional held-out evaluations.** Empire win rates, Wilson 95% intervals, and two-sided exact binomial tests against 0.5. Counts are drawn from the evaluation logs; the test compares Empire and Nomads wins among decisive games (Appendix H). The default batch is separate from the authors' initial configuration. NTBEA/TPE replace the search component; GP-LCB replaces the acquisition rule.

Configuration	Games	Empire (%)	95% CI (%)	p
Default (additional evaluation)	300	11.3	[8.2, 15.4]	2.9×10^{-45}
Optimized configuration (Table 10)	500	47.6	[43.3, 52.0]	0.30
GP run 1	300	51.7	[46.0, 57.3]	0.60
GP run 2	300	49.3	[43.7, 55.0]	0.86
GP run 3	300	52.7	[47.0, 58.2]	0.39
NTBEA	300	48.0	[42.4, 53.6]	0.53
TPE	300	54.3	[48.7, 59.9]	0.15
GP-LCB (single run)	300	57.7	[52.0, 63.1]	0.009

Acquisition rule. We also ran the loop once with the GP lower confidence bound (LCB, with scikit-optimize's default $\kappa = 1.96$) in place of EI, leaving everything else unchanged. The run drives the in-loop loss to zero, but its selected configuration is not near-balanced on held-out games: Empire wins 57.5% of 200 games (Wilson 95% CI [50.6, 64.1]) and 57.7% of 300 games ([52.0, 63.1]). With a single run and a configuration selected from small in-loop batches, this does not rank the two rules; we keep EI because its value also sets the adaptive game count (Section 3.3).

At 100 games near 50%, a 95% interval spans roughly ± 10 percentage points; the other ablations remain single-run studies. Full 200-/300-game results and statistical definitions are in Appendix H.

Qualitative behavior. Figure 3 illustrates a Nomads conquest through mobility and sustained city pressure; further rollouts show both factions winning through different tactics (Appendix G.2).

Table 4: **Cross-LLM-family transfer.** Empirical win rates meet the near-balance criterion ($50\% \pm 5\%$) when parameters optimized with one LLM family are evaluated by the other. All results use 100 evaluation games.

Optimization LLM	Evaluation LLM	Empire%	Nomads%	$\mathcal{L}$
InternVL3.5-8B	InternVL3.5-8B	51	49	0.02
InternVL3.5-8B	Qwen3.5-9B	50	50	0.00
Qwen3.5-9B	InternVL3.5-8B	49	51	0.02
Qwen3.5-9B	Qwen3.5-9B	52	48	0.04

4.5 Ablation Studies

Different optimization methods. Table 5 compares adaptive BO with Random Search, a (1+1) Evolution Strategy (ES), and fixed-budget GP-BO. Random Search samples uniformly from the discrete rule space; ES perturbs a single incumbent with isotropic Gaussian mutations and accepts improvements. Both use $N = 64$ games per candidate, while fixed GP-BO uses $N = 16$, 32, or 64. All methods run for 100 iterations and receive a 100-game held-out evaluation with 8B agents on both sides. Adaptive BO and fixed GP-BO with $N = 64$ reach empirical near-balance, using about 3500 and 6400 optimization games, respectively. The other variants do not meet this criterion. These single-run comparisons suggest a benefit from adaptive replication in this setup, though they do not establish a general optimizer ranking.

Table 5: **Original search and replication ablation.** Each method runs for 100 iterations and is evaluated on 100 held-out games. Random Search and (1+1)-ES use 64 games per candidate; GP-BO uses either fixed replication or 16–64 games based on EI. These are single optimization runs.

	Random Search	(1+1)-ES	BO (adaptive)
Win Rate	13|87	26|74	**51|49**
	BO ($N = 16$)	**BO** ($N = 32$)	**BO** ($N = 64$)
Win Rate	34|66	61|39	**48|52**

Stronger search components. We additionally evaluate the N-Tuple Bandit Evolutionary Algorithm (NTBEA) [18] and the Tree-structured Parzen Estimator (TPE) [47] within the same engine, rulebook, and LLM self-play interface. At 200/300 evaluation games, Empire win rates are 49.0%/48.0% for NTBEA and 51.5%/54.3% for TPE, compared to means of 50.7%/51.2% across the three GP runs, which are comparable within uncertainty (Tables 3 and 19). All three reach comparable balance with the engine and self-play objective unchanged, showing that RuleSmith works with different surrogate-based search components; because they converge to similar balance on CivMini, this testbed does not rank them. The fidelity mechanism behind BOHB and Hyperband corresponds here to running more self-play games per candidate, which adaptive sampling already provides.

Proxy player quality. LLM agents outperform random legal-action agents and provide meaningful competition to scripted agents, although such outcomes do not by themselves establish human-like or optimal play. Execution-level conflicts account for 3.3% of attempted unit actions for 8B and

4.6% for 2B agents; the engine converts these to PASS. An additional 300-game analysis finds 229 distinct three-turn openings and a mean turn-5 board-occupancy Jaccard distance of 0.62, supporting trajectory diversity under stochastic decoding (Table 6).

Table 6: **Trajectory diversity** under identical rules (8B self-play, NTBEA configuration; Appendix I). The 100-game column is a subset of the 300-game column, not an independent replication.

Metric	First 100 games	All 300 games
Distinct three-turn opening sequences	86/100	229/300
Distinct first-turn action combinations	6	11
Action entropy, Empire / Nomads (bits)	2.03 / 1.65	2.04 / 1.64
Mean turn-5 Jaccard distance	0.62	0.62
Ended before the turn cap (a city destroyed)	63%	62%
Game length, mean (SD) in turns	10.8 (4.4)	11.1 (4.4)

Three participants played 120 games against the agents, with randomized faction and opponent scale and blinded scale assignment. Humans won 55/60 against 2B and 48/60 against 8B. Yet Empire won 45.0% and 43.3% of these games, and 53.3% in the separate 60-game human-versus-human study; none of the three departs detectably from an even split (all $p \geq 0.37$). The configuration optimized by LLM self-play is consistent with balance in these populations; given the few, repeated participants, this provides preliminary evidence (Appendices E.4 and E.5).

4.6 Generalization and Scalability

Scalability and extended configurations (Appendix F). In the tested settings, RuleSmith finds near-even empirical outcomes across map sizes from 5×5 to 11×11 (Table 7; Appendix F.1). It also scales to three factions (Empire, Nomads, Theocracy), balancing all three simultaneously (balance loss 0.013–0.033). Grid size works as a discrete optimization variable; results stay near-balanced.

Table 7: **Extended configurations.** Empire|Nomads win rates (with Theocracy third for three factions) and balance loss for four map sizes with InternVL3.5-8B (turn limits in parentheses), three-faction balancing with four added Theocracy parameters ($\mathcal{L}_3$), and map size (5–9) as an additional optimizable parameter (Appendix F). Map-size losses follow Eq. (3) from the reported win rates.

	Map size (turn limit)				**Three factions**		**Map size optimized**	
	5×5 (16)	7×7 (16)	9×9 (32)	11×11 (32)	8B	2B	8B	2B
Win rate (%)	53|47	51|49	48|52	51|49	32|33|35	34|33|33	53|47	51|49
Balance loss	0.06	0.02	0.04	0.02	0.033	0.013	0.06	0.02

Parameter analysis (Appendix G). Optimized parameters vary with agent pairing. Symmetric pairings yield similar TTK values in both directions; asymmetric pairings show coordinated changes in combat, economy, cost, and scoring, consistent with compensating for capacity differences. Distinct pairings yield different configurations with similar empirical balance scores (Table 17).

5 Conclusion, Limitations, and Impact Statement

RuleSmith automates asymmetric game balancing through LLM self-play and Bayesian optimization over interpretable rule parameters. EI-based adaptive replication focuses evaluation on promising candidates. Three independent GP runs and larger held-out evaluations strengthen the balance results, while NTBEA/TPE demonstrate compatibility with other search components. Cross-model and small human studies suggest transfer across the tested populations. Balance remains population-dependent, and since CivMini is a controlled proof of concept, scaling to larger rule spaces, partial observability, and population shifts remains open (Appendices A and B).

Acknowledgments and Disclosure of Funding

The authors declare no competing interests.

References

[1] Zelai Xu, Chao Yu, Fei Fang, Yu Wang, and Yi Wu. Language agents with reinforcement learning for strategic play in the werewolf game. *arXiv preprint arXiv:2310.18940*, 2023.

[2] Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. Exploring large language models for communication games: An empirical study on werewolf. *arXiv preprint arXiv:2309.04658*, 2023.

[3] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023.

[4] Yiwei Huang, Shiwei Zhao, Runze Wu, Jiale Wei, Zhipeng Hu, Ping Wang, Tangjie Lv, Le Li, and Changjie Fan. AI as game designer: Achieving optimal level balance through LLM-based system. In *Behavioural and Social Computing: BESC 2025, Proceedings, Part III*, volume 16433 of *Lecture Notes in Computer Science*, pages 173–182. Springer, 2026. doi: 10.1007/978-981-95-7144-4_15.

[5] Fernando de Mesentier Silva, Scott Lee, Julian Togelius, and Andy Nealen. AI-based playtesting of contemporary board games. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, pages 1–10, 2017.

[6] Yunqi Zhao, Igor Borovikov, Fernando de Mesentier Silva, Ahmad Beirami, Jason Rupert, Caedmon Somers, Jesse Harder, John Kolen, Jervis Pinto, Reza Pourabolghasem, et al. Winning is not everything: Enhancing game development with intelligent agents. *IEEE Transactions on Games*, 12(2):199–212, 2020.

[7] Joakim Bergdahl, Camilo Gordillo, Konrad Tollmar, and Linus Gisslén. Augmenting automated game testing with deep reinforcement learning. In *2020 IEEE Conference on Games (CoG)*, pages 600–603. IEEE, 2020.

[8] Carlos Celemin. Bayesian optimization-based search for agent control in automated game testing. In *2024 IEEE Conference on Games (CoG)*, pages 1–4. IEEE, 2024.

[9] Kirthevasan Kandasamy, Gautam Dasarathy, Junier B Oliva, Jeff Schneider, and Barnabás Póczos. Gaussian process bandit optimisation with multi-fidelity evaluations. *Advances in neural information processing systems*, 29, 2016.

[10] Noor Shaker, Julian Togelius, and Mark J Nelson. Procedural content generation in games. In *Procedural content generation in games*, Computational Synthesis and Creative Systems. Springer, 2016.

[11] Mark Hendrikx, Sebastiaan Meijer, Joeri Van Der Velden, and Alexandru Iosup. Procedural content generation for games: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 9(1):1–22, 2013.

[12] Julian Togelius, Alex J Champandard, Pier Luca Lanzi, Michael Mateas, Ana Paiva, Mike Preuss, and Kenneth O Stanley. Procedural content generation: Goals, challenges and actionable steps, 2013.

[13] Adam Summerville, Sam Snodgrass, Matthew Guzdial, Christoffer Holmgård, Amy K Hoover, Aaron Isaksen, Andy Nealen, and Julian Togelius. Procedural content generation via machine learning (pcgml). *IEEE Transactions on Games*, 10(3):257–270, 2018.

[14] Isaac Karth and Adam M Smith. Addressing the fundamental tension of pcgml with discriminative learning. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, pages 1–9, 2019.

[15] Sam Snodgrass and Santiago Ontanón. Player movement models for platformer game level generation. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 757–763, 2017.

[16] Sam Snodgrass, Adam Summerville, and Santiago Ontañón. Studying the effects of training data on machine learning-based procedural content generation. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 13, pages 122–128, 2017.

[17] Kamolwan Kunanusont, Raluca D. Gaina, Jialin Liu, Diego Perez-Liebana, and Simon M. Lucas. The N-tuple bandit evolutionary algorithm for automatic game improvement, 2017. URL https://arxiv.org/abs/1705.01080.

[18] Simon M. Lucas, Jialin Liu, and Diego Pérez-Liébana. The N-tuple bandit evolutionary algorithm for game agent optimisation. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–9, 2018. doi: 10.1109/CEC.2018.8477869.

[19] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[20] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[21] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[22] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[23] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[24] Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. In *International conference on machine learning*, pages 793–802. PMLR, 2019.

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

[26] Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*, 2016.

[27] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

[28] Jakob N Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. *arXiv preprint arXiv:1709.04326*, 2017.

[29] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

[30] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024.

[31] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101, 2025.

[32] Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24, 2024.

[33] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

[34] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face. *Advances in Neural Information Processing Systems*, 36:38154–38180, 2023.

[35] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008, 2023.

[36] Mingchen Zhuge, Haozhe Liu, Francesco Faccio, Dylan R Ashley, Róbert Csordás, Anand Gopalakrishnan, Abdullah Hamdi, Hasan Abed Al Kader Hammoud, Vincent Herrmann, Kazuki Irie, et al. Mindstorms in natural language-based societies of mind. *arXiv preprint arXiv:2305.17066*, 2023.

[37] Shaoguang Mao, Yuzhe Cai, Yan Xia, Wenshan Wu, Xun Wang, Fengyi Wang, Tao Ge, and Furu Wei. ALYMPICS: LLM agents meet game theory – exploring strategic decision-making with AI agents. *arXiv preprint arXiv:2311.03220*, 2023.

[38] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*, 2024.

[39] Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 17889–17904, 2024.

[40] Samuel Arnesen, David Rein, and Julian Michael. Training language models to win debates with self-play improves judge accuracy. *arXiv preprint arXiv:2409.16636*, 2024.

[41] Sid Meier and MicroProse. Civilization. Video game, 1991. MicroProse Software, Inc.

[42] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

[43] Tim Head, Manoj Kumar, Holger Nahrstaedt, Gilles Louppe, and Iaroslav Shcherbatyi. scikit-optimize: Sequential model-based optimization in Python. https://github.com/scikit-optimize/scikit-optimize, 2018.

[44] Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task bayesian optimization. *Advances in neural information processing systems*, 26, 2013.

[45] Kirthevasan Kandasamy, Gautam Dasarathy, Jeff Schneider, and Barnabás Póczos. Multi-fidelity bayesian optimisation with continuous approximations. In *International conference on machine learning*, pages 1799–1808. PMLR, 2017.

[46] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. InternVL3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

[47] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, volume 24, pages 2546–2554, 2011. URL https://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.

[48] Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1015–1022, 2010.

[49] Adam D. Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12:2879–2904, 2011.

[50] Ruyi Liu, Joshua L. Warren, Yuki Ohnishi, Donna Spiegelman, Liangyuan Hu, and Fan Li. Calibrated Bayes analysis of cluster-randomized trials. *arXiv preprint arXiv:2511.20833*, 2025.

[51] Ruyi Liu, Yuki Ohnishi, Xi Fang, Michael J. Daniels, and Fan Li. Calibrated Bayesian inference for overall and subgroup-specific causal effects in stepped-wedge cluster-randomized trials. *arXiv preprint arXiv:2610.20858*, 2026.

[52] Melody Owen, Fan Li, Ruyi Liu, and Donna Spiegelman. A comparison of methods for designing hybrid type 2 cluster-randomized trials with continuous effectiveness and implementation endpoints. *arXiv preprint arXiv:2510.20741*, 2025.

[53] Ruyi Liu, Fan Li, Denise Esserman, and Mary M. Ryan. Group sequential two-stage preference designs. *Statistics in Medicine*, 2023. doi: 10.1002/sim.9962.

[54] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631, 2019.

A Limitations

LLM-agent-centric balance criteria. RuleSmith optimizes for balance as measured by LLM self-play win rates. Cross-LLM evaluations and the small human study provide encouraging evidence of transfer to the tested populations (Appendix E). LLM win rates of 54–78% against rule-based opponents establish an intermediate opponent baseline while also showing sensitivity to player pairing (Appendix D.2). Within these studies, faction outcomes show no detectable imbalance: human players defeat the agents in most games, yet faction outcomes stay statistically even against both agent scales and between humans (Appendix E.4). Formal guarantees of transfer to human gameplay remain absent, and these studies are limited in scale. Systematic human evaluation and player modeling would strengthen the transfer evidence.

Action validity, strategy bias, and skill. Validation ensures executable games but not good strategies. The measured 3.3%/4.6% execution-rejection rates reflect within-turn conflicts rather than a complete measure of hallucination or parse failure. Both factions are played by the same model, so population-level biases, such as a general preference for aggression, affect both sides and largely cancel in the win-rate disparity; what remains is that balance is defined relative to the evaluated population. The released prompts include strategy guides, so the evaluation population is defined jointly by the model, decoding, and prompts. Humans substantially outperform these agents; how large the strength gap between optimization and evaluation players can become before the optimized rules lose balance remains an open question for future work.

Statistical and reproducibility scope. The three-seed experiment covers the main GP setting. The other ablations and the additional NTBEA, TPE, and LCB searches are single runs. Selecting the final configuration by its lowest in-loop loss can favor a lucky small-batch evaluation; the held-out re-evaluations guard against this, as the single LCB run illustrates. Binomial uncertainty and failure to reject equality do not establish equivalence within five percentage points. Human games share a few participants and should not be treated as 60 independent draws from a broad human population. Seeds and optimizer settings for the additional runs are listed in Table 22.

Designer-specified rule space. The engine, the choice of tunable versus fixed rules, parameter bounds, and the objective require designer input. A balanced solution must be accessible within those bounds. Several reported settings lie at boundaries; widening ranges and re-optimizing is a practical extension, while automated range discovery remains future work.

Simplified game environment. CivMini is a deliberately simplified testbed with a fully observable state, small action spaces, and a compact parameter set. Real commercial strategy games involve thousands of interacting parameters, imperfect information, long-horizon dependencies, and emergent player behaviors that our current framework does not address. Extending RuleSmith to richer, partially observable, or multi-stage environments is a key direction for future work.

No formal optimality guarantees. Bayesian optimization with a Gaussian process surrogate provides no guarantee of finding the globally optimal balanced configuration. The surrogate model may misfit in high-curvature regions, and the discrete projection step introduces approximation errors. Robustness under distribution shifts (such as changes in the player population or game meta) is not explicitly modeled. Incorporating robustness objectives or adversarial evaluation into the optimization loop is a promising avenue for future research.

B Broader Impact

This work advances automated design tools and multi-agent simulation by enabling LLM-based evaluation and optimization of rule-driven systems. By treating large language models as flexible evaluators rather than deployed decision-makers, RuleSmith offers a scalable and interpretable approach to analyzing complex, parameterized environments.

Potential applications of this framework extend beyond games to domains such as economic modeling, policy analysis, and resource allocation, where outcomes depend on structured rules and multi-actor interactions. At the same time, we recognize that automated simulation and optimization of

competitive or adversarial systems may raise ethical considerations if misapplied. In particular, such tools could be misused to over-optimize asymmetric rules in ways that disadvantage certain participants or reduce transparency in decision-making processes.

To mitigate these risks, RuleSmith is designed as an offline analysis and design-time tool operating on abstract, simplified environments, rather than a system for real-time decision execution. We emphasize that any real-world deployment of rule-optimization frameworks should incorporate human oversight, domain-specific constraints, and ethical review. Overall, we believe the primary impact of this work lies in supporting safer, more transparent, and more systematic design of rule-based systems, while encouraging responsible use in sensitive application domains.

C Theoretical Analysis

This section provides an idealized analysis of two assumptions behind RuleSmith: when exact balance is attainable, and why Gaussian-process Bayesian optimization (GP-BO) is a suitable search procedure for the balance loss. These results should be read as theoretical context rather than as a finite-budget optimality guarantee for the full RuleSmith implementation, which uses Expected Improvement, adaptive evaluation budgets, and a discrete projection step.

Proposition C.1 (Balanced Optimum under Role Symmetry)**.** *Consider a two-faction game parameterized by $\theta \in \Theta$. Suppose Θ contains a role-exchange-invariant configuration θ_{sym}, and suppose the proxy-player policy distribution is invariant under swapping the two faction roles. Then*

$$w_E(\theta_{\mathrm{sym}}) = w_N(\theta_{\mathrm{sym}}). \tag{7}$$

Consequently, for the balance loss in Eq. (3)*,*

$$\mathcal{L}(\theta_{\mathrm{sym}}) = 1.5\, w_D(\theta_{\mathrm{sym}}). \tag{8}$$

In particular, if $w_D(\theta_{\mathrm{sym}}) = 0$, then

$$\mathcal{L}(\theta_{\mathrm{sym}}) = 0, \tag{9}$$

and the optimal balance loss obeys

$$\min_{\theta \in \Theta} \mathcal{L}(\theta) = 0. \tag{10}$$

Proof. At a role-exchange-invariant configuration, swapping the faction labels leaves the game distribution unchanged. By the policy-symmetry assumption, the induced distribution over trajectories is also unchanged after the swap, except that Empire and Nomads exchange labels. Therefore, the probability of an Empire win before the swap equals the probability of a Nomads win after the swap, giving

$$w_E(\theta_{\mathrm{sym}}) = w_N(\theta_{\mathrm{sym}}).$$

Since $w_E = w_N = (1 - w_D)/2$ at this configuration, substituting into Eq. (3) yields

$$\mathcal{L}(\theta_{\mathrm{sym}}) = |w_E(\theta_{\mathrm{sym}}) - 0.5| + |w_N(\theta_{\mathrm{sym}}) - 0.5| + 0.5\, w_D(\theta_{\mathrm{sym}}) = 1.5\, w_D(\theta_{\mathrm{sym}}).$$

Thus, if the symmetric configuration has no draw probability, $w_D(\theta_{\mathrm{sym}}) = 0$, then $\mathcal{L}(\theta_{\mathrm{sym}}) = 0$. Since the balance loss is nonnegative by definition, this implies $\min_{\theta \in \Theta} \mathcal{L}(\theta) = 0$. □

Remark. This proposition does not claim that every asymmetric ruleset can be made perfectly balanced. Role symmetry guarantees equal faction win probabilities, while exact zero loss for the draw-penalized objective additionally requires zero draw probability. RuleSmith targets the harder practical case where the searchable parameter subspace is asymmetric, discrete, and evaluated through noisy LLM self-play.

Lemma C.2 (Concentration of the empirical projected balance loss)**.** *Let $\widetilde{\Theta}$ denote the continuous relaxation used by the optimizer, let $D : \widetilde{\Theta} \to \Theta_{\mathrm{disc}}$ be the deterministic discretization operator, and define the projected balance objective*

$$F(\tilde{\theta}) := \mathcal{L}(D(\tilde{\theta})).$$

At round t, let $\tilde{\theta}_t \in \widetilde{\Theta}$ be the continuous proposal selected by the optimizer, and let

$$\theta_t = D(\tilde{\theta}_t) \in \Theta_{\mathrm{disc}}$$

be the corresponding discretized rule configuration evaluated by the game engine. Suppose N_t is chosen using only information available before observing the round-t self-play outcomes. From N_t independent rollouts under θ_t, define

$$\hat{w}_{E,t}, \hat{w}_{N,t}, \hat{w}_{D,t}$$

as the empirical Empire-win, Nomads-win, and draw rates, respectively, and let

$$y_t = |\hat{w}_{E,t} - 0.5| + |\hat{w}_{N,t} - 0.5| + 0.5\hat{w}_{D,t}$$

be the empirical balance loss observed by the optimizer. Then, for any fixed horizon T and failure probability $\delta \in (0,1)$, with probability at least $1-\delta$, simultaneously for all $t = 1, \dots, T$,

$$|y_t - F(\tilde{\theta}_t)| = |y_t - \mathcal{L}(D(\tilde{\theta}_t))| \le \eta_t, \qquad \eta_t = \frac{5}{2}\sqrt{\frac{\log(6T/\delta)}{2N_t}}.$$

Proof. Condition on the history before round t. Since $\tilde{\theta}_t$ and N_t are chosen before observing the round-t outcomes, the projected ruleset $\theta_t = D(\tilde{\theta}_t)$ is also fixed before these outcomes are observed. Therefore, the N_t rollouts are independent samples from the outcome distribution induced by the evaluated ruleset θ_t.

For each outcome type $j \in \{E, N, D\}$, Hoeffding's inequality gives

$$|\hat{w}_{j,t} - w_j(\theta_t)| \le \sqrt{\frac{\log(6T/\delta)}{2N_t}}$$

simultaneously for all $j \in \{E, N, D\}$ and all $t = 1, \dots, T$ with probability at least $1-\delta$, by a union bound.

On this event, the inequality $||a-c| - |b-c|| \le |a-b|$ gives

$$\begin{aligned} |y_t - F(\tilde{\theta}_t)| &\le |\hat{w}_{E,t} - w_E(\theta_t)| + |\hat{w}_{N,t} - w_N(\theta_t)| + 0.5|\hat{w}_{D,t} - w_D(\theta_t)| \\ &\le \frac{5}{2}\sqrt{\frac{\log(6T/\delta)}{2N_t}}. \end{aligned}$$

This proves the claim. □

Proposition C.3 (Idealized GP-LCB for projected self-play evaluations)**.** *Let $\widetilde{\Theta}$ denote the continuous relaxation used by the optimizer, let $D : \widetilde{\Theta} \to \Theta_{\text{disc}}$ be the deterministic discretization operator, and define the projected objective*

$$F(\tilde{\theta}) := \mathcal{L}(D(\tilde{\theta})).$$

Let

$$\tilde{\theta}^\star \in \arg\min_{\tilde{\theta} \in \widetilde{\Theta}} F(\tilde{\theta})$$

be an optimal continuous representative of a best projected ruleset.

At round t, an idealized optimizer selects a continuous proposal $\tilde{\theta}_t \in \widetilde{\Theta}$, evaluates the projected ruleset $D(\tilde{\theta}_t)$ using N_t independent self-play rollouts, and observes the empirical projected loss y_t defined in Lemma C.2. For this idealized projected-objective analysis, the surrogate is indexed by the pair $(\tilde{\theta}_t, y_t)$; the implementation stores evaluated discrete coordinates instead. This distinction is part of why this proposition is not a guarantee for the implemented pipeline.

Assume a bounded kernel diagonal $k(x,x) \le 1$, nondecreasing β_t, and a GP observation model with variances bounded above and bounded away from zero (the finite replication range permits such bounds). Assume that the GP surrogate is calibrated for the projected objective F in the standard GP bandit sense: there exist confidence parameters $\{\beta_t\}_{t \ge 1}$ such that, with probability at least $1-\delta$,

$$|F(\tilde{\theta}) - \mu_{t-1}(\tilde{\theta})| \le \sqrt{\beta_t}\sigma_{t-1}(\tilde{\theta})$$

for all $\tilde{\theta} \in \widetilde{\Theta}$ and all $t \ge 1$. Here μ_{t-1} and σ_{t-1} are the GP posterior mean and posterior standard deviation after $t-1$ observations. Lemma C.2 motivates a heteroscedastic observation-uncertainty proxy scaling as $\eta_t^2 = O(1/N_t)$, although the GP posterior remains an idealized modeling approximation.

Consider the idealized GP lower-confidence-bound rule

$$\tilde{\theta}_t \in \arg\min_{\tilde{\theta}\in\widetilde{\Theta}} \left[\mu_{t-1}(\tilde{\theta}) - \sqrt{\beta_t}\sigma_{t-1}(\tilde{\theta})\right].$$

Then, on the confidence event above, the cumulative projected regret

$$R_T = \sum_{t=1}^{T} \left(F(\tilde{\theta}_t) - F(\tilde{\theta}^\star)\right)$$

satisfies

$$R_T = O\left(\sqrt{T\beta_T\gamma_T^{(N)}}\right),$$

where $\gamma_T^{(N)}$ is the maximum information gain after T observations under the corresponding heteroscedastic observation uncertainties. Consequently, whenever $\beta_T\gamma_T^{(N)} = o(T)$, the average projected regret R_T/T vanishes.

Proof. The proof follows the standard GP-UCB/LCB regret argument [48], applied to the minimization of the projected objective F. On the confidence event, for the selected point $\tilde{\theta}_t$,

$$F(\tilde{\theta}_t) \le \mu_{t-1}(\tilde{\theta}_t) + \sqrt{\beta_t}\sigma_{t-1}(\tilde{\theta}_t),$$

while for the optimal projected point $\tilde{\theta}^\star$,

$$\mu_{t-1}(\tilde{\theta}^\star) - \sqrt{\beta_t}\sigma_{t-1}(\tilde{\theta}^\star) \le F(\tilde{\theta}^\star).$$

Since $\tilde{\theta}_t$ minimizes the lower confidence bound,

$$\mu_{t-1}(\tilde{\theta}_t) - \sqrt{\beta_t}\sigma_{t-1}(\tilde{\theta}_t) \le \mu_{t-1}(\tilde{\theta}^\star) - \sqrt{\beta_t}\sigma_{t-1}(\tilde{\theta}^\star).$$

Combining the three inequalities gives

$$F(\tilde{\theta}_t) - F(\tilde{\theta}^\star) \le 2\sqrt{\beta_t}\sigma_{t-1}(\tilde{\theta}_t).$$

Summing over $t = 1, \dots, T$ and applying the Cauchy–Schwarz inequality yields

$$\begin{aligned} R_T &\le 2\sum_{t=1}^{T} \sqrt{\beta_t}\sigma_{t-1}(\tilde{\theta}_t) \\ &\le 2\sqrt{T\beta_T \sum_{t=1}^{T} \sigma_{t-1}^2(\tilde{\theta}_t)}. \end{aligned}$$

The standard information-gain argument for GP bandits bounds the cumulative posterior variance by a constant multiple of $\gamma_T^{(N)}$, yielding

$$R_T = O\left(\sqrt{T\beta_T\gamma_T^{(N)}}\right).$$

Dividing by T gives vanishing average projected regret whenever $\beta_T\gamma_T^{(N)} = o(T)$. □

Remark. Proposition C.3 should be interpreted as an idealized analysis of RuleSmith's continuous-proposal, discrete-evaluation procedure. The optimizer proposes $\tilde{\theta} \in \widetilde{\Theta}$, but the environment evaluates only the projected ruleset $D(\tilde{\theta})$. Consequently, the objective observed by the optimizer is the projected loss

$$F(\tilde{\theta}) = \mathcal{L}(D(\tilde{\theta})).$$

This objective is generally piecewise constant and can be discontinuous at projection-cell boundaries, so the GP calibration assumption is not guaranteed solely by choosing a smooth kernel such as Matérn $5/2$. Rather, it is a standard modeling idealization used to provide theoretical context for GP-based search under noisy self-play evaluations.

The proposition analyzes an LCB rule, whereas the implemented system uses Expected Improvement together with adaptive replication. The result is therefore not a regret guarantee for the implementation. LCB carries the bound because the classical argument [48] requires an explicit confidence-bound exploration term: the objective is sandwiched by the confidence interval, per-step regret is bounded by the posterior standard deviation at the queried point, and the sum is controlled by the maximum information gain. EI has no comparably standard cumulative-regret bound; its classical guarantee is a convergence rate in the noiseless case [49], and results for noisy EI do not cover a discrete projection combined with EI-dependent replication. The two rules nonetheless use the same posterior quantities. Writing $\mathrm{EI} = \sigma\, h(z)$ with $z = (y^* - \mu)/\sigma$ and $h(z) = z\Phi(z) + \phi(z)$, the partial derivatives of EI with respect to the predicted improvement $y^* - \mu$ and σ are $\Phi(z)$ and $\phi(z)$, respectively; because both are positive, EI increases in the exact two quantities that LCB trades off, and the rules differ in how they weight them. Lemma C.2 justifies the concentration of the self-play estimates, Proposition C.3 shows that GP-based search is well posed for this objective under idealized calibration, and the choice among acquisition rules is settled empirically, including the LCB run in Section 4.4.

D Validating the Proxy Player

D.1 LLM Agents vs. Random Agent Baseline

To establish a lower bound on proxy-player quality, Table 8 evaluates InternVL3.5-8B and InternVL3.5-2B against a **random agent** that selects uniformly from all legal actions. The prompted LLM agents win 100% of games in every listed configuration, regardless of faction. This establishes effective play relative to random action selection and supports their use as proxy players. As with the other agent comparisons, performance reflects the model together with its prompts and strategy guidance.

Table 8: **LLM agent vs. random agent.** LLM agents (InternVL3.5-8B and 2B) are evaluated against a uniformly random legal-action agent using parameters optimized by the respective LLM. The prompted LLM agents win 100% of games in every listed configuration.

Empire	**Nomads**	**Empire%**	**Nomads%**
InternVL3.5-8B	Random	100	0
Random	InternVL3.5-8B	0	100
InternVL3.5-2B	Random	100	0
Random	InternVL3.5-2B	0	100

D.2 LLM Agents vs. Rule-Based Agent

To provide a stronger proxy-player baseline than random play, we design a **rule-based agent** that executes fixed greedy heuristics with random tie-breaking, without language understanding.

Empire rule-based agent. Farmers always `GATHER` if possible; otherwise they `PASS`. Soldiers `BATTLE` any adjacent enemy (prioritizing the enemy city), or `MOVE` toward the nearest enemy by Manhattan distance; if multiple enemies are equidistant, one is chosen uniformly at random. The city alternates each turn between `PRODUCE_UNIT` (farmer) and `PRODUCE_UNIT` (soldier) whenever resources suffice and an adjacent cell is free; otherwise it uses `PRODUCE_RESOURCE`.

Nomads rule-based agent. Cavalry `BATTLE` the enemy city if adjacent; otherwise they `BATTLE` the nearest adjacent enemy unit; otherwise they `MOVE` toward the enemy city. The city uses `PRODUCE_UNIT` (cavalry) when resources allow; otherwise it uses `PASS`.

These heuristics capture the core asymmetry of each faction: Empire prioritizes economy and incremental defense, whereas Nomads prioritize fast aggression and city destruction.

Table 9 reports win rates when LLM agents face the rule-based agent under two parameter sets: parameters optimized with LLM self-play (Table 2) and parameters optimized with rule-based self-play (Table 16).

LLM-agent win rates range from 54% to 78% in Table 9, below their 100% rate against random agents. The rule-based agent therefore provides meaningful competition for both model sizes. Under LLM-optimized parameters, 8B agents win 78% as Nomads and 75% as Empire. Under rule-based-optimized parameters, these rates are 62% and 73%, respectively. The latter configuration has cheaper Empire units (4), higher farmer gathering (2), and lower Nomads kill gain (5), compatible with an economy favoring frequent Empire production. These descriptive comparisons show interactions between rules and player policies; they do not isolate the contribution of individual mechanics. Together, the random and scripted baselines support the use of prompted LLM agents as a more capable evaluation population in these settings.

Table 9: **LLM agent vs. rule-based agent.** Win rates when LLM agents (InternVL3.5-8B and 2B) play against the rule-based agent under parameters optimized via LLM self-play (left) and rule-based self-play (right). All results use 100 evaluation games.

		LLM-optimized params		**Rule-optimized params**	
Empire	**Nomads**	**Empire%**	**Nomads%**	**Empire%**	**Nomads%**
InternVL3.5-8B	Rule-based	75	25	73	27
Rule-based	InternVL3.5-8B	22	78	38	62
InternVL3.5-2B	Rule-based	68	32	65	35
Rule-based	InternVL3.5-2B	31	69	46	54

D.3 Comparison with Human-Designed Parameters

Table 10 compares the authors' initial configuration ("Initial θ") against the configuration discovered by RuleSmith using InternVL3.5-8B. This starting point produces a completely one-sided outcome in its original evaluation (Empire 100% | Nomads 0%, loss 1.00). Its low Nomads damage (1) relative to Empire damage (4) and Soldier HP (13) plausibly contributes to this advantage. This interpretation is descriptive; joint parameter changes do not isolate a causal effect for any one coordinate.

RuleSmith's Bayesian optimizer jointly adjusts all parameter categories to recover empirical balance. Nomads' damage triples ($1 \rightarrow 3$), their production cost rises ($3 \rightarrow 8$), initial resources increase ($6 \rightarrow 10$), and Empire Soldier HP falls ($13 \rightarrow 9$). These changes are consistent with stronger Nomads combat, costlier cavalry production, and greater initial spending capacity. The optimized combat statistics imply three attacks to eliminate either faction's main combat unit. The empirical balance loss drops from 1.00 to 0.02; these mechanistic interpretations concern the joint configuration rather than isolated parameter effects.

Table 10: **Human-designed vs. RuleSmith-optimized parameters.** The authors' initial configuration is heavily skewed toward Empire (balance score 1.00). RuleSmith reduces the balance score to 0.02 through coordinated adjustments across all parameter categories.

Parameter	**Initial**	**Opt.**	**Parameter**	**Initial**	**Opt.**
Economy			*Production*		
Initial resources	6	10	Empire unit cost	7	9
Empire farmer gather	2	1	Nomads unit cost	3	8
Nomads kill resource gain	4	7			
Combat			*Scoring*		
Empire damage	4	4	Score per resource	0.1	0.1
Nomads damage	1	3	Score per battle won	4	3
Empire soldier HP	13	9	Score per surviving unit	4	5
Nomads cavalry HP	10	12			
Empire win rate: 100% $\rightarrow$ 51%			**Nomads win rate**: 0% $\rightarrow$ 49%		
Balance score $\mathcal{L}$: 1.00 $\rightarrow$ 0.02					

E Generalization of Balanced Configurations

A central question for RuleSmith is whether balanced game parameters reflect the structural properties of the game itself or are merely artifacts of a specific LLM's strategy distribution. We present five complementary experiments to answer this question.

E.1 Cross-LLM-Family Transfer

We optimize game parameters using either InternVL3.5-8B or Qwen3.5-9B as the proxy player and then evaluate them with both models. As reported in Table 4 (main paper), all four combinations remain near-balanced (balance score ≤ 0.04), providing evidence of transfer across these LLM families at the tested sample size.

E.2 Evaluation with Gemini Flash and Pro

Table 11 evaluates configurations with **Gemini 2.5 Flash** and **Gemini 2.5 Pro**. Parameters optimized with InternVL3.5-8B or Qwen3.5-9B meet the empirical near-balance criterion under both evaluators. This provides evidence of cross-model transfer at the tested sample size; comparing distinct models does not isolate the effect of a reasoning mode.

Table 11: **Evaluation with Gemini Flash and Pro.** Parameters optimized with InternVL3.5-8B or Qwen3.5-9B meet the empirical near-balance criterion when evaluated with Gemini 2.5 Flash or Pro. The comparison varies model identity and does not isolate a reasoning mode.

Optimization LLM	Evaluation LLM	Empire%	Nomads%	$\mathcal{L}$
InternVL3.5-8B	Gemini 2.5 Flash	51	49	0.02
InternVL3.5-8B	Gemini 2.5 Pro	47	53	0.06
Qwen3.5-9B	Gemini 2.5 Flash	45	55	0.10
Qwen3.5-9B	Gemini 2.5 Pro	53	47	0.06

E.3 Asymmetric Cross-Model Matchups

Table 12 reports *asymmetric* matchups using parameters optimized with symmetric InternVL3.5-8B self-play, placing a different LLM on each side. With InternVL3.5-8B playing Empire, Gemini 2.5 Pro as Nomads wins 59% versus 55% for Flash; these single evaluations do not establish a statistically reliable ordering. InternVL3.5-8B obtains 41–55% win rates across the listed matchups, depending on its faction and opponent. These results show that outcomes remain sensitive to the player pairing, so transfer is a property to measure rather than assume.

Table 12: **Asymmetric cross-model matchups.** Parameters optimized with symmetric InternVL3.5-8B self-play are evaluated with different models on the two sides. Faction outcomes vary with the pairing, while both factions retain wins in the tested matchups.

Empire	Nomads	Empire%	Nomads%	$\mathcal{L}$
InternVL3.5-8B	Gemini 2.5 Flash	45	55	0.10
InternVL3.5-8B	Gemini 2.5 Pro	41	59	0.18
Gemini 2.5 Flash	InternVL3.5-8B	45	55	0.10
Gemini 2.5 Pro	InternVL3.5-8B	49	51	0.02

E.4 Human User Study

To evaluate whether the game parameters balanced via LLM self-play also produce fair outcomes for human players, we conducted a small-scale user study using the parameters optimized by RuleSmith with InternVL3.5-8B.

Setup. Four of the authors (graduate students with prior experience in turn-based strategy games) took part. We arranged all $\binom{4}{2} = 6$ unique pairwise matchups; each pair played with both faction assignments (Empire vs. Nomads, then Nomads vs. Empire), yielding 12 game sessions in total. Each session consisted of 5 games, for a total of **60 games**. Participants received the same natural-language rulebook provided to the LLM agents, with no additional strategic guidance. We recorded only aggregate win/loss outcomes and did not collect personal or sensitive data.

Results. Table 13 summarizes the outcomes. Across 60 games, Empire won 32 times and Nomads won 28 times (53.3% vs. 46.7%), with no draws. Both individual win rates (53.3% and 46.7%) fall within the $50\% \pm 5\%$ near-balanced range used as our criterion throughout this work.

Table 13: **Human user study results.** 4 participants, 6 pairwise matchups, 2 faction assignments each, 5 games per session, 60 games total. Empire win rate 53.3% and Nomads win rate 46.7% both fall within the near-balanced range ($50\% \pm 5\%$).

	Empire	**Nomads**	**Total**
Wins	32	28	60
Win Rate	53.3%	46.7%	N/A

These results provide preliminary evidence of transfer to these human participants; the 60 games do not establish equivalence within five percentage points or population-wide human balance. We acknowledge that this study is small in scale and that a larger, pre-registered human study would provide stronger evidence; we leave this to future work.

E.5 Human versus LLM Agents

Three of the authors each played 40 games under the configuration optimized with InternVL3.5-8B, giving 120 games in total. Each participant played ten games in each of the four combinations of opponent scale (2B/8B) and human faction (Empire/Nomads) in a randomized order, without knowing which opponent was faced. Randomizing the order spreads participant learning evenly across the conditions, and blinding removes expectation effects from knowing the opponent's size. Per-participant outcomes are reported in Table 14.

Table 14: Human wins by participant and faction against InternVL3.5 agents. Each participant/faction cell has 10 games; totals pool the three participants.

Participant	Human faction	vs. 2B	vs. 8B
P1	Empire	9/10	8/10
P1	Nomads	10/10	8/10
P2	Empire	8/10	7/10
P2	Nomads	9/10	9/10
P3	Empire	9/10	7/10
P3	Nomads	10/10	9/10
Total human wins		55/60	48/60
Human wins as Empire		26/30	22/30
Human wins as Nomads		29/30	26/30

Humans win 91.7% against 2B agents and 80.0% against 8B, with nominal Wilson 95% intervals of [81.9, 96.4]% and [68.2, 88.2]%. As Empire, humans win 26/30 and 22/30; as Nomads, they win 29/30 and 26/30, respectively. Agent wins increase from 8.3% to 20.0% with model scale, a descriptive ordering consistent with the lower execution-conflict rate of the larger model. These outcomes reflect a substantial human–agent skill gap in both faction assignments.

Each participant wins 17–19/20 against 2B and 16/20 against 8B. Most agent wins came in a participant's first games, before the participant attended to the scoring rules, and were decided on points at final scoring. Faction outcomes stay close to even in every population across a wide range

Table 15: **Aggregate faction outcomes across small human-player studies.** Wilson intervals and binomial p values summarize game-level totals under a working independence assumption. Repeated participants and learning limit population-level interpretation.

Population	Empire wins	E (%)	E 95% CI (%)	p
Human vs. human	32/60	53.3	[40.9, 65.4]	0.70
Human vs. 2B	27/60	45.0	[33.1, 57.5]	0.52
Human vs. 8B	26/60	43.3	[31.6, 55.9]	0.37

of playing strength (Table 15): Empire wins 45.0% against the 2B agent ($p = 0.52$), 43.3% against the 8B agent ($p = 0.37$), and 53.3% between humans ($p = 0.70$), with none deviating detectably from an even split. Larger studies with more participants would sharpen these estimates.

Participants, instructions, and consent. All participants in both human studies were authors of this paper. They took part voluntarily and without compensation, played through a text interface showing the game state, and received the natural-language rulebook (Appendix J) as their only instructions; no strategic guidance was provided. Only game outcomes were recorded, and no personal or identifying data were collected. Because the participants were the investigators themselves and the sessions posed no risk beyond ordinary gameplay, the study was not submitted for institutional review.

F Scalability and Extended Configurations

F.1 Robustness to Map Size and Turn Limit

Table 7 in the main text evaluates RuleSmith across four map sizes with corresponding turn limits (16 turns for 5×5 and 7×7; 32 turns for 9×9 and 11×11). Near-balanced outcomes hold in all settings, providing initial evidence across these game scales.

F.2 Three-Faction Balancing

To evaluate scalability beyond two factions, we introduce a third faction, *Theocracy*, to CivMini. Theocracy produces `Priest` units that can `MOVE` and `CONVERT` adjacent enemy Farmers, Soldiers, or Cavalry over 1–5 game turns (optimizable); Priests sacrifice themselves upon conversion. Four new parameters are added to the search space: Priest production cost, Priest HP, Theocracy city attack damage, and conversion time. The balance loss is extended to the three-faction objective $\mathcal{L}_3(\theta) = |w_E - 1/3| + |w_N - 1/3| + |w_T - 1/3|$, targeting equal win rates among all three factions.

Table 7 shows that RuleSmith successfully balances all three factions simultaneously under both InternVL3.5-8B and InternVL3.5-2B, reaching losses of 0.033 and 0.013, respectively, based on the reported faction win rates.

F.3 Grid Size as a Rule-Level Parameter

Beyond numerical tuning, we test whether RuleSmith handles rule-level structural changes by adding the map grid size (integer range $[5, 9]$) as an optimizable parameter alongside the standard 12 numeric parameters. Table 7 shows that RuleSmith converges to near-balanced outcomes under both model sizes, demonstrating that the framework naturally extends to structural game decisions, not just fine-grained numerical adjustments.

F.4 Rule Parameters: LLM Agents, Random Agents, and Rule-Based Agents

Table 16 compares parameters optimized by BO under LLM, random, and rule-based self-play. All configurations are subsequently evaluated with InternVL3.5-8B; mixed LLM/rule-based evaluations are reported separately in Table 9. The distinct configurations illustrate how the choice of proxy population affects the discovered ruleset.

LLM-optimized parameters include initial resources of 10, a kill resource gain of 7, and Nomads HP of 12, compatible with sustained combat and unit replacement. **Random-agent parameters** use

initial resources of 4 and a lower battle-scoring weight, placing relatively more scoring weight on resource inventories. **Rule-based-agent parameters** feature cheaper Empire units (4 versus 9) and greater Nomads damage (4 versus 3). The latter reduces Nomads' TTK against Empire Soldiers from three attacks to two. Cheaper production and faster combat may help the scripted heuristics remain competitive. These numerical contrasts provide interpretable descriptions of the joint solutions; isolating individual causal effects would require separate parameter ablations.

Table 16: **Optimized parameters: LLM agents, random agents, and rule-based agents.** The three proxy player types yield substantially different balanced configurations, reflecting their distinct decision dynamics.

Parameter	LLM Agents	Random Agents	Rule-Based Agents
Economy			
Initial resources	10	4	6
Empire farmer gather	1	1	2
Nomads kill resource gain	7	5	5
Combat			
Empire damage	4	2	3
Nomads damage	3	5	4
Empire soldier HP	9	13	8
Nomads cavalry HP	12	6	9
Production			
Empire unit cost	9	8	4
Nomads unit cost	8	4	5
Scoring			
Score per resource	0.1	0.3	0.2
Score per battle won	3	1	2
Score per surviving unit	5	4	3

G Supplementary Analysis

G.1 Optimized Parameters Comparison

Table 17 summarizes the parameters that yield near-even empirical win rates under the different LLM pairings. The optimized configurations vary with agent capacity and asymmetry.

For symmetric-capacity pairings (E_{2B} vs N_{2B} and E_{8B} vs N_{8B}), the implied time-to-kill (TTK) is three attacks in both directions. These similar attack counts are compatible with competitive combat, while faction mobility and economic roles remain asymmetric. TTK describes the main combat units and does not by itself measure strategic diversity.

For E_{8B} vs N_{2B}, Nomads require four attacks to eliminate an Empire Soldier, whereas Empire requires eight to eliminate Cavalry. This favors the lower-capacity Nomads side in direct combat. Empire units also cost more than Nomads units (6 versus 4), and battle and unit scoring weights are lower than in the symmetric 8B configuration. Together, these edits are consistent with compensating for the capacity difference.

For E_{2B} vs N_{8B}, TTK still slightly favors Nomads (five versus six attacks), but initial resources fall from 10 in the symmetric 8B setting to 2. This constrains early production for both factions. Higher farmer gathering (2 versus 1) and cheaper Empire units (4 versus 5 for Nomads) may help the lower-capacity Empire remain competitive. These interpretations concern the combined configuration, whose observed outcomes are near-balanced.

These comparisons illustrate coordinated edits across economy, combat, production, and scoring. RuleSmith discovers distinct near-balanced configurations for different player pairings, retaining faction asymmetry while adjusting numerical strength. This supports flexibility across the evaluated populations without assuming a single canonical balanced ruleset.

Table 17: **Optimized game parameters under different LLM agent configurations.** We report optimized parameters that achieve approximately 50/50 win rates across different LLM agent configurations. We also include the implied time-to-kill (TTK; number of attacks required to eliminate the opponent's main combat unit) derived from damage and HP.

Parameter	$\boldsymbol{E}_{2B}$ **vs** $\boldsymbol{N}_{2B}$	$\boldsymbol{E}_{2B}$ **vs** $\boldsymbol{N}_{8B}$	$\boldsymbol{E}_{8B}$ **vs** $\boldsymbol{N}_{2B}$	$\boldsymbol{E}_{8B}$ **vs** $\boldsymbol{N}_{8B}$
Initial resources	10	2	10	10
Empire farmer gather	1	2	4	1
Nomads kill resource gain	6	9	1	7
Empire damage	4	2	2	4
Nomads damage	3	3	5	3
Empire soldier HP	9	14	16	9
Nomads cavalry HP	10	11	16	12
$\mathbf{TTK_{N\to E}}$ ($\lceil \mathbf{HP}_E/\mathbf{DMG}_N \rceil$)	**3**	**5**	**4**	**3**
$\mathbf{TTK_{E\to N}}$ ($\lceil \mathbf{HP}_N/\mathbf{DMG}_E \rceil$)	**3**	**6**	**8**	**3**
Empire unit cost	4	4	6	9
Nomads unit cost	4	5	4	8
Score per resource	0.4	0.2	0.3	0.1
Score per battle won	3	4	1	3
Score per surviving unit	3	4	1	5

G.2 Representative Rollout

Rollout configurations. The qualitative examples use two archived 8B/8B optimization checkpoints: R94 (iteration 94) and R98 (iteration 98). Figure 3 shows R94 game 13, while Figures 4–7 show R98 games 4, 1, 5, and 13, respectively. Table 18 lists their parameters alongside the reported 8B/8B configuration. These examples illustrate tactics at selected checkpoints; population-level balance is assessed by the separate held-out evaluations. Turn labels in these figures start at 0, so a game ending on turn 12 spans 13 rounds and one ending on turn 6 spans 7.

Table 18: **Configurations used in the qualitative rollouts.** R94 and R98 are archived optimization checkpoints, distinct from the reported 8B/8B configuration in Table 17.

Parameter	R94	R98	Reported 8B/8B
Initial resources	8	4	10
Empire farmer gather	1	1	1
Nomads kill resource gain	8	7	7
Empire damage	2	2	4
Nomads damage	4	4	3
Empire soldier HP	14	14	9
Nomads cavalry HP	8	8	12
Empire unit cost	3	6	9
Nomads unit cost	5	4	8
Score per resource	0.4	0.2	0.1
Score per battle won	3	4	3
Score per surviving unit	4	4	5

For R94 and R98, the battle multiplier is 1 and each city produces 3 resources per `PRODUCE_RESOURCE` action. In addition to their cities, Empire starts with one Farmer and one Soldier, and Nomads start with two Cavalry.

Figure 3 shows a Nomads conquest on turn 12. Cavalry mobility allows a unit to bypass the Empire's central formation and reach its city. The Empire redeploys Soldiers to intercept, but sustained city pressure ends the game. This illustrates how a narrow opening can become a decisive conquest even when the Empire retains units elsewhere on the board.

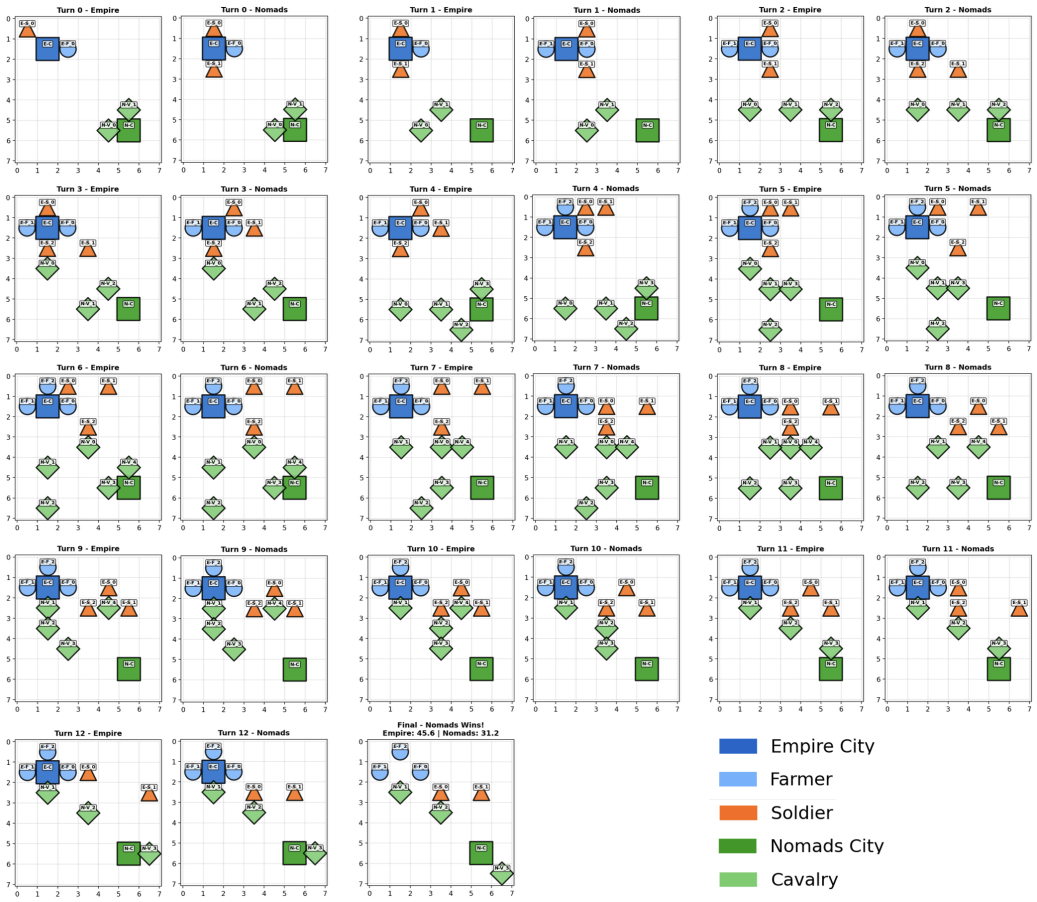


Figure 3: **Visualization of a CivMini Game.** Nomads conquer the Empire's city on turn 12 (13 rounds, counting from 0). Cavalry bypass the Empire's defense and sustain city pressure. Both agents use InternVL3.5-8B. This example is R94 game 13 (Table 18); city destruction determines victory irrespective of the displayed scores.

G.3 Extra Visualization for Game Rollouts

We provide additional qualitative visualizations with InternVL3.5-8B agents on both sides under checkpoint R98 (Table 18). Figures 4–7 show both factions winning through distinct tactical paths, illustrating early aggression, unit composition, and economic decisions under identical rules.

Figure 4 shows a fast Nomads victory. Cavalry Unit 0 repeatedly attacks the Empire's city from turn 3 through turn 6, dealing 4 HP per strike. Four attacks destroy the 14-HP city, ending the game in its seventh round. This example illustrates a concentrated city attack succeeding before local defenses can stop it.

In contrast, Figure 5 shows a longer game in which the Nomads secure a high-score victory through sustained expansion and force dominance. By continuously producing cavalry units, the Nomads overwhelm the Empire's defensive formation. Although the Empire attempts to stabilize the board with farmers and soldiers, its inability to eliminate cavalry units leads to gradual territorial collapse and a board state fully dominated by Nomad forces.

Figures 6 and 7 show the Empire countering early Nomads aggression. In Figure 6, Soldiers hold a defensive line through a prolonged engagement, while Farmers generate resources. The Empire wins at the turn cap through its final resources, battle outcomes, and surviving units.

Similarly, Figure 7 shows a failed Nomad city rush. Although the Nomads initiate a direct attack on the Empire's city as early as turn 5, the Empire's soldiers systematically eliminate the attacking

cavalry units. As the Nomad force diminishes, the Empire maintains both a military presence and economic growth, ultimately winning by preserving units and accumulating resources.

These selected rollouts illustrate different tactical paths under the optimized rules: the Nomads win through city pressure or numerical advantage, while Empire victories combine defense, resource production, and unit survival. They complement the larger trajectory diagnostics in Appendix I. These examples demonstrate observed behavioral variety; establishing a causal effect of optimization on strategic diversity or player enjoyment would require a controlled comparison.

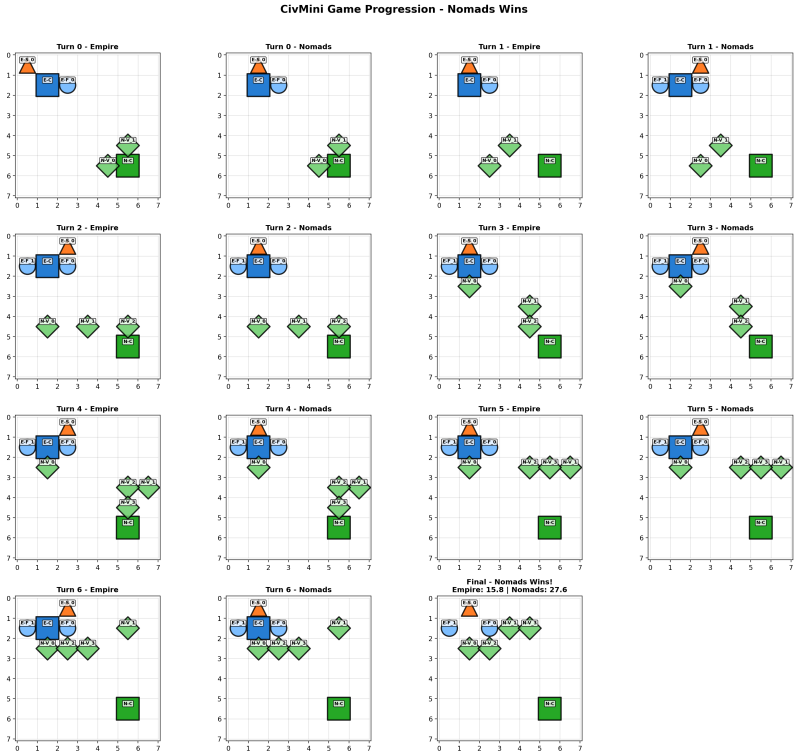


Figure 4: A rapid Nomads conquest on turn 6 (7 rounds, counting from 0). Cavalry Unit 0 strikes the Empire's city on turns 3–6, dealing 4 HP per attack and destroying it. Checkpoint R98, game 4. (Final score: Empire 15.8 | Nomads 27.6)

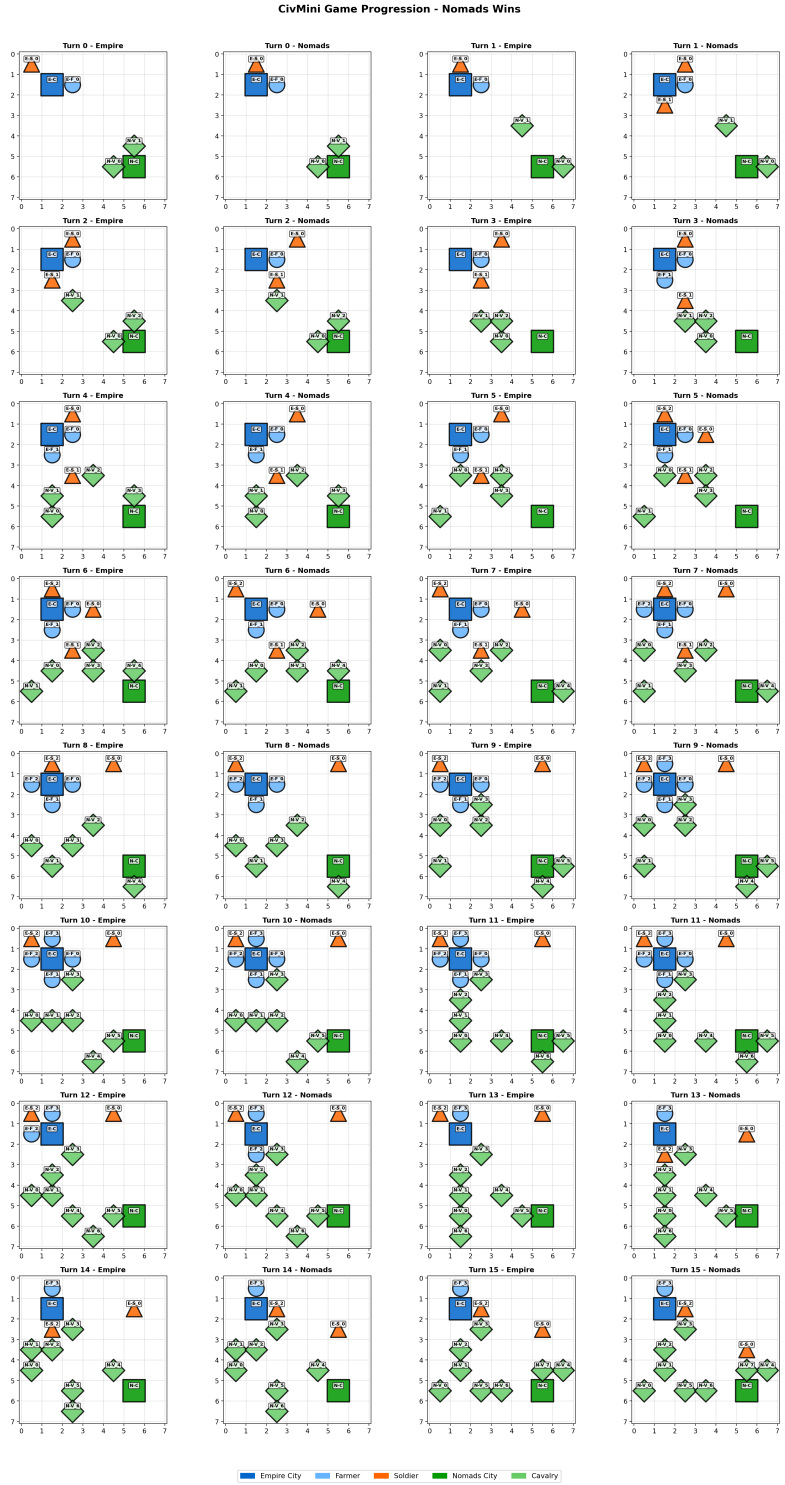


Figure 5: A Nomads score victory after 16 rounds (turns 0–15). Continued cavalry production builds a numerical advantage, while the Empire deploys Farmers and Soldiers and loses units during the fighting. Checkpoint R98, game 1. (Final score: Empire 14.2 | Nomads 67.6)



Figure 6: The Empire withstands early Nomads aggression and wins at the 16-round turn cap. Soldiers defend through fighting on turns 3–15, while Farmers supply resources. Two Cavalry remain at the end; the Empire wins through the combined final scoring terms. Checkpoint R98, game 5. (Final score: Empire 31.6 | Nomads 23.4)

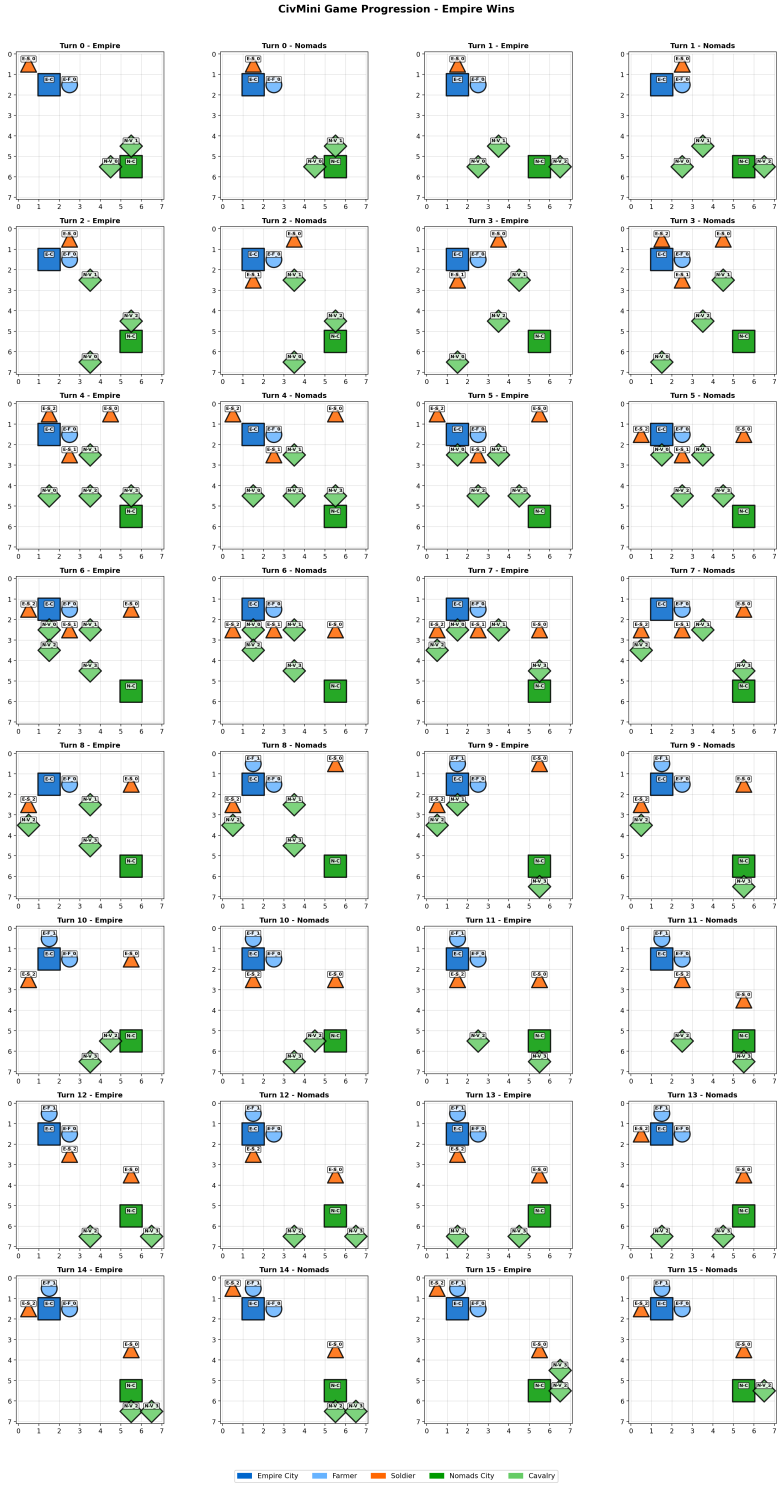


Figure 7: The Empire defends against a Nomads city rush beginning on turn 5. By turn 15, one Cavalry remains, while the Empire retains two Soldiers and two Farmers. Unit survival and resources contribute to its score victory. Checkpoint R98, game 13. (Final score: Empire 34.4 | Nomads 12.6)

H Additional Statistical Evaluations

The additional evaluations include three independent GP optimization runs, each evaluated on separate 200- and 300-game batches, and one selected configuration each from NTBEA, TPE, and a GP-LCB run, evaluated at the same two sample sizes. The originally reported optimized configuration is also evaluated on 500 games. These are final-configuration evaluations, not per-iteration confidence intervals or comparisons of convergence speed. Table 19 reports every additional 200-/300-game batch.

Table 19: **Full 200-/300-game re-evaluations.** k_E is the number of Empire wins from the evaluation logs; none of these batches contains a draw. Losses use counts rather than rounded percentages. Each configuration's two batches use separate rollout seeds.

Search / run	n	k_E	E (%)	N (%)	$\widehat{\mathcal{L}}$	E 95% CI	p
GP run 1	200	103	51.5	48.5	0.030	[44.6, 58.3]	0.72
GP run 1	300	155	51.7	48.3	0.033	[46.0, 57.3]	0.60
GP run 2	200	98	49.0	51.0	0.020	[42.2, 55.9]	0.83
GP run 2	300	148	49.3	50.7	0.013	[43.7, 55.0]	0.86
GP run 3	200	103	51.5	48.5	0.030	[44.6, 58.3]	0.72
GP run 3	300	158	52.7	47.3	0.053	[47.0, 58.2]	0.39
NTBEA	200	98	49.0	51.0	0.020	[42.2, 55.9]	0.83
NTBEA	300	144	48.0	52.0	0.040	[42.4, 53.6]	0.53
TPE	200	103	51.5	48.5	0.030	[44.6, 58.3]	0.72
TPE	300	163	54.3	45.7	0.087	[48.7, 59.9]	0.15
GP-LCB	200	115	57.5	42.5	0.150	[50.6, 64.1]	0.04
GP-LCB	300	173	57.7	42.3	0.153	[52.0, 63.1]	0.009

Interval and test definitions. For k Empire wins in n games, write $\hat{p} = k/n$ and $z = 1.9599639845$. The Wilson interval is

$$\frac{\hat{p} + z^2/(2n) \;\pm\; z\sqrt{\hat{p}(1-\hat{p})/n + z^2/(4n^2)}}{1 + z^2/n}.$$

For no-draw games, a two-sided exact binomial test of $p_E = 0.5$ uses

$$p_{\text{test}} = \min\left\{1,\; 2\sum_{j=0}^{\min(k,n-k)} \binom{n}{j} 2^{-n}\right\}.$$

Counts are taken from the evaluation logs. When a batch contains draws, the test compares Empire and Nomads wins among decisive games, and draws are reported separately; the marginal Empire interval uses all games.

The sample SD of the loss across the three GP runs is computed with denominator $3 - 1$, separately for each evaluation size. It includes optimization-run and evaluation variability; it is not an error bar on every iteration of the optimization trajectory. No multiple-comparison-adjusted claim of superiority is made. Failure to reject an even split does not establish equivalence within $[0.45, 0.55]$. Bayesian analyses calibrated to frequentist operating characteristics [50, 51] are an alternative for small evaluation batches; we report frequentist intervals and exact tests for transparency.

Finite-sample loss. Because the loss contains absolute values, its empirical estimate has finite-sample bias as well as sampling variance, both of which depend on the game count. Selected configurations are therefore assessed on independent held-out batches at fixed reported sample sizes, separately from the losses observed during adaptive-budget optimization. Choosing how many games to spend on an evaluation is a sample-size design question, as in clinical trial design [52], and allocating games as evidence accumulates during optimization parallels group sequential designs [53]; the held-out evaluations use sample sizes fixed in advance.

Table 20: Wilson 95% intervals for every distinct win rate in the 100-game evaluations of Tables 2, 4, 5, 8, 9, 10, 11, 12, and 7, under the 100-game protocol of Section 4.2. A rate of $r\%$ corresponds to r wins. For no-draw two-faction outcomes, the Nomads interval is $[100-u, 100-l]$ when the Empire interval is $[l, u]$; for three factions, each faction's rate is listed.

Rate (%)	95% CI (%)	Rate (%)	95% CI (%)	Rate (%)	95% CI (%)
0	[0.0, 3.7]	6	[2.8, 12.5]	13	[7.8, 21.0]
22	[15.0, 31.1]	24	[16.7, 33.2]	26	[18.4, 35.4]
29	[21.0, 38.5]	31	[22.8, 40.6]	32	[23.7, 41.7]
33	[24.6, 42.7]	34	[25.5, 43.7]	35	[26.4, 44.7]
37	[28.2, 46.8]	38	[29.1, 47.8]	41	[31.9, 50.8]
45	[35.6, 54.8]	46	[36.6, 55.7]	47	[37.5, 56.7]
48	[38.5, 57.7]	49	[39.4, 58.7]	50	[40.4, 59.6]
51	[41.3, 60.6]	52	[42.3, 61.5]	53	[43.3, 62.5]
55	[45.2, 64.4]	61	[51.2, 70.0]	65	[55.3, 73.6]
68	[58.3, 76.3]	73	[63.6, 80.7]	75	[65.7, 82.5]
81	[72.2, 87.5]	91	[83.8, 95.2]	100	[96.3, 100.0]

Starting configurations. The default configuration in the released code sets initial resources to 5, Farmer gather to 2, Nomads kill gain to 3, damage to 1 (Empire) and 2 (Nomads), HP to 8 (Soldier) and 6 (Cavalry), unit cost to 4 for both factions, and score weights to 0.1, 1, and 1. Its 300-game evaluation with 8B agents gives 34 Empire wins, 264 Nomads wins, and 2 draws (Empire 11.3%, Wilson interval [8.2, 15.4]%; exact binomial test of Empire against Nomads wins among the 298 decisive games $p = 2.9 \times 10^{-45}$). The qualitative rollout checkpoints R94/R98 are identified separately in Table 18.

I Agent Validity and Trajectory Diversity

I.1 Execution-Level Conflicts

All accepted state transitions obey the engine's rules. A joint action proposal can still fail at execution when an earlier action changes its destination occupancy or target. The statistics recorded below count attempted unit actions converted to `PASS` at execution; they are not a comprehensive rate of unparseable model outputs or hallucinated rule statements.

Table 21: Execution-level rejection statistics from additional evaluations.

Setting	Attempted actions	Converted to `PASS`	Rate
8B agents, 500 games	33,834	1,126	3.33%
2B agents, 200 games	21,871	1,012	4.63%

In the 8B evaluation, 963 rejections involve movement and 163 involve battle; most result from within-turn resolution conflicts. The 2B rejections are mostly movement conflicts. These rates show a capability-dependent burden from joint action planning, but do not isolate model scale from every other source of variation in these two evaluation batches. Legality enforcement also cannot detect a legal but strategically poor action.

I.2 Trajectory Diversity

We analyze the 300-game evaluation of the NTBEA configuration with 8B agents (Empire 48.0%) and its first 100 games. The analysis uses the executed per-unit actions. Openings are the sequence of all actions by both factions in turns 0–2; entropy is $-\sum_a p(a)\log_2 p(a)$ over action types per faction; board divergence is the mean pairwise Jaccard distance $1-|A\cap B|/|A\cup B|$ between the sets of (cell, unit type) pairs after the Nomads' turn 5 (or the last recorded state for shorter games). The analysis script is included in the release, and the results are in Table 6 of the main text.

The first turn offers few legal action combinations, but trajectories diverge over subsequent decisions; the most frequent three-turn opening accounts for 15 of the 300 games. Games end both by conquest and at the turn cap. The diagnostics show that evaluation games do not replay a single line; they do not establish strategic optimality or coverage of the strategy space.

J Implementation, Prompts, and Settings

The MIT-licensed repository https://github.com/Adonis-galaxy/RuleSmith provides the game engine, prompts, self-play, the GP, ES, random-search, NTBEA, and TPE optimizers, evaluation, and visualization.

J.1 Settings

Table 22: Settings for the additional runs (Sections 4.4 and 4.5). The main runs used the same game and agent settings.

Component	Setting
Agents	InternVL3.5-2B/8B, FP16, one call per player turn, at most 300 new tokens, temperature 0.7, top-p 0.9, no regeneration; rule checker disabled
Retrieval	TF-IDF with cosine similarity over the 10 rulebook entries, query "multi-unit actions turn structure", top 3
Prompt	Up to 5 own units listed per decision
Game	7×7 map, 16-turn cap, Farmer HP 5, city HP equal to the faction's combat-unit HP, battle multiplier 1, city production 3 per action, movement 1 (Empire) and 2 (Cavalry), Empire starts with one Farmer and one Soldier and Nomads with two Cavalry
GP optimizer	scikit-optimize 0.10.2: Gaussian process with constant amplitude times ARD Matérn-5/2 kernel, Gaussian noise, normalized targets; 10 random initial points; acquisition optimized with L-BFGS; EI ($\xi = 0.01$); LCB runs use $\kappa = 1.96$
Adaptive budget	Eq. (6), $N_{\min} = 16$, $N_{\max} = 64$, $\epsilon = 10^{-3}$, integer truncation
Final selection	Configuration with the lowest observed in-loop loss (ties: earliest)
GP runs	Seeds 1, 2, 3 for both the BO random state and the per-game seed base (game i of iteration t uses seed $10^5 s + 100t + i$)
LCB run	Seed 1, otherwise identical to GP run 1, so the two share their initial design; 100 iterations (2,899 games)
NTBEA	Seed 11; 1-tuples, 2-tuples, and the full 12-tuple over the discretized grid; UCB constant $C = 1$; 50 mutated neighbors per step; 68 evaluations of 16 games (1,088 games); uniformly random start
TPE	Optuna 4.9.0 [54] `TPESampler` defaults, seed 12; 68 trials of 16 games (1,088 games)
Held-out evaluation	Per-game seed bases disjoint from in-loop seeds: NTBEA 311 and 611, TPE 312 and 612, GP-LCB 901 and 1001 (200 and 300 games), default configuration 1100; the batches behind Table 21 use 1000 (8B) and 1200 (2B)
Software	PyTorch 2.6.0, Transformers 4.57.6, scikit-learn 1.8.0

J.2 Complete Rulebook

The rulebook has ten entries, reproduced verbatim (with whitespace normalized). Entries name rule parameters (for example, `empire_unit_production_cost`) but never state their values. Two entries are inconsistent with the engine: Rule 1 gives a typical length of 10 turns, whereas all main experiments use a 16-turn cap that is shown in the prompt, and Rule 2 states that Cavalry can `GATHER`, whereas Rule 6 and the engine do not allow it. The legal-action lists in the prompt follow the engine.

```
Rule 1 - Game Setup: CivMini is played on a 7x7 grid map between two civilizations: Empire and Nomads.
     Empire starts
with one city at position (1,1), Nomads starts with one city at position (5,5). Both cities are immobile
      units that
```

```
can be attacked. The game lasts for a maximum number of turns (typically 10). Players take turns
      alternating.
IMPORTANT: Only ONE unit can occupy a cell at any time - no two units (friendly or enemy) can share the
      same cell.

Rule 2 - Unit Types: Empire has THREE unit types: (1) City - immobile, can produce resources or units,
      can be
attacked; (2) Farmer - can only GATHER resources and MOVE, cannot battle; (3) Soldier - can only BATTLE
      and MOVE,
cannot gather. Nomads has TWO unit types: (1) City - same as Empire city; (2) Cavalry - versatile units
      that can
GATHER, BATTLE, and MOVE. All units (including cities) have hit points and can be destroyed.

Rule 3 - Map: The 7x7 map is a simple grid with no terrain or resource types. All cells are identical.
      IMPORTANT: Each
cell can contain AT MOST ONE unit (no stacking). Units cannot move to occupied cells.

Rule 4 - Turn Structure: Each turn consists of: (1) Player action phase where EACH UNIT (including
      cities) of the
current player performs ONE action (multi-action mode), (2) Switch to next player. After both players
      have acted, the
turn counter increments and all units reset their movement points. In single-action mode, only one unit
      acts per turn.

Rule 5 - City Actions: Cities are special immobile units that can perform two types of actions: (1)
      PRODUCE_RESOURCE -
generate resources (empire_city_resource_production for Empire, nomads_city_resource_production for
      Nomads), (2)
PRODUCE_UNIT - spend resources to create a new unit. Unit costs differ by civilization (
      empire_unit_production_cost
and nomads_unit_production_cost). Empire cities can produce Farmers (gather only) or Soldiers (combat
      only). Nomads
cities can only produce Cavalry (both gather and combat). New units appear in an adjacent empty cell.

Rule 6 - Resource Economy: Empire gains resources through GATHER action (Farmers only). Farmers have
empire_farmer_gather_amount gathering rate. Nomads CANNOT gather - instead, Nomads gain resources by
      KILLING enemy
units. When Nomads destroy an enemy, they receive nomads_kill_resource_gain resources. This creates
      asymmetric
economies: Empire is peaceful gatherers, Nomads are aggressive raiders.

Rule 7 - Movement Rules: MOVE action allows a unit to move up to its move_points in Manhattan distance (
      up, down,
left, right - no diagonals). Empire units (Farmers and Soldiers) have empire_unit_move_points (typically
       1). Cavalry
have nomads_cavalry_move_points (typically 2, meaning they can move UP TO 2 cells per turn for greater
      mobility).
Cities have 0 movement points (immobile). CRITICAL: A cell can only contain ONE unit - moves are blocked
       if ANY unit
(friendly or enemy) is at the destination. Units cannot stack.

Rule 8 - Combat System: BATTLE action attacks an enemy unit in an ADJACENT cell (range 1, up/down/left/
      right). Only
Soldiers, Cavalry, and Cities can battle. Farmers cannot battle. Damage calculation: base_damage
(empire_battle_base_damage for Empire, nomads_battle_base_damage for Nomads) multiplied by battle_bonus
      (unified for
both sides). When a unit's HP reaches 0, it is destroyed and removed from the map. Cities have the same
      HP as their
civilization's combat units (empire_soldier_hp or nomads_cavalry_hp). The attacker gains a battle
      victory count.

Rule 9 - Victory Conditions: The game ends immediately if one player's CITY is destroyed - the other
      player wins
instantly. If both cities survive to max_turns, the winner is determined by final score. Final score =
      resources *
score_per_resource + battles_won * score_per_battle_won + surviving_units * score_per_surviving_unit.
      Cities are not
scored separately since each side has exactly one.

Rule 10 - Strategic Considerations: Empire has peaceful economy (Farmers gather resources) but must
      protect Farmers
with Soldiers. Nomads have aggressive economy (gain resources ONLY by killing enemies) forcing them into
       combat from
the start. Cavalry can move 2 cells per turn, allowing rapid strikes and retreats. Empire can choose
      defensive play,
but Nomads MUST attack to gain resources for army expansion. This asymmetry creates distinct playstyles:
       Empire builds
economy then army, Nomads use mobility to raid immediately. Destroying enemy city is instant victory.
      Cities produce
resources OR units each turn.
```

J.3 System Prompts

Each faction receives a fixed system prompt; the Empire version is shown below, and the Nomads version differs only in the faction name and the subsequent trait block.

```
You are playing CivMini as the Empire civilization.

Your goal is to WIN the game by DEFEATING your opponent!

CRITICAL: Try your best to attack the units and city of the opponent to win the game!
The enemy city is your PRIMARY TARGET - destroying it means INSTANT VICTORY!

Strategy:
- ATTACK enemy units and cities aggressively
- Gather resources to build an army
- Expand your forces to overwhelm the enemy
- Protect your own city from enemy attacks

You will be given the current game state and legal actions.
Choose actions based on your civilization's strengths:

Empire Traits:
- STRONGER in combat (battle bonus)
- Economy: GATHER resources with farmers
- Your city at (1,1) TOP-LEFT. Enemies come from (5,5) BOTTOM-RIGHT.

ACTIONS:
- SOLDIERS: MOVE right/down (X+1 or Y+1) to intercept enemies
- FARMERS: GATHER resources when safe
- If enemy adjacent: BATTLE immediately!

Respond with a valid JSON action object. Prioritize military aggression!
```

Nomads trait block:

```
Nomads Traits:
- FAST cavalry (mobile raiders) - CAN MOVE 2 CELLS PER TURN!
- Economy: KILL enemies for resources
- Your city at (5,5) BOTTOM-RIGHT. Empire city at (1,1) TOP-LEFT.

STRATEGY:
- PRIMARY GOAL: Destroy Empire city at (1,1)!
- CAVALRY: Move towards Empire CITY (not just left!)
- Use your SPEED ADVANTAGE - Empire units only move 1 cell!
- If enemy adjacent: BATTLE immediately!
- DEFEND: If Empire soldiers approach (5,5), intercept them!
- Target priority: City > Farmers > Soldiers
```

J.4 Example Decision Prompt

The user prompt below is rendered by the released code for the Empire's first decision under the configuration in Table 10; emojis in the strategy-guide banner are replaced by `[emoji]`. The "Cities: 0" line is a display defect in the released code (the city is listed as a unit). The block after "COPY THESE EXACT ACTIONS" is produced by the heuristic strategy guide from unit positions alone.

```
RELEVANT RULES:
1. Rule 4 - Turn Structure: Each turn consists of: (1) Player action phase where EACH UNIT
    (including cities) of the current player performs ONE action (multi-action mode), (2) Switch
    to next player. After both players have acted, the turn counter increments and all units
    reset their movement points. In single-action mode, only one unit acts per turn.

2. Rule 5 - City Actions: Cities are special immobile units that can perform two types of
    actions: (1) PRODUCE_RESOURCE - generate resources (empire_city_resource_production for Empire,
    nomads_city_resource_production for Nomads), (2) PRODUCE_UNIT - spend resources to create a
    new unit. Unit costs differ by civilization (empire_unit_production_cost and
      nomads_unit_production_cost).
    Empire cities can produce Farmers (gather only) or Soldiers (combat only). Nomads cities can
    only produce Cavalry (both gather and combat). New units appear in an adjacent empty cell.

3. Rule 10 - Strategic Considerations: Empire has peaceful economy (Farmers gather resources)
    but must protect Farmers with Soldiers. Nomads have aggressive economy (gain resources ONLY
    by killing enemies) forcing them into combat from the start. Cavalry can move 2 cells per turn,
    allowing rapid strikes and retreats. Empire can choose defensive play, but Nomads MUST attack
    to gain resources for army expansion. This asymmetry creates distinct playstyles: Empire builds
    economy then army, Nomads use mobility to raid immediately. Destroying enemy city is instant
```

```
    victory. Cities produce resources OR units each turn.

CURRENT GAME STATE:
Turn: 0/16
Your Civilization: Empire
Resources: 10.0
Cities: 0
Battles Won: 0

ENEMY (Nomads) UNITS:
  nomads_city: (5,5) Type=city HP=12.0
  nomads_cavalry_0: (4,5) Type=cavalry HP=12.0
  nomads_cavalry_1: (5,4) Type=cavalry HP=12.0

==================================================
STRATEGY GUIDE (FOLLOW THIS!):
==================================================
[emoji] EMPIRE STRATEGY [emoji]

YOUR CITY: (1,1) - PROTECT IT!
ENEMY CITY: (5,5) - ATTACK TARGET!

[emoji] No immediate threats - push towards enemy city!

COPY THESE EXACT ACTIONS:

"empire_city": {"action_type": "PRODUCE_UNIT", "produce_unit_type": "soldier", "to": {"x": 1, "y": 2}}
  ^ Produce soldier for defense!
"empire_farmer_0": {"action_type": "GATHER"}
  ^ Gather resources safely
"empire_soldier_0": {"action_type": "MOVE", "to": {"x": 1, "y": 0}}
  ^ ATTACK: Move towards Nomads city! (9 cells away)

REMEMBER: COPY THE EXACT JSON ABOVE!
GOAL: Protect city at (1,1) AND attack enemy city at (5,5)!
==================================================

YOUR UNITS (each needs an action):

empire_city:
  Position: (1, 1)
  HP: 9.0, Move Points: 0.0
  Legal actions: PRODUCE_UNIT, PRODUCE_RESOURCE
  Available production positions: (1,2), (1,0), (0,1)

empire_farmer_0:
  Position: (2, 1)
  HP: 5.0, Move Points: 1.0
  Legal actions: MOVE, GATHER

empire_soldier_0:
  Position: (0, 0)
  HP: 9.0, Move Points: 1.0
  Legal actions: MOVE

OUTPUT FORMAT:
Return a JSON object with actions for ALL units:
{
  "unit_id_1": {"action_type": "GATHER"},
  "unit_id_2": {"action_type": "MOVE", "to": {"x": 1, "y": 1}},
  "unit_id_3": {"action_type": "BATTLE", "to": {"x": 2, "y": 2}, "target_civ": "Nomads"},
  "empire_city": {"action_type": "PRODUCE_UNIT", "produce_unit_type": "soldier", "to": {"x": 1, "y": 2}}
}

NOTE: For PRODUCE_UNIT, specify the target position with 'to': {'x': X, 'y': Y}
      The position must be adjacent to the city and empty!

YOUR ACTIONS:
```

J.5 Models and Figure Tools

InternVL3.5 [46] (2B/8B) and Qwen3.5-9B are released under Apache-2.0.[2] Gemini 2.5 Flash and Pro were accessed through Google's API under its terms of service. The explanatory illustrations in Figures 1 and 2 were generated with Nano Banana Pro (Google) and ChatGPT-5.2 (OpenAI),

[2] https://huggingface.co/OpenGVLab/InternVL3_5-2B; https://huggingface.co/OpenGVLab/InternVL3_5-8B; https://huggingface.co/Qwen/Qwen3.5-9B.

respectively, under the providers' terms; they convey the pipeline only, and no data figure or reported number was produced with these tools. Game-rollout figures are rendered by the released visualization code.

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 introduction state the framework, the CivMini testbed, and the empirical scope. Claims are limited to the tested agent populations and sample sizes.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 and Appendix A discuss population dependence of balance, the role of the prompt's strategy guide in the evaluated population, statistical scope, designer-specified rule spaces, the simplified game, and the absence of optimality guarantees.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

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: Appendix C states all assumptions and gives proofs for Proposition C.1, Lemma C.2, and Proposition C.3; the remarks state that the regret bound covers an idealized LCB rule, not the implemented acquisition or adaptive replication.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and crossreferenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

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: Section 4.2, Table 1, Table 22, and Appendix J give the models, decoding, prompts and rulebook verbatim, optimizer settings, seeds, budgets, and evaluation protocol.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm. (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully. (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset). (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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: Code for the main experiments, the prompts, and the analysis scripts are released under the MIT license at https://github.com/Adonis-galaxy/RuleSmith, with instructions to run the optimization and the held-out evaluations.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/ public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https: //neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

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: Section 4.2 and Table 22 specify the optimizer, acquisition, budgets, adaptive-sampling bounds, discretization, and the held-out evaluation protocol, and state which settings were chosen by convention rather than tuned.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

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: Win rates of the additional evaluations carry Wilson 95% intervals and exact two-sided binomial tests; repeatability is assessed with three independent optimization runs, and the 100-game original results are given Wilson intervals in Table 20. Definitions and assumptions are in Appendix H. Original ablations remain single runs, as stated.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

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: [Yes]

   Justification: Section 4.2 states the hardware of the original experiments (8 NVIDIA A100 80GB) and of the added runs (NVIDIA RTX 6000 Ada), and the wall-clock time of a 100-iteration optimization.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

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 work uses a synthetic game and small gameplay studies in which the authors themselves were the players; only game outcomes were recorded, and no personal or sensitive data were collected (Appendix E.4).

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: Appendix B discusses positive uses (design tools, analysis of rule-driven systems) and risks (over-optimizing rules to the disadvantage of participants), with mitigations.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

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 paper releases a game environment and an optimization pipeline, not pretrained models or scraped data with high misuse risk.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

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: [Yes]

Justification: InternVL3.5 is cited, and InternVL3.5 and Qwen3.5 are released under Apache-2.0 (model cards in Appendix J); Gemini 2.5 was accessed under Google's API terms; scikit-optimize (BSD-3-Clause) and Optuna (MIT) are cited; the figure-generation tools are credited (Appendix J).

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the link of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: CivMini and the RuleSmith pipeline are documented in Section 3 and Appendix J (complete rulebook, prompts, and settings) and in the repository README, under the MIT license.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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: [Yes]

    Justification: Appendix E.4 describes both human studies: the participants were authors of this paper, their only instructions were the natural-language rulebook reproduced in Appendix J, they played through a text interface showing the game state, and they received no compensation.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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: [Yes]

Justification: Appendix E.4 states that all participants were authors of this paper, that the sessions posed no risk beyond ordinary game play and collected no personal data, and that the study was therefore not submitted for institutional review.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

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: LLMs are the game-playing agents whose self-play defines the balance objective; their use, prompts, and decoding are described in Sections 3.2 and 4.2 and Appendix J.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.