

Where to Search: Measure the Prior-Structured Search Space of LLM Agents

Zhuo-Yang Song¹

¹School of Physics, Peking University, Beijing 100871, China

Abstract

The generate-filter-refine (iterative paradigm) based on large language models (LLMs) has achieved progress in reasoning, programming, and program discovery in AI+Science. However, the effectiveness of search depends on where to search, namely, how to encode the domain prior into an operationally structured hypothesis space. To this end, this paper proposes a compact formal theory that describes and measures LLM-assisted iterative search guided by domain priors. We represent an agent as a fuzzy relation operator on inputs and outputs to capture feasible transitions; the agent is thereby constrained by a fixed safety envelope. To describe multi-step reasoning/search, we weight all reachable paths by a single continuation parameter and sum them to obtain a coverage generating function; this induces a measure of reachability difficulty; and it provides a geometric interpretation of search on the graph induced by the safety envelope. We further provide the simplest testable inferences and validate them via two instantiation. This theory offers a workable language and operational tools to measure agents and their search spaces, proposing a systematic formal description of iterative search constructed by LLMs.

1 Introduction

The generate-filter-refine iterative paradigm centered on large language models (LLMs) is rapidly expanding its application boundary—from reasoning and programming [1–5], to planning and tool use [6–11], and further to program/function search in AI+Science [12–17]. The common structure of such paradigms embeds tasks or hypotheses into an operational space and performs multi-round generation, evaluation, and update on that space. Although this approach has performed well in many cases, its effectiveness is fundamentally constrained by where to search [17–21]: that is, how the prior is encoded into the agent-operable space. In practice, agents based on LLMs often do not wander blindly in the orig-

inal space, but iterate within a smaller semantic space defined by priors and constraints; the geometry and boundary of this space determine efficiency and stability [22].

Long-horizon tasks raise higher demands for understanding such search. First, safety is the primary constraint: in real systems or sensitive scenarios, LLMs must operate within verifiable and controllable boundaries [23–29]. Intuitively, this requires formally confining the model within a safety envelope, allowing only constraint-satisfying transitions. Second, complexity requires a systematic characterization of the search process: long-horizon problems often involve combinatorial explosion and sparse rewards; purely heuristic or 0/1 scoring is insufficient to quantify reachability difficulty, com-

pare the coverage capability of different agents, or guide sampling budgets and staged training [18, 30, 31]. Therefore, a concise, computable, model-agnostic formal theory is needed: one that unifies safety and reachability under the same set of measures, and provides testable predictions and engineering-usable design principles.

Current practice mostly relies on engineering heuristics (prompt design, filters, scoring functions, temperatures, and sampling budgets), lacking a unified language and quantitative tools for agent-space-search. Concretely, it is difficult to measure, in a comparable way, the trade-offs between reachability and safety across agents, and there is a lack of clear characterization and explanation of long-horizon behavioral features of agents. This theoretical gap may be the key deficiency in moving LLM-driven complex tasks from usable to controllable and measurable.

To address this, the paper proposes a compact formal theory to characterize and measure LLM-assisted iterative search. Specifically, we formalize agents as fuzzy relation operators; in iterative applications we feed outputs back into inputs to form an iterated agent, and introduce a critical parameter as a unified quantification of reachability difficulty. On the directed graph induced by the safety envelope, we discuss geometric features of the search space. To validate the abstract concepts, we provide two minimal instantiation. The instantiation yields evidence consistent with the hypotheses, providing an initial external validation of the formalization.

The significance of this theory is that it establishes a measurement system in which safety and reachability are measured by the same symbols and geometric quantities; this enables operational metrics for several questions—for example, whether an intermediate waypoint can significantly reduce overall difficulty can be localized by the compositional lower bound for coverage (transitivity) of the coverage index. This formalization offers a consistent baseline for comparing agents, designing search strategies, and setting training signals.

The paper is organized as follows. Section 2

presents the formal theory, including the fuzzy relation operator representation of agents, the coverage generating function and critical parameters, geometric quantities and inequalities on the safety envelope-induced graph, and several hypotheses for iterated agents constructed by LLMs. Section 3 provides the majority-vote instantiation and tests the inequalities involving shortest distance and the number of shortest paths, as well as the approximately unidirectional search hypothesis. Section 4 concludes and discusses prospective directions for connecting the proposed measures to evaluation, search policy, and reinforcement learning rewards.

2 Formal Theory

2.1 Conventions and Objects of Study

This section introduces the minimal mathematical objects for characterizing the LLM-driven generate-filter-refine process and defines reachability and search geometry using a unified generating-function language.

Notation 1 (Search space and empty-product convention). *Let C_1, C_2 be nonempty sets representing the input space and output space of an agent, respectively. In iterative scenarios, we assume $C_2 \subseteq C_1$ so that outputs can be fed back as inputs for the next step. For any finite product, the empty product is defined to be 1.*

Definition 1 (Ideal agent and fuzzy relation operator). *An ideal agent $\mathcal{T}$ is a mapping $f \mapsto \mu_f(\cdot)$, where each $f \in C_1$ is associated with a membership function*

$$\mu_f : C_2 \rightarrow [0, 1], \quad g \mapsto \mu_f(g). \quad (1)$$

This can be equivalently viewed as a fuzzy relation operator $\mathcal{T}(f, g) := \mu_f(g)$ [32].

Definition 2 (Crisp idealized agent and safety envelope). *If for all $f \in C_1, g \in C_2$, we have $\mu_f(g) \in \{0, 1\}$, then $\mathcal{T}$ is called a crisp idealized*

agent. Fix a crisp idealized agent and denote it by the safety envelope $\mathcal{T}_0$. An ideal agent $\mathcal{T}$ is said to be constrained by $\mathcal{T}_0$ in safety if

$$0 \leq \mathcal{T}(f, g) \leq \mathcal{T}_0(f, g), \quad \forall f, g. \quad (2)$$

In this case, each feasible transition of $\mathcal{T}$ is limited to the reachable edges allowed by $\mathcal{T}_0$, so execution proceeds only within the safety envelope.

Definition 3 (Iterated agent and search trajectory). When $C_2 \subseteq C_1$, $\mathcal{T}$ is called an iterated agent. A finite sequence $ST = (f^{(0)}, f^{(1)}, \dots, f^{(n)})$ is called a search trajectory from $f^{(0)}$ to $f^{(n)}$ (of length n) if for all $i = 0, \dots, n-1$,

$$\mu_{f^{(i)}}(f^{(i+1)}) > 0. \quad (3)$$

2.2 Coverage generating function

To uniformly measure the reachability of iterated agents across problem difficulties, we introduce a coverage generating function based on a continuation parameter without aftereffects.

Definition 4 (Coverage generating function and continuation parameter). Let a single parameter $p \in [0, 1]$ denote the weight for continuing iteration (the continuation parameter), understood as a scalar weight assigned to trajectory length; it is not a probability but a bookkeeping factor. Thus a trajectory of length n is assigned weight p^n . Define the coverage generating function from f to g as

$$P_{f,g}(p) := \sum_{n=0}^{\infty} \sum_{\substack{ST: f^{(0)}=f, \\ f^{(n)}=g}} p^n \prod_{i=0}^{n-1} \mu_{f^{(i)}}(f^{(i+1)}), \quad (4)$$

where for $n = 0$ the inner product is the empty product, and this term exists and contributes 1 if and only if $f = g$.

Remark 1 (Operator viewpoint and spectral radius). If C_1, C_2 are countable, let the matrix

(kernel) M satisfy $M_{f,g} = \mathcal{T}(f, g)$. Then

$$P(p) = \sum_{n \geq 0} p^n M^n, \quad (5)$$

whose (f, g) entry is exactly (4). When $p\rho(M) < 1$ (with $\rho(M)$ the spectral radius), the series converges in the operator sense and

$$P(p) = (I - pM)^{-1}. \quad (6)$$

In general, $P_{f,g}(p)$ is a power series (generating function) with nonnegative coefficients, monotone nondecreasing in p . The boundary value satisfies $P_{f,g}(0) = \mathbf{1}\{f = g\}$.

Notation 2 (Continuation-induced search). Given an iterated agent $\mathcal{T}$ and parameter $p \in [0, 1]$, define the continuation-induced ideal agent

$$\mathcal{T}^{(p)}(f, g) := \min(1, P_{f,g}(p)), \quad (7)$$

which can be viewed as an agent formed by multi-round iterative feedback through $\mathcal{T}$. We will refer to $\mathcal{T}^{(p)}$ as the search or the continuation-induced (search) agent. This clipping induces a $[0, 1]$ -valued membership, not a probability measure; alternative normalizations are possible, but we adopt unit clipping for threshold analysis.

2.3 Geometry under the crisp safety envelope

On the directed graph induced by the crisp idealized agent (safety envelope), natural geometric quantities can be defined.

Definition 5 (Generating function under the crisp idealized agent and path counting). If $\mathcal{T}$ is a crisp idealized agent, then

$$P_{f,g}^{\text{ideal}}(p) = \sum_{n=0}^{\infty} N_n(f, g) p^n, \quad (8)$$

where $N_n(f, g)$ is the number of reachable paths of length n from f to g .

Definition 6 (Shortest distance). *On the directed graph induced by the crisp idealized agent, define the shortest distance*

$$d_0(f, g) := \inf \{n \in \mathbb{N} : N_n(f, g) \geq 1\}, \quad (9)$$

and set $d_0(f, g) = +\infty$ if g is unreachable from f .

Lemma 1 (Shortest distance and low-order terms of the generating function). *If $d_0(f, g) < \infty$, then*

$$\lim_{p \rightarrow 0^+} \frac{P_{f,g}^{\text{ideal}}(p)}{p^{d_0(f,g)}} = N_{d_0(f,g)}(f, g) \in \mathbb{N} \setminus \{0\}. \quad (10)$$

Remark 2 (Insufficient search under small continuation parameter). *When the continuation parameter p is small, search is in an insufficient expansion regime (particularly when many edges have low membership or the graph is approximately unidirectional). By Lemma 1, the shortest-path term dominates the behavior of $P_{f,g}(p)$, so the shortest distance d_0 and the corresponding number of shortest paths N_{d_0} control the early reachable set.*

2.4 Critical parameter and coverage index

We characterize the reachability difficulty from f to g by the critical value at which the generating function reaches the unit threshold.

Definition 7 (Coverage index and critical parameter). *Define*

$$p_c(f, g) := \inf \{p \in [0, 1] : P_{f,g}^{\text{ideal}}(p) \geq 1\}, \quad (11)$$

and set $p_c(f, g) = 1$ if the set is empty (unreachable). *Define the coverage index*

$$R_c(f, g) := 1 - p_c(f, g) \in [0, 1]. \quad (12)$$

A larger $R_c(f, g)$ indicates reaching unit coverage at smaller weights, i.e., easier to reach. Increasing the number of reachable paths or shortening path length both increase R_c .

Definition 8 (Intermediate node). *A node h is called an intermediate node for (f, g) if at least one reachable path from f to g passes through h .*

Proposition 1 (Transitivity of the coverage index). *If h is an intermediate node for (f, g) , then for all $p \in [0, 1]$,*

$$P_{f,g}^{\text{ideal}}(p) \geq P_{f,h}^{\text{ideal}}(p) \cdot P_{h,g}^{\text{ideal}}(p). \quad (13)$$

Therefore,

$$\begin{aligned} p_c(f, g) &\leq \max(p_c(f, h), p_c(h, g)), \\ R_c(f, g) &\geq \min(R_c(f, h), R_c(h, g)). \end{aligned} \quad (14)$$

If h is not an intermediate node for (f, g) , then at least one of $f \rightarrow h$ or $h \rightarrow g$ is unreachable; in this case Eq. 14 still holds.

Definition 9 (Epoch and the lower-bound meaning of shortest distance). *An epoch refers to one expansion step that applies the crisp safety envelope $\mathcal{T}_0$ to all outputs from the previous round and performs set-wise deduplication. Clearly, starting at f , reaching g requires at least $d_0(f, g)$ epochs.*

2.5 Threshold hypotheses and testable inequalities

We provide an empirically common and testable hypothesis for LLM-induced approximately unidirectional search.

Assumption 1 (Approximate structure of LLM induced agent). *Empirically, for iterated agents constructed by LLMs:*

1. *Closed walks (nonzero-length paths whose start and end coincide) are rare; the crisp envelope is approximately unidirectional, so $P_{f,g}^{\text{ideal}}(p)$ has a much simpler denominator compared to its complete numerator.*
2. *Overly long trajectories are relatively rare; equivalently, the generating function is essentially dominated by its low-order terms.*

3 Instantiation and Experiments

This section provides two minimal, reproducible instantiation that aligns one-to-one with the formal objects above.

3.1 Majority-vote walking agent

On a two-dimensional grid, we construct an ideal agent and its corresponding crisp agent induced by LLM majority vote, and test the observable hypotheses and inferences in Assumption 1.

To make abstract concepts concrete, we give a minimal construction following objects-mappings-geometry.

Task space and transition syntax: Consider a two-dimensional grid $G_N := \{0, \dots, N-1\}^2$, hence $C_1 = C_2 = G_N$. Given a start-target pair $(f, t) \in G_N \times G_N$, we allow unit-step transitions up, down, left, and right, staying within the board as the syntax of feasible local transitions.

Ideal agent induced by LLMs: For any $f \in G_N$ and fixed target t , we query a given LLM ($\mathcal{L}$) under prompts that combine constraints and goals to output the next position g ; the complete prompt is in Appendix App. A. This yields an empirical decision distribution $\hat{P}_f^{(\mathcal{L}, t)}(g)$ (approximated via multiple samples) for from f to g .

For each f , independently sample m times and take the mode g^* of $\hat{P}_f^{(\mathcal{L}, t)}(g)$. If its frequency exceeds $m/2$, define the agent

$$\mu_f^{(\mathcal{L}, t)}(g) := \mathbf{1}\{g = g^*\};$$

otherwise regard it as no strict majority, i.e., $\mu_f^{(\mathcal{L}, \sqcup)}(g) = 0$.

Aggregate majority-vote results from n different models uniformly to obtain the ideal agent

$$\mu_f^{(t)}(g) := \frac{1}{n} \sum_{\mathcal{L}} \mu_f^{(\mathcal{L}, t)}(g) \in [0, 1].$$

Crisp agent (safety envelope) and induced graph: Binarize the support of the ideal agent to obtain a crisp idealized agent

$$\mu_f^{0, (t)}(g) := \mathbf{1}\{\mu_f^{(t)}(g) > 0\} \in \{0, 1\},$$

and define the directed graph $\mathcal{G}_t$: nodes are G_N , and if $\mu_f^{0, (t)}(g) = 1$ draw an edge $f \rightarrow g$. This construction is the directed graph induced by the ideal/crisp idealized agents.

The construction corresponds respectively to: $C_1 = C_2 = G_N$ as the search space; $\mu^{(t)}$ as the iterated ideal agent; $\mu^{0, (t)}$ as its safety envelope; $\mathcal{G}_t$ as the geometry induced by the safety envelope.

Experimental setup, model list, grid sizes and target points, and full prompts are provided in Appendix App. A. Under this setup, we construct the ideal agent $\mu^{(t)}$ for each target t , and obtain the crisp agent and the induced directed graph $\mathcal{G}_t$ from its support. Figure 1 shows a representative case ($N = 5$, $t = (3, 4)$) of $\mathcal{G}_t$: under semantic constraints, the graph exhibits a unidirectional structure (strictly decreasing Manhattan distance to the target) with anisotropic preferences over allowed edges, consistent with the finite terms premise in Assumption 1.

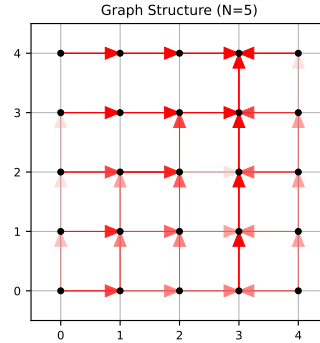


Figure 1: Visualization of $\mathcal{G}_{(3,4)}$ on a 5×5 grid. Red arrows denote reachable directed edges, and transparency encodes the membership on the ideal agent $\mu^{(t)}$. The graph is unidirectional, strictly decreasing the Manhattan distance to the target.

3.2 IdeaSearch agent

To further test the observability of Assumption 1 in real complex tasks, we instantiate an iterated agent and its safety envelope in a symbolic regression setting. Specifically, we adopt the IdeaSearchFitter framework proposed by Song et al., configure hyperparameters under "expert mode" and set `example_num` to 1 [17]. Under this configuration, the search space $C_1 = C_2$ is the numexpr hypothesis space; each state f represents a candidate expression (merging equal expressions(ideas) by string-level equality), and $f \rightarrow g$ denotes the next-step update retained by the framework under constraints and scoring filters.

On the PMLB dataset `strogatz_bacres2` [33], we conduct 10 independent runs, and aggregate 34,651 input-output pairs $f \rightarrow g$. To avoid misclassifying occasional random temperature perturbations as input-output edges of an agent constructed by large language models (LLMs), we tally all observed input-output pairs and introduce a threshold parameter p_0 for binarization (note: p_0 is unrelated to the continuation parameter p in Section 2). Let $n_{\text{in}}(f)$ be the number of occurrences where f appears as the input in all pairs, and let $n_{\text{pair}}(f, g)$ be the number of occurrences of the pair $f \rightarrow g$; define the relative frequency $r(f, g) := n_{\text{pair}}(f, g)/n_{\text{in}}(f)$. Based on this, we construct the corresponding crisp agent (safety envelope)

$$\mathcal{T}_{\text{ideasearch}}^{0, (p_0)}(f, g) = \begin{cases} 0, & n_{\text{in}}(f) \leq 1 \\ 0, & r(f, g) \leq p_0 \\ 1, & \text{otherwise.} \end{cases} \quad (15)$$

This binarization ensures that an input is included in the envelope only when it has a statistically stable output, thereby filtering out occasional input-output pairs as much as possible.

To test approximate unidirectionality, we contract the few nodes that participate in closed walks into strongly connected components, yielding a supernode graph; Figure 2 shows the directed graph under this contraction, and Figure 3 shows the internal landscape of one such supernode.

Statistics indicate that when $p_0 = 0.1$, the total size of all supernodes accounts for about 2% of all nodes, indicating that closed walks are rare and the induced graph is overall close to acyclic, consistent with the rareness of closed walks in Assumption 1.

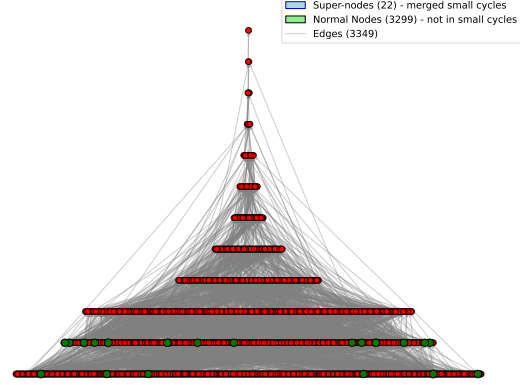


Figure 2: Directed graph after supernode contraction ($p_0=0.1$). Red denotes ordinary nodes, totaling 3299; green denotes supernodes, totaling 22; in sum there are 3349 distinct input-output pairs at the string level. Overall, closed walks are rare, and the graph structure is approximately acyclic.

Furthermore, we examine the effect of the threshold p_0 on the graph structure. Define

$$\eta(p_0) := \frac{\text{number of nodes contained in supernodes}}{\text{total number of nodes in the graph}},$$

as an observable measure of the proportion of closed walks. Figure 4 shows η as a function of p_0 : as p_0 increases, low-frequency edges (more likely originating from random temperature perturbations) are systematically pruned; the input-output pairs of the LLM-based reasoning agent gradually dominate the structure, and the number of closed walks drops rapidly. This continuous process exhibits a transition from "random guessing" to "stable reasoning", and macroscopically yields an approximately unidirectional

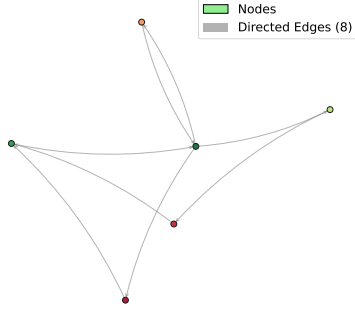


Figure 3: Internal landscape of a supernode ($p_0=0.1$). This supernode contains 6 nodes and 8 internal edges and is strongly connected internally; node color encodes goodness of fit on the dataset (green indicates a lower chi-square goodness-of-fit statistic).

directed graph (approximately acyclic), consistent with the approximate unidirectionality in Assumption 1.

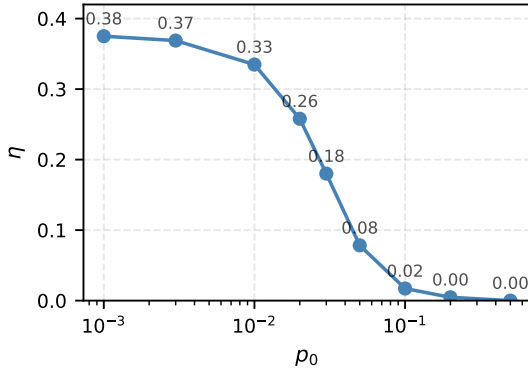


Figure 4: Closed-walk proportion η versus threshold p_0 . η is defined as the ratio of the number of nodes contained in supernodes to the total number of nodes; as p_0 increases, low-frequency edges are pruned, and η rapidly decreases toward zero, indicating the emergence of an approximately unidirectional structure.

This instantiation corresponds, at the formal level, to the objects in Section 2: $C_1 = C_2$ is

the search space of the iterated agent; $\mathcal{T}_{\text{ideasearch}}^{0,(p_0)}$ is the safety envelope obtained via frequency-threshold binarization; the graph induced by the envelope, after contracting closed walks, exhibits an approximately acyclic geometric structure, providing external evidence for the empirical hypothesis of "rare closed walks and rare overly long trajectories".

4 Conclusion

This paper proposes a compact formal theory to unify the description and measurement of LLM-assisted iterative search. The core is to represent agents as fuzzy relation operators μ on inputs and outputs; aggregate the contributions of all reachable paths via the coverage generating function $P_{f,g}(p)$; and characterize reachability difficulty by the critical parameter $p_c(f, g)$ (with coverage index $R_c = 1 - p_c$). On the graph induced by the crisp agent (safety envelope), the shortest distance d_0 and the number of shortest paths N_{d_0} provide a geometric interpretation of the search process. The low-order dominance seen in iterated agents constructed by LLMs supplies a computable, testable, and model-agnostic language for how to measure. The instantiation shows that the safety envelope induced by LLMs yields a unidirectional structure.

The theory offers testable predictions and quantifiable trade-offs. First, the safety-reachability trade-off can be quantified via μ and $P_{f,g}(p)$: tightening the safety envelope (reducing reachable edges) decreases path diversity, increases d_0 , lowers $P_{f,g}(p)$, raises p_c , and reduces R_c ; relaxing constraints has the opposite effect, but must respect safety [27]. Second, the multiplicative lower bound and the propagation inequality for critical parameters in the presence of intermediate nodes provide possible guidance for constructing intermediate waypoints to reduce overall difficulty [31]. Practically, this theory provides quantitative guidelines for agent design and training on complex tasks. For example, evaluation and training signals can be designed

around p_c/R_c , d_0 , and N_{d_0} so that reachability difficulty and safety compliance are simultaneous optimization goals; conversely, the theory can guide the design of agents for executing complex tasks: in early stages, prefer models with stricter safety envelopes to shrink the envelope and ensure compliance and stability; once the running epochs approach the reachable limit, gradually introduce looser safety envelopes to increase the coverage index [14, 17].

This paper presents an implementable theory; detailed experimental validation is left to future work, including further testing of the effectiveness of these measures and connecting the above indicators to reinforcement learning rewards and training procedures. Overall, by formalizing agents as computable fuzzy relation operators and unifying safety and reachability under the same measurement, the theory serves as a foundational tool for understanding and improving LLM-driven long-horizon search and complex-task agents.

References

- [1] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL <https://arxiv.org/abs/2210.03629>.
- [2] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023. URL <https://arxiv.org/abs/2305.10601>.
- [3] Jason et. al. Wei. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022. URL <https://papers.baulab.info/papers/also/Wei-2022b.pdf>.
- [4] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023. URL <https://arxiv.org/abs/2203.11171>.
- [5] Maciej et. al. Besta. Graph of thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17682–17690, Mar. 2024. doi: 10.1609/aaai.v38i16.29720. URL <https://ojs.aaai.org/index.php/AAAI/article/view/29720>.
- [6] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models, 2023. URL <https://arxiv.org/abs/2305.04091>.
- [7] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023. URL <https://arxiv.org/abs/2302.04761>.
- [8] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL <https://arxiv.org/abs/2305.16291>.
- [9] Wenhui Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks, 2023. URL <https://arxiv.org/abs/2211.12588>.
- [10] Luyu et. al. Gao. PAL: Program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202

- of *Proceedings of Machine Learning Research*, pages 10764–10799. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/gao23f.html>.
- [11] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control, 2023. URL <https://arxiv.org/abs/2209.07753>.
 - [12] Daniel J et. al. Mankowitz. Faster sorting algorithms discovered using deep reinforcement learning. *Nature*, 618(7964):257–263, 2023. URL <https://www.nature.com/articles/s41586-023-06004-9>.
 - [13] Bernardino et. al. Romera-Paredes. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. URL <https://www.nature.com/articles/s41586-023-06924-6>.
 - [14] Alexander Novikov et. al. Alphaevolve: A coding agent for scientific and algorithmic discovery, 2025. URL <https://arxiv.org/abs/2506.13131>.
 - [15] Zhuo-Yang Song, Tong-Zhi Yang, Qing-Hong Cao, Ming xing Luo, and Hua Xing Zhu. Explainable ai-assisted optimization for feynman integral reduction, 2025. URL <https://arxiv.org/abs/2502.09544>.
 - [16] Qing-Hong Cao et. al. Quantum state preparation via large-language-model-driven evolution, 2025. URL <https://arxiv.org/abs/2505.06347>.
 - [17] Zhuo-Yang Song et. al. Iterated agent for symbolic regression, 2025. URL <https://arxiv.org/abs/2510.08317>.
 - [18] Tad Hogg, Bernardo A. Huberman, and Colin P. Williams. Phase transitions and the search problem. *Artificial Intelligence*, 81(1):1–15, 1996. ISSN 0004-3702. doi: [https://doi.org/10.1016/0004-3702\(95\)00044-5](https://doi.org/10.1016/0004-3702(95)00044-5). URL <https://www.sciencedirect.com/science/article/pii/0004370295000445>. Frontiers in Problem Solving: Phase Transitions and Complexity.
 - [19] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. doi: 10.1109/TPAMI.2013.50. URL <https://ieeexplore.ieee.org/abstract/document/6472238/>.
 - [20] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning, 2020. URL <https://arxiv.org/abs/2012.13255>.
 - [21] Zhuo-Yang Song, Zeyu Li, Qing-Hong Cao, Ming xing Luo, and Hua Xing Zhu. Bridging the dimensional chasm: Uncover layer-wise dimensional reduction in transformers through token correlation, 2025. URL <https://arxiv.org/abs/2503.22547>.
 - [22] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 2002. URL <https://ieeexplore.ieee.org/abstract/document/585893>.
 - [23] Eitan Altman. *Constrained Markov decision processes*. Routledge, 2021. URL <https://doi.org/10.1201/9781315140223>.
 - [24] Long et. al. Ouyang. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc., 2022. URL <https://dl.acm.org/doi/abs/10.5555/3600270.3602281>.

- [25] Yuntao Bai et. al. Constitutional ai: Harmlessness from ai feedback, 2022. URL <https://arxiv.org/abs/2212.08073>.
- [26] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017. URL <https://papers.baulab.info/papers/also/Christiano-2017.pdf>.
- [27] Geoffrey Irving, Paul Christiano, and Dario Amodei. Ai safety via debate, 2018. URL <https://arxiv.org/abs/1805.00899>.
- [28] Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. Prompting is programming: A query language for large language models. *Proc. ACM Program. Lang.*, 7(PLDI), June 2023. doi: 10.1145/3591300. URL <https://doi.org/10.1145/3591300>.
- [29] Traian Rebedea, Razvan Dinu, Makesh Sreedhar, Christopher Parisien, and Jonathan Cohen. Nemo guardrails: A toolkit for controllable and safe llm applications with programmable rails, 2023. URL <https://arxiv.org/abs/2310.10501>.
- [30] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998. URL <http://unbox.org/wisp/doc/sutton98.pdf>.
- [31] Yoshua et. al. Bengio. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. URL <https://doi.org/10.1145/1553374.1553380>.
- [32] Radim Belohlavek. *Fuzzy relational systems: foundations and principles*, volume 20. Springer Science & Business Media, 2012. URL <https://link.springer.com/book/10.1007/978-1-4615-0633-1>.
- [33] Randal S Olson, William La Cava, Patryk Orzechowski, Ryan J Urbanowicz, and Jason H Moore. Pmlb: a large benchmark suite for machine learning evaluation and comparison. *BioData mining*, 10(1):36, 2017. URL <https://link.springer.com/article/10.1186/s13040-017-0154-4>.

A Experimental Setup and Prompts

For reproducibility, this appendix provides detailed experimental setup and prompts.

A.1 Grid sizes and target points

number	N	t
1	3	(1, 2)
2	5	(3, 4)
3	8	(6, 7)

Table 1: Three grid sizes and corresponding target points t .

A.2 Model list and sampling settings

Model set: gpt-5-mini, gpt-5, qwen3, qwen-plus, gemini-2.5-flash, deepseek-v3, grok-4, doubao.

Number of samples: For each input position f under a given target t , independently sample $m = 5$ times.

A.3 Prompts

The prompt used to drive each model to output the next position is as follows:

```

1 # example input:
2 # N = 5
3 # f = (0, 0)

```

```

4 # t = (3, 4)
5
6 prompt = f"""
7     You are an ant on a {N}x{N}
      grid. Your current
      position is {list(f)},
      and the target position
      (food) is {list(t)}.
8     You can move up, down, left,
      or right by one unit,
      but cannot move outside
      the grid.
9     Based on the current state,
      decide the next position
      to move to, and return
      the result in JSON
      format with the field "
      next_position".
10
11     Note:
12     - Only choose legal move
      positions
13     - Choose the position that
      gets you closer to the
      target
14     - Example return format: {{"
      next_position": [x_g,
      y_g]}}
15     - Write down the json only,
      no other text
16     """
17
18 messages = [
19     {"role": "system", "content":
      "You are a helpful
      assistant that helps
      people find information.
      "},
20     {"role": "user", "content":
      prompt}
21 ]

```