DECOMPOSING EXTRAPOLATIVE PROBLEM SOLVING: SPATIAL TRANSFER AND LENGTH SCALING WITH MAP WORLDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Someone who learns to walk shortest paths in New York can, upon receiving a map of Paris, immediately apply the same rule to navigate, despite never practicing there. This ability to recombine known rules to solve novel problems exemplifies compositional generalization (CG), a hallmark of human cognition. Yet our understanding of what drives the success or failure of such extrapolative problem solving, particularly the roles of training data properties and optimization paradigms, remains limited. In this work, we introduce a controlled map-navigation testbed that cleanly separates two dimensions of CG: ***spatial transfer*** (systematicity across environments) and ***length scaling*** (productivity along problem difficulty). Through quantitative experiments, we show that transfer is enabled by sufficient distinct questions with high coverage and modest diversity, while scaling critically depends on exposure to neighboring-but-longer examples. Finally, we find that reinforcement learning (RL) stabilizes optimization but does not surpass the ceiling set by supervised fine-tuning (SFT). Together, these results provide principled insights into how data properties and training paradigms shape extrapolative problem solving.

1 INTRODUCTION

The field is currently excited by strong evidence of LLMs' ability to tackle truly novel problems—solving IMO 2025 questions (Huang & Yang, 2025) and discovering algorithms that surpass state-of-the-art solutions (Novikov et al., 2025). To solve such novel questions, a model must compose the words and rules learned, echoing a fundamental hallmark of human cognition: compositional generalization (CG)—the ability to make "*infinite use of finite means*" (Chomsky, 1957).

Despite this promise, our understanding of extrapolative and compositional problem solving remains limited. Since it is hard to cleanly separate "novel" problems in natural language, prior work has turned to synthetic challenges/puzzles to test whether foundation models can solve problems not present in training (Ramesh et al., 2023; Xu et al., 2024; Dziri et al., 2023). These studies reached mixed conclusions: sometimes models succeed, sometimes they fail. We view this inconsistency as evidence that LLMs generalize along some dimensions more readily than others. This motivates our work. Rather than asking for a brute-force CG-or-not answer, we aim to **decompose "novel problem solving" into concrete, well-defined extrapolation dimensions**, and study **how data properties and training paradigms drive success or failure along each.**

Concretely, we focus on two fundamental dimensions of CG (Sinha et al., 2024), while restricting ourselves to a single problem class to avoid entanglement: (1) ***Transfer (systematicity in CG)***: the ability to solve the same class of problems in entirely new environments. For example, a model trained on English problems should also succeed in German or French; in mathematics, this corresponds to learning induction in algebra and aplying the same inductive structure in number theory (e.g., divisibility proofs) or combinatorics (e.g., binomial identities); (2) ***Scaling (productivity in CG)***: the ability to solve harder (e.g., longer) problems after having seen simpler ones. For example, once a model has learned induction, it should then be able to solve problems requiring induction *recursively*.

We use navigation tasks on 2D sparse grid maps as our testbed. This setup offers two key advantages. (1) **Orthogonal factors:** path data separates cleanly into spatial (where the path is) and length (how long it is) components, enabling controlled measurement of each dimension of generalization. This is far harder in natural language or arithmetic, where vocabulary and sequence length are deeply entangled. (2) **True systematicity:** When we speak of "infinite use of finite means" in linguistics, we expect rules learned in one domain to apply even to a disjoint one (e.g., transferring from English to German). In natural data, however, primitives are embedded in unknown high-dimensional spaces, making it nearly impossible to enforce completely disjoint test domains, or design cross-lingual or cross-topic evaluations. Grid maps, by contrast, allow us to build arbitrarily many disjoint worlds, providing a clean test of whether rules generalize to entirely novel primitives. Note that, unlike graph-based generalization tests (Cai et al., 2025; Zhang et al., 2024) that feed the full graph structure upfront into pretrained models and reduce the task to explicit rule application, our setup uses map data to simulate a language-like world. The model must infer the map's structure from its training corpus of paths, much as language models learn word relations from text during training.

In the remainder of this paper, we examine how data selection and training paradigms (i.e., SFT and RL) influence the emergence of generalization along the two dimensions (transfer and scaling). We defer detailed discussions of research gaps and motivations to the beginning of each section, and related work can be found in Section B. Our main conclusions are: (1) problem-solving transfer is primarily enabled by distinct path prompts with high coverage and modest diversity (Section 3); (2) length scaling critically depends on exposure to neighboring-but-longer examples, and can only be locally mitigated regardless of the training paradigm (Section 4); and (3) RL effectively stabilizes optimization but does not provide additional gains beyond the ceiling established by SFT (Section 5).

2 Preliminaries and Experimental Setup

Spatial Transfer (Systematicity). Following the classic definition of systematicity in compositional generalization (Wiedemer et al., 2023b; Fu et al., 2024), we define it as the ability to correctly apply a known rule to new compositions of primitives that lie outside the training support. Formally, let $G = (V, A)$ be a *sparse grid map* (i.e., with edges blocked) with node set V and adjacency A. A mobility *rule* $f(i, j \mid G)$ returns a mobility path from node i to node j under G. The *training support* is the set of ordered start–end node pairs used in training, $\text{supp}(\mathcal{D}_{\text{train}}) \subseteq V \times V \setminus \{(i, i)\}$. We **evaluate systematicity of a model θ trained on $\mathcal{D}_{\text{train}}$ as its performance in applying rule f to novel ordered pairs** $(i, j) \sim \mathcal{D}_{\text{test}}$, where all node pairs in the test set are disjoint from those in training, i.e., $\text{supp}(\mathcal{D}_{\text{test}}) \cap \text{supp}(\mathcal{D}_{\text{train}}) = \emptyset$. In our case, $\mathcal{D}_{\text{test}}$ is drawn from a disjoint novel map $\hat{G} = (\hat{V}, \hat{A})$ with $\hat{V} \cap V = \emptyset$ and $\hat{A} \neq A$, i.e., irrelevant to G in nodes, edges, sparsity or size.

Such a truly disjoint test space is rarely achievable in natural language, where systematicity is often evaluated by holding out primitives within the same domain. This can yield overly optimistic estimates, since semantically similar primitives (e.g., "run" vs. "walk") may lie close in embedding space. Our spatial setup therefore provides a more faithful measure of systematic generalization.

Length scaling (Productivity). Problem-solving scaling corresponds to productivity (or length generalization) in CG (Sinha et al., 2024; Cai et al., 2025). Within the same notation, it can be viewed as a constrained form of Systematicity, where novelty is enforced along the path-length axis. Let $l(\mathcal{D})$ denote the set of path lengths for the mobility pairs in dataset $\mathcal{D}$. Then, in addition to the disjointness condition $\text{supp}(\mathcal{D}_{\text{test}}) \cap \text{supp}(\mathcal{D}_{\text{train}}) = \emptyset$, productivity further requires $\max l(\mathcal{D}_{\text{train}}) \leq \min l(\mathcal{D}_{\text{test}})$, i.e., all test pairs must involve strictly longer paths than any seen in training.

Metric. Let $\hat{f}_\theta(i, j \mid G)$ denote the path predicted by the model θ. We measure extrapolative problem-solving performance using the *success rate (SR)*:

$$\text{SR} = \Pr_{(i,j)\sim\mathcal{D}_{\text{test}}} \left[\hat{f}_\theta(i, j \mid G) = f(i, j \mid G) \right] \tag{1}$$

In our experiments, we adopt the shortest-path rule for f, which makes path length precisely controllable.[1] Our goal is to study the properties of the data and training paradigm rather than the inherent

[1] Many other common mobility rules, such as DFS, yield unconstrained lengths.

learnability of the task itself, and shortest-path is a canonical pathfinding problem that is theoretically regarded leanable by language models (Cohen et al., 2025; Dai et al., 2024). In shortest-path, $f(i, j \mid G)$ may return a *set of valid paths* whenever multiple paths exist between i and j. During evaluation in Equation (1), we deem $\hat{f}_\theta(i, j \mid G)$ successful if it belongs to the set $f(i, j \mid G)$.

Empirical Setup. We trained 8-layer, 8-head Transformer models from scratch following the LLaMA architecture (AI@Meta, 2024), which employs Rotary Positional Embeddings (RoPE) (Su et al., 2021) for position encoding. The models were pretrained on random-walk paths over all maps (G and $\hat{G}$), simulating the autoregressive pretraining phase of large language models (LLMs). This pretraining enables the model to acquire the primitives (nodes) and their semantics, defined by their adjacency relationships. To prevent interference with downstream mobility-rule learning tasks, we bias the pretraining distribution by constraining random-walk paths to have a minimum length substantially longer than any path in the fine-tuning distribution. (We also validate this non-interference in Section C.3.) This mirrors common practice in LLM pretraining, where models are exposed to much longer sequences than those used in fine-tuning or evaluation.

For evaluation, we fine-tune the models on shortest paths on the training map $G = (V, A)$. We split the node set V into training and test regions: the training region contains 80% of the nodes (from which a subset of nodes V_{train} used to form $\mathcal{D}_{\text{train}}$ is sampled) and the remaining 20% for length scaling evaluation. We test spatial transfer on different disjoint test maps $\hat{G} = (\hat{V}, \hat{A})$.

Training paradigms and data format. We study two training paradigms: supervised fine-tuning (SFT) and reinforcement learning (RL).

For SFT, each training sample is represented as a sequence of the form `<s> i j : i E S E E ... N E S W W j </s>`, where `i` and `j` denote the start and end nodes, `<s>` and `</s>` are special tokens, and the path is encoded as a sequence of movement directions (E, W, N, S). Using directions instead of node indices prevents the model from trivially memorizing n-gram sequences of node identifiers. The prompt prefix `<s> i j :`, which we refer to as the *question* such that the path itself is the *answer*, is excluded from the loss during SFT. At test time, we feed this prompt to the model and evaluate the generated continuation, i.e., asking the question "what is the shortest path from i to j?".

Our path setup also naturally lends itself to RL for two reasons. (1) The shortest paths are inherently verifiable: a generated sequence either forms a valid shortest path or not, allowing us to define a binary reward without additional heuristics; (2) Although the model is not explicitly designed to "think", the path-generation process itself resembles a step-by-step reasoning procedure, making RL a natural training paradigm for this setting. We adopt the Dr.GRPO (Liu et al., 2025) algorithm (an unbiased variant of GRPO and the de facto standard in recent implementations of RL with LLMs), with a binary reward of 1 if the generated sequence forms a valid shortest path between `i` and `j` and 0 otherwise. The model is trained on the same prompt prefix `<s> i j :`, and we vary the number of rollouts per prompt (4, 8, and 16) during training.

3 Effects of Data Selection on Problem-Solving Transfer

We start by analyzing the effects of data selection for the classic SFT paradigm. A model exhibits systematic generalization if it can solve problems built from disjoint primitives. In our setting, this means generating valid mobility paths in a map never seen during training. We ask here: how to allocate a fixed training budget of records to best support such transfer? Should the budget go toward collecting diverse answers for each question, or toward covering as many distinct questions as possible (Section 3.1)? And if more questions are preferable, what kinds of questions should be prioritized (Section 3.2)?

3.1 More questions vs. more answers

In many domains of current interest (e.g., mathematics, program synthesis, navigation), a single problem naturally admits multiple valid solutions. This makes budget allocation an important consideration in SFT, especially since collecting high-quality solutions often requires significant efforts (Cobbe et al., 2021; Hendrycks et al., 2021). The question is not trivial: the model requires

sufficiently diverse questions to capture the underlying rules; but if each problem is paired with only one solution, the model may overfit to surface patterns rather than acquiring the underlying rule, potentially harming transfer. We therefore investigate whether allocating budget to solution diversity improves extrapolation, or if prioritizing distinct questions is more effective.

Experiment Design. We consider five training budgets $B \in \{5\%, 10\%, 20\%, 60\%, 80\%\}$ of the total possible training records, where the total is determined by the maximum number of directed start–end pairs within the designated training region (80% of the nodes in the training map G, as illustrated in Section 2). We use a 50×40 sparse grid map with $|V| = 2000$ nodes as G. For each budget, we vary the number of distinct questions (unique start–end pairs) and the average number of answers per question (distinct valid shortest paths between each node pair), subject to the constraint $N_{\text{questions}} \times N_{\text{answers per question}} = B$.[2] Problem-solving transfer is measured by the success rate (SR, Equation (1)) on disjoint test maps, restricted to paths within the training length (i.e., excluding length-scaling). We evaluate on three spatially disjoint maps of varying size (30×30, 40×40, 50×50), sparsity (25%–75%), and adjacency. We report the average SR across them.

Unique questions drive transfer. We first confirm that the model can spatial transfer: even when trained on a limited subset of the training map (e.g., 20% of the 80% training region, i.e., 16% of the full training map), it achieves an average success rate of 94% over three spatially disjoint test maps. As shown in Figure 1, under a fixed budget, training on more distinct questions consistently improves transfer, even at the cost of reducing answer diversity. For example, with a low budget, allocating all data to distinct questions with one solution each yields an SR of 94%, compared to only 82% when using fewer questions but 32 solutions per question. This pattern holds across all budget levels, showing that unique questions provide higher marginal value than unique solutions. (This does not imply that solutions are unimportant; rather, one high-quality solution per question appears sufficient under SFT.) However, the benefit of adding more questions quickly saturates: at very high budgets, training with hundreds of thousands of additional questions produces almost no gain over low budgets.

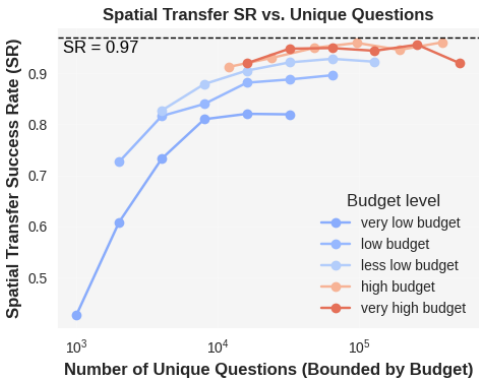


Figure 1: Spatial transfer success rate (SR) improves consistently with more budget allocated to unique questions (log scale). Curves show five budget levels (very low to very high: 5%, 10%, 20%, 60%, 80% of all possible records). Dashed line marks the SR ceil.

Takeaway 1 (Data efficiency guideline under SFT): Spatial transfer (systematic CG) is best supported by covering as many distinct questions as possible. This yields the most effective use of the training budget, especially when collecting solutions is expensive.

3.2 Coverage vs. Diversity in Questions

If more distinct questions are more valuable, a second question arises: **which kinds of questions should be prioritized?** Prior work on CG rarely considered solution diversity, but it has long emphasized the importance of training distribution properties such as **coverage** and **diversity**. These factors have been discussed since early seq2seq RNN and CNN models Lake & Baroni (2018); Bahdanau et al. (2018); Keysers et al. (2019), and continue to play a central role for decoder-only Transformers Lippl & Stachenfeld; Ahuja & Mansouri (2024); Levy et al. (2023). However, while commonly believed to matter, their precise role remains unclear: are higher coverage and diversity always beneficial? How do they interact? In this section, we empirically vary these two classic factors in a controlled and decoupled way to measure their effect on systematic transfer.

We begin by defining these notions in our setting. Following Chang et al. (2025), we define coverage and diversity in questions over node primitives.

[2] If a question admits fewer distinct solutions than required, we include all available solutions and allocate the remaining budget to other questions without repetition.

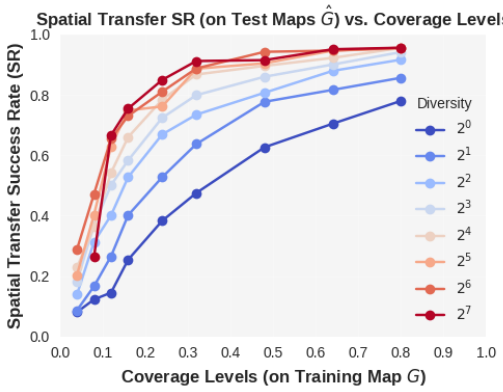


Figure 2: Spatial transfer success rate (SR) measured on the disjoint maps as the nodes coverage ratio in question in the training map increases. Each curve corresponds to a fixed diversity level.

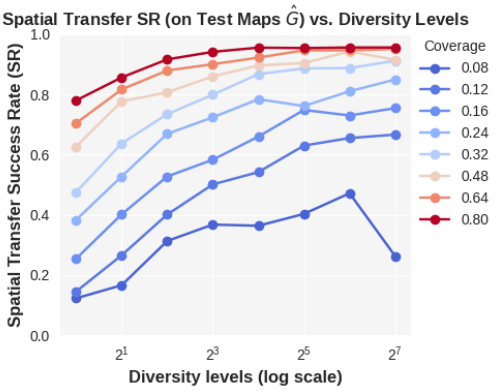


Figure 3: Spatial transfer success rate (SR) measured on the disjoint maps as the node pair composition diversity level increases. Each curve corresponds to a fixed coverage ratio.

(Local) Coverage. Coverage measures the fraction of unique nodes (i.e., primitives) in the *local training map* $G = (V, A)$ that appear in the training set. Formally, following Section 2, let $V_{\text{train}} \subseteq V$ denote the set of nodes included in $\mathcal{D}_{\text{train}}$. We define $\mathrm{c} = |V_{\text{train}}|/|V|$, which ranges between 0 and 1.

Remark. We stress that coverage is defined **only locally relative to the training map, not the global universe.** Even $\mathrm{c} = 0.8$ corresponds only to a tiny fraction of the universe of possible primitives. Since the model is expected (and observed) to spatially transfer to (infinitely) many disjoint maps $\hat{G} = (\hat{V}, \hat{A})$, including nodes from all such maps in the denominator would only dilute the fraction and make coverage misleadingly small. For comparability, we therefore compute coverage solely with respect to the training map; any nodes in any disjoint map $\hat{G}$ are in fact uncovered.

Diversity. Diversity measures how richly the observed nodes are combined with each other in training. Formally, recall from Section 2 that $\text{supp}(\mathcal{D}_{\text{train}})$ denotes the set of ordered node pairs included in training. We define $\mathrm{d} = |\text{supp}(\mathcal{D}_{\text{train}})|/|V_{\text{train}}|$, which ranges from 1 to $|V|-1$. Intuitively, d corresponds to the average number of distinct endpoints j that each start node $i \in V_{\text{train}}$ is paired with. In practice, we control diversity explicitly by constraining, for each i, the number of distinct j's that appear in pairs (i, j).

Remark. Note that we intentionally do not normalize d by $|V_{\text{train}}| - 1$, since $|V_{\text{train}}| = c|V|$; this would couple diversity with coverage and prevent the two from being varied independently.

Experiment Design. To disentangle the roles of coverage and diversity, we design controlled experiments where one factor is varied while the other is fixed. Coverage is defined as $c = |V_{\text{train}}|/|V|$ and is varied by ***linearly*** increasing the fraction of nodes included in the training questions from as low as 4% up to 80% of the nodes in the training map. Diversity d is varied by controlling how many distinct endpoints j each start node i is connected to, ranging ***exponentially*** from 2^0 to 2^7. We control the total number of question–answer records to remain fixed across conditions. We use the same evaluation protocol as before, with one training map G and three independent and disjoint maps $\hat{G}$, and report the average performance (measured by the success rate) over the three test maps.

3.2.1 Coverage Quantification at Fixed Diversity

We draw three key observations from Figure 2:

(1) Coverage determines the ceiling of spatial transfer. Across a wide range of diversity levels (from $d = 2^2$ to s^7), the curves converge to a similar maximum SR once coverage is high. This shows that coverage ultimately sets the upper bound of systematic generalization, while diversity only influences how quickly, as coverage increases, this ceiling is approached.

(2) Minimal diversity is required to unlock efficient use of coverage. However, at very low diversity ($d = 1, 2$), SR grows slowly and saturates at a noticeably lower level. Only when diversity passes a small threshold ($d \geq 4$ here) does coverage begin to unlock its full effect.

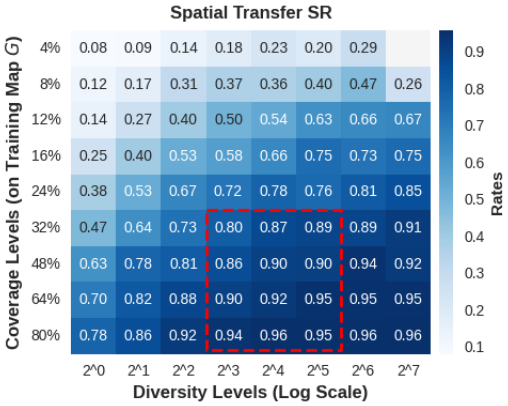


Figure 4: Interaction between coverage and diversity on problem-solving transfer.

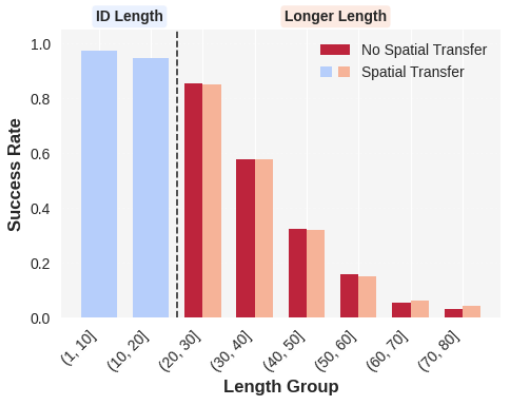


Figure 5: Spatial transfer does not imply length scaling.

(3) Threshold and plateau phenomenon. With sufficient diversity, SR exhibits a sharp inflection around mid-to-low coverage (≈ 0.2–0.25). Beyond this point, additional coverage yields diminishing returns, whereas below it, generalization remains poor.

Takeaway 2: Coverage in question sets the ceiling of spatial transfer, but minimal diversity is required to unlock it efficiently. Coverage also creates a sharp inflection point in SR, indicating a cost-efficient region at low values.

3.2.2 Diversity effects at fixed coverage

We next vary diversity while keeping coverage constant. Results in Figure 3 show two main patterns:

(1) Log-linear gains at mid-to-high coverage. At mid-to-high coverage, performance grows roughly linearly in $\log(d)$, indicating that the marginal benefit of additional diversity decreases as d grows. In other words, exposing the model to a few diverse combinations is highly beneficial, but each further doubling of diversity yields progressively smaller gains.

(2) High diversity can hurt when coverage is low. At low coverage, adding diversity sometimes reduces success rates. This likely occurs because exhaustively combining a tiny set of primitives encourages memorization rather than rule abstraction. For example, if a model is trained on a very small set $1, 2, 3$ and exposed to all possible addition combinations, it can simply memorize the resulting facts without grasping the general rule of addition.

Takeaway 3: Diversity can bring rapid early gains but quickly flattens out, and may even harm transfer when coverage is low.

3.3 Coverage–diversity interaction

Jointly analyzing coverage and diversity (Figure 4) reveals a clear interaction. At low coverage, even exponentially high diversity cannot rescue the performance (e.g., SR rises only from 0.06 to 0.19 when coverage is 4%). By contrast, at higher coverage (e.g., 64%), diversity strongly amplifies performance (raising SR from 0.42 at low diversity to above 0.65 at moderate–to-high diversity). High coverage can also partially compensate for low diversity (e.g., performance increasing from 0.08 to 0.70 when $d = 1$).

Because diversity grows in cost exponentially, **a resource-efficient regime (highlighted in red, Figure 4) is to target mid-to-high coverage** ($\geq 32\%$) **with modest diversity** (8–32). This achieves strong performance at a much lower computational cost than maximizing both dimensions. Beyond this regime, both coverage and diversity show diminishing returns. Note that dataset size is approximately controlled across conditions by varying the number of answers, which actually makes high-coverage/low-diversity settings relatively disadvantaged (Section 3.1). The fact that such settings still outperform low-coverage/high-diversity extremes underscores the strength of the result.

Takeaway 4: Low coverage in question cannot be rescued even with extreme diversity, but low diversity can be compensated by high coverage. Moderate–to-high coverage with modest diversity achieves the best efficiency–performance trade-off.

Intuitively, in broad problem-solving scenarios, primitives can be seen as the concepts that appear in questions. Coverage reflects how many distinct concepts are actually mentioned in training questions (e.g., in geometry, whether training questions touch only the triangle sum rule, or also include parallel-line angle rules, even though both are basic known primitives to the model). Diversity reflects how flexibly these concepts co-occur within questions (e.g., whether the triangle sum rule always appears alone, or also together with different angle rules across problems). The results in this section provide concrete guidance for dataset selection: to enable systematic transfer under a limited budget, one should prioritize broad coverage of concepts in question, combine it with only modest diversity in their combinations, and spend the least effort on solution diversity.

Mechanistic explanation for the success of spatial transfer. We observed strong generalization to disjoint maps, akin to how a language model, once having internalized a rule in English, can seamlessly apply the same rule to other languages it already knows. This suggests that the model does not merely memorize surface-level node n-gram, but rather encodes structured latent operators that can be flexibly reused across domains—for instance, "move to an adjacent node towards the end node" heuristic (which we probed in Section C.2). This interpretation aligns with recent theoretical progress framing attention as a hypernetwork (Schug et al., 2024), where attention scores serve as latent codes parameterizing reusable computations.

3.4 A Case Study in the Math Domain

To examine whether the conclusions drawn from our controlled navigation environment transfer to a more realistic setting, we conduct a case study on mathematical word problems using the **MathQA** dataset (Amini et al., 2019). Each MathQA problem is annotated with a *linearized operation program* containing primitive mathematical operations. These programs serve as direct analogues of the primitives used in our navigation experiments. For instance, the problem:

> "The ratio between the length and breadth of a rectangular park is 3 : 2. A man cycles around the boundary at 12 km/hr and completes one round in 8 minutes. What is the area of the park?"

has the corresponding operation chain:

> divide(12, 60) | multiply(#0, 8) | multiply(#1, 1000) | add(3, 2) | multiply(2, #3) | divide(#2, #4) | multiply(#5, 3) | multiply(#5, 2) | rectangle_area(#6, #7)

We convert each program into an *unordered multiset of operations*, which defines the conceptual skill set of the problem. This allows us to operationalize:

- **Coverage:** number of distinct skill sets (operation-sets) in training;
- **Diversity:** number of distinct program structures that instantiate the same skill set.

Setup. We fine-tune Qwen2.5-7B-Instruct (Team, 2024) on three representative categories—`probability` (easy), `gain` (medium), and `physics` (hard)—under a strict data budget of roughly 1,000 samples per category (and only ~ 200 for `probability` due to its very small size). We use DeepSeek-R1 (DeepSeek-AI et al., 2025) to generate high-quality chain-of-thought solutions for supervision. Following our earlier observations, we compare the following data allocation strategies (more details are provided in Section D):

- **More Questions:** one solution per question, maximizing the number of distinct questions. This strategy has two variants:
  - **High Coverage**: maximize the number of distinct operation-sets;
  - **High Diversity**: increase the number of questions per operation-set (tenfold), and therefore operate under a smaller coverage.
- **More Solutions:** ten solutions per question and reducing the number of distinct questions.

Table 1: Performance of different data regimes across MathQA categories.

		probability (*easy*)	gain (*medium*)	physics (*hard*)
Qwen2.5-7B-Instruct	–	0.729	0.70	0.68
More questions	High Coverage	**0.792**	**0.82**	**0.77**
	High Diversity	0.792	0.74	0.74
More solutions	–	0.771	0.72	0.70

More questions consistently outperform more solutions. Across all three categories, both *More Questions* regimes (High Coverage and High Diversity) achieve better generalization than *More Solutions*. Notably, these improvements appear under an extremely small training budget: roughly 1,000 samples for category `gain` and `physics`, and only ~ 200 for `probability`). Despite such limited supervision, allocating more budget to distinct questions still produces clear performance gains. For instance, in the `gain` category, accuracy rises from 0.70 to 0.82 under `High Coverage`, and a similar increase appears in the harder `physics` category ($0.68 \rightarrow 0.77$).

Coverage plays the dominant role. Within the *More Questions* group, *High Coverage* consistently outperforms *High Diversity* (e.g., 0.82 vs. 0.74 in `gain`; 0.77 vs. 0.74 in `physics`). This suggests that encountering a broader range of conceptual skills matters more than exposing the model to many different ways of applying or combining the seen skills. Taken together, these findings reinforce a simple intuition: ***under realistic data budgets, breadth matters more than depth***.

4 Effects of Data Selection on Problem-Solving Scaling

In addition to problem-solving transfer, a fundamental dimension of extrapolation is *problem-solving scaling* (or productivity in CG) (Sinha et al., 2024; Hupkes et al., 2020; Cai et al., 2025). While transfer asks whether rules can be applied spatially to infinitely many node pairs within the same length regime, scaling tests whether these rules extend to node pairs that require longer paths than those seen during training (see Section 2 for a formal definition). This raises a natural question: do the same conditions that enable spatial transfer (i.e., sufficient training questions and primitive coverage) also support scaling under SFT, or does this setting demand additional data conditions?

Length scaling fails irrespective of the map. In Figure 5, we show the length scaling performance of the strongest spatial-transfer model (selected under the high-budget setting with all budget allocated to questions and high primitive coverage–diversity). Results for other budgets are shown in Section C.6. We report success rate (SR) on both holdout nodes within the training map (*No spatial transfer*) and spatially disjoint maps (*Spatial transfer*). For each length group, evaluation is performed on 3,000 randomly sampled unseen node pairs. The trends are nearly identical: while the model achieves near-perfect generalization within the training length regime (blue region), SR rapidly deteriorates once path length exceeds the training maximum (red region). This indicates that **even when spatial transfer succeeds, length scaling can fail**.

Rescuing with neighboring-and-longer paths. Surprisingly, we found that adding even a handful of training examples randomly sampled from lengths at or above the target length can substantially rescue performance, whereas adding shorter paths provides much less benefit. For instance, in Figure 6, we evaluate performance on target length $= 30$, where the model exhibits suboptimal generalization. Augmenting the training set with a very small fraction ($\approx 1\%$ of the training data) of neighboring-but-longer paths (e.g., $l = 32, 34$) raises success rates to nearly 90%. By contrast, adding shorter paths (e.g., $l = 22, 24$) yields small gains—even when added in large amounts (12%)—while much longer paths (e.g., $l = 80$) confuse the model and degrade performance. These results suggest that, under SFT, curriculum-like exposure to neighboring-and-longer examples provides the critical adaptation signal that neither shorter nor excessively long paths can supply.

Takeaway 5: Length generalization can be rescued by adding *neighboring-and-longer* paths; shorter ones give little benefit, and overly long ones can even harm performance.

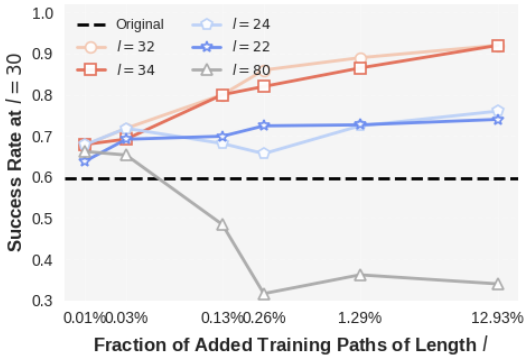


Figure 6: Effect of adding paths of different lengths on SR to length $= 30$. A few neighboring-and-longer paths (e.g., $l = 32, 34$) rescue performance, shorter ones ($l = 22, 24$) give little gain, and overly long ones ($l = 80$) degrade it. Dashed line: no augmentation.

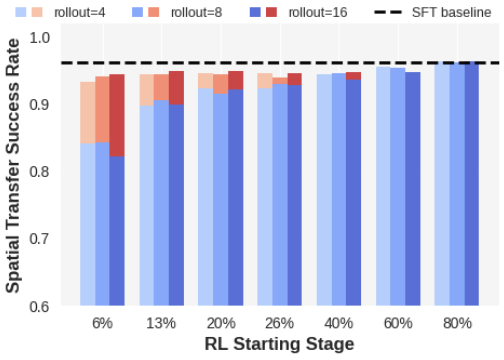


Figure 7: RL does not further improve spatial transfer: performance bounded by the SFT baseline. Each group of bars corresponds to a different SFT checkpoint used to initialize RL. Blue bars denote one-pass RL and red bars denote multi-pass RL.

5 Effects of Training Paradigm on Problem-Solving

While the previous sections focus on how data properties shape problem-solving skills, another natural question is whether the training paradigm itself can provide further gains. A recent line of work presents compelling empirical evidence that reinforcement learning (RL) can enable extrapolative generalization beyond supervised fine-tuning (SFT) (Chu et al., 2025; Chen et al., 2025; Huang et al., 2025). At the same time, other studies argue that RL primarily unlocks capabilities already present in SFT rather than introducing new ones (Yue et al., 2025; Ma et al., 2025). We therefore test whether RL adds value on top of SFT for spatial transfer and length scaling.

Spatial transfer setup. As detailed in Section 2, we train the RLVR model using an unbiased GRPO variant with a binary reward of 1 if the generated sequence forms a valid shortest path, and 0 otherwise. The training data budget is set to high, under which the model is capable of spatially transferring (see Figure 1). RL is warm-started from different SFT checkpoints, ranging from 6% to 80% of SFT training progress. For each warm-start, we vary the number of rollouts per prompt in $4, 8, 16$. We report two types of RL outcomes: (i) **one-pass RL** (blue bars), where the model is trained on the remaining data for a single pass; and (ii) **multi-pass RL** (red bars), where RL is allowed to repeatedly reuse the same remaining data. This disentangles the effects of warm-start quality, data availability, and rollout compute.

RL does not improve spatial transfer. As shown in Figure 7, RL does not confer additional capabilities beyond what can be achieved by a fully trained SFT for spatial transfer: the best RL curves are always bounded by the SFT upper line. Early warm-starts perform poorly in one-pass RL (blue bars), but multi-pass training (red bars) can recover the gap

Length scaling setup. To test whether RL can address the known weakness of SFT in length scaling under "unlimited" passes, we continue RL training for up to $\sim$20 epochs on the same dataset (rollout fixed at 8), with the model warm-started from an early SFT checkpoint (after 1 epoch, 400 steps). For comparison, we also extend SFT training for the same number of epochs.

RL stabilizes training but cannot exceed the best SFT. Figure 8 compares SFT and RL under the same 10-epoch progress and show a clear pattern: SFT initially improves with more steps but quickly overfits, leading to sharp degradation. RL curves, in contrast, remain tightly clustered across steps, indicating stable training even after many epochs. However, RL never exceeds the best SFT bound, confirming that additional training (whether SFT or RL) cannot resolve the fundamental limitation in length scaling. Results for extended RL training up to 20 epochs are provided Section C.7 and show the same stable trend.

Across both settings, RL's role is primarily to ***stabilize training*** and avoid overfitting during prolonged training, rather than to unlock new reasoning capabilities. As shown in Section F, the *SFT and RL models exhibit the same error types with nearly identical error distributions*, indicating that

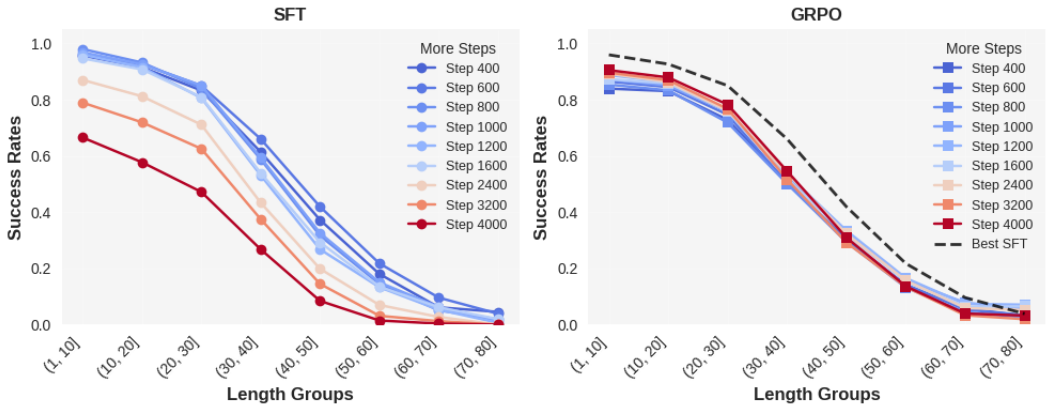


Figure 8: Length scaling under extended training (1 epoch $\approx$ 400 steps). Left: SFT improves at first but quickly overfits with more epochs. Right: RL (GRPO) remains stable across epochs but never exceeds the best SFT bound (dashed line).

RL cannot bypass the errors made by the corresponding SFT model. Consequently, the performance ceiling is always set by the best SFT model. This behavior is consistent with recent analyses of the ***generation–verification gap (Swamy et al., 2025)***: RL provides benefits when generating good continuations is difficult but verifying them is easy. In our setting, the optimal path can be computed explicitly, making generation nearly as easy as verification and effectively closing this gap. When the data are sufficient and high-quality, and data usage is carefully designed to avoid overfitting, SFT is more efficient at fitting available information. In contrast, RL serves as a more robust "safe default", trading peak efficiency for stability when principled data selection is missing.

Takeaway 6: RL stabilizes training and prevents overfitting but does not unlock new transfer or scaling capabilities. The performance ceiling is always set by the best SFT model. SFT is more efficient with sufficient, high-quality data, while RL provides a safer default when principled data selection is missing.

Remark. Our findings do not suggest that RL is useless. In practice, training data are often noisy, heterogeneous, or subject to domain shifts, where SFT may overfit or struggle. RL, by exploiting sequence-level rewards, provides stability and robustness under such conditions. Our findings therefore reconcile two perspectives in the literature: RL may not add fundamentally new capabilities beyond SFT, but it can serve as a stability-enhancing paradigm when training on messy or poorly curated data. In such unfavorable settings for SFT, RL can appear to generalize better—not because it extends the capability frontier, but because it maintains robustness where SFT struggles.

6 Conclusion and Limitations

In this work, we introduce a controlled map-navigation testbed to dissect extrapolative problem-solving. spatial transfer is primarily enabled by sufficient distinct questions with high coverage and modest diversity, while length scaling critically depends on exposure to neighboring-but-longer examples. For training paradigms, we find that RL effectively stabilizes optimization but cannot surpass the ceiling established by SFT. Overall, we provide clear guidelines for how data and training choices shape spatial and length generalization.

Limitations The main limitation of our study is that all conclusions are based on a synthetic testbed with small models, which naturally raises questions about practical relevance. We first want to highlight that this is an unavoidable trade-off: narrowing down the problem enables rigorous and well-controlled tests, but inevitably comes at the cost of practical realism. Despite the abundance of practical large-scale benchmarks, the insights gained so far remain limited. Our work therefore narrows the scope to a smaller, more concrete setting. To increase practical relevance, we ground our setup in problem-solving tasks that widely exist in real-world scenarios, employ path data that is closely tied to reasoning and math, and incorporate RL paradigms. In realistic contexts, training a series of RL models is mostly infeasible since RLVR typically requires large base models and substantial compute, whereas our synthetic task (with the easy-to-verify reward) makes such systematic experimentation tractable.

REFERENCES

Amirhesam Abedsoltan, Huaqing Zhang, Kaiyue Wen, Hongzhou Lin, Jingzhao Zhang, and Mikhail Belkin. Task generalization with autoregressive compositional structure: Can learning from d tasks generalize to d^t tasks? *arXiv preprint arXiv:2502.08991*, 2025.

Kartik Ahuja and Amin Mansouri. On provable length and compositional generalization. *arXiv preprint arXiv:2402.04875*, 2024.

AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2357–2367, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1245. URL https://aclanthology.org/N19-1245.

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 38546–38556, 2022.

Dzmitry Bahdanau, Shikhar Murty, Michael Noukhovitch, Thien Huu Nguyen, Harm de Vries, and Aaron Courville. Systematic generalization: what is required and can it be learned? *arXiv preprint arXiv:1811.12889*, 2018.

Francesco Cagnetta, Leonardo Petrini, Umberto M Tomasini, Alessandro Favero, and Matthieu Wyart. How deep neural networks learn compositional data: The random hierarchy model. *Physical Review X*, 14(3):031001, 2024.

Ziyang Cai, Nayoung Lee, Avi Schwarzschild, Samet Oymak, and Dimitris Papailiopoulos. Extrapolation by association: Length generalization transfer in transformers. *arXiv preprint arXiv:2506.09251*, 2025.

Hoyeon Chang, Jinho Park, Hanseul Cho, Sohee Yang, Miyoung Ko, Hyeonbin Hwang, Seungpil Won, Dohaeng Lee, Youbin Ahn, and Minjoon Seo. The coverage principle: A framework for understanding compositional generalization. *arXiv preprint arXiv:2505.20278*, 2025.

Hardy Chen, Haoqin Tu, Fali Wang, Hui Liu, Xianfeng Tang, Xinya Du, Yuyin Zhou, and Cihang Xie. Sft or rl? an early investigation into training r1-like reasoning large vision-language models. *arXiv preprint arXiv:2504.11468*, 2025.

Noam Chomsky. *Syntactic Structures*. Mouton, 1957.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Andrew Cohen, Andrey Gromov, Kaiyu Yang, and Yuandong Tian. Spectral journey: How transformers predict the shortest path. *arXiv preprint arXiv:2502.08794*, 2025.

Róbert Csordás, Kazuki Irie, and Jürgen Schmidhuber. Ctl++: Evaluating generalization on never-seen compositional patterns of known functions, and compatibility of neural representations. *arXiv preprint arXiv:2210.06350*, 2022.

Xinnan Dai, Qihao Wen, Yifei Shen, Hongzhi Wen, Dongsheng Li, Jiliang Tang, and Caihua Shan. Revisiting the graph reasoning ability of large language models: Case studies in translation, connectivity and shortest path. *arXiv preprint arXiv:2408.09529*, 2024.

Verna Dankers, Elia Bruni, and Dieuwke Hupkes. The paradox of the compositionality of natural language: A neural machine translation case study. *arXiv preprint arXiv:2108.05885*, 2021.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Yann Dubois, Gautier Dagan, Dieuwke Hupkes, and Elia Bruni. Location attention for extrapolation to longer sequences. *arXiv preprint arXiv:1911.03872*, 2019.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36:70293–70332, 2023.

Ying Fan, Yilun Du, Kannan Ramchandran, and Kangwook Lee. Looped transformers for length generalization. *arXiv preprint arXiv:2409.15647*, 2024.

Jingwen Fu, Zhizheng Zhang, Yan Lu, and Nanning Zheng. A general theory for compositional generalization. *arXiv preprint arXiv:2405.11743*, 2024.

Daniel Furrer, Marc van Zee, Nathan Scales, and Nathanael Schärli. Compositional generalization in semantic parsing: Pre-training vs. specialized architectures. *arXiv preprint arXiv:2007.08970*, 2020.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, et al. Measuring coding challenge competence with apps. *arXiv preprint arXiv:2105.09938*, 2021.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.

Yichen Huang and Lin F Yang. Gemini 2.5 pro capable of winning gold at imo 2025. *arXiv preprint arXiv:2507.15855*, 2025.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.

Dieuwke Hupkes, Mario Giulianelli, Verna Dankers, Mikel Artetxe, Yanai Elazar, Tiago Pimentel, Christos Christodoulopoulos, Karim Lasri, Naomi Saphra, Arabella Sinclair, et al. A taxonomy and review of generalization research in nlp. *Nature Machine Intelligence*, 5(10):1161–1174, 2023.

Samy Jelassi, Stéphane d'Ascoli, Carles Domingo-Enrich, Yuhuai Wu, Yuanzhi Li, and François Charton. Length generalization in arithmetic transformers. *arXiv preprint arXiv:2306.15400*, 2023.

Mason Kamb and Surya Ganguli. An analytic theory of creativity in convolutional diffusion models. *arXiv preprint arXiv:2412.20292*, 2024.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36:24892–24928, 2023.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, et al. Measuring compositional generalization: A comprehensive method on realistic data. *arXiv preprint arXiv:1912.09713*, 2019.

Najoung Kim and Tal Linzen. Cogs: A compositional generalization challenge based on semantic interpretation. *arXiv preprint arXiv:2010.05465*, 2020.

Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pp. 2873–2882. PMLR, 2018.

Michael Lepori, Thomas Serre, and Ellie Pavlick. Break it down: Evidence for structural compositionality in neural networks. *Advances in Neural Information Processing Systems*, 36:42623–42660, 2023.

Itay Levy, Ben Bogin, and Jonathan Berant. Diverse demonstrations improve in-context compositional generalization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1401–1422, 2023.

Martha Lewis, Nihal V Nayak, Peilin Yu, Qinan Yu, Jack Merullo, Stephen H Bach, and Ellie Pavlick. Does clip bind concepts? probing compositionality in large image models. *arXiv preprint arXiv:2212.10537*, 2022.

Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. *ICLR*, 2023.

Qiyao Liang, Daoyuan Qian, Liu Ziyin, and Ila Fiete. Compositional generalization requires more than disentangled representations. *arXiv e-prints*, pp. arXiv–2501, 2025.

Samuel Lippl and Kim Stachenfeld. When does compositional structure yield compositional generalization? a kernel theory. In *The Thirteenth International Conference on Learning Representations*.

Adam Liška, Germán Kruszewski, and Marco Baroni. Memorize or generalize? searching for a compositional rnn in a haystack. *arXiv preprint arXiv:1802.06467*, 2018.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Joao Loula, Marco Baroni, and Brenden M Lake. Rearranging the familiar: Testing compositional generalization in recurrent networks. *arXiv preprint arXiv:1807.07545*, 2018.

Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Bin Cui, et al. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. *arXiv preprint arXiv:2506.07527*, 2025.

Benjamin Newman, John Hewitt, Percy Liang, and Christopher D Manning. The eos decision and length extrapolation. *arXiv preprint arXiv:2010.07174*, 2020.

Yaniv Nikankin, Anja Reusch, Aaron Mueller, and Yonatan Belinkov. Arithmetic without algorithms: Language models solve math with a bag of heuristics. *arXiv preprint arXiv:2410.21272*, 2024.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Maya Okawa, Ekdeep S Lubana, Robert Dick, and Hidenori Tanaka. Compositional abilities emerge multiplicatively: Exploring diffusion models on a synthetic task. *Advances in Neural Information Processing Systems*, 36:50173–50195, 2023.

Santiago Ontanon, Joshua Ainslie, Vaclav Cvicek, and Zachary Fisher. Making transformers solve compositional tasks. *arXiv preprint arXiv:2108.04378*, 2021.

Jackson Petty, Sjoerd van Steenkiste, Ishita Dasgupta, Fei Sha, Dan Garrette, and Tal Linzen. The impact of depth on compositional generalization in transformer language models. *arXiv preprint arXiv:2310.19956*, 2023.

Philip Quirke and Fazl Barez. Understanding addition in transformers. *arXiv preprint arXiv:2310.13121*, 2023.

Philip Quirke, Clement Neo, and Fazl Barez. Arithmetic in transformers explained. *arXiv preprint arXiv:2402.02619*, 2024.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Rahul Ramesh, Ekdeep Singh Lubana, Mikail Khona, Robert P Dick, and Hidenori Tanaka. Compositional capabilities of autoregressive transformers: A study on synthetic, interpretable tasks. *arXiv preprint arXiv:2311.12997*, 2023.

Simon Schug, Seijin Kobayashi, Yassir Akram, João Sacramento, and Razvan Pascanu. Attention as a hypernetwork. *arXiv preprint arXiv:2406.05816*, 2024.

Sania Sinha, Tanawan Premsri, and Parisa Kordjamshidi. A survey on compositional learning of ai models: Theoretical and experimental practices. *arXiv preprint arXiv:2406.08787*, 2024.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 115–124. Association for Computational Linguistics, 2021.

Gokul Swamy, Sanjiban Choudhury, Wen Sun, Zhiwei Steven Wu, and J Andrew Bagnell. All roads lead to likelihood: The value of reinforcement learning in fine-tuning. *arXiv preprint arXiv:2503.01067*, 2025.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`, 2020.

Yuxiang Wang, Xinnan Dai, Wenqi Fan, and Yao Ma. Exploring graph tasks with pure llms: A comprehensive benchmark and investigation. *arXiv preprint arXiv:2502.18771*, 2025a.

Zehong Wang, Zheyuan Liu, Tianyi Ma, Jiazheng Li, Zheyuan Zhang, Xingbo Fu, Yiyang Li, Zhengqing Yuan, Wei Song, Yijun Ma, et al. Graph foundation models: A comprehensive survey. *arXiv preprint arXiv:2505.15116*, 2025b.

Thaddäus Wiedemer, Jack Brady, Alexander Panfilov, Attila Juhos, Matthias Bethge, and Wieland Brendel. Provable compositional generalization for object-centric learning. *arXiv preprint arXiv:2310.05327*, 2023a.

Thaddäus Wiedemer, Prasanna Mayilvahanan, Matthias Bethge, and Wieland Brendel. Compositional generalization from first principles. *Advances in Neural Information Processing Systems*, 36:6941–6960, 2023b.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

Zhuoyan Xu, Zhenmei Shi, and Yingyu Liang. Do large language models have compositional ability? an investigation into limitations and scalability. *arXiv preprint arXiv:2407.15720*, 2024.

Gilad Yehudai, Ethan Fetaya, Eli Meirom, Gal Chechik, and Haggai Maron. From local structures to size generalization in graph neural networks. In *International Conference on Machine Learning*, pp. 11975–11986. PMLR, 2021.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Tian Yun, Usha Bhalla, Ellie Pavlick, and Chen Sun. Do vision-language pretrained models learn composable primitive concepts? *arXiv preprint arXiv:2203.17271*, 2022.

Qifan Zhang, Nuo Chen, Zehua Li, Miao Peng, Jing Tang, and Jia Li. Improving llms' generalized reasoning abilities by graph problems. *arXiv preprint arXiv:2507.17168*, 2025.

Yizhuo Zhang, Heng Wang, Shangbin Feng, Zhaoxuan Tan, Xiaochuang Han, Tianxing He, and Yulia Tsvetkov. Can llm graph reasoning generalize beyond pattern memorization? *arXiv preprint arXiv:2406.15992*, 2024.

Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Josh Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? a study in length generalization. *arXiv preprint arXiv:2310.16028*, 2023.

A LLM USAGE

We use ChatGPT and Gemini to support writing and formatting, such as grammar and style refinement, polishing figure and table captions, and other surface-level edits. Some of the code is written with the help of GitHub Copilot and Claude, for example, in code auto-completion and providing debugging suggestions.

B Related works

Compositional and length generalization Our notion of extrapolative problem-solving is closely tied to systematicity in compositional generalization (CG) and to length generalization, sometimes referred to as productivity in CG. Compositional Generalization (CG) plays a central role in generalization studies, which underpins the ability to extend learning to unseen situations (Hupkes et al., 2023). *Systematicity*, the most common definition of CG, refers to the capacity to systematically recombine known primitives and rules (Dankers et al., 2021). While Systematicity has long been regarded as a fundamental challenge for neural networks (Liška et al., 2018; Lake & Baroni, 2018; Loula et al., 2018; Csordás et al., 2022; Ontanon et al., 2021; Keysers et al., 2019; Lewis et al., 2022), recent work has increasingly provided evidence that modern generative models exhibit non-trivial Systematic CG abilities (Lepori et al., 2023; Yun et al., 2022; Okawa et al., 2023; Ramesh et al., 2023; Abedsoltan et al., 2025; Xu et al., 2024). Further progress in understanding Systematic CG comes from multiple perspectives: the structural side (Lepori et al., 2023; Schug et al., 2024; Quirke & Barez, 2023; Li et al., 2023), the task side (Abedsoltan et al., 2025; Zhou et al., 2023), and the data side (Lippl & Stachenfeld; Ahuja & Mansouri, 2024; Kamb & Ganguli, 2024; Chang et al., 2025; Cagnetta et al., 2024). For example, (Schug et al., 2024) shows that multi-head attention can function as a hypernetwork supporting compositional behavior (e.g., encouraging learning functions as reusable components). From the data perspective, (Ahuja & Mansouri, 2024) derives provable guarantees for length and compositional generalization under sufficient training-set diversity, while (Chang et al., 2025) frames training data coverage as a key factor in a model's ability to generalize to unseen combinations.

Progress on compositionality in vision object learning has become increasingly well characterized, both empirically (Yun et al., 2022) and theoretically (Wiedemer et al., 2023b;a). In language, however, understanding remains fragmented: studies have pointed to a variety of factors (e.g., from model-side (Kazemnejad et al., 2023; Petty et al., 2023) to data-side (Ahuja & Mansouri, 2024; Chang et al., 2025)), but lacking an integrated account. Our work takes a data-centric perspective, unifying the recurring factors into a coherent view of how they jointly shape the model's systematic extrapolation. Inspired by progress in vision, where disentangled primitives and rules have enabled clearer advances, and to avoid prior inconclusive results in language (Lake & Baroni, 2018; Furrer et al., 2020; Dziri et al., 2023), we design map-navigation tasks in which primitives (nodes) and rules (mobilities) are cleanly disentangled, allowing us to directly assess the influence of data properties on generalization performance (Liang et al., 2025).

Length generalization, or *Productivity*, is another notion within the broader study of compositional generalization Sinha et al. (2024). It has been widely discussed as a central challenge (Dubois et al., 2019; Newman et al., 2020; Cai et al., 2025; Fan et al., 2024; Jelassi et al., 2023; Anil et al., 2022), and is sometimes framed as a form of recursive composition or extrapolation Kim & Linzen (2020); Hupkes et al. (2020); Dziri et al. (2023). For instance, in natural language tasks, longer input sequences may correspond to recursive or nested structures of previously seen phrases Kim & Linzen (2020). In our setting, path length provides a directly controllable axis for studying this phenomenon: extrapolating to longer paths mirrors the core difficulty of length generalization, while allowing us to systematically manipulate the data properties and training paradigm to probe its limits.

Graph navigation and other capabilities While our work may appear related to prior studies that evaluate models' graph navigation abilities Zhang et al. (2024); Wang et al. (2025a), build powerful graph models Wang et al. (2025b); Yehudai et al. (2021), or use graph data to enhance LMs' reasoning Zhang et al. (2025), it is in fact fundamentally different in both task setting and goal. First, rather than treating the graph as the task itself (i.e., providing the model with many small graphs in prompts and training it to solve specific navigation task on future graphs), our work considers the large map and treats each map as an independent vocabulary world. Instead of explicitly describing the graph structure, we require the model to learn the connections and the map itself, analogous to how LLMs acquire word semantics during pretraining. The map is sufficiently complex that it cannot be memorized or learned within a single prompt. Second, our goal is not to test whether models can perform navigation tasks, nor to improve navigation performance by modifying architectures or training pipelines. Instead, we seek to understand models' compositionality/extrapolation under varying data distributional properties. To ensure that our focus remains on distributional effects, we even restrict ourselves to tasks that are already proven to be learnable Cohen et al. (2025);

Dai et al. (2024). Therefore, our work is also orthogonal to studies that examine whether models can perform specific capabilities with certain heuristics under narrowly defined tasks Quirke et al. (2024); Nikankin et al. (2024); Cohen et al. (2025).

C Additional results

C.1 Implementation and Licensing.

Our LLaMA-style models are based on the standard implementations in the Hugging Face `transformers` library (Apache 2.0 license) Wolf et al. (2020). Reinforcement learning with Dr.GRPO is conducted using the `GRPOTrainer` from the Hugging Face TRL library (Apache 2.0 license) (von Werra et al., 2020).

Pretrain Specifications. We pretrain the model on random-walk trajectories to provide basic "map semantics" without leaking any shortest-path information. The pretraining data consists of long random walks sampled uniformly across the grid.

We adopt a pretraining corpus of **10M random-walk trajectories** (approximately 1.3B tokens), trained for **124,999** steps. This number was chosen based on preliminary runs with smaller datasets (2M, 5M, and 8M trajectories), where we observed that the model's valid-path rate increases steadily with data size and saturates at the 10M scale. As reported in Table 3, at this final budget, the pretrained model achieves a **valid-path rate of** 1.0 while retaining zero shortest-path capability, confirming that pretraining captures structural map knowledge without imparting any optimal navigation behavior.

C.2 Probing: Model tracks distance to the end node

We investigate whether the model encodes the remaining shortest-path distance to the end node, which would allow it to apply heuristics such as "move towards the goal". For probing, we apply a 2-layer MLP, $p_\theta(x_t^k) = \text{softmax}\big(W_1 \,\text{ReLU}(W_2 x_t^k)\big)$, where x_t^k denotes the hidden representation of the t-th token at the k-th layer. The probe outputs a probability distribution over discretized distance classes ($C = 10$). Although we probe at a *fixed token position*, the hidden state at this position already integrates information from all previous tokens, including the traversed path. We thus train a probe on paths of varied length from the training map for each layer, and test it on paths from a disjoint map, grouping path lengths from 1–20 into 10 classes (granularity of 2). As shown in Table 2, the nonlinear probe achieves high accuracy, especially in middle-to-late layers, supporting the hypothesis that the model encodes distance-based heuristics as reusable operators for spatial transfer. While a linear probe would provide a stronger conclusion, we have not yet identified one that performs well in this setting.

Table 2: Probe accuracy (%) across layers.

Layer	Accuracy (%)
0	35.94
1	32.78
2	57.85
3	76.58
4	83.14
5	**86.29**
6	**85.77**
7	81.55

C.3 Pretraining does not interfere with downstream shortest-path learning

To ensure that our pretraining stage does not leak or overlap with the downstream shortest-path task, we evaluate pretrained models directly on shortest-path generation. Both the loss distribution

Table 3: Performance on shortest-path generation. Pretrained models cannot generate valid shortest paths, confirming that pretraining does not interfere with downstream learning. The Avg. Length Ratio measures the ratio between the true shortest-path length and the generated path length.

Model Trained on	Valid Path Rate↑	Shortest Path Rate↑	Avg. Length Ratio↑
Pretrain	1.0	0.00	0.0707
Finetune	1.0	0.9726	0.9983

analysis Figure 9 and generation performance Table 3 confirm that pretraining does not endow the model with shortest-path capabilities, thereby ruling out interference.

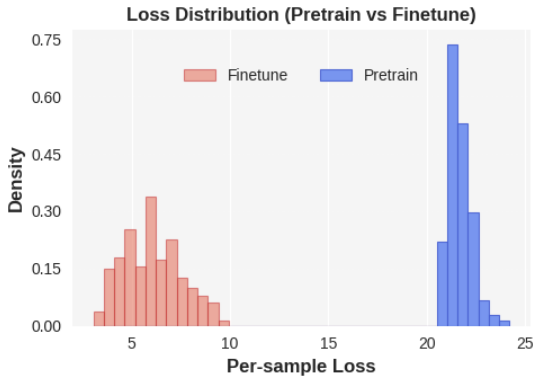


Figure 9: Loss distributions of the pretrained and fine-tuned models on test (*i.e.* unseen) shortest paths. The distributions are completely disjoint, indicating that pretraining alone does not prepare the model with shortest-path generation capabilities.

C.4 Training Path-Length Distribution Under Varying Coverage Values

To examine whether the shortest-path distance distribution shifts as coverage increases, potentially contributing to performance gains, we plot the shortest-path length histograms for different coverage ratios (under fixed diversity). Figure 10 reports the relative-frequency distributions (x: path length, y: proportion of samples) for coverage values $\{0.01, 0.05, 0.1, 0.2, 0.6, 0.8\}$ on the training map G.

Although the total number of sampled start–end pairs increases with coverage, the **shape of the path-length distribution remains highly stable** across all settings. The mean shortest-path length is consistently around 13.25 with a standard deviation of approximately 4.85, and the proportion of samples near the maximum observed training length ($L_{\max} = 20$) shows minimal variation.

These statistics confirm that increasing coverage does not introduce systematic changes to the length distribution, ensuring that our analyses isolate the effect of coverage itself rather than incidental differences in distance exposure.

C.5 Length Scaling Under a Relaxed Feasibility Metric

We additionally evaluate navigation under a relaxed metric, valid rates, that counts any trajectory reaching the goal (without using invalid edges) as correct, rather than requiring shortest-path optimality. As shown in Figure 11, feasibility remains near perfect for in-distribution lengths but still degrades substantially for longer paths. Relaxing the objective, therefore, does not remove the length-scaling failure. Instead, the higher feasible-path rates relative to shortest-path success suggest that the drop in performance arises from a combination of producing invalid trajectories and producing valid but non-optimal (non-shortest) ones.

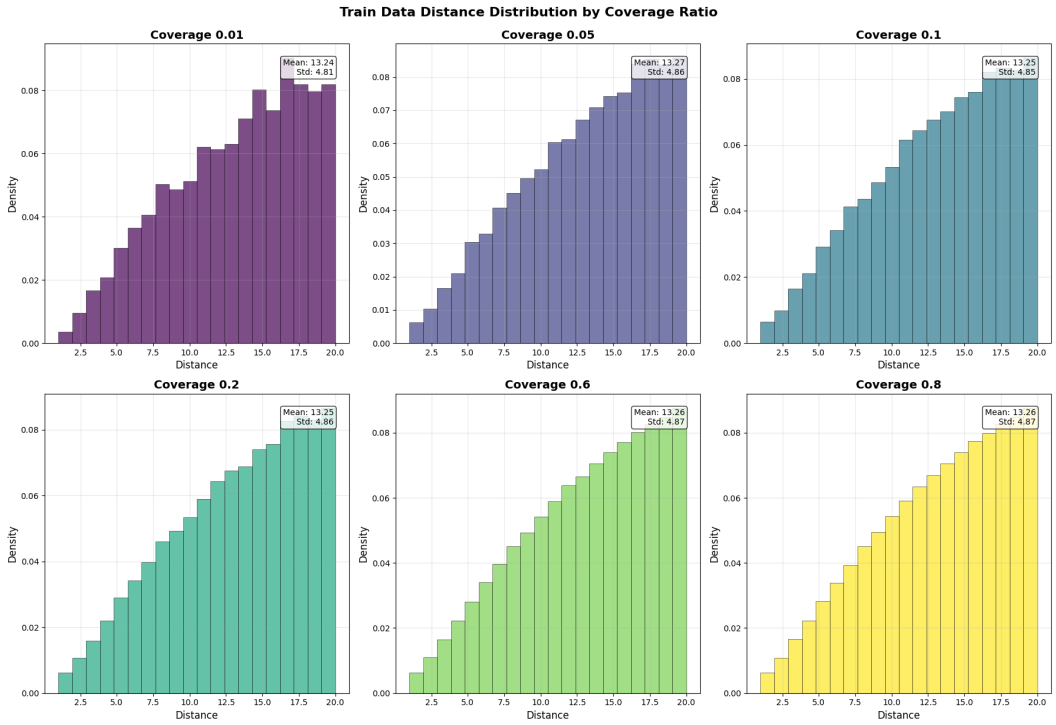


Figure 10: Shortest-path lengths distribution under varying coverage ratios (fixed diversity). The distributions remain stable across settings, indicating that coverage does not alter distance exposure.

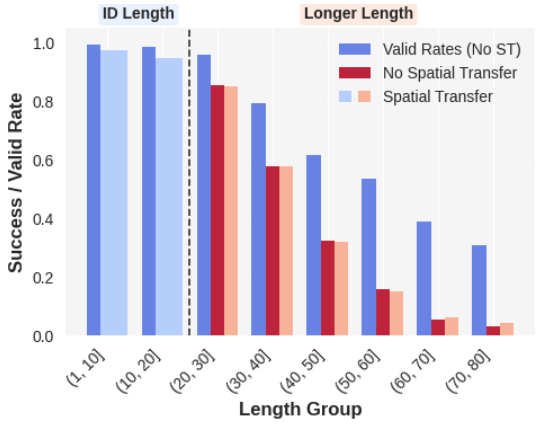


Figure 11: Valid-path rates across length groups. Although absolute performance improves, the same length-scaling failure persists.

C.6 Length Scaling Performance under Different Budgets

For completeness, we also evaluate length scaling across different data budgets (Figure 12). For each budget, we select the best-performing spatial-transfer model and report success rates (SR) on holdout node pairs with longer paths between them within the training map.

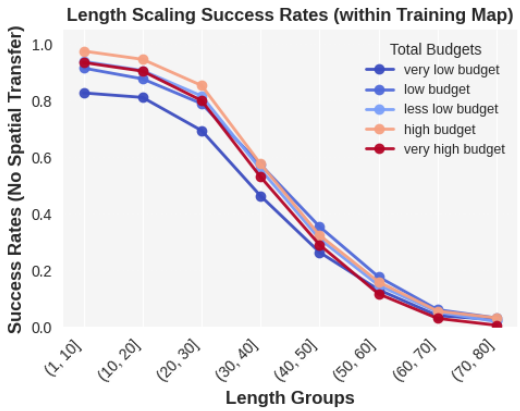


Figure 12: Length scaling performance of the best spatial-transfer model under different data budgets. All evaluations are conducted on holdout nodes within the training map (i.e., without spatial transfer). Despite variation in budgets, success rate (SR) consistently deteriorates as path length exceeds the training regime, showing that length scaling fails universally.

C.7 RL PERFORMANCE FOR MORE TRAINING STEPS

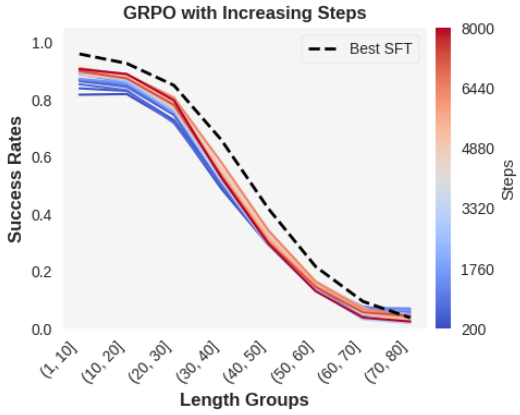


Figure 13: Length scaling for RL under extended training for 20 epochs (1 epoch $\approx$ 400 steps).

D PRACTICAL IMPLICATION: EVIDENCE FROM MATH DOMAIN

To assess the practical relevance of our findings beyond the synthetic map-navigation setting, we conduct a complementary study in the math domain. Specifically, we aim to examine whether, in a more realistic setting, *seeing more questions* remains more impactful than *seeing more solutions*, and whether a question's *coverage* plays a more dominant role than its *diversity*.

D.1 SETUP

Dataset We select the **MathQA** dataset (Amini et al., 2019) for evaluation because: (1) it contains six well-separated conceptual categories (gain, geometry, probability, physics, general, other), spanning a range of difficulties and posing greater challenges to commonly used 7B models than simpler math benchmarks such as GSM8K (Qwen et al., 2025); and (2) critically, it provides *linearized operation programs*. These programs act as direct analogues of the primitives in our navigation setting.

For instance, the problem:

> "The ratio between the length and breadth of a rectangular park is 3 : 2. A man cycles around the boundary at 12 km/hr and completes one round in 8 minutes. What is the area of the park?"

has the corresponding operation chain:

> divide(12, 60) | multiply(#0, 8) | multiply(#1, 1000) | add(3, 2) | multiply(2, #3) | divide(#2, #4) | multiply(#5, 3) | multiply(#5, 2) | rectangle_area(#6, #7)

Each operation chain can be converted into an unordered multiset of primitive operations, which captures the conceptual skills required by the problem and serves as a natural analogue to the primitives in our map setting. This representation allows us to define both *coverage* and *diversity* directly on mathematical problems. For datasets without human-provided formulas, we verify that a modern LLM can reliably extract the underlying primitive operations from natural-language questions using a short instruction prompt (e.g., `["compute rectangle area", "multiply", ...]`). A sample prompt–response pair is provided in Section E.1.

Definitions of terms We restate the core terms (questions, solutions, coverage, diversity) in the context of this math setting:

- **Questions:** Each distinct math word problem.
- **Solutions:** For each question, multiple high-quality reasoning traces may exist, and each trace is counted as a solution. We use DeepSeek-R1 (DeepSeek-AI et al., 2025) to produce such traces (i.e., explicit chain-of-thought outputs in the form of "Let's think step-by-step.").
- **Coverage and diversity in the math domain.** Each MathQA problem is paired with a linearized operation program specifying the sequence of primitive mathematical operations used to solve the problem. To align these programs with the coverage–diversity framework introduced in our map-navigation setting, we decompose them into two orthogonal components:
  - **Coverage.** A single primitive operation (e.g., `add`, `multiply`) is too coarse to characterize mathematical problem types, as most problems reuse the same small set of basic operations. What distinguishes one problem type from another is the *combination* of operations required. Therefore, we treat the *primitive operation-set*—the unordered multiset of operations appearing in an operation program—as the atomic semantic unit of a mathematical problem. This operation-set captures the underlying conceptual skills (or knowledge points) and serves as the minimal distinguishing signature of a problem type. Under this abstraction, coverage measures how many distinct operation-sets the model encounters during training.
  - **Diversity.** In the map setting, diversity measures how many distinct composition patterns exist under the same primitive support—that is, how flexibly a primitive participates in different relational structures. In the math domain, operations are composed sequentially rather than graphically, but the analogous notion remains: *the number of distinct program structures (operator orderings) that instantiate the same primitive operation-set*. This measures how flexibly a fixed conceptual skill set can be composed into different reasoning chains, without introducing new skills.

  For example, the two operation programs below share the same primitive operation-set but differ in ordering; thus, they contribute to diversity but not coverage:

  > [divide, multiply, add, rectangle_area]
  >
  > [add, divide, multiply, rectangle_area]

We fine-tune Qwen2.5-7B-Instruct (Team, 2024) across three representative difficulty categories in MathQA: `probability` (easy), `gain` (medium), and `physics` (hard). For each category, we fix a tight training budget of roughly 20% of its available samples (approximately 1,000 examples), except for the `probability` split, which uses 50% due to its extremely small size. All models are evaluated on the test set corresponding to the same category. As described above, we use DeepSeek-R1 to generate high-quality chain-of-thought traces for each question and construct three training regimes:

Table 4: Performance of different data regimes across three MathQA categories.

		probability	gain	physics
Qwen2.5-7B-Instruct	–	0.729	0.70	0.68
More questions	High Coverage	**0.792**	**0.82**	**0.77**
	High Diversity	0.792	0.74	0.74
More solutions	–	0.771	0.72	0.70

- **More Questions:** each question is paired with exactly *one* solution, enabling a larger number of distinct questions to be included under the same training budget. This includes two cases:
  - **High Coverage:** we include as many distinct primitive operation-sets as possible, resulting n questions per set;
  - **High Diversity:** for each operation-set we include $10n$ distinct questions, which increases structural diversity but necessarily reduces the number of covered operation-sets under the same training budget;
- **More Solutions:** each question is paired with ten independently generated solutions;

D.2 Results Analysis

The results in Table 4 demonstrate that the core principles identified in our controlled navigation setting apply to the MathQA domain, even though these practical tasks contain heterogeneous natural-language formulations and lack clearly separable generalization axes (e.g., spatial or length extrapolation).

More questions consistently outperform more solutions. Across all three categories (i.e., `gain`, `probability`, and `physics`), both *More Questions* regimes (High Coverage and High Diversity) achieve better generalization than *More Solutions*. Notably, these improvements appear under an extremely small training budget: roughly 1,000 samples for category `gain` and `physics`, and only ~ 200 for `probability`). Despite such limited supervision, allocating more budget to distinct questions still produces clear performance gains. For instance, in the `gain` category, accuracy rises from 0.70 to 0.82 under `High Coverage`, and a similar increase appears in the harder `physics` category ($0.68 \rightarrow 0.77$).

Coverage (operation-set variety) remains the dominant factor. Within the *More Questions* groups, `High Coverage` consistently outperforms `High Diversity` (e.g., 0.82 vs. 0.74 in `gain`, 0.77 vs. 0.74 in `physics`). This suggests that the model gains more from being exposed to a broader set of conceptual skills than from seeing many different applications, orderings, or compositional variants of the same skill set. This echoes the threshold behavior observed in the navigation setting: once the model has seen the *right set of conceptual skills*, generalization improves sharply—even without extensive practice on each skill.

Taken together, these findings reinforce a simple intuition: *under realistic data budgets, breadth matters more than depth.*

E Prompts Used in the Pipeline

E.1 Prompt for Extracting Primitive Operations from Word Problems

For natural-language math questions without human annotation, we can directly prompt an LLM to extract the underlying primitive operations. Below we provide an example prompt–response pair. The extracted operations are highly reasonable—often comparable to, or even clearer than, the human-provided operation programs used in Section 3.4.

Prompt

```
You are an expert at breaking down math word problems into primitive
mathematical operations.
Given a math word problem, output only a JSON list of short operation
verbs (1-3 words) that describe the steps needed to solve it, such as:
"add", "multiply", "divide", "use ratio", "convert units", "compute
area".
List them in the order they would be used.
Do NOT show explanations or numbers.
Output only the JSON list.
```

Model Response

```
["convert minutes to hours", "multiply", "divide", "use ratio",
"multiply", "compute rectangle area"]
```

E.2 Prompt for Collecting Chain-of-Thought Reasoning Traces

To obtain high-quality chain-of-thought reasoning traces for each math question, we prompt a stronger LLM (DeepSeek-R1 in our implementation) with an instruction that encourages explicit step-by-step reasoning followed by a clearly formatted final answer. Below we provide the exact templates used in our data construction pipeline.

System Prompt

```
You are a helpful math tutor who explains things step-by-step and
always finishes with a clearly formatted final answer.
```

User Prompt

```
Break down your reasoning process step by step, and show your thought
process explicitly.
Separate each step using \n\n.

At the end, conclude with a single line in the exact format:
The answer choice is: <option>.

Now solve the following multiple-choice math problem:

[Question]
{question_text}
[Solution]
```

E.3 Prompt for Qwen2.5-7B-Instruct

We use the same prompt for finetuning and evaluation of Qwen2.5-7B-Instruct and our finetuned variants. In all cases, the model is instructed to first produce a step-by-step reasoning trace and then output a clearly formatted final answer, as shown below.

System Prompt

```
You are a helpful assistant.
```

User Prompt

```
Break down your reasoning process step by step, and show your thought
process explicitly.
Separate each step with \n\n.
Conclude with a single line in the exact format:
The answer choice is: [insert answer choice].

[Question]
{question_text}
[Solution]
```

Chat template. The human-readable prompts described above correspond to the `system` and `user` messages used during both fine-tuning and evaluation. In practice, all messages are serialized

using Qwen2.5's official tokenizer chat template (via `apply_chat_template` function). This ensures that both fine-tuning and evaluation use the exact prompt format expected by Qwen2.5-7B-Instruct models, including all special tokens (e.g., `<|im_start|>`) and role indicators required by the tokenizer.

F Qualitative Analysis of Navigation Failure Cases

To complement the quantitative results in the main text, we provide qualitative examples and a systematic summary of failure modes for both SFT (coverage $= 0.6$, diversity $= 64$, and maximizing the number of questions) and the corresponding GRPO model (16 rollouts) across two representative length groups: **(10, 20)** (within the training-length regime) and **(40, 50)** (longer length regime). The model's prediction errors consistently fall into the following three categories; we did not observe additional or unexpected behaviors (e.g., producing no trajectory or starting from the wrong initial node):

- **Valid but non-shortest path**
- **Did not reach target**
- **Invalid move**

Table 5 summarizes error statistics. Representative visualizations are presented in Figures Figures 14 to 17.

These qualitative findings show that SFT and GRPO exhibit nearly identical failure modes, reinforcing our conclusion that GRPO stabilizes training but does not surpass the performance ceiling established by the best SFT model.

Table 5: Error-type statistics for SFT and GRPO across length groups.

Length Group	Method	Non-Shortest	Not Reach	Invalid Move
(10, 20)	SFT	80.0%	20.0%	0%
(10, 20)	GRPO	88.9%	11.1%	0%
(40, 50)	SFT	45.0%	49.0%	6.0%
(40, 50)	GRPO	43.0%	50.0%	7.0%

Figure 14: Representative failure cases (SFT) for the (10, 20) length group.

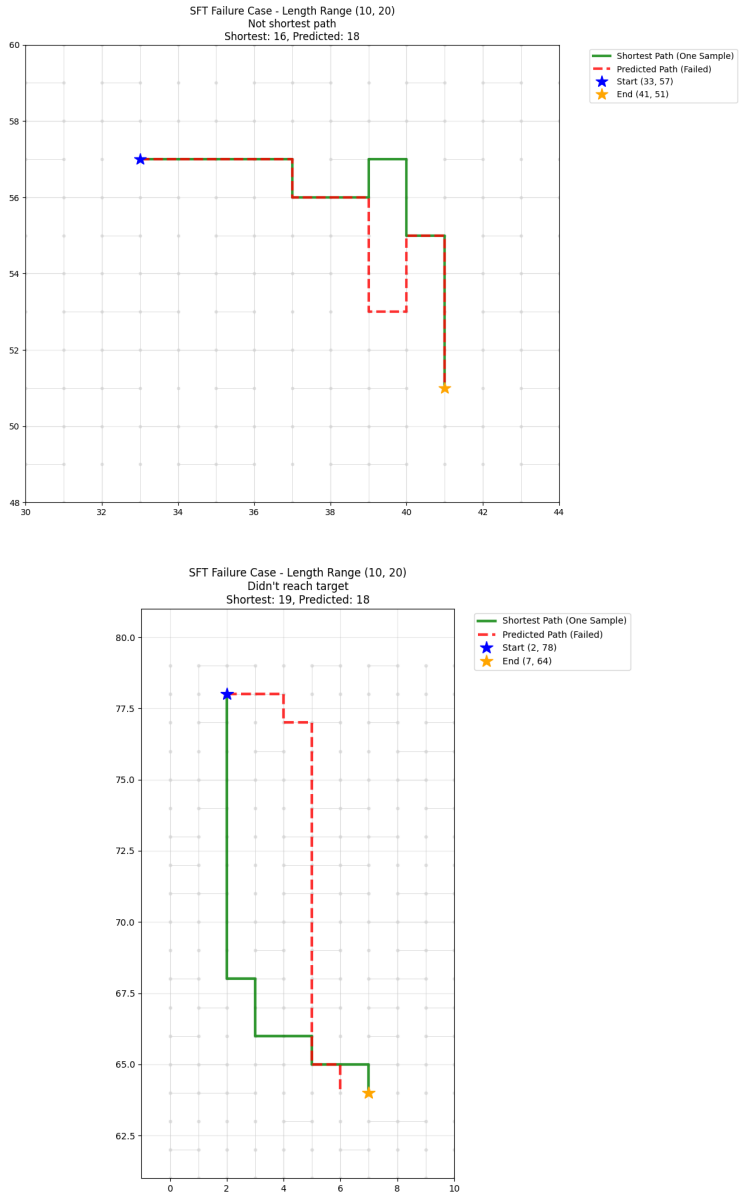


Figure 15: Representative failure cases (GRPO) for the (10, 20) length group.

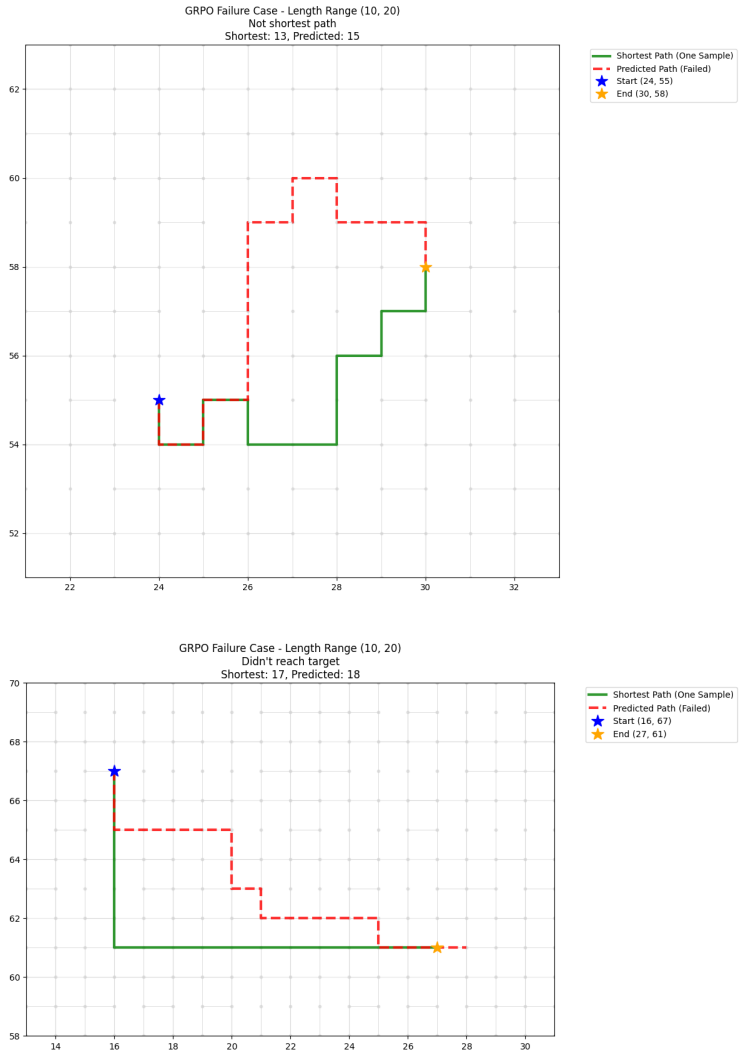
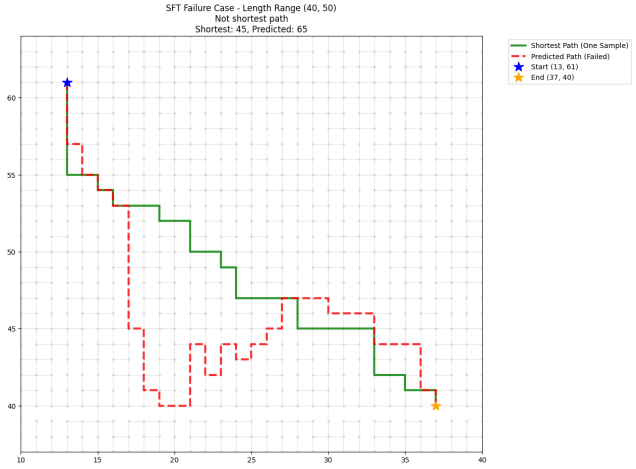
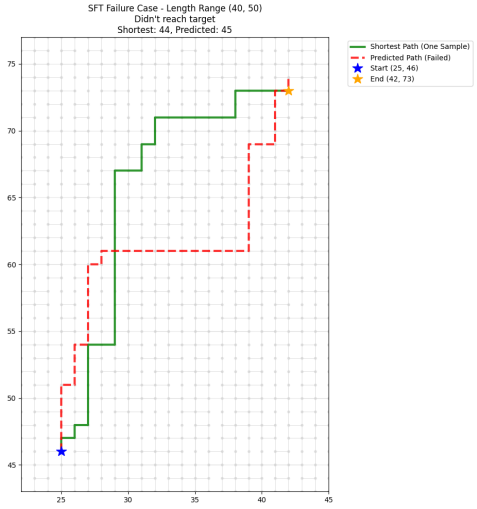


Figure 16: Representative failure cases (SFT) for the (40, 50) length group.





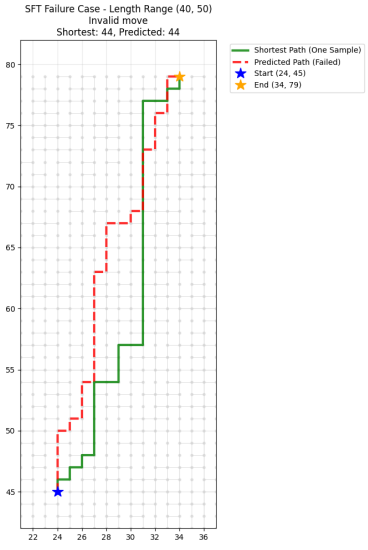


Figure 17: Representative failure cases (GRPO) for the (40, 50) length group.