MUX: Continuous Reasoning via Multiplexed Tokens

Ayhan Suleymanzade[1], **Halil Alperen Gozeten**[2], **Michael Bronstein**[3,1]
İsmail İlkan Ceylan[4,1,3,†], **Jinwoo Kim**[5,†]

[1]AITHYRA [2]University of Michigan [3]University of Oxford [4]TU Wien [5]KAIST
[†]Equal advising

Abstract. Language models solve complex problems by articulating intermediate reasoning steps in natural language. While effective, this process is computationally bottlenecked: each reasoning step conveys only a single subword, and many are spent expressing a thought instead of carrying out computation. We propose MUX, a simple method for high-bandwidth and compact reasoning based on distillation of discrete reasoning into continuous multiplexed tokens in a latent space. Here, each latent token is trained to represent a weighted linear superposition (multiplexing) of a span of discrete reasoning subwords, where this superposition is lossless by construction and the span can be fully recovered (demultiplexing). We prove that simple position-dependent weightings, such as suitable geometric decay, support lossless multiplexing, which in turn prevents shortcut behaviors caused by latent collapse. We further show that multiplexed reasoning can perform parallel exploration in problems that require search. Across 32 evaluation settings spanning four language models, MUX outperforms strong latent reasoning baselines. Ablation and probing analyses further show that the learned latent tokens encode faithful and interpretable reasoning. Our results suggest that lossless superposition as local learning targets constitutes a sufficient condition for achieving strong and efficient latent continuous reasoning.

Code: https://github.com/MisakiTaro0414/mux

1 Introduction

Modern language models are capable of solving complex problems in domains such as mathematics, coding, and commonsense tasks through their *reasoning* mechanism (Hurst et al., 2024; Anil et al., 2023; Touvron et al., 2023). In autoregressive language models, this mechanism typically involves verbalizing intermediate solution steps in natural language before producing the final answer (Nye et al., 2021; Wei et al., 2022; Kojima et al., 2022). However, this mode of operation imposes a strict constraint on the computational bandwidth since each reasoning step transmits only a single subword. Moreover, many of these steps are redundant (Xia et al., 2025; Li et al., 2026b), since the model mirrors problem-solving patterns learned from human-generated corpora, which are inherently more optimized for communication than computation. These limitations motivate the development of approaches that enable higher-bandwidth and more compact reasoning in language models.

Reasoning in continuous latent spaces has emerged as an alternative paradigm, where a language model sequentially predicts continuous vectors instead of subwords before answering (Hao et al., 2025; Xu et al., 2025b). These latent reasoning approaches have high bandwidths, since each step can convey multiple subwords simultaneously by encoding them *in superposition*. This notably enables exploring different problem-solving paths in parallel, offering potential improvements in planning and search tasks (Zhu et al., 2026; Gozeten et al., 2026). Despite such potential, latent reasoning methods have not yet been widely adopted, in part because they are notoriously hard to learn. One class of methods relies on temporal backpropagation of *trajectory-level losses* (Hao et al., 2025; Shen et al., 2025), which tend to produce shortcut or uninformative latent tokens (Zhang et al., 2025c; Cui et al., 2026). Other approaches define *local distillation losses* for each latent token based on discrete reasoning traces (Wei et al., 2026; Kuzina et al., 2026). These methods avoid shortcuts, but at the cost of additional technical complexity such as autoregressive decoders or cache compression, as well as potentially restricting the ability to maintain diverse hypotheses needed for search (Cui et al., 2026). Such apparent tradeoff motivates our key question: *What should constitute the supervision target for continuous latent reasoning?*

We address this question with MUX, a simple and novel training method for high-bandwidth, compact latent

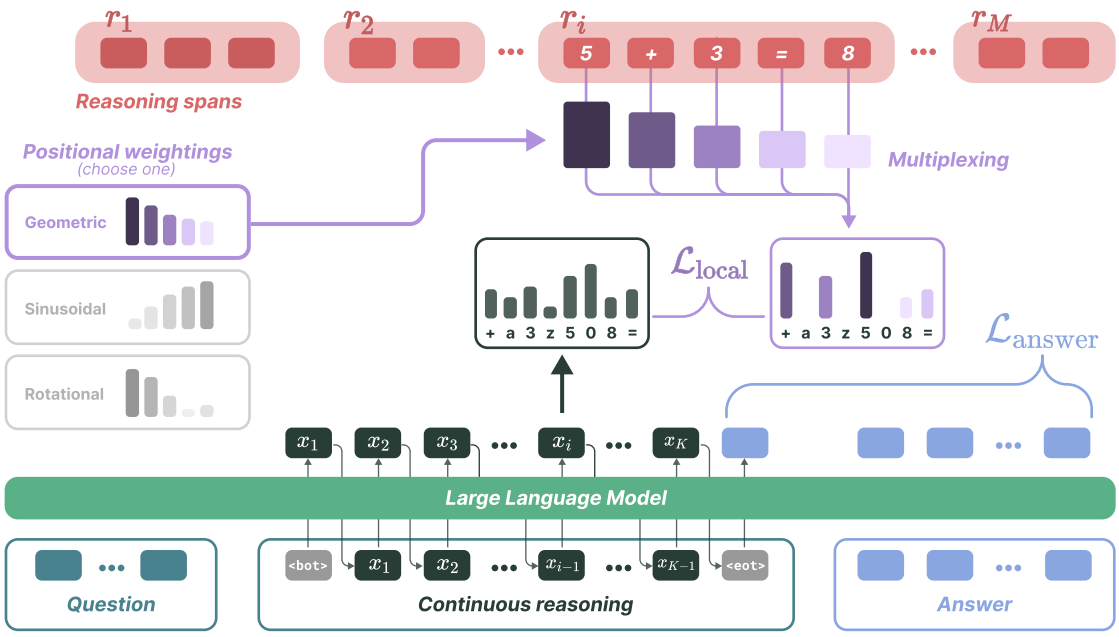


Figure 1 Overview of MUX. Given a question, the language model predicts a sequence of continuous latent reasoning tokens. Each token is linearly projected to the vocabulary space and is trained to represent a local span of discrete reasoning steps. We construct the learning target from each discrete span as a weighted average of one-hot encodings, and train each latent token with local KL divergence loss. The answer is trained with standard cross-entropy loss.

reasoning in language models. Here, we view the local distillation setting, where each latent token is supervised to represent a span of discrete reasoning steps, as learning a fixed-dimensional, continuous signal that encodes a varying-length categorical signal. This naturally connects to *multiplexing* in communication systems, which allows multiple logical signals to share a common physical medium. Drawing inspiration from code-division multiplexing (Fan et al., 2020), we propose to leverage a multiplexed encoding for variable-length categorical signals based on *linear superpositions* of one-hot encodings. Our hypothesis is that (i) local distillation from such multiplexed targets induces faithful latent reasoning, provided that these targets are *lossless* encodings of discrete reasoning spans, while (ii) natively supporting joint encoding of multiple possibilities, thanks to their superposed construction. This is also conceptually simpler than prior methods, as it does not require an auxiliary autoregressive decoder or a compressed cache target. Our main contributions can be summarized as:

(1) **Latent reasoning via multiplexed tokens.** We introduce MUX, a local distillation method for continuous latent reasoning based on multiplexed targets (Figure 1). For each latent token, we define a vocabulary-space target by taking a position-weighted linear superposition of one-hot encodings in its corresponding discrete reasoning span. The model is trained to match this target through a linear-softmax head with a KL loss.

(2) **Lossless multiplexing.** We identify simple classes of positional weightings that guarantee lossless multiplexing, such that each superposed target fully preserves the discrete reasoning span it represents. These include geometric, sinusoidal, and rotary weightings, characterized by a subset-sum separation condition. We show that lossless multiplexing prevents shortcut behaviors found in prior methods caused by latent collapse.

(3) **Parallel search via multiplexing.** We show that, in problems requiring breadth-first search (BFS), multiplexed tokens are expressive enough to represent and update multiple hypotheses simultaneously, owing to their natively superposed construction, thereby implementing each BFS step using a single latent token. The result implies that parallel search can naturally emerge from serial supervision via multiplexing.

(4) **Empirical results.** MUX is the best latent reasoning method across 32 mathematical reasoning settings spanning two training corpora, four language models, and four test sets. It also surpasses strong discrete and continuous reasoning baselines on two search benchmarks. Through probing analysis, we show that the learned latent tokens encode interpretable reasoning content and contribute meaningfully to final prediction.

2 Related work

We provide an overview of related work. An extended discussion can be found in Appendix A.5.

Reasoning in language models. Prior work has shown that language models benefit from making intermediate computations explicit. Early scratchpad methods (Nye et al., 2021) showed that learning intermediate computation steps improves algorithmic problem solving. Chain-of-thought (CoT) prompting (Wei et al., 2022; Kojima et al., 2022) established natural language reasoning as a general mechanism for arithmetic, logical, and commonsense problem solving. Recent work has identified inefficiencies in language reasoning, showing that many reasoning tokens can be pruned (Li et al., 2026b; Zhang et al., 2025a) or compressed (Xia et al., 2025; Li et al., 2026a) with small degradation in accuracy. We share this motivation, but instead of shortening discrete reasoning at inference time, we distill their traces into compact continuous reasoning through training.

Reasoning in continuous latent spaces. A growing line of work explores reasoning in continuous latent spaces, broadly categorized into global and local methods. Global methods supervise the answer token or trajectory endpoint and learn latent reasoning via temporal backpropagation, as in Coconut (Hao et al., 2025) and CODI (Shen et al., 2025). Local methods instead supervise each latent token to represent a span of discrete steps by aligning them in some choice of representation space, typically via auxiliary modules: SIM-CoT (Wei et al., 2026) aligns in autoregressively decoded text space, KaVa (Kuzina et al., 2026) in a key-value cache space. We question their complexity, and instead leverage linearly superposed representations in vocabulary space. While some prior work autoregress vocabulary-space vectors (Zhang et al., 2026; Deng et al., 2025; Tang et al., 2026), we use vocabulary projection only for supervision and autoregress directly in latent space at inference.

Theoretical works studied benefits of continuous latent reasoning, in particular superposition and parallel search (Gozeten et al., 2026; Zhu et al., 2026; Wu et al., 2025). On the other hand, recent analyses caution that latent reasoning need not automatically encode faithful computations, sometimes acting as uninterpretable placeholders or exploiting shortcuts (Zhang et al., 2025c). Cui et al. (2026) find pervasive shortcut behavior in global methods, and report that existing local methods mitigate shortcuts but trade off the ability to maintain diverse hypotheses in latent tokens. Dilgren and Wiegreffe (2026) propose vocabulary projection as a tool for interpreting latent tokens, arguing interpretability itself is a signal of reasoning correctness. Together with earlier probing studies (Cywiński et al., 2025; Liang and Pan, 2026), these motivate supervision methods whose targets are locally decodable, tied to explicit reasoning, and compatible with parallel search by design.

3 MUX: Continuous reasoning via multiplexed tokens

3.1 Problem setup

Language reasoning. Let $\mathcal{V}$ be a discrete vocabulary of subwords and let $\mathcal{V}^*$ be its associated text space. We denote text of length L by $\mathbf{y} = (\mathbf{y}^1, ..., \mathbf{y}^L)$ with each $\mathbf{y}^l \in \mathcal{V}$. Language models generate continuations of a text by autoregressively predicting the next subword. While a language model may directly answer a given question $\mathbf{q} \mapsto \hat{\mathbf{a}}$ by continuation, prompting an intermediate reasoning $\mathbf{r} \in \mathcal{V}^*$ before answering $(\mathbf{q}, \mathbf{r}) \mapsto \hat{\mathbf{a}}$ improves performance. This is, however, computationally inefficient.

Continuous latent reasoning. To overcome the efficiency limitations of discrete reasoning, we reason in a choice of continuous vector space $X = \mathbb{R}^d$, where we denote by X^* the set of vector sequences. For each question $\mathbf{q}$, we would like to train a language model to articulate latent reasoning $\mathbf{x} \in X^*$ by autoregressing on continuous tokens $\mathbf{x}_1, \mathbf{x}_2, ... \in X$ before answering $(\mathbf{q}, \mathbf{x}) \mapsto \hat{\mathbf{a}}$. In practice, we treat X^* as $X^K = \mathbb{R}^{K \times d}$ for a choice of K, which restricts each reasoning to a sequence of K vectors. Following prior work, we assume availability of triples $(\mathbf{q}, \mathbf{r}, \mathbf{a})$ containing discrete reasoning traces $\mathbf{r}$, and use them to learn latent reasoning $\mathbf{x}$ via distillation.

Local distillation. We focus on local distillation, where latent tokens are supervised with local spans of discrete reasoning steps. We assume each discrete trace $\mathbf{r}$ is chunked into spans $(\mathbf{r}_1, ..., \mathbf{r}_M)$ where $\mathbf{r}_i = (r_i^1, \ldots, r_i^{S_i}) \in \mathcal{V}^{S_i}$. For example, in algorithmic and mathematical tasks, each span can be a step of computation, and in natural language, each span can be a sentence. If a trace has more spans than latent tokens, $M > K$, some of the spans

are merged heuristically (Appendix E.3); we thus assume $M \leq K$ onward. If $M < K$, some latent tokens have no aligned span. We let $\mathcal{K}$ denote the subset of latent token positions with nonempty span, noting $|\mathcal{K}| = M$.

In local distillation, latent reasoning $\mathbf{x}$ is trained so that each token $\mathbf{x}_i \in \mathbb{R}^d$ matches a span $\mathbf{r}_i \in \mathcal{V}^*$ in some representation space $\mathcal{Z}$. This goal can be formalized as $f(\mathbf{x}_i) = g(\mathbf{r}_i)$ for some choice of maps $f : \mathbb{R}^d \to \mathcal{Z}$ and $g : \mathcal{V}^* \to \mathcal{Z}$. The representation space and the maps constitute the core design decision of local distillation methods. SIM-CoT (Wei et al., 2026) uses $\mathcal{Z} = \mathcal{V}^*$ with an autoregressive $f : \mathbb{R}^d \to \mathcal{V}^*$ and $g = \mathrm{id}$, and KaVa (Kuzina et al., 2026) takes as $\mathcal{Z}$ the space of key-value cache and performs cache distillation. In contrast, we simply choose $\mathcal{Z}$ as the vocabulary simplex $\Delta^{|\mathcal{V}|-1}$, or the space of $|\mathcal{V}|$-dimensional probability vectors.

3.2 Local distillation by multiplexing

We now present our method for continuous latent reasoning via local distillation. To motivate it, we consider the case where $\mathcal{Z}$ is a fixed-dimensional vector space. Then, $g : \mathcal{V}^* \to \mathcal{Z}$ can be viewed as an operator that combines a variable-dimensional categorical signal $i \mapsto \mathbf{r}_i$ into one, fixed-dimensional continuous signal $i \mapsto g(\mathbf{r}_i)$, which defines the learning target $f^*(\mathbf{x}_i) = g(\mathbf{r}_i)$ for each latent token $\mathbf{x}_i$ via an optimal decoder f^*. Given this observation, it is natural to conceptualize g as a type of *multiplexed* encoding of variable-length categorical signal, $g = \mathsf{mux}$. We now identify the core requirement for local distillation based on multiplexing as follows.

Definition 1 (Multiplexing). A map $\mathsf{mux} : \mathcal{V}^* \to \mathcal{Z}$ is *spanwise injective* if $\mathsf{mux}|_{\mathcal{V}^S}$ is injective for any span length $S \geq 1$. We say latent reasoning $(\mathbf{x}_1, \ldots, \mathbf{x}_K)$ under an optimal decoder $f^* : \mathbb{R}^d \to \mathcal{Z}$ *multiplexes* discrete reasoning spans $(\mathbf{r}_1, \ldots, \mathbf{r}_M)$ if there exists a spanwise injective mux satisfying

$$f^*(\mathbf{x}_i) = \mathsf{mux}(\mathbf{r}_i), \quad \forall i \in \mathcal{K}. \tag{1}$$

Equation (1) requires that each latent token represents a local span of a discrete reasoning trace through multiplexing. As a training objective, it is used to drive $f(\mathbf{x}_i)$ toward $f^*(\mathbf{x}_i) = \mathsf{mux}(\mathbf{r}_i)$ by jointly learning the latent reasoning $\mathbf{x}$ and the decoder f. Injectivity means that the representation mux is lossless, admitting an inverse (demultiplexing). Under lossless multiplexing, the local distillation target $\mathsf{mux}(\mathbf{r}_i)$ is fixed-dimensional, allowing for scalable optimization, and encodes full information of each span $\mathbf{r}_i$, enabling faithful reasoning.

Multiplexing via linear superposition. Constructing a spanwise lossless multiplexer is nontrivial, as it must handle variable-length categorical signals. Here, inspired by code-division schemes in communication systems, we propose a class of simple and training-free multiplexers based on linear superposition of one-hot encodings in the vocabulary space. Concretely, for each discrete reasoning span $\mathbf{r}_i = (r_i^1, \ldots, r_i^{S_i})$, we define

$$\mathsf{mux}(\mathbf{r}_i) := \sum_{j=1}^{S_i} \alpha_j^{(i)} \,\mathrm{onehot}(r_i^j), \qquad \alpha_j^{(i)} := \frac{w_j}{\sum_{\ell=1}^{S_i} w_\ell}, \tag{2}$$

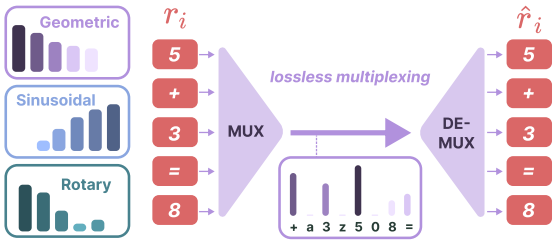


Figure 2 Lossless multiplexing of a span «5+3=8» through position-weighted linear superposition.

where $w_{[\cdot]} : \mathbb{N} \to \mathbb{R}_+$ is a choice of positional weighting.

Because the coefficients $\alpha_j^{(i)}$ are positive and normalized, $\mathsf{mux}(\mathbf{r}_i)$ always lies in the vocabulary simplex $\Delta^{|\mathcal{V}|-1}$. To match such targets following (1), we decode each latent token $\mathbf{x}_i$ through a linear-softmax head

$$f(\mathbf{x}_i) := \mathrm{softmax}(W\mathbf{x}_i/\tau), \qquad W \in \mathbb{R}^{|\mathcal{V}|\times d}, \tag{3}$$

where W is the pretrained language model's unembedding layer and $\tau > 0$ is a temperature variable. At inference time, the model still autoregresses latent tokens $\mathbf{x}_i$ in hidden space, and the vocabulary projection is needed only for their supervision.

Positional weighting. We propose three families of positional weightings $w_{[\cdot]}$ (Figure 2):

(1) *Geometric.* $w_j = \rho^{j-1}$ with decay rate $\rho \in (0, 1)$. Earlier positions receive exponentially more weight, producing a monotonically decaying profile.

(2) *Sinusoidal.* $w_j = \exp(\lambda s_j)$ with scores $s_j = \sin(\frac{\pi}{2} \cdot \frac{j-1}{\max(S-1,1)})$ and scale $\lambda > 0$. This induces a monotonically increasing weighting that peaks near the end of a span.

(3) *Rotary.* $w_j = \exp(\lambda s_j)$ with scores $s_j = \frac{1}{P} \sum_{p=1}^{P} \cos(\theta_p(j-1))$, where $(\theta_p)_{p \le P}$ is a set of positive frequencies analogous to rotary position embeddings (Su et al., 2024). Averaging cosine components across frequencies yields an expressive positional weighting.

All of the above weightings yield spanwise lossless multiplexing when configured properly, as we show in Section 4.1. This outcome is not trivial. The sum $\mathsf{mux}(\mathbf{r}_i)$ records only the *total* mass of each unique subword in a span $\mathbf{r}_i$, so if a subword appears more than once, its positions in the span can be ambiguous. For example, with uniform weighting $\alpha_j = 1/S$, the mass of a subword only counts how many times it appears, dropping the positions.

Therefore, the key to lossless multiplexing is to choose positional weights $\alpha_1, ..., \alpha_S$ such that different sets of positions always produce different total masses. Our theory in Section 4.1 formalizes this as a subset-sum separation condition on the weights, satisfied by all of our weightings for proper hyperparameters. Then, even if a subword appears multiple times, its total mass uniquely determines which positions it occupies, so the original span can be recovered exactly from its multiplexing. In Section 4.2, we show that this property consequently prevents shortcut behaviors that are caused by the collapse of latent tokens.

Training objective. In order to train a language model to perform continuous latent reasoning, we use a composite loss $\mathcal{L} = \mathcal{L}_{\text{answer}} + \beta \mathcal{L}_{\text{local}} + \gamma \mathcal{L}_{\text{global}}$ with weights $\beta, \gamma \ge 0$, where each term corresponds to answer prediction loss, local distillation loss under multiplexed targets (2), and an optional trajectory-level loss, detailed as follows. The answer loss is standard cross entropy $\mathcal{L}_{\text{answer}} = -\log p_\theta(\mathbf{a} \mid \mathbf{q}, \mathbf{x}_1, \ldots, \mathbf{x}_K)$, where p_θ is the likelihood evaluated by the language model. For the local distillation loss, recall that the decoder $f : \mathbb{R}^d \to \Delta^{|\mathcal{V}|-1}$ is a linear projection followed by tempered softmax (3). For each latent token $\mathbf{x}_{i \in \mathcal{K}}$ aligned with a nonempty span $\mathbf{r}_i$, we minimize the KL divergence between the model prediction $f(\mathbf{x}_i)$ and the multiplexed target $\mathsf{mux}(\mathbf{r}_i)$:

$$\mathcal{L}_{\text{local}} = \frac{1}{|\mathcal{K}|} \sum_{i \in \mathcal{K}} \mathrm{KL}\big(\mathsf{mux}(\mathbf{r}_i) \,\big\|\, f(\mathbf{x}_i)\big)\,. \tag{4}$$

Lastly, following Shen et al. (2025), we use an optional trajectory-level loss that aligns the hidden features at the answer token in the language model with continuous reasoning, with respect to those from a model with discrete reasoning, trained with standard next-token prediction on discrete reasoning traces. We employ parameter sharing between the two models, which offers efficiency. The trajectory-level loss provides learning signal for the "spare" tokens $\mathbf{x}_{M+1}, ..., \mathbf{x}_K$ when $M < K$, which lack local targets. For the respective hidden features $\mathbf{h}^{\text{cont}}_{\text{answer}}$ and $\mathbf{h}^{\text{disc}}_{\text{answer}}$, we use $\mathcal{L}_{\text{global}} = \|\mathbf{h}^{\text{cont}}_{\text{answer}} - \mathrm{sg}(\mathbf{h}^{\text{disc}}_{\text{answer}})\|_2^2$, where $\mathrm{sg}(\cdot)$ denotes the stop-gradient operator. Together, the composite loss provides direct learning signal for every latent token as well as answer prediction.

4 Theoretical analysis

We organize the theory around the utility of multiplexing for latent reasoning. In Section 4.1, we ask when multiplexing is lossless, and which positional weightings satisfy this criterion. Losslessness guarantees that every latent token encodes faithful computation without degrading into uninformative placeholders. In Section 4.2, we make this precise and show that multiplexing prevents latent collapse. In Section 4.3, we prove that multiplexed tokens can implement parallel search by encoding an entire search frontier. All proofs are in Appendix C.1.

4.1 Lossless multiplexing

Consider multiplexing a discrete reasoning span $\mathbf{r}_i = (r_i^1, ..., r_i^S)$ into a continuous token $\mathsf{mux}(\mathbf{r}_i)$ using normalized masses $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_S)$ (2). Each span $\mathbf{r}_i$ is a short sequence of reasoning tokens, and the weights α_j determine how much each position contributes to the resulting latent representation. Our goal is to identify conditions that make mux spanwise lossless or injective (Definition 1). The following quantity will be central in our results:

Definition 2 (**Subset-sum separation**). Let $\mathcal{C}_S$ be the set of all nonzero sequences $\mathbf{c} = (c_1, ..., c_S)$ taking values in $\{-1, 0, 1\}$ and consider the following measure of subset-sum collisions:

$$\mathcal{E}(\boldsymbol{\alpha}) \coloneqq \min_{\mathbf{c} \in \mathcal{C}_S} \left| \sum_{j=1}^{S} c_j \alpha_j \right| \tag{5}$$

Intuitively, $\mathcal{E}(\boldsymbol{\alpha}) > 0$ if and only if there are no distinct subsets of $\{1, ..., S\}$ having an identical total mass. We now characterize the exact criterion for lossless multiplexing as follows.

Proposition 3 (**Span-level lossless multiplexing**). *Assume $|\mathcal{V}| > 1$ and fix a span length S. Then the map $\mathsf{mux} : \mathcal{V}^S \to \Delta^{|\mathcal{V}|-1}$ is injective if and only if $\mathcal{E}(\boldsymbol{\alpha}) > 0$.*

The result shows that every finite span of discrete reasoning can be recovered exactly (demultiplexed) from its weighted linear superposition if the subset sums of the weights never collide. Based on this single-span case, we now consider an extension to full reasoning trace; the only additional ingredient is that a full reasoning trace is chunked into several spans, possibly of different lengths.

Corollary 4 (**Trace-level lossless multiplexing**). *Let $\mathbf{r} = (\mathbf{r}_1, \ldots, \mathbf{r}_M)$ be a discrete reasoning trace where each $\mathbf{r}_i$ has length S_i and normalized masses $\boldsymbol{\alpha}^{(i)}$. If $\mathcal{E}\big(\boldsymbol{\alpha}^{(i)}\big) > 0$ for every i, then $\mathbf{r}$ is uniquely recoverable from the collection of multiplexed targets $\mathsf{mux}(\mathbf{r}_i)$ together with S_i and $\boldsymbol{\alpha}^{(i)}$.*

We now identify the hyperparameter choices for our positional weightings (Section 3.2) that support lossless multiplexing. This can be characterized compactly: geometric weights admit an exact algebraic criterion, and the remaining exponential weights are injective if the scores are distinct.

Proposition 5 (**Weightings for lossless multiplexing**).

(i) Geometric. *For $w_j = \rho^{j-1}$, multiplexing is injective iff ρ is not a root of any nonzero polynomial $\sum_{j=1}^{S} c_j x^{j-1}$ with coefficients $c_j \in \{-1, 0, 1\}$. If $\rho \in (0, 1)$ is rational, this holds for all finite S.*

(ii) Exponential. *For $w_j = \exp(\lambda s_j)$, if span length $S \geq 2$ and $s_1, ..., s_S$ are pairwise distinct, then multiplexing is injective for all but finitely many values of λ.*

The sinusoidal and rotary weightings are special cases of the exponential $w_j = \exp(\lambda s_j)$, and so the above result implies that sinusoidal weighting is generally lossless. For rotary weightings, we prove a simple sufficient condition that all of its frequencies lie on the first decreasing branch of cosine. Together, these results show that simple weightings can achieve spanwise lossless multiplexing.

Corollary 6 (**Sinusoidal and rotary weightings for lossless multiplexing**).

(i) For any $S \geq 2$, sinusoidal weighting yields an injective multiplexing for all but finitely many λ.

(ii) If $0 < \theta_p(S-1) < \pi \; \forall p$, then rotary weighting yields an injective multiplexing for all but finitely many λ.

Finite precision. The results above are under exact arithmetic. In practice, multiplexing is done in finite precision, so it is natural to ask whether our findings remain meaningful. Let ε_{fp} be the worst-case error between multiplexed target and its finite-precision rounding. In Appendix C.2, we show that the separation margin $\mathcal{E}(\boldsymbol{\alpha})$ also governs numerical error: if $\varepsilon_{\text{fp}} < \mathcal{E}(\boldsymbol{\alpha})/2$, then the original span remains exactly recoverable. Under the standard unit-roundoff model (Goldberg, 1991; Higham, 2002), ε_{fp} admits an $O(Su)$ bound for span length S and arithmetic precision u. For our default geometric weighting $\rho = 0.9$ in `float32`, multiplexing is lossless for all span lengths faced in experiments ($S \leq 11$).

4.2 Latent diversity

Intuitively, lossless multiplexing is beneficial as it enforces latent tokens to hold meaningful computation. We make this intuition precise and show that MUX guarantees diversity of latent tokens, avoiding semantic homogenization of latent reasoning shown by Wei et al. (2026) for global methods.

Definition 7 (**Latent collapse**). A continuous reasoning $\mathbf{x}$ exhibits *collapse at level* $\varepsilon \geq 0$ if

$$\frac{1}{|\mathcal{K}|^2} \sum_{i,j \in \mathcal{K}} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \leq \varepsilon.$$

Definition 8 (**Target diversity**). The *target diversity* of discrete reasoning $\mathbf{r}$ under multiplexing is $\mathcal{D} := \min_{i,j \in \mathcal{K}, i \neq j} \|\mathsf{mux}(\mathbf{r}_i) - \mathsf{mux}(\mathbf{r}_j)\|_1$.

Whenever two discrete spans are distinct $\mathbf{r}_i \neq \mathbf{r}_j$ and the positional weighting is lossless $\mathcal{E}(\boldsymbol{\alpha}) > 0$, Proposition 3 guarantees $\mathsf{mux}(\mathbf{r}_i) \neq \mathsf{mux}(\mathbf{r}_j)$, so $\mathcal{D} > 0$. We now show that local distillation on diverse targets forces diversity in latent tokens. This is empirically supported in Appendix B.2.

Proposition 9 (**Non-collapsing guarantee for multiplexed distillation**). *Let* $\widetilde{W} := W/\tau$ *be the scaled readout matrix. Suppose* $\mathcal{D} > 0$, $\|\widetilde{W}\|_{\mathrm{op}} > 0$, *and* $\mathcal{L}_{\mathrm{local}} \leq \delta < \mathcal{D}^2/8$. *Then*

$$\frac{1}{|\mathcal{K}|^2} \sum_{i,j \in \mathcal{K}} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \;\geq\; \frac{|\mathcal{K}| - 1}{|\mathcal{K}|} \left(\frac{\mathcal{D} - 2\sqrt{2\delta}}{\|\widetilde{W}\|_{\mathrm{op}}\, C_{|\mathcal{V}|}} \right)^2, \tag{6}$$

where $C_{|\mathcal{V}|}$ *depends only on* $|\mathcal{V}|$. *Thus, latent tokens cannot collapse at any level below the right-hand side.*

4.3 Parallel search with multiplexed reasoning

We now consider search problems where each latent token has to represent a *set* of hypotheses. In a graph reachability problem, there may be several nodes that have been explored and are waiting to be expanded. A continuous token can, in principle, carry such a set all at once in superposition, instead of forcing the model to commit to one possibility. We show that MUX preserves this advantage.

As a setup, consider the depth-H reachability problem on a finite directed graph $G = (\mathcal{N}, E)$: given a source node $s \in \mathcal{N}$ and a target node $t \in \mathcal{N}$, the task is to determine whether there is a directed path $s \to t$ of length $\leq H$. A standard breadth-first search (BFS) maintains two sets at each step k: the frontier node set F_k discovered for the first time, and the node set U_k discovered so far. Denoting by $N^+(B)$ the out-neighborhood of a node set B, each BFS step updates, from $F_0 = \{s\}, U_0 = \{s\}$:

$$F_{k+1} = N^+(F_k) \setminus U_k, \qquad U_{k+1} = U_k \cup F_{k+1}, \qquad k = 0, \ldots, H-1.$$

Suppose the discrete reasoning at step k is $\mathbf{r}_k = (r_k^1, \ldots, r_k^{|F_k|})$ that lists the elements of F_k in an arbitrary order. In this setting, the object of interest is which nodes are in F_k, which can be fully encoded with multiplexing $\mathsf{mux}(\mathbf{r}_k) = \frac{1}{|F_k|} \sum_{j=1}^{|F_k|} \mathsf{onehot}(r_k^j)$ as a uniform distribution over F_k. We now prove that this target is expressive enough to carry and expand an entire frontier F_k together with the discovered set U_k, thus implementing BFS.

Proposition 10 (**Parallel BFS with multiplexing**). *There exists a sequence of continuous tokens* $(\mathbf{x}_0, \ldots, \mathbf{x}_H)$ *such that, for every* $k \leq H$*:*

(i) $\mathbf{x}_k$ *is a deterministic function of* $\mathbf{x}_{k-1}$ *and* G,

(ii) F_k *and* U_k *can be recovered from* $\mathbf{x}_k$, *and so reachability* $\mathbf{1}(t \in U_H)$ *can be recovered from* $\mathbf{x}_H$,

(iii) whenever $F_k \neq \varnothing$, $\mathsf{mux}(\mathbf{r}_k)$ *can be recovered from* $\mathbf{x}_k$, *up to arbitrary precision with softmax.*

The result implies that parallel search can naturally emerge from serial supervision via multiplexing.

Table 1 Mathematical reasoning test accuracies (%). † and ‡ are from Shen et al. (2025) and Kuzina et al. (2026), respectively. We underline MUX when it outperforms SFT-CoT. MUX reports ±1 std. over 3 seeds. We did not conduct iCoT/Coconut OOD tests on NL due to their low ID scores.

Method	GSM8K-AUG				GSM8K-AUG-NL			
	ID	SVAMP	GSM-Hard	MultiArith	ID	SVAMP	GSM-Hard	MultiArith
GPT-2								
SFT-CoT	44.1†	41.8†	9.8†	90.7†	34.2	36.9	7.1	88.7
No-CoT†	19.1	16.4	4.3	41.1	19.1	16.4	4.3	41.1
Latent reasoning								
iCoT	30.1†	29.4†	5.7†	55.5†	3.2	–	–	–
Coconut	34.1†	36.4†	7.9†	82.2†	24.9	–	–	–
CODI	43.7	42.9	9.9	92.8	34.1	30.8	6.8	58.9
SIM-CoT	42.6	42.6	9.4	92.8	30.9	27.5	6.5	53.9
MUX	**48.1** ±0.3	**45.0** ±0.7	**10.6** ±0.5	**93.0** ±0.8	**37.4** ±0.2	**36.7** ±0.7	**8.9** ±0.4	**72.4** ±1.6
LLaMA 3.2 1B-Instruct								
SFT-CoT	61.6†	66.7†	15.6†	99.3†	53.2	62.9	13.3	98.5
No-CoT†	30.9	44.1	7.1	70.9	30.9	44.1	7.1	70.9
Latent reasoning								
iCoT	19.0†	40.9†	4.4†	39.0†	15.2	–	–	–
Coconut	45.3†	48.8†	9.9†	90.1†	24.2	–	–	–
CODI	55.6	61.1	12.8	96.1	47.9	55.3	11.3	96.7
SIM-CoT	56.1	61.5	12.7	96.2	28.4	43.0	6.6	59.4
MUX	**56.7** ±0.5	**63.6** ±1.0	**13.0** ±0.2	**98.5** ±0.9	**50.3** ±0.3	**57.5** ±0.6	**11.6** ±0.2	**96.9** ±0.6
Latent reasoning via Jacobi iterations								
PCCoT	53.5	57.6	**12.9**	97.2	50.1	54.6	12.2	96.8
KaVa‡	56.5	58.9	12.7	–	55.7	58.6	12.8	–
MUX	**58.0** ±0.5	**61.8** ±0.5	**12.9** ±0.4	**98.7** ±0.7	**57.2** ±0.6	**60.6** ±2.0	**13.4** ±0.5	**99.2** ±0.3

5 Experiments

We evaluate MUX on mathematical reasoning (Section 5.1), verify its parallel search capabilities (Section 5.2), and analyze the role of key design choices (Section 5.3). Interpretability and attention analysis can be found in Appendices B.2 and B.3, and training cost analysis can be found in Appendix B.4.

5.1 Mathematical reasoning

Setup. We follow the protocol of prior work and, for training, use two reasoning-augmented mathematical corpora built upon GSM8K (Cobbe et al., 2021). GSM8K-AUG (Shen et al., 2025) includes structured reasoning from GPT-4 (Achiam et al., 2023), whereas GSM8K-AUG-NL (Deng et al., 2023) includes informal linguistic reasoning. We use four test sets: GSM8K test split which is in-domain, and out-of-domain arithmetic datasets SVAMP (Patel et al., 2021), GSM-Hard (Gao et al., 2023), and MultiArith (Roy and Roth, 2015) to test for transferability under distribution shift. We mainly use GPT-2 (Radford et al., 2019) and LLaMA 3.2 1B-Instruct (Meta, 2024) as backbone language models for MUX and baselines, and post-train them via LoRA (Hu et al., 2022). To assess scalability, we also test larger backbones LLaMA 3.2 3B and 3.1 8B on GSM8K-AUG following the protocol of Wei et al. (2026); we were unable to train them on GSM8K-AUG-NL due to resource limits, as reasoning traces therein are considerably longer. For MUX and baselines, we mainly follow the setup of Shen et al. (2025), generating six latent tokens sequentially. For improved scalability, we also experiment with the setup of Wu et al. (2025) where 24 latent tokens are generated in parallel via three Jacobi iterations (Ortega and Rheinboldt, 2000), using LLaMA 3.2 1B-Instruct as backbone. CommonsenseQA (Talmor et al., 2019) and StrategyQA (Geva et al., 2021), which are non-mathematical, have been tested in prior work (Shen et al., 2025; Wu et al., 2025; Wei et al., 2026), but are known to produce high-variance, unreliable results for latent reasoning methods (Shen et al., 2025; Wu et al., 2025). We therefore omit them.

Table 2 Scaling to larger backbones on GSM8K-AUG (%). ◇ results from Wei et al. (2026). We underline MUX when it outperforms SFT-CoT. Single runs due to resource limits.

Method	LLaMA 3.2 3B				LLaMA 3.1 8B			
	ID	SVAMP	GSM-Hard	MultiArith	ID	SVAMP	GSM-Hard	MultiArith
SFT-CoT◇	71.5	71.0	17.0	98.3	71.7	73.1	16.5	98.3
No-CoT◇	38.3	52.9	9.5	88.7	39.5	55.3	9.8	88.0
CODI◇	60.8	73.3	14.3	98.7	61.1	78.1	15.5	99.5
SIM-CoT	62.3	74.9	14.6	98.8	64.1	79.4	16.3	**100.0**
MUX	**65.0**	**77.1**	**15.2**	**100.0**	**68.1**	**80.1**	**17.1**	**100.0**

We compare against non-reasoning, discrete-reasoning, and latent-reasoning baselines. SFT-CoT is supervised on discrete reasoning; No-CoT predicts only the answer; iCoT (Deng et al., 2023) internalizes discrete reasoning into a forward pass; Coconut (Hao et al., 2025) and CODI (Shen et al., 2025) rely on trajectory-level losses for latent reasoning; SIM-CoT (Wei et al., 2026) adds local distillation via an autoregressive decoder. In the parallel decoding setting, we test latent methods PCCoT (Wu et al., 2025) and KaVa (Kuzina et al., 2026) which are developed in the setting.

Results. Tables 1 and 2 show the results. MUX achieves the best latent reasoning performance in all 32 settings, surpassing both global (iCoT, Coconut, CODI, PCCoT) and local (SIM-CoT, KaVa) distillation methods for latent reasoning often by a large margin. Strikingly, MUX even outperforms discrete-reasoning SFT-CoT in 15 cases spanning all model scales and both in-domain and out-of-domain evaluations. This result is surprising since it shows that MUX is able to outperform the target of distillation, in a computationally efficient manner since generating six latent reasoning tokens corresponds to roughly 2.4× and 5.9× fewer reasoning tokens than SFT-CoT on GSM8K-AUG and GSM8K-AUG-NL, respectively. We conjecture that multiplexing for local distillation regularizes the language models to exhibit good generalization behaviors, while acquiring efficiency via compact superposed reasoning. Overall, the results suggest that MUX is a simple method that learns strong, generalizable, and efficient latent reasoning that scales with language model sizes. We further disentangle the effect of local supervision from that of global distillation in Appendix B.1, where the multiplexed target with γ=0 consistently surpasses SIM-CoT under the same regime.

5.2 Parallel search

Setup. We evaluate MUX on tasks that require search, aiming to verify our theoretical results in Section 4.3. We consider two benchmarks, each naturally cast as a depth-H reachability problem on a finite directed graph so that the BFS frontier and discovered set are well defined. We set the number of latent tokens equal to graph depth. In this setting, a sequential strategy tracking a single hypothesis per token cannot explore all states within this budget. Therefore, accuracy gains reflect the model's ability to represent and update multiple candidates in parallel. Further details are in Appendix D.

MNNS. The minimum nonnegative sum (MNNS) task (Gozeten et al., 2026) asks, given a set of integers $a_1, \ldots, a_H$, for the smallest nonnegative value of $\sigma_1 a_1 + \ldots + \sigma_H a_H$ over signs $\sigma_k = \pm 1$. This can be viewed as a search problem over a directed graph $G = (\mathcal{N}, E)$ whose node set is $\mathcal{N} = \{(k, z) : k \in \{0, \ldots, H\},\ z \text{ a reachable partial sum}\}$, with edges $((k, z), (k{+}1, z \pm a_{k+1})) \in E$. The source is $s = (0, 0)$ and the answer is the minimum nonnegative z such that $(H, z) \in U_H$. At each depth k, the frontier F_k contains all partial-sum states discovered for the first time.

Table 3 Search accuracies (%).

Method	MNNS	Game24
No-CoT	68.4	74.4±2.1
SFT-CoT	84.6±2.1	84.3±1.5
Coconut	92.8±0.6	78.6±2.0
CoT2	98.9±0.3	85.0±1.5
MUX	**99.6**±0.3	**88.7**±1.1

Game of 24. We introduce a new arithmetic search benchmark based on the Game of 24 (Yao et al., 2023). Given C cards drawn from $\{1, \ldots, D\}$ and an operator set $\mathcal{O} \subseteq \{+, -, \times\}$, the task is to determine whether the value 24 is reachable by folding the cards left to right, accumulating with an operator from $\mathcal{O}$. This defines a layered directed graph $G = (\mathcal{N}, E)$ whose nodes at depth k are all reachable values after accumulating the k-th

Table 4 Ablations (LLaMA 1B; GSM8K-AUG).

Method	ID	SVAMP	GSM-Hard	MultiArith
Local distillation only ($\gamma = 0$)				
SIM-CoT	31.6	44.0	7.5	69.5
MUX	**48.7**	**51.2**	**10.6**	**98.9**
Chunking strategy				
None	54.3	**61.3**	12.5	96.5
Deterministic	55.7	60.6	12.6	97.3
Random	**56.6**	**61.3**	**13.0**	**98.3**
Positional weighting				
Uniform	54.2	62.4	12.8	96.1
Rotary	55.0	**64.4**	12.4	98.3
Sinusoidal	55.2	61.1	12.6	**99.4**
Geometric	**57.2**	63.1	**12.9**	98.3

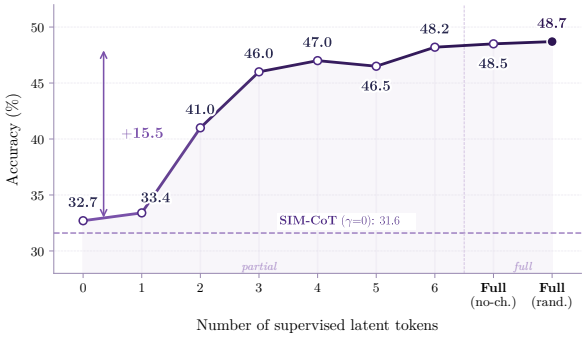


Figure 3 Effect of local distillation loss.

card, and edges correspond to the available operations. The task is binary reachability: $y = \mathbf{1}(24 \in U_{C-1})$. We use $C = 5$ cards, digits $\{1, \ldots, 5\}$, and $\mathcal{O} = \{+, -, \times\}$.

Results. Table 3 shows the results, averaged over 3 seeds. MUX achieves the best search performance in both tasks, directly verifying our claims in Section 4.3 that multiplexed supervision can give rise to latent reasoning that performs parallel search. This supports that MUX is capable of exploring multiple hypotheses in superposition, a core advantage of latent reasoning methods.

5.3 Ablation studies

Contribution of local loss. MUX learns from local and global distillation losses $\beta\,\mathcal{L}_{\text{local}} + \gamma\,\mathcal{L}_{\text{global}}$ with $\beta, \gamma \geq 0$ (Section 3.2). To isolate the role of local distillation via multiplexing, we set $\gamma = 0$, and compare against SIM-CoT, which also uses loss $\beta\,\mathcal{L}'_{\text{local}} + \gamma\mathcal{L}_{\text{global}}$ where $\mathcal{L}'_{\text{local}}$ learns local distillation in autoregressively decoded text space, by setting its $\gamma = 0$. This allows comparing the quality of local loss in a controlled manner. Table 4 shows that MUX indeed has a higher-quality local loss, although being simpler and not requiring an auxiliary autoregressive decoder. A more comprehensive comparison can be found in Appendix B.1.

As a complementary analysis, we take MUX and vary how many latent tokens receive local loss. Figure 3 shows accuracy rises from 32.7% with no distilled tokens to 48.2% with six. The model without distillation still performs latent reasoning, but no tokens are matched with discrete reasoning. This result shows that the gains of MUX come from local distillation, not merely from effective depth of latent reasoning.

Chunking strategy. When $M > K$, the M discrete spans must be merged into K spans before being distilled into K latent tokens (Section 3.1). We compare randomized chunking, fixed deterministic chunking, and no chunking (truncation). Table 4 shows that randomized chunking performs best. We attribute this to its resampled boundaries, which act as a form of structured data augmentation.

Positional weighting. We test the role of positional weighting, which theoretically affects multiplexing losslessness (Sections 3.2 and 4.1). We compare our weightings, proven to be lossless, against (lossy) uniform weighting. Table 4 shows that while lossless weighting is better, the gap is modest, which could be attributed to the fact that many reasoning spans in GSM8K-AUG are highly structured, e.g., «60/2 = 30», so ordering of subwords can often be inferred even from bags of subwords. Nevertheless, the overall gains suggest that losslessness is beneficial in practice. We further train a small MLP to demultiplex spans $\mathbf{r}_i$ from $\mathsf{mux}(\mathbf{r}_i)$. Figure 4 shows that uniform weighting allows nontrivial accuracy, confirming that ordering of subwords is partially recoverable from occurrences. Lossless weightings are still better, agreeing with latent reasoning performances.

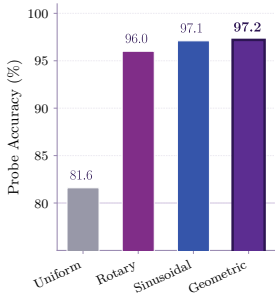


Figure 4 Probe accuracy

6 Conclusion

We introduced MUX, a simple local distillation method for continuous latent reasoning based on position-weighted superposition in vocabulary space. Each latent token is trained to represent an aligned span of discrete reasoning via a multiplexed target that is easy to compute, theoretically grounded, and empirically effective. We showed that suitable positional weightings support exact span recovery, and that multiplexed targets can express parallel search dynamics. Across multiple models and benchmarks, MUX consistently improved upon strong baselines. These results suggest that simple, interpretable local targets can make latent reasoning stronger and easier to train.

Acknowledgments

The authors would like to thank Xingyue Huang, Louis Tichelman, and Angelo Gnazzo for valuable discussions.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. (page 8)

Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. (page 1)

Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*, 2024. (page 16)

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. (page 8)

Yingqian Cui, Zhenwei Dai, Bing He, Zhan Shi, Hui Liu, Rui Sun, Zhiji Liu, Yue Xing, Jiliang Tang, and Benoit Dumoulin. How do latent reasoning methods perform under weak and strong supervision? *arXiv preprint arXiv:2602.22441*, 2026. (pages 1, 3, 16, 18)

Bartosz Cywiński, Emil Ryd, Senthooran Rajamanoharan, and Neel Nanda. Towards eliciting latent knowledge from llms with mechanistic interpretability. *arXiv preprint arXiv:2505.14352*, 2025. (page 3)

Jingcheng Deng, Liang Pang, Zihao Wei, Shichen Xu, Zenghao Duan, Kun Xu, Yang Song, Huawei Shen, and Xueqi Cheng. Latent reasoning in llms as a vocabulary-space superposition. *arXiv preprint arXiv:2510.15522*, 2025. (page 3)

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*, 2023. (pages 8, 9, 16)

Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step. *arXiv preprint arXiv:2405.14838*, 2024. (page 16)

Connor Dilgren and Sarah Wiegreffe. Are latent reasoning models easily interpretable? *arXiv preprint arXiv:2604.04902*, 2026. (pages 3, 16, 18, 19)

Jinping Fan, Yujie Gu, Masahiro Hachimori, and Ying Miao. Signature codes for weighted binary adder channel and multimedia fingerprinting. *IEEE Transactions on Information Theory*, 67(1):200–216, 2020. (page 2)

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International conference on machine learning*, pages 10764–10799. PMLR, 2023. (pages 8, 16)

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021. (page 8)

David Goldberg. What every computer scientist should know about floating-point arithmetic. *ACM computing surveys (CSUR)*, 23(1):5–48, 1991. (page 6)

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *The Twelfth International Conference on Learning Representations*, 2024. (page 16)

Halil Alperen Gozeten, Muhammed Emrullah Ildiz, Xuechen Zhang, Hrayr Harutyunyan, Ankit Singh Rawat, and Samet Oymak. Continuous chain of thought enables parallel exploration and reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. (pages 1, 3, 9, 17, 38, 40)

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Second Conference on Language Modeling*, 2025. (pages 1, 3, 9, 16, 19)

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. (page 40)

David Herel and Tomas Mikolov. Thinking tokens for language modeling. *arXiv preprint arXiv:2405.08644*, 2024. (page 16)

Nicholas J Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002. (page 6)

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. (page 8)

Wei Huang, Yizhe Xiong, Xin Ye, Zhijie Deng, Hui Chen, Zijia Lin, and Guiguang Ding. Fast quiet-STaR: Thinking without thought tokens. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 18771–18781, 2025. (page 16)

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. (page 1)

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. (page 21)

Richard M Karp. Reducibility among combinatorial problems. In *50 Years of Integer Programming 1958-2008: from the Early Years to the State-of-the-Art*, pages 219–241. Springer, 2009. (page 38)

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022. (pages 1, 3, 16)

Anna Kuzina, Maciej Pióro, and Babak Ehteshami Bejnordi. Kava: Latent reasoning via compressed KV-cache distillation. In *The Fourteenth International Conference on Learning Representations*, 2026. (pages 1, 3, 4, 8, 9, 16, 18, 21, 40)

Juncai Li, Ru Li, Yuxiang Zhou, Boxiang Ma, and Jeff Z Pan. Chain of thought compression: A theoritical analysis. *arXiv preprint arXiv:2601.21576*, 2026a. (pages 3, 16)

Zeju Li, Jianyuan Zhong, Ziyang Zheng, Xiangyu Wen, Zhijian Xu, Yingying Cheng, Fan Zhang, and Qiang Xu. Making slow thinking faster: Compressing LLM chain-of-thought via step entropy. In *The Fourteenth International Conference on Learning Representations*, 2026b. (pages 1, 3, 16)

Jia Liang and Liangming Pan. Do latent-cot models think step-by-step? a mechanistic study on sequential reasoning tasks. *arXiv preprint arXiv:2602.00449*, 2026. (page 3)

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. (page 38)

Meta. Llama 3.2: Open-Source AI Models by Meta. https://www.llama.com/docs/model-cards-and-prompt-formats/llama3_2/, 2024. Accessed: 2026-05-13. (page 8)

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021. (pages 1, 3, 16)

James M Ortega and Werner C Rheinboldt. *Iterative solution of nonlinear equations in several variables*. SIAM, 2000. (page 8)

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 2080–2094, 2021. (page 8)

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. (page 8)

Subhro Roy and Dan Roth. Solving general arithmetic word problems. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1743–1752, 2015. (page 8)

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 677–693, 2025. (pages 1, 3, 5, 8, 9, 16, 17, 18, 39, 40)

DiJia Su, Hanlin Zhu, Yingchen Xu, Jiantao Jiao, Yuandong Tian, and Qinqing Zheng. Token assorted: Mixing latent and text tokens for improved language model reasoning. In *Forty-second International Conference on Machine Learning*, 2025. (page 17)

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. (page 5)

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. 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)*, pages 4149–4158, 2019. (page 8)

Yao Tang, Li Dong, Yaru Hao, Qingxiu Dong, Furu Wei, and Jiatao Gu. Multiplex thinking: Reasoning via token-wise branch-and-merge. *arXiv preprint arXiv:2601.08808*, 2026. (pages 3, 17)

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. (page 1)

Martin Tutek, Fateme Hashemi Chaleshtori, Ana Marasović, and Yonatan Belinkov. Measuring chain of thought faithfulness by unlearning reasoning steps. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9946–9971, 2025. (page 36)

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. (page 16)

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. (pages 1, 3, 16)

Xilin Wei, Xiaoran Liu, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Jiaqi Wang, Xipeng Qiu, and Dahua Lin. SIM-cot: Supervised implicit chain-of-thought. In *The Fourteenth International Conference on Learning Representations*, 2026. (pages 1, 3, 4, 7, 8, 9, 16, 18, 21, 40)

Haoyi Wu, Zhihao Teng, and Kewei Tu. Parallel continuous chain-of-thought with jacobi iteration. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 914–926, 2025. (pages 3, 8, 9, 17)

Junhong Wu, Jinliang Lu, Zixuan Ren, Gangqiang Hu, Zhi Wu, Dai Dai, and Hua Wu. LLMs are single-threaded reasoners: Demystifying the working mechanism of soft thinking. In *The Fourteenth International Conference on Learning Representations*, 2026. (page 17)

Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. Tokenskip: Controllable chain-of-thought compression in llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3351–3363, 2025. (pages 1, 3, 16)

Jingxian Xu, Mengyu Zhou, Weichang Liu, Hanbing Liu, Shi Han, and Dongmei Zhang. TwT: Thinking without tokens by habitual reasoning distillation with multi-teachers' guidance. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 16475–16489, 2025a. (page 17)

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot: Soft chain-of-thought for efficient reasoning with llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23336–23351, 2025b. (pages 1, 17)

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot++: Test-time scaling with soft chain-of-thought reasoning. *arXiv preprint arXiv:2505.11484*, 2025c. (page 17)

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023. (pages 9, 16, 38)

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022. (page 16)

Eric Zelikman, Georges Raif Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah Goodman. Quiet-STar: Language models can teach themselves to think before speaking. In *First Conference on Language Modeling*, 2024. (page 16)

Jintian Zhang, Yuqi Zhu, Mengshu Sun, Yujie Luo, Shuofei Qiao, Lun Du, Da Zheng, Huajun Chen, and Ningyu Zhang. Lightthinker: Thinking step-by-step compression. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 13318–13339, 2025a. (pages 3, 16)

Jue Zhang, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. From reasoning to answer: Empirical, attention-based and mechanistic insights into distilled deepseek r1 models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3985–4002, 2025b. (page 36)

Yuyi Zhang, Boyu Tang, Tianjie Ju, Sufeng Duan, and Gongshen Liu. Do latent tokens think? a causal and adversarial analysis of chain-of-continuous-thought. *arXiv preprint arXiv:2512.21711*, 2025c. (pages 1, 3, 18)

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Eric Wang. Soft thinking: Unlocking the reasoning potential of LLMs in continuous concept space. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. (pages 3, 17)

Zhi Zheng, Yu Gu, Wei Liu, Yee Whye Teh, and Wee Sun Lee. Soft-grpo: Surpassing discrete-token llm reinforcement learning via gumbel-reparameterized soft-thinking policy optimization. *arXiv preprint arXiv:2511.06411*, 2025. (page 17)

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*, 2023. (page 16)

Hanlin Zhu, Shibo Hao, Zhiting Hu, Jiantao Jiao, Stuart Russell, and Yuandong Tian. Reasoning by superposition: A theoretical perspective on chain of continuous thought. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. (pages 1, 3, 17)

Appendix

A Extended related work

We expand the related-work discussion from the main text and then summarize the main distinctions in Table 5.

A.1 Reasoning in language

Chain-of-thought (CoT) prompting (Wei et al., 2022; Kojima et al., 2022) showed that asking a language model to articulate intermediate reasoning steps dramatically improves performance on arithmetic, symbolic, and commonsense tasks. Nye et al. (2021) introduced scratchpads as a training-time analogue, where intermediate tokens serve as an explicit computation buffer. Subsequent methods refine how this buffer is generated, verified, or searched, including self-consistency (Wang et al., 2023), STaR (Zelikman et al., 2022), least-to-most prompting (Zhou et al., 2023), Tree of Thoughts (Yao et al., 2023), and PAL (Gao et al., 2023).

At the same time, recent work has shown that such traces are often far more verbose than what the underlying computation requires. TokenSkip (Xia et al., 2025), step-entropy pruning (Li et al., 2026b), LightThinker (Zhang et al., 2025a), and ALiCoT (Li et al., 2026a) all indicate that explicit CoTs contain redundant linguistic overhead. These findings motivate MUX. Our goal is not to compress a generated reasoning at inference time, but to use the redundancy of discrete traces to train a smaller number of continuous reasoning states.

A.2 Implicit reasoning and internalization

A related line of work tries to keep the benefits of intermediate computation while removing the need to emit intermediate language at inference time. iCoT (Deng et al., 2023) and its stepwise extension (Deng et al., 2024) distill explicit reasoning into computations within a single forward pass. Pause tokens (Goyal et al., 2024), Quiet-STaR (Zelikman et al., 2024), Fast Quiet-STaR (Huang et al., 2025), and thinking tokens (Herel and Mikolov, 2024) increase internal compute by inserting special positions that need not correspond to normal language. Compressed CoT (Cheng and Van Durme, 2024) similarly move toward denser reasoning representations. These methods increase effective compute depth without proportionally increasing output length.

A.3 Continuous latent reasoning

Continuous reasoning methods go further by operating in a latent vector space and feeding latent tokens back to the model. We organize the literature by the type of supervision employed.

Global supervision. Coconut (Hao et al., 2025) is the foundational method, replacing discrete reasoning with latent recurrence, forming a "chain of continuous thought." Its training uses a curriculum that gradually transitions from discrete reasoning to fully latent reasoning. Coconut demonstrated that continuous reasoning can support breadth-first-style exploration, but intermediate states are trained only from the final answer loss, which leaves the intermediate latent trajectory unsupervised. CODI (Shen et al., 2025) strengthened this with self-distillation, aligning the continuous and discrete reasoning modes in terms of the hidden state used to predict the final answer. Both methods supervise the reasoning process mainly through the final answer or trajectory endpoint. In our terminology, they are *global* supervision methods that do not supervise what each latent token should represent. Recent empirical analyses show that this lack of intermediate supervision typically leads to shortcut behavior, as globally supervised models can achieve high accuracy without meaningfully relying on the latent reasoning tokens (Cui et al., 2026; Dilgren and Wiegreffe, 2026).

Local supervision via auxiliary components. MUX is closer to *local* supervision methods, which supervise each latent reasoning token with a choice of target. SIM-CoT (Wei et al., 2026) identifies a critical limitation of global supervision: as the number of latent tokens increases, they become homogeneous and training collapses. To address this, SIM-CoT uses an auxiliary autoregressive decoder during training that forces each latent token to encode its aligned discrete reasoning span, providing local supervision. KaVa (Kuzina et al., 2026) takes a different approach by distilling the teacher's compressed key-value (KV) cache into the student model layer by layer. The supervision target is the teacher's cache dynamics, providing a rich but structurally complex signal. Both show that local supervision is effective in mitigating the failure mode of global supervision methods, but each requires additional components, which are an auxiliary decoder or a KV compression module.

Table 5 Qualitative comparison of reasoning methods. ✓ = favorable, ✗ = unfavorable.

Method	Supervision	Lossless	Shortcut-free	Train eff.	Infer. eff.	Interpretable
SFT-CoT	Discrete	✓	✓	✓	✗	✓
CODI	Global	✗	✗	✓	✓	✗
SIM-CoT	Local	✓	✓	✗	✓	✓
KaVa	Local	✗	✓	✓	✓	✓
MUX	Local	✓	✓	✓	✓	✓

Parallelization and efficiency. PCCoT (Wu et al., 2025) improves the efficiency of continuous reasoning by parallelizing sequential predictions of latent tokens via Jacobi iterations, reducing inference latency while maintaining accuracy. SoftCoT (Xu et al., 2025b) generates soft reasoning tokens from a frozen model using a trained projection layer, and SoftCoT++ (Xu et al., 2025c) extends this to test-time compute scaling. Token Assorted (Su et al., 2025) mixes discrete and continuous tokens in a hybrid reasoning trace, allowing the model to choose when to reason in language and when to reason in latent space. TWT (Xu et al., 2025a) distills reasoning from multiple teacher models into habitual latent computation.

Inference-time continuous reasoning. Another related line of work considers reasoning in a continuous space only at *inference time* by modifying the decoding procedure of a pretrained language model. Soft Thinking (Zhang et al., 2026) replaces discrete subword selection with probability-weighted mixtures of vocabulary embeddings. Subsequent work studies its limitations and variants (Wu et al., 2026; Tang et al., 2026; Zheng et al., 2025). Multiplex Thinking (Tang et al., 2026) samples a set of subwords at each reasoning step and aggregates their embeddings into a single continuous *multiplex token*, maintaining vocabulary embedding priors while enabling on-policy RL. These methods are complementary to us. They modify the decoding procedure of a pretrained model, whereas MUX is a training-time method for latent reasoning distillation. This separation lets us use vocabulary space for interpretable supervision without necessarily committing to it during inference time.

A.4 Theoretical foundations of continuous reasoning

A growing theoretical literature formalizes the advantages of continuous over discrete reasoning. Zhu et al. (2026) prove that a two-layer transformer with diameter(G) steps of continuous reasoning can solve directed graph reachability on graph G. The key mechanism is *superposition*: each continuous thought vector encodes multiple search frontiers simultaneously, enabling parallel breadth-first search. In contrast, discrete reasoning requires $O(|V(G)|^2)$ steps with constant-depth transformers. CoT2 (Gozeten et al., 2026) provides complementary results for search problems, showing that supervision against latent token distributions induces parallel exploration. Our work connects to this line in two ways. We prove that our multiplexed targets are *lossless* under standard positional weightings, and that a latent recurrence over such targets can implement exact parallel breadth-first exploration.

A.5 Positioning of MUX

Table 5 summarizes the positioning of MUX relative to representative baselines along five desirable properties: whether the training signal preserves the full discrete reasoning trace (*lossless*), whether latent tokens avoid collapsing into uninformative placeholders (*shortcut-free*), whether training and inference are efficient (*train/infer. eff.*), and whether intermediate latent states can be decoded into human-readable content (*interpretable*).

SFT-CoT directly supervises every discrete reasoning subword via cross-entropy, so losslessness and shortcut avoidance hold by construction. Training is efficient as it processes only the discrete reasoning sequence, but inference requires generating the full trace, which is 2.4–5.9× more subwords than the compact budgets used by continuous methods (Section 5.1). The output is natural language, making it inherently interpretable.

CODI (Shen et al., 2025) performs trajectory-level global distillation, aligning the student's hidden state to the teacher's at the answer position. Whether this preserves the full reasoning content depends on how much the teacher's hidden state actually encodes about reasoning span. Since there is no structural guarantee, we mark it

Table 6 Test accuracies (%) with local distillation only (γ=0).

Method	GSM8K-AUG				GSM8K-AUG-NL			
	ID	SVAMP	GSM-Hard	MultiArith	ID	SVAMP	GSM-Hard	MultiArith
GPT-2								
SIM-CoT (γ=0)	29.5	26.5	6.8	48.9	21.4	23.4	4.9	34.4
MUX (γ=0)	**38.6**	**34.4**	**9.2**	**75.3**	**31.9**	**28.1**	**7.0**	**48.3**
LLaMA 3.2 1B-Instruct								
SIM-CoT (γ=0)	31.6	44.0	7.5	69.5	30.1	44.0	6.7	62.2
MUX (γ=0)	**48.7**	**51.2**	**10.6**	**98.9**	**38.9**	**45.4**	**9.6**	**77.0**
	LLaMA 3.2 3B (GSM8K-AUG)				**LLaMA 3.1 8B** (GSM8K-AUG)			
SIM-CoT (γ=0)	49.1	64.6	12.4	**100.0**	45.6	69.7	11.9	95.0
MUX (γ=0)	**51.5**	**65.4**	**12.7**	97.2	**50.6**	**71.3**	**13.2**	**95.6**

as not lossless. Supervision acts only at the trajectory endpoint, and recent analyses show that this leads to pervasive shortcut behavior. Models trained with global losses can achieve high accuracy without meaningfully relying on intermediate latent tokens (Zhang et al., 2025c; Cui et al., 2026). Training and inference are both efficient, but without per-token supervision, individual latent tokens are hard to interpret (Appendix B.2).

SIM-CoT (Wei et al., 2026) and KaVa (Kuzina et al., 2026) represent two flavors of local supervision. SIM-CoT attaches an auxiliary autoregressive decoder that reconstructs the full aligned reasoning span from each latent token, providing a lossless training signal. KaVa instead distills compressed key-value cache states from the teacher via an importance-based eviction mechanism (R-KV) that selectively discards KV pairs, making its supervision lossy. Both methods mitigate shortcut behavior through step-level local supervision (Cui et al., 2026). KaVa's auxiliary cost is negligible, whereas SIM-CoT's decoder adds 16–32% training overhead (Appendix B.4). Both can produce interpretable latent tokens: SIM-CoT through its decoder output, and KaVa through vocabulary projection of its distilled representations.

MUX is the only method in this comparison that satisfies all five properties. The multiplexed targets are provably lossless under suitable positional weightings (Propositions 3 and 5), and the non-collapsing guarantee (Proposition 9) prevents shortcut behavior. Training adds only a KL divergence over vocabulary distributions (<0.01% of the base cost (Appendix B.4)), and inference uses the same compact latent token budget as other continuous methods. Each latent token can be read out through the pretrained unembedding layer, giving a vocabulary distribution that reflects the aligned reasoning content (Appendix B.2).

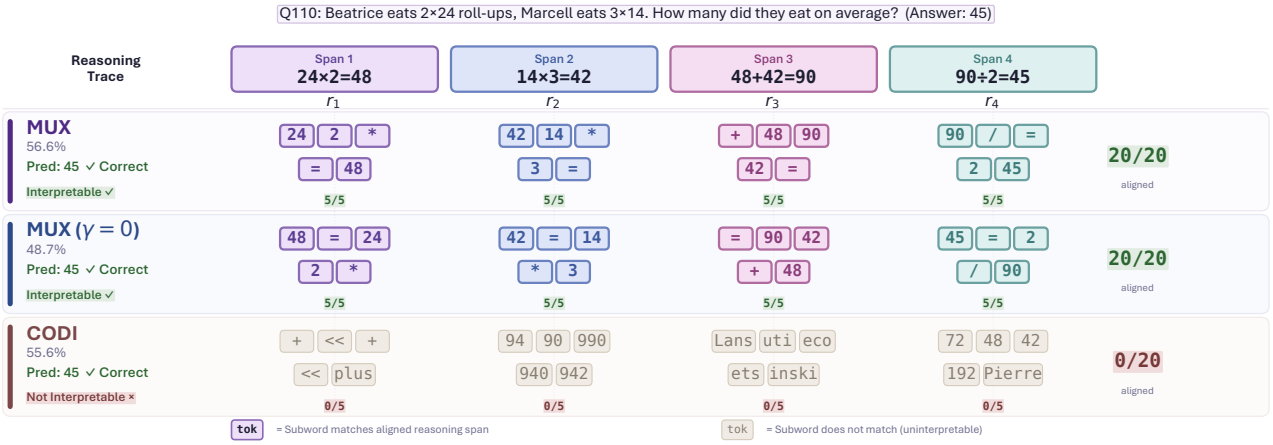
B Supplementary results

B.1 Contribution of local distillation

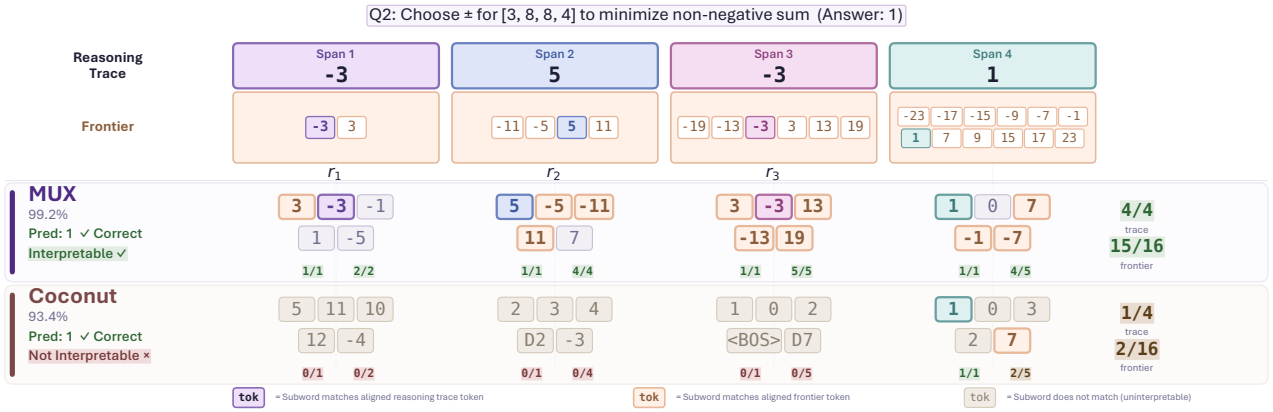
To isolate the contribution of the local supervision, we remove the global trajectory-level distillation loss by setting γ=0 in both MUX and SIM-CoT. Table 6 reports the results across all model-dataset combinations. MUX (γ=0) outperforms SIM-CoT (γ=0) in 23 of 24 settings. These results imply that the multiplexed target is the source of performance gain of MUX, as without any trajectory-level supervision signal, multiplexed local supervision outperforms SIM-CoT's autoregressive decoder-based local supervision. Notably, MUX achieves this with a simpler architecture, since no auxiliary decoder is needed.

B.2 Interpretability analysis

Prior works interpret latent reasoning by projecting latent tokens back into vocabulary space (Shen et al., 2025; Wei et al., 2026; Kuzina et al., 2026), arguing that interpretability itself signals quality of latent reasoning (Dilgren and Wiegreffe, 2026). Figure 5 shows representative examples of this analysis applied to our method. MUX produces interpretable latent tokens in both mathematical reasoning and parallel search settings. For math reasoning, the top decoded subwords correspond to the operands, operators, and intermediate results of each



(a) Mathematical reasoning (GSM8K-AUG)



(b) Parallel search (MNNS)

Figure 5 Top-5 LM-head decoded subwords per latent token.

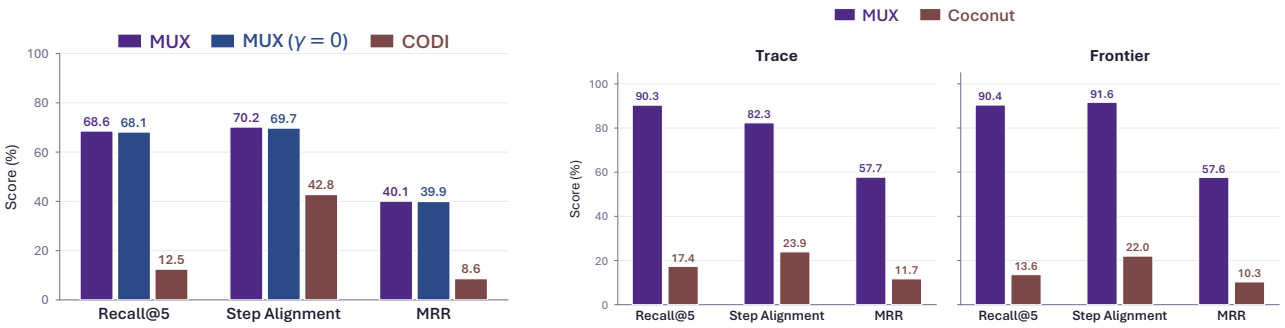
step. For parallel search, the decoded tokens recover the BFS frontier at each depth, and confirm that a single latent token maintains multiple hypotheses in superposition.

In contrast, Coconut and CODI predict the correct answers, but their decoded tokens are uninformative and do not align with the reasoning spans. The latent tokens from MUX are not only useful for prediction, but also easier to read out.

We complement this with quantitative metrics measured across all test examples. Following the vocabulary-projection probing approach used in prior work (Hao et al., 2025; Dilgren and Wiegreffe, 2026), for each latent token $\mathbf{x}_i$ we project it through pretrained unembedding layer of the language model to obtain a distribution over the vocabulary and extract the top-N decoded subwords. We compare these against the reference discrete reasoning span $\mathbf{r}_i$ aligned to slot i and report three metrics:

- *Recall@N*: the fraction of tokens in $\mathbf{r}_i$ that appear among the top-N decoded subwords.
- *Step Alignment*: the fraction of slots for which the top-N decoded set has its highest token overlap with the correct (diagonally aligned) reasoning step.
- *MRR* (Mean Reciprocal Rank): the average of 1/rank over all tokens in $\mathbf{r}_i$, where rank is determined by the decoded vocabulary distribution.

All metrics are micro-averaged over slots and examples. We use N=5 throughout.



(a) Mathematical reasoning (GSM8K-AUG) **(b)** Parallel search (MNNS)

Figure 6 Quantitative interpretability results.

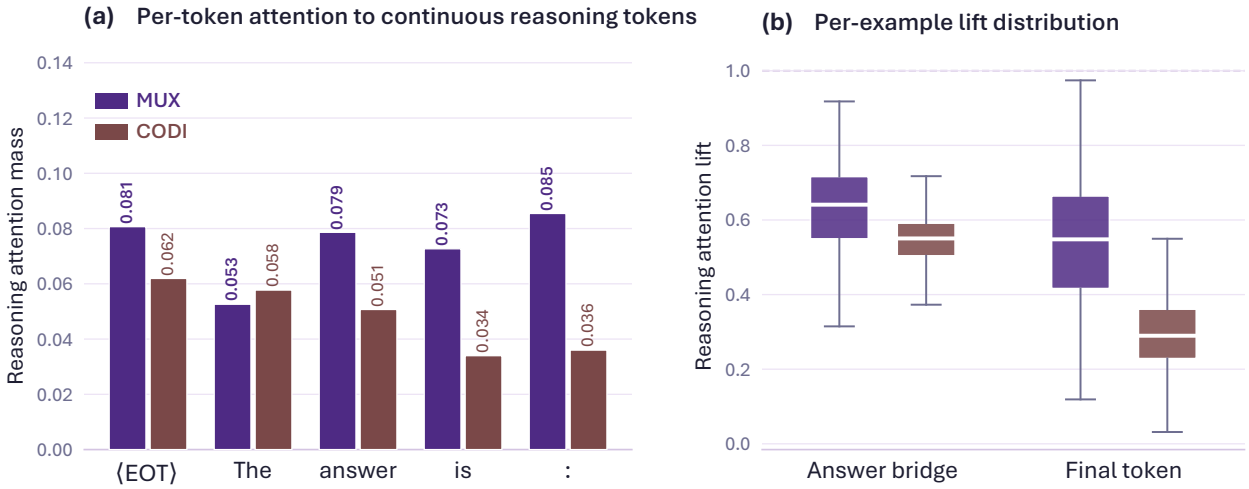


Figure 7 Attention analysis on GSM8K-AUG.

Mathematical reasoning. Figure 6a reports results on LLaMA 3.2 1B-Instruct trained on GSM8K-AUG, evaluated over all test examples. MUX recovers 68.6% of reference tokens (Recall@5) and achieves 70.2% Step Alignment, confirming that latent tokens encode both the content and position of their aligned spans. Removing the global loss (MUX (γ=0)) yields nearly identical scores (68.1%, 69.7%), pointing to local multiplexed supervision as the driver of interpretability. CODI, trained with only global distillation, reaches 12.5% Recall@5 and 8.6% MRR, consistent with its latent tokens not preserving readable reasoning content.

Parallel search. To evaluate whether latent tokens encode search structure, we apply the same metrics to the MNNS task. We define two reference targets for each latent token at depth k: the *trace*, which is the single partial sum along the optimal reasoning path at step k, and the *frontier*, which is the complete set of all partial sums reachable at depth k over every possible sign assignment. Trace metrics test whether the model recovers the particular solution path; frontier metrics test whether the latent state encodes the full distribution of reachable states at each depth. Figure 6b shows the results. MUX achieves 90.3% trace Recall@5 and 82.3% trace Step Alignment, compared to Coconut's 17.4% and 23.9%. The pattern is equally strong for frontier metrics (90.4% vs. 13.6% Recall@5; 91.6% vs. 22.0% Step Alignment). These numbers show that MUX latent tokens encode both the solution path and the reachable states at each depth, consistent with the analysis in Section 4.3.

Table 7 Training cost relative to CODI (LLaMA-1B)

Method	Relative training cost (vs. CODI)	
	GSM8K-AUG	GSM8K-AUG-NL
SFT-CoT	0.56×	0.64×
Coconut	0.44×	0.36×
CODI	1.00×	1.00×
SIM-CoT	1.16×	1.32×
KaVa	≈1.00×	≈1.00×
MUX	≈1.00×	≈1.00×

B.3 Attention analysis

We add an attention-based diagnostic to test whether latent reasoning meaningfully contributes to final answer. On LLaMA 3.2 1B-Instruct trained on GSM8K-AUG, we measure how much the model attends to its latent tokens when producing the answer. We extract last-layer attention on all test examples at the answer-interface tokens {<EOT>, The, answer, is, :} and compute two quantities. Let α_{lat} be the total attention weight on all K latent tokens, and let N_{pre} be the number of total preceding tokens. We define *reasoning attention mass* $= \alpha_{\text{lat}}$, and *reasoning attention lift* $= \alpha_{\text{lat}} \,/\, (K/N_{\text{pre}})$. A lift of 1.0 means the model distributes attention uniformly and values above 1.0 mean latent tokens receive more attention than their share within the context.

Figure 7 reports the results. Panel (a) shows that MUX assigns higher reasoning attention mass than CODI across almost every answer-prediction step. Panel (b) compares the per-example distribution of reasoning attention lift at two scopes: the full answer bridge (all five interface tokens) and the final prediction token (:). MUX achieves higher lift in both cases (0.633 vs. 0.542 at the answer bridge; 0.544 vs. 0.295 at the final token). In addition, on 85.1% of examples MUX routes more attention through latent reasoning tokens at the answer bridge, rising to 91.8% at the final prediction token. Thus, relative to CODI, MUX more effectively utilizes its learned latent reasoning when generating the answer. We leave a theoretical explanation in Appendix C.3. Figure 8 shows a representative last-layer attention map. MUX forms a clear autoregressive chain among its latent tokens before the answer bridge; CODI's attention is diffuse and largely bypasses the latent tokens.

B.4 Training cost analysis

We compare the per-step training FLOPs of each method using the standard approximation (Kaplan et al., 2020): the forward pass costs $\approx 2PL$ and the backward pass $\approx 4PL$, giving a total of $\approx 6PL$ FLOPs per training step, where P is the number of model parameters and L is the sequence length. All self-distillation methods (CODI, SIM-CoT, KaVa, and MUX) process both a teacher sequence of length $L_t = L_q + L_c + L_a$ (question, full chain-of-thought, answer) and a student sequence of length $L_s = L_q + K + L_a$ (question, K continuous tokens, answer). Since the teacher cross-entropy loss is included in the total training loss and gradients flow through both paths, each incurs the full $6PL$ training cost. SFT-CoT trains only on the teacher sequence, and Coconut trains only on the student sequence.

The methods differ only in their auxiliary losses. SIM-CoT (Wei et al., 2026) trains a full auxiliary decoder with $P_{\text{dec}} = P$ parameters on the chain-of-thought tokens, adding $6P_{\text{dec}} \cdot L_c$ FLOPs per step. KaVa (Kuzina et al., 2026) adds a KV-cache matching loss (Eq. 7 in their paper) with cost $\mathcal{O}(MHLd)$ where M is the number of retained KV pairs, H the number of KV heads, L the number of layers, and d the head dimension. MUX adds a multiplexed KL divergence over K vocabulary distributions, costing $\mathcal{O}(K|\mathcal{V}|)$. Both KaVa's and MUX's auxiliary costs are negligible (<0.01% of the base cost). However, KaVa's KV-cache distillation requires an importance-based eviction mechanism (R-KV) that scores and selectively discards teacher KV pairs before matching, introducing additional architectural complexity and a lossy compression step that is absent in MUX.

Table 7 reports the total relative training cost for LLaMA-1B on both GSM8K-AUG (L_q=55, L_c=25, L_a=8, K=6) and GSM8K-AUG-NL (L_q=55, L_c=62, L_a=8, K=6). CODI, KaVa, and MUX have effectively identical training cost on both datasets. SIM-CoT is 16% more expensive on AUG and 32% on AUG-NL, as its decoder overhead scales with chain length.

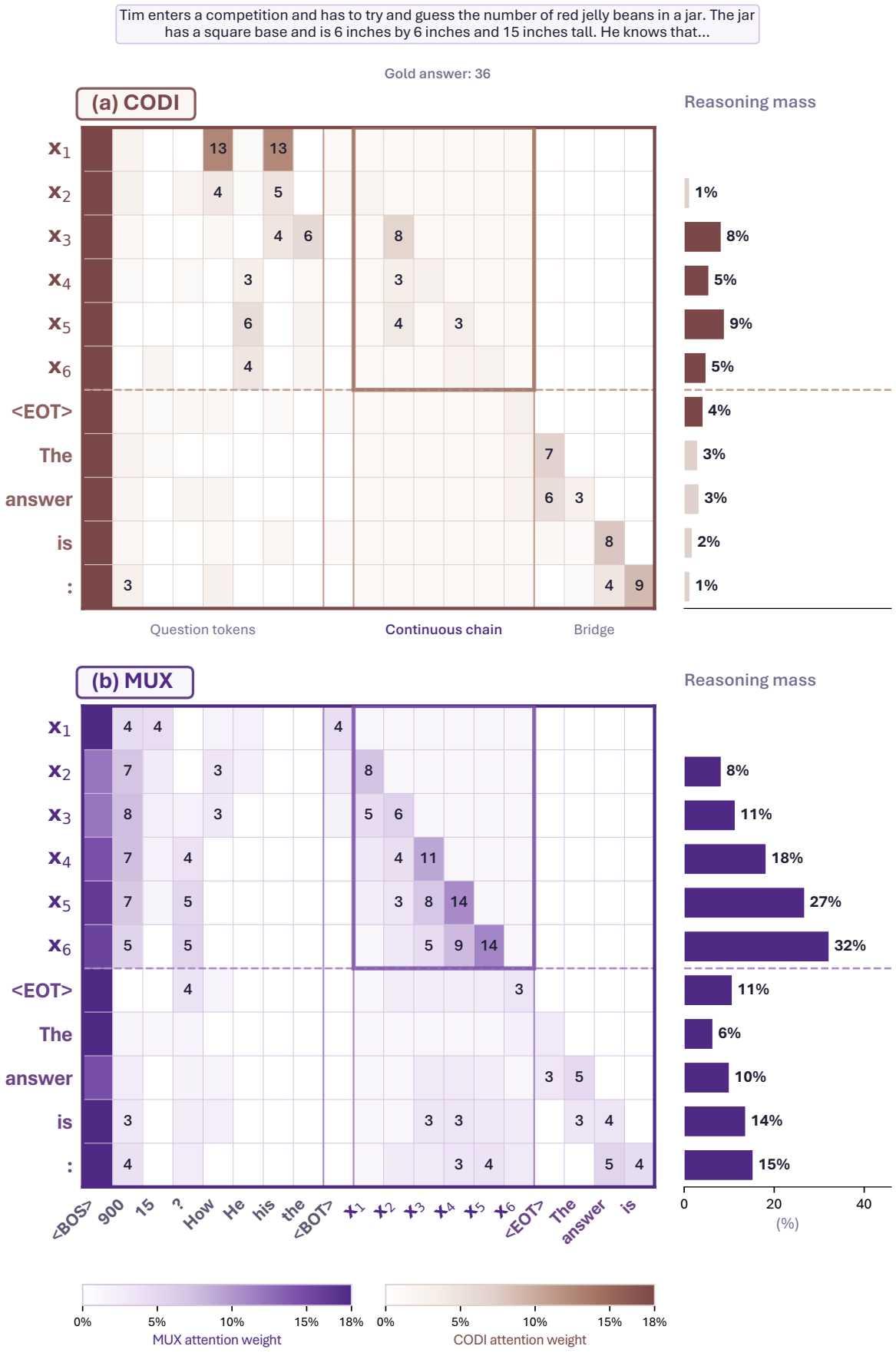


Figure 8 Attention routing through continuous reasoning tokens.

C Proofs and theoretical details

C.1 Proofs of the main results

Proposition 3 (Span-level lossless multiplexing). *Assume $|\mathcal{V}| > 1$ and fix a span length S. Then the map $\mathsf{mux} : \mathcal{V}^S \to \Delta^{|\mathcal{V}|-1}$ is injective if and only if $\mathcal{E}(\boldsymbol{\alpha}) > 0$.*

Proof. We prove both directions.

Sufficiency. Assume $\mathcal{E}(\boldsymbol{\alpha}) > 0$. We will show that mux is injective.

Take two token sequences

$$(r^1, \dots, r^S), \qquad (r'^1, \dots, r'^S)$$

such that

$$\mathsf{mux}(r^1, \dots, r^S) = \mathsf{mux}(r'^1, \dots, r'^S).$$

This means that the two sequences induce exactly the same target distribution over the vocabulary.

For each vocabulary token $v \in \mathcal{V}$, define the set of positions at which v appears:

$$A_v = \{j \in \{1, \dots, S\} : r^j = v\}, \qquad B_v = \{j \in \{1, \dots, S\} : r'^j = v\}.$$

Because the two target distributions are equal, for every $v \in \mathcal{V}$ we have

$$\sum_{j \in A_v} \alpha_j = \sum_{j \in B_v} \alpha_j.$$

Suppose, for contradiction, that $A_v \neq B_v$ for some v. Then the coefficient vector defined by

$$c_j = \begin{cases} 1, & j \in A_v \setminus B_v, \\ -1, & j \in B_v \setminus A_v, \\ 0, & \text{otherwise} \end{cases}$$

is nonzero and satisfies

$$\sum_{j=1}^{S} c_j \alpha_j = \sum_{j \in A_v} \alpha_j - \sum_{j \in B_v} \alpha_j = 0.$$

This contradicts $\mathcal{E}(\boldsymbol{\alpha}) > 0$. Therefore $A_v = B_v$ for every token v.

Now fix any position $j \in \{1, \dots, S\}$. There is exactly one vocabulary token v such that $j \in A_v$, namely $v = r^j$. Since $A_v = B_v$, we also have $j \in B_v$, so $r'^j = v = r^j$. As this holds for every position j, the two sequences are identical. Hence mux is injective.

Necessity. Assume $\mathcal{E}(\boldsymbol{\alpha}) = 0$. Then, by definition, there exists a nonzero coefficient vector

$$\mathbf{c} = (c_1, \dots, c_S) \in \{-1, 0, 1\}^S$$

such that

$$\sum_{j=1}^{S} c_j \alpha_j = 0.$$

Define two subsets

$$A = \{j : c_j = 1\}, \qquad B = \{j : c_j = -1\}.$$

Since $\mathbf{c} \neq \mathbf{0}$, at least one of A or B is non-empty. Moreover,

$$\sum_{j \in A} \alpha_j = \sum_{j \in B} \alpha_j.$$

Choose two distinct vocabulary tokens $u, v \in \mathcal{V}$. Construct two sequences by

$$r^j = \begin{cases} u, & j \in A, \\ v, & j \notin A, \end{cases} \qquad r'^j = \begin{cases} u, & j \in B, \\ v, & j \notin B. \end{cases}$$

Because $A \neq B$, the sequences are different. However, the probability mass of token u under the first sequence is $\sum_{j \in A} \alpha_j$, while under the second sequence it is $\sum_{j \in B} \alpha_j$; these are equal. The same is true for token v, since both distributions sum to 1, and all other tokens have probability 0. Therefore the two sequences induce exactly the same target distribution. Hence mux is not injective.

We have shown that mux is injective if and only if $\mathcal{E}(\boldsymbol{\alpha}) > 0$. □

Corollary 4 (Trace-level lossless multiplexing). *Let $\mathbf{r} = (\mathbf{r}_1, \ldots, \mathbf{r}_M)$ be a discrete reasoning trace where each $\mathbf{r}_i$ has length S_i and normalized masses $\boldsymbol{\alpha}^{(i)}$. If $\mathcal{E}\big(\boldsymbol{\alpha}^{(i)}\big) > 0$ for every i, then $\mathbf{r}$ is uniquely recoverable from the collection of multiplexed targets $\mathsf{mux}(\mathbf{r}_i)$ together with S_i and $\boldsymbol{\alpha}^{(i)}$.*

Proof. Fix $i \in \{1, \ldots, M\}$. By assumption,

$$\mathcal{E}(\boldsymbol{\alpha}^{(i)}) > 0.$$

Therefore, by Proposition 3, the multiplexed target $\mathsf{mux}(\mathbf{r}_i)$, together with the span length S_i and the corresponding masses $\boldsymbol{\alpha}^{(i)}$, uniquely determines the full aligned span

$$\mathbf{r}_i = (r_i^1, \ldots, r_i^{S_i}).$$

This is true for every span $i = 1, \ldots, M$.

Once all spans $\mathbf{r}_i$ have been recovered, the original reasoning trace is obtained by concatenating them in the same order. Thus the ordered tuple

$$\Big((S_i, \boldsymbol{\alpha}^{(i)}, \mathsf{mux}(\mathbf{r}_i))\Big)_{i=1}^{M}$$

determines the full reasoning trace uniquely. □

Proposition 5 (Weightings for lossless multiplexing).

(i) Geometric. *For $w_j = \rho^{j-1}$, multiplexing is injective iff ρ is not a root of any nonzero polynomial $\sum_{j=1}^{S} c_j x^{j-1}$ with coefficients $c_j \in \{-1, 0, 1\}$. If $\rho \in (0, 1)$ is rational, this holds for all finite S.*

(ii) Exponential. *For $w_j = \exp(\lambda s_j)$, if span length $S \geq 2$ and $s_1, ..., s_S$ are pairwise distinct, then multiplexing is injective for all but finitely many values of λ.*

Proof. For geometric weighting,

$$\alpha_j = \frac{\rho^{j-1}}{\sum_{l=1}^{S} \rho^{l-1}}.$$

Let

$$Z_S = \sum_{l=1}^{S} \rho^{l-1}.$$

Since $0 < \rho < 1$, we have $Z_S > 0$.

Take any coefficient vector $\mathbf{c} \in \{-1, 0, 1\}^S$. Then

$$\sum_{j=1}^{S} c_j \alpha_j = \sum_{j=1}^{S} c_j \frac{\rho^{j-1}}{Z_S} = \frac{1}{Z_S} \sum_{j=1}^{S} c_j \rho^{j-1}.$$

Because $Z_S > 0$, this quantity is zero if and only if

$$\sum_{j=1}^{S} c_j \rho^{j-1} = 0.$$

Therefore

$$\mathcal{E}(\boldsymbol{\alpha}) > 0 \quad \Longleftrightarrow \quad \sum_{j=1}^{S} c_j \rho^{j-1} \neq 0 \quad \text{for every } \mathbf{c} \in \{-1, 0, 1\}^S \setminus \{\mathbf{0}\}.$$

Proposition 3 now gives the stated equivalence.

To prove that if $\rho \in (0, 1)$ is rational, then geometric weighting is injective for every finite span length S, suppose $\rho = p/q \in (0, 1)$ is rational in lowest terms and that geometric weighting were not injective. We proved that there would exist a nonzero polynomial

$$P(x) = \sum_{j=1}^{S} c_j x^{j-1}, \qquad c_j \in \{-1, 0, 1\},$$

such that $P(\rho) = 0$. If necessary, divide out the largest power of x so that the constant term is nonzero. The resulting polynomial still has integer coefficients, is nonzero, has constant term ± 1, and has leading coefficient ± 1. By the rational root theorem, any rational root must be an integer divisor of the constant term divided by an integer divisor of the leading coefficient, hence must belong to $\{\pm 1\}$. This contradicts $\rho \in (0, 1)$. Therefore no such polynomial exists, and the weighting is injective.

To prove part (ii), let us introduce the following lemma on exponential polynomials first.

Lemma 11. *Let $\lambda_1, \ldots, \lambda_n \in \mathbb{R}$ be pairwise distinct, and let*

$$f(x) = \sum_{m=1}^{n} b_m e^{\lambda_m x}$$

with real coefficients b_m, not all zero. Then f has at most $n - 1$ real zeros.

Proof. We use induction on n.

If $n = 1$, then

$$f(x) = b_1 e^{\lambda_1 x}$$

with $b_1 \neq 0$, so $f(x) \neq 0$ for all x. Thus the claim holds.

Assume the statement holds for $n - 1$, and consider

$$f(x) = \sum_{m=1}^{n} b_m e^{\lambda_m x} \quad \text{with} \quad \lambda_1 < \lambda_2 < \cdots < \lambda_n.$$

Define

$$g(x) = e^{-\lambda_1 x} f(x) = b_1 + \sum_{m=2}^{n} b_m e^{(\lambda_m - \lambda_1)x}.$$

The functions f and g have the same zeros because $e^{-\lambda_1 x}$ is never zero.

Suppose g has N distinct real zeros. By Rolle's theorem, g' has at least $N - 1$ distinct real zeros. But

$$g'(x) = \sum_{m=2}^{n} b_m (\lambda_m - \lambda_1) e^{(\lambda_m - \lambda_1)x}$$

is again an exponential polynomial, now with $n - 1$ pairwise distinct exponents. By the induction hypothesis, g' has at most $n - 2$ real zeros. Therefore $N - 1 \leq n - 2$, which implies $N \leq n - 1$.

Hence g, and therefore also f, has at most $n - 1$ real zeros. □

Now suppose the scores $s_1, \dots, s_S$ are pairwise distinct, and define

$$\alpha_j(\lambda) = \frac{e^{\lambda s_j}}{\sum_{l=1}^{S} e^{\lambda s_l}}.$$

By Proposition 3, injectivity fails if and only if there exists a nonzero coefficient vector

$$\mathbf{c} = (c_1, \dots, c_S) \in \{-1, 0, 1\}^S$$

such that

$$\sum_{j=1}^{S} c_j \alpha_j(\lambda) = 0.$$

Since the denominator $\sum_l e^{\lambda s_l}$ is strictly positive, this is equivalent to

$$\sum_{j=1}^{S} c_j e^{\lambda s_j} = 0.$$

For fixed nonzero $\mathbf{c}$, the function

$$f_{\mathbf{c}}(\lambda) = \sum_{j=1}^{S} c_j e^{\lambda s_j}$$

is a nonzero exponential polynomial with pairwise distinct exponents s_j. By Lemma 11, $f_{\mathbf{c}}$ has only finitely many real zeros.

There are only finitely many nonzero coefficient vectors in $\{-1, 0, 1\}^S$. Therefore the union of the zero sets of all such functions $f_{\mathbf{c}}$ is finite. Call this union D. If $\lambda \notin D$, then no nontrivial signed sum vanishes, so $\mathcal{E}(\boldsymbol{\alpha}) > 0$. By Proposition 3, the encoding is injective. □

Corollary 6 (Sinusoidal and rotary weightings for lossless multiplexing).

(i) For any $S \geq 2$, sinusoidal weighting yields an injective multiplexing for all but finitely many λ.

(ii) If $0 < \theta_p(S-1) < \pi \; \forall p$, then rotary weighting yields an injective multiplexing for all but finitely many λ.

Proof. Part (i) is immediate because the function

$$u \mapsto \sin\left(\frac{\pi}{2} u\right)$$

is strictly increasing on $[0, 1]$, so the sinusoidal scores are pairwise distinct. For part (ii), assume

$$0 < \theta_p(S-1) < \pi \qquad \text{for every } p = 1, \dots, P.$$

Fix $p \in \{1, \dots, P\}$. For each $j = 1, \dots, S-1$,

$$0 \leq (j-1)\theta_p < j\theta_p < \pi.$$

The cosine function is strictly decreasing on the interval $[0, \pi]$. Hence

$$\cos\big(\theta_p(j-1)\big) > \cos(\theta_p j) \qquad \text{for } j = 1, \dots, S-1.$$

Averaging these inequalities over $p \in \{1, \dots, P\}$ gives

$$\frac{1}{P}\sum_{p=1}^{P} \cos\big(\theta_p(j-1)\big) > \frac{1}{P}\sum_{p=1}^{P} \cos(\theta_p j),$$

that is,

$$s_j > s_{j+1} \qquad \text{for } j = 1, \dots, S-1.$$

Thus the rotary scalar scores are strictly decreasing and therefore pairwise distinct. The injectivity claim then follows immediately from Proposition 5. □

Proposition 9 (**Non-collapsing guarantee for multiplexed distillation**). *Let $\widetilde{W} := W/\tau$ be the scaled readout matrix. Suppose $\mathcal{D} > 0$, $\|\widetilde{W}\|_{\mathrm{op}} > 0$, and $\mathcal{L}_{\mathrm{local}} \le \delta < \mathcal{D}^2/8$. Then*

$$\frac{1}{|\mathcal{K}|^2} \sum_{i,j \in \mathcal{K}} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \;\ge\; \frac{|\mathcal{K}| - 1}{|\mathcal{K}|} \left(\frac{\mathcal{D} - 2\sqrt{2\delta}}{\|\widetilde{W}\|_{\mathrm{op}}\, C_{|\mathcal{V}|}} \right)^2, \tag{6}$$

where $C_{|\mathcal{V}|}$ depends only on $|\mathcal{V}|$. Thus, latent tokens cannot collapse at any level below the right-hand side.

Proof. Let

$$n := |\mathcal{K}|, \qquad \widetilde{W} := W/\tau, \qquad m_i := \mathsf{mux}(\mathbf{r}_i), \qquad q_i := f(\mathbf{x}_i) = \mathrm{softmax}(\widetilde{W}\mathbf{x}_i), \qquad e_i := \|m_i - q_i\|_1.$$

By Pinsker's inequality,

$$e_i \le \sqrt{2\, D_{\mathrm{KL}}(m_i \,\|\, q_i)} \qquad \text{for every } i \in \mathcal{K}.$$

Hence, by Jensen's inequality and the assumption $\mathcal{L}_{\mathrm{local}} \le \delta$,

$$\frac{1}{n} \sum_{i \in \mathcal{K}} e_i \le \frac{1}{n} \sum_{i \in \mathcal{K}} \sqrt{2\, D_{\mathrm{KL}}(m_i \,\|\, q_i)} \le \sqrt{2 \cdot \frac{1}{n} \sum_{i \in \mathcal{K}} D_{\mathrm{KL}}(m_i \,\|\, q_i)} \le \sqrt{2\delta}.$$

For any distinct $i, j \in \mathcal{K}$, Definition 8 and the triangle inequality give

$$\|q_i - q_j\|_1 \ge \|m_i - m_j\|_1 - \|m_i - q_i\|_1 - \|m_j - q_j\|_1 \ge \mathcal{D} - e_i - e_j.$$

Averaging over all ordered pairs $i \ne j$ yields

$$\frac{1}{n(n-1)} \sum_{i \ne j} \|q_i - q_j\|_1 \ge \mathcal{D} - \frac{1}{n(n-1)} \sum_{i \ne j} (e_i + e_j).$$

Since

$$\frac{1}{n(n-1)} \sum_{i \ne j} (e_i + e_j) = \frac{2}{n} \sum_{i \in \mathcal{K}} e_i,$$

we obtain

$$\frac{1}{n(n-1)} \sum_{i \ne j} \|q_i - q_j\|_1 \ge \mathcal{D} - \frac{2}{n} \sum_{i \in \mathcal{K}} e_i \ge \mathcal{D} - 2\sqrt{2\delta}.$$

Because $\delta < \mathcal{D}^2/8$, the right-hand side is strictly positive.

Let σ denote the softmax map. For $\mathbf{z} \in \mathbb{R}^{|\mathcal{V}|}$, write

$$J_\sigma(\mathbf{z}) = \mathrm{Diag}(\sigma(\mathbf{z})) - \sigma(\mathbf{z})\sigma(\mathbf{z})^\top$$

for its Jacobian matrix, and define

$$C_{|\mathcal{V}|} := \sup_{\mathbf{z} \in \mathbb{R}^{|\mathcal{V}|}} \|J_\sigma(\mathbf{z})\|_{2 \to 1}, \qquad \|A\|_{2 \to 1} := \sup_{\|\mathbf{v}\|_2 = 1} \|A\mathbf{v}\|_1.$$

Then $C_{|\mathcal{V}|}$ depends only on $|\mathcal{V}|$. By the mean value theorem, for every i, j,

$$\|q_i - q_j\|_1 = \|\sigma(\widetilde{W}\mathbf{x}_i) - \sigma(\widetilde{W}\mathbf{x}_j)\|_1 \le C_{|\mathcal{V}|}\, \|\widetilde{W}(\mathbf{x}_i - \mathbf{x}_j)\| \le C_{|\mathcal{V}|}\, \|\widetilde{W}\|_{\mathrm{op}}\, \|\mathbf{x}_i - \mathbf{x}_j\|.$$

Therefore

$$\frac{1}{n(n-1)} \sum_{i \ne j} \|\mathbf{x}_i - \mathbf{x}_j\| \ge \frac{\mathcal{D} - 2\sqrt{2\delta}}{\|\widetilde{W}\|_{\mathrm{op}}\, C_{|\mathcal{V}|}}.$$

Applying Jensen's inequality,

$$\frac{1}{n(n-1)}\sum_{i\neq j}\|\mathbf{x}_i-\mathbf{x}_j\|^2 \geq \left(\frac{1}{n(n-1)}\sum_{i\neq j}\|\mathbf{x}_i-\mathbf{x}_j\|\right)^2 \geq \left(\frac{\mathcal{D}-2\sqrt{2\delta}}{\|\widetilde{W}\|_{\mathrm{op}}\, C_{|\mathcal{V}|}}\right)^2.$$

Since the diagonal terms vanish,

$$\frac{1}{n^2}\sum_{i,j\in\mathcal{K}}\|\mathbf{x}_i-\mathbf{x}_j\|^2 = \frac{n-1}{n}\cdot\frac{1}{n(n-1)}\sum_{i\neq j}\|\mathbf{x}_i-\mathbf{x}_j\|^2 \geq \frac{n-1}{n}\left(\frac{\mathcal{D}-2\sqrt{2\delta}}{\|\widetilde{W}\|_{\mathrm{op}}\, C_{|\mathcal{V}|}}\right)^2.$$

This proves the claimed lower bound on the average pairwise squared distance. Hence the continuous tokens cannot exhibit representation collapse at any level ε below this quantity. Since the average of these nonnegative squared distances is at least this quantity, there exists a distinct pair $i, j \in \mathcal{K}$ such that

$$\|\mathbf{x}_i-\mathbf{x}_j\| \geq \sqrt{\frac{n-1}{n}}\,\frac{\mathcal{D}-2\sqrt{2\delta}}{\|\widetilde{W}\|_{\mathrm{op}}\, C_{|\mathcal{V}|}}.$$

□

Proposition 10 (Parallel BFS with multiplexing). *There exists a sequence of continuous tokens $(\mathbf{x}_0, \ldots, \mathbf{x}_H)$ such that, for every $k \leq H$:*

(i) $\mathbf{x}_k$ is a deterministic function of $\mathbf{x}_{k-1}$ and G,

(ii) F_k and U_k can be recovered from $\mathbf{x}_k$, and so reachability $\mathbf{1}(t \in U_H)$ can be recovered from $\mathbf{x}_H$,

(iii) whenever $F_k \neq \varnothing$, $\mathsf{mux}(\mathbf{r}_k)$ can be recovered from $\mathbf{x}_k$, up to arbitrary precision with softmax.

Proof. We construct a recurrence over continuous tokens and verify that it exactly implements breadth-first search.

For a set $B \subseteq \mathcal{N}$, let $1_B \in \{0,1\}^n$ denote its indicator vector, where $n = |\mathcal{N}|$. At step k, let the continuous token be the pair

$$(f_k, u_k) \in \{0,1\}^{2n},$$

where

$$f_k = 1_{F_k}, \qquad u_k = 1_{U_k}.$$

Thus the token stores the current frontier and the set of visited nodes.

Initialize

$$f_0 = 1_{\{s\}}, \qquad u_0 = 1_{\{s\}}.$$

Let $A \in \{0,1\}^{n\times n}$ be the adjacency matrix of the graph, with

$$A_{uv} = 1 \iff (u,v) \in E.$$

Given the token (f_k, u_k), define the next token by

$$g_{k+1} = 1[A^\top f_k > 0],$$

$$f_{k+1} = g_{k+1} \odot (1 - u_k),$$

$$u_{k+1} = u_k + f_{k+1}.$$

We claim that for every $k \leq H$,

$$f_k = 1_{F_k}, \qquad u_k = 1_{U_k}.$$

The claim is immediate at $k = 0$. Assume it holds at step k. For any node $v \in \mathcal{N}$,

$$(g_{k+1})_v = 1 \iff (A^\top f_k)_v > 0 \iff \exists\, u \in F_k \text{ such that } (u, v) \in E \iff v \in N^+(F_k).$$

Therefore

$$g_{k+1} = 1_{N^+(F_k)}.$$

Hence

$$f_{k+1} = 1_{N^+(F_k)} \odot (1 - 1_{U_k}) = 1_{N^+(F_k) \setminus U_k} = 1_{F_{k+1}}.$$

Also, by the BFS update, $F_{k+1} \cap U_k = \varnothing$, so

$$u_{k+1} = u_k + f_{k+1} = 1_{U_k} + 1_{F_{k+1}} = 1_{U_k \cup F_{k+1}} = 1_{U_{k+1}}.$$

This proves the claim by induction.

It follows that the recurrence exactly tracks the breadth-first frontier and visited set at every step. In particular, the final answer is exact:

$$y = 1[t \in U_H].$$

Indeed, since $u_H = 1_{U_H}$ is part of the final token, the answer head can read the coordinate corresponding to t and output the correct answer.

It remains to recover the frontier distribution. If $F_k = \varnothing$, one may use a designated null distribution. Assume now that $F_k \neq \varnothing$. Since

$$f_k = 1_{F_k},$$

the frontier is explicitly encoded in the token, so define

$$p_k(v) = \frac{(f_k)_v}{\|f_k\|_1}.$$

Then

$$p_k(v) = \begin{cases} 1/|F_k|, & v \in F_k, \\ 0, & v \notin F_k. \end{cases}$$

By the setup of Section 5.2, this is exactly $\mathrm{mux}(r_k)$.

Finally, if one insists on a standard softmax readout with finite logits, exact zeros outside F_k are impossible, but arbitrarily good approximation is still possible. For $B > 0$, define

$$\ell_k(v) = B\big((f_k)_v - 1\big).$$

Then

$$\ell_k(v) = \begin{cases} 0, & v \in F_k, \\ -B, & v \notin F_k. \end{cases}$$

Let $m = |F_k|$. The corresponding softmax distribution is

$$p_k^{(B)}(v) = \frac{e^{\ell_k(v)}}{\sum_{u \in \mathcal{N}} e^{\ell_k(u)}}.$$

Since

$$\sum_{u \in \mathcal{N}} e^{\ell_k(u)} = m + (|\mathcal{N}| - m)e^{-B},$$

we obtain

$$p_k^{(B)}(v) = \begin{cases} \dfrac{1}{m + (|\mathcal{N}| - m)e^{-B}}, & v \in F_k, \\ \dfrac{e^{-B}}{m + (|\mathcal{N}| - m)e^{-B}}, & v \notin F_k. \end{cases}$$

Therefore

$$p_k^{(B)} \to \mathrm{mux}(r_k) \qquad \text{as } B \to \infty.$$

So the frontier distribution is recoverable from the continuous token exactly, and realizable by a standard softmax readout up to arbitrarily small error. □

C.2 Multiplexing under finite precision

Proposition 3 characterizes lossless multiplexing in exact arithmetic: for a fixed span length S, injectivity of $\mathsf{mux} : \mathcal{V}^S \to \Delta^{|\mathcal{V}|-1}$ is equivalent to $\mathcal{E}(\boldsymbol{\alpha}) > 0$. We now make the finite-precision version of this statement explicit. The argument has two steps. First, the same margin $\mathcal{E}(\boldsymbol{\alpha})$ is the minimum coordinatewise separation between distinct exact targets. Second, under the standard unit-roundoff model, target construction introduces an $O(Su)$ perturbation for span length S and unit roundoff u. We state everything for one fixed span length S; for full traces with varying span lengths, the argument applies spanwise exactly as in Corollary 4. We use the ℓ_∞ norm because each coordinate of $\mathsf{mux}(\mathbf{r})$ is a subset sum of the masses, and the separation margin in Definition 2 is coordinatewise.

We first identify $\mathcal{E}(\boldsymbol{\alpha})$ as the minimum ℓ_∞-distance between two distinct exact multiplexed targets.

Proposition 12 (Separation between distinct exact multiplexed targets)**.** *Assume* $|\mathcal{V}| > 1$. *Then*

$$\min_{\mathbf{r}\neq\mathbf{r}'\in\mathcal{V}^S} \|\mathsf{mux}(\mathbf{r}) - \mathsf{mux}(\mathbf{r}')\|_\infty = \mathcal{E}(\boldsymbol{\alpha}).$$

Proof. Take any distinct $\mathbf{r}, \mathbf{r}' \in \mathcal{V}^S$. For each vocabulary symbol $v \in \mathcal{V}$,

$$\big(\mathsf{mux}(\mathbf{r}) - \mathsf{mux}(\mathbf{r}')\big)_v = \sum_{j=1}^{S} c_j^{(v)} \alpha_j, \qquad c_j^{(v)} = \mathbf{1}[r^j = v] - \mathbf{1}[(r')^j = v] \in \{-1, 0, 1\}.$$

If $\mathbf{r} \neq \mathbf{r}'$, then for at least one v the coefficient vector $(c_1^{(v)}, \ldots, c_S^{(v)})$ is nonzero. By Definition 2,

$$\big|\big(\mathsf{mux}(\mathbf{r}) - \mathsf{mux}(\mathbf{r}')\big)_v\big| \geq \mathcal{E}(\boldsymbol{\alpha}),$$

and hence

$$\|\mathsf{mux}(\mathbf{r}) - \mathsf{mux}(\mathbf{r}')\|_\infty \geq \mathcal{E}(\boldsymbol{\alpha}).$$

Taking the minimum over all distinct pairs yields

$$\min_{\mathbf{r}\neq\mathbf{r}'} \|\mathsf{mux}(\mathbf{r}) - \mathsf{mux}(\mathbf{r}')\|_\infty \geq \mathcal{E}(\boldsymbol{\alpha}).$$

For the reverse inequality, choose a nonzero vector $\mathbf{c} = (c_1, \ldots, c_S) \in \{-1, 0, 1\}^S$ attaining the minimum in (5). Since $|\mathcal{V}| > 1$, pick distinct symbols $u, v \in \mathcal{V}$, and define $\mathbf{r}, \mathbf{r}' \in \mathcal{V}^S$ by

$$r^j = \begin{cases} u, & c_j = 1, \\ v, & c_j \in \{-1, 0\}, \end{cases} \qquad (r')^j = \begin{cases} u, & c_j = -1, \\ v, & c_j \in \{1, 0\}. \end{cases}$$

Then the only possibly nonzero coordinates of $\mathsf{mux}(\mathbf{r}) - \mathsf{mux}(\mathbf{r}')$ are the u- and v-coordinates, equal to

$$\sum_{j=1}^{S} c_j \alpha_j \qquad \text{and} \qquad -\sum_{j=1}^{S} c_j \alpha_j,$$

respectively. Therefore

$$\|\mathsf{mux}(\mathbf{r}) - \mathsf{mux}(\mathbf{r}')\|_\infty = \left|\sum_{j=1}^{S} c_j \alpha_j\right| = \mathcal{E}(\boldsymbol{\alpha}),$$

which proves the reverse inequality. □

Proposition 12 is the exact-arithmetic separation statement. It shows that any perturbation smaller than half of this margin preserves unique demultiplexing.

Corollary 13 (Stable demultiplexing under bounded perturbation)**.** *Assume* $|\mathcal{V}| > 1$*. Let* $\mathbf{r} \in \mathcal{V}^S$*, and let* $y \in \mathbb{R}^{|\mathcal{V}|}$ *satisfy*

$$\|y - \mathsf{mux}(\mathbf{r})\|_\infty < \frac{\mathcal{E}(\boldsymbol{\alpha})}{2}.$$

Then $\mathbf{r}$ *is the unique minimum-*ℓ_∞ *demultiplexing of* y*, i.e.*

$$\arg\min_{\mathbf{s}\in\mathcal{V}^S} \|y - \mathsf{mux}(\mathbf{s})\|_\infty = \{\mathbf{r}\}.$$

Proof. Take any competitor $\mathbf{s} \neq \mathbf{r}$. By Proposition 12,

$$\|\mathsf{mux}(\mathbf{r}) - \mathsf{mux}(\mathbf{s})\|_\infty \geq \mathcal{E}(\boldsymbol{\alpha}).$$

Hence the triangle inequality gives

$$\|y - \mathsf{mux}(\mathbf{s})\|_\infty \geq \|\mathsf{mux}(\mathbf{r}) - \mathsf{mux}(\mathbf{s})\|_\infty - \|y - \mathsf{mux}(\mathbf{r})\|_\infty > \mathcal{E}(\boldsymbol{\alpha}) - \frac{\mathcal{E}(\boldsymbol{\alpha})}{2} = \frac{\mathcal{E}(\boldsymbol{\alpha})}{2}.$$

On the other hand,

$$\|y - \mathsf{mux}(\mathbf{r})\|_\infty < \frac{\mathcal{E}(\boldsymbol{\alpha})}{2}.$$

Therefore

$$\|y - \mathsf{mux}(\mathbf{r})\|_\infty < \|y - \mathsf{mux}(\mathbf{s})\|_\infty \qquad \text{for every } \mathbf{s} \neq \mathbf{r},$$

so $\mathbf{r}$ is the unique minimizer. □

Corollary 13 applies to any perturbation y near an exact multiplexed target, regardless of its source. In this paper we use it for finite-precision target construction. Let $\widetilde{\mathsf{mux}}(\mathbf{r})$ denote the target materialized by the implementation, and define the worst-case target-construction error

$$\varepsilon_{\mathrm{fp}} := \sup_{\mathbf{r}\in\mathcal{V}^S} \|\widetilde{\mathsf{mux}}(\mathbf{r}) - \mathsf{mux}(\mathbf{r})\|_\infty.$$

This is a target-side quantity: it can include rounding of the masses, approximate normalization, and summation error. We now make $\varepsilon_{\mathrm{fp}}$ explicit under the standard unit-roundoff model. Let u denote the unit roundoff, and define

$$\gamma_n(u) := \frac{nu}{1-nu}, \qquad nu < 1.$$

Assume the exact normalized masses α_j are fixed first, and that:

(i) each α_j is stored once in the working format as $\widehat{\alpha}_j$, with

$$|\widehat{\alpha}_j - \alpha_j| \leq u\alpha_j;$$

(ii) each coordinate of $\widetilde{\mathsf{mux}}(\mathbf{r})$ is formed by naively summing the relevant stored masses $\widehat{\alpha}_j$ in the same arithmetic.

This isolates the floating-point error after the exact masses are fixed.

Corollary 14 (Floating-point sufficient condition)**.** *Under the model above,*

$$\varepsilon_{\mathrm{fp}} \leq \eta_S(u) := u + (1+u)\gamma_{S-1}(u) = \frac{Su}{1-(S-1)u}.$$

Consequently, exact recovery is guaranteed whenever

$$\eta_S(u) < \frac{\mathcal{E}(\boldsymbol{\alpha})}{2}.$$

Proof. Fix $\mathbf{r} \in \mathcal{V}^S$ and a vocabulary symbol $v \in \mathcal{V}$. Let

$$I_v := \{j : r^j = v\}, \qquad x_v := \sum_{j \in I_v} \alpha_j, \qquad \widehat{x}_v := \sum_{j \in I_v} \widehat{\alpha}_j.$$

If $I_v = \varnothing$, then $x_v = \widehat{x}_v = 0$, so the bound is trivial. Assume $I_v \neq \varnothing$. Since all terms are nonnegative,

$$|\widehat{x}_v - x_v| \leq \sum_{j \in I_v} |\widehat{\alpha}_j - \alpha_j| \leq u \sum_{j \in I_v} \alpha_j = u x_v.$$

Let $\widetilde{x}_v$ be the value obtained by naively summing the stored masses $\widehat{\alpha}_j$. Standard floating-point summation bounds give

$$\widetilde{x}_v = \widehat{x}_v(1 + \theta_{|I_v|-1}), \qquad |\theta_{|I_v|-1}| \leq \gamma_{|I_v|-1}(u) \leq \gamma_{S-1}(u).$$

Therefore

$$|\widetilde{x}_v - \widehat{x}_v| \leq \gamma_{S-1}(u)\, \widehat{x}_v \leq (1+u)\gamma_{S-1}(u)\, x_v,$$

where we used $\widehat{x}_v \leq (1+u)x_v$. Combining the two bounds yields

$$|\widetilde{x}_v - x_v| \leq \big(u + (1+u)\gamma_{S-1}(u)\big)x_v \leq u + (1+u)\gamma_{S-1}(u).$$

Taking the maximum over v proves

$$\varepsilon_{\mathrm{fp}} \leq u + (1+u)\gamma_{S-1}(u) = \frac{Su}{1-(S-1)u}.$$

The recovery condition then follows from Corollary 13. □

For round-to-nearest arithmetic,

$$u_{\mathrm{FP32}} = 2^{-24} \approx 5.96 \times 10^{-8}.$$

Hence, for $2 \leq S \leq 32$,

$$\eta_S(u_{\mathrm{FP32}}) \leq 1.91 \times 10^{-6},$$

Geometric weights. The floating-point bound above is independent of the weighting family. The weighting enters only through $\mathcal{E}(\boldsymbol{\alpha})$. For rational geometric weights, this margin admits an exact integer-arithmetic representation.

Corollary 15 (Rational geometric weights)**.** *Suppose $\rho = p/q \in (0,1)$ is rational in lowest terms and*

$$\alpha_j = \frac{\rho^{j-1}}{\sum_{\ell=0}^{S-1} \rho^\ell}, \qquad j = 1, \ldots, S.$$

Then

$$\mathcal{E}(\boldsymbol{\alpha}) = \frac{q-p}{q^S - p^S}\, m_S,$$

where

$$m_S := \min_{\mathbf{c} \in \{-1,0,1\}^S \setminus \{0\}} \left| \sum_{j=1}^{S} c_j p^{j-1} q^{S-j} \right|.$$

Moreover $m_S \geq 1$, and therefore

$$\mathcal{E}(\boldsymbol{\alpha}) \geq \frac{q-p}{q^S - p^S}.$$

Consequently, a sufficient condition for exact recovery is

$$\eta_S(u) < \frac{q-p}{2(q^S - p^S)}.$$

Proof. Using

$$\sum_{\ell=0}^{S-1}\left(\frac{p}{q}\right)^{\ell}=\frac{q^S-p^S}{q^{S-1}(q-p)},$$

we can rewrite the normalized masses as

$$\alpha_j=\frac{(q-p)p^{j-1}q^{S-j}}{q^S-p^S}.$$

Hence, for any nonzero $\mathbf{c}\in\{-1,0,1\}^S$,

$$\sum_{j=1}^{S}c_j\alpha_j=\frac{q-p}{q^S-p^S}\sum_{j=1}^{S}c_jp^{j-1}q^{S-j}.$$

Taking absolute values and then the minimum over all nonzero $\mathbf{c}$ gives the exact formula for $\mathcal{E}(\boldsymbol{\alpha})$. The quantity inside the absolute value is an integer. It is nonzero for every nonzero $\mathbf{c}$, because otherwise $\sum_{j=1}^{S}c_j\rho^{j-1}=0$, contradicting Proposition 5(a). Therefore $m_S\geq 1$, which yields the lower bound. The final condition follows by combining this lower bound with Corollary 14. □

For our default choice $\rho=9/10$,

$$\mathcal{E}(\boldsymbol{\alpha})=\frac{m_S}{10^S-9^S},\qquad m_S=\min_{\mathbf{c}\in\{-1,0,1\}^S\setminus\{0\}}\left|\sum_{j=1}^{S}c_j\,9^{j-1}10^{S-j}\right|.$$

This quantity can be evaluated exactly offline by integer arithmetic for each span length S used in practice. For $S\leq 30$, exact evaluation gives

$$\mathcal{E}(\boldsymbol{\alpha})\approx 3.98\times10^{-6}\text{ at }S=11,\qquad \mathcal{E}(\boldsymbol{\alpha})\approx 1.20\times10^{-6}\text{ at }S=12,$$

By contrast,

$$\eta_{11}(u_{\mathrm{FP32}})\approx 6.56\times10^{-7},\qquad \eta_{12}(u_{\mathrm{FP32}})\approx 7.15\times10^{-7},$$

Therefore, for the default geometric choice $\rho=0.9$, the conservative certificate from Corollary 14 holds in FP32 up to $S=11$.

C.3 Why local distillation preserves answer-side use of latent reasoning

This section gives an objective-level explanation for the attention pattern observed in Appendix B.2. The two auxiliary terms in $\mathcal{L}=\mathcal{L}_{\text{answer}}+\beta\,\mathcal{L}_{\text{local}}+\gamma\,\mathcal{L}_{\text{global}}$ constrain different objects. The local term $\mathcal{L}_{\text{local}}$ constrains each latent reasoning token $\mathbf{x}_i$ toward its own aligned target $\mathsf{mux}(\mathbf{r}_i)$, whereas $\mathcal{L}_{\text{global}}$ constrains only the aggregate hidden state used to produce the answer. We show that only the former yields a tokenwise lower bound on answer-side routing through previous latent reasoning tokens. Our positive result is a *routing-transfer* statement. We do not claim that answer-side routing through latent reasoning appears automatically. Instead, we isolate the regime in which the aligned multiplexed targets already have an answer-side advantage over non-reasoning context, and ask whether local distillation preserves that advantage after those targets are replaced by actual latent tokens.

Fix an answer-interface token t. In Appendix B.2 these are the tokens

$$\{\texttt{<EOT>},\ \texttt{The},\ \texttt{answer},\ \texttt{is},\ \texttt{:}\}.$$

Let $\mathcal{B}_t$ denote the set of non-reasoning positions visible to t that are shared by the discrete and continuous reasoning modes, namely question tokens and answer-bridge tokens. Recall that $\mathcal{K}\subseteq\{1,\dots,K\}$ is the set of latent-token positions whose aligned span is non-empty.

For each $i\in\mathcal{K}$, let

$$s_{t,i}:\Delta^{|\mathcal{V}|-1}\to\mathbb{R}$$

denote the attention logit assigned by token t in the *continuous reasoning mode* to position i as a function of the represented content $f(\mathbf{x}_i)$. For each background position $b \in \mathcal{B}_t$, let $\xi_t(b) \in \mathbb{R}$ denote its corresponding attention logit in the same mode. We define the total attention mass assigned by t to previous latent reasoning tokens by

$$A_t(\mathbf{x}) = \frac{\sum_{i\in\mathcal{K}} \exp(s_{t,i}(f(\mathbf{x}_i)))}{\sum_{i\in\mathcal{K}} \exp(s_{t,i}(f(\mathbf{x}_i))) + \sum_{b\in\mathcal{B}_t} \exp(\xi_t(b))}. \tag{7}$$

To state the routing bound, it is enough to summarize the answer-side geometry at token t by two intrinsic quantities. First, define the aligned-target margin

$$\Delta_t^{\mathrm{ref}} := \min_{i\in\mathcal{K},\, b\in\mathcal{B}_t} \Big(s_{t,i}(\mathsf{mux}(\mathbf{r}_i)) - \xi_t(b) \Big).$$

This is the worst-case logit margin, in the continuous reasoning mode, between an aligned multiplexed target and a background position. Second, for $\delta > 0$, define the local score-drift modulus

$$\omega_t(\delta) := \max_{i\in\mathcal{K}} \sup_{\substack{p\in\Delta^{|\mathcal{V}|-1}:\\ D_{\mathrm{KL}}(\mathsf{mux}(\mathbf{r}_i)\,\|\,p)\le\delta}} \Big(s_{t,i}(\mathsf{mux}(\mathbf{r}_i)) - s_{t,i}(p) \Big).$$

This quantity measures the largest downward change in routing score caused by replacing the aligned target with any content inside a KL-ball of radius δ.

Proposition 16 (Local distillation preserves answer-side routing)**.** *Fix an answer-interface token t and $\delta > 0$. Define the set of well-aligned latent-token positions by*

$$\mathcal{K}_\delta = \{ i \in \mathcal{K} : D_{\mathrm{KL}}(\mathsf{mux}(\mathbf{r}_i) \,\|\, f(\mathbf{x}_i)) \le \delta \}.$$

Then

$$|\mathcal{K}_\delta| \ge |\mathcal{K}| \left(1 - \frac{\mathcal{L}_{\mathrm{local}}}{\delta} \right), \tag{8}$$

and

$$A_t(\mathbf{x}) \ge \frac{|\mathcal{K}_\delta| \, \exp(\Delta_t^{\mathrm{ref}} - \omega_t(\delta))}{|\mathcal{K}_\delta| \, \exp(\Delta_t^{\mathrm{ref}} - \omega_t(\delta)) + |\mathcal{B}_t|}. \tag{9}$$

In particular, if $\Delta_t^{\mathrm{ref}} > \omega_t(\delta)$, then every $i \in \mathcal{K}_\delta$ satisfies

$$s_{t,i}(f(\mathbf{x}_i)) > \xi_t(b), \qquad \forall b \in \mathcal{B}_t,$$

so each well-aligned latent reasoning token individually outranks every background position at token t.

Proof. We first prove (8).

For each $i \in \mathcal{K}$, define

$$d_i := D_{\mathrm{KL}}(\mathsf{mux}(\mathbf{r}_i) \,\|\, f(\mathbf{x}_i)).$$

By (4),

$$\mathcal{L}_{\mathrm{local}} = \frac{1}{|\mathcal{K}|} \sum_{i\in\mathcal{K}} d_i.$$

Let

$$\mathcal{E}_\delta := \{ i \in \mathcal{K} : d_i > \delta \}.$$

Each index in $\mathcal{E}_\delta$ contributes more than δ to the sum, hence

$$\sum_{i\in\mathcal{K}} d_i \ge \sum_{i\in\mathcal{E}_\delta} d_i > |\mathcal{E}_\delta| \, \delta.$$

Dividing by $|\mathcal{K}|$ gives

$$\mathcal{L}_{\text{local}} > \frac{|\mathcal{E}_\delta|}{|\mathcal{K}|}\,\delta,$$

so

$$|\mathcal{E}_\delta| < |\mathcal{K}|\,\frac{\mathcal{L}_{\text{local}}}{\delta}.$$

Since $\mathcal{K}_\delta = \mathcal{K} \setminus \mathcal{E}_\delta$, we obtain

$$|\mathcal{K}_\delta| = |\mathcal{K}| - |\mathcal{E}_\delta| \geq |\mathcal{K}| \left(1 - \frac{\mathcal{L}_{\text{local}}}{\delta}\right),$$

which proves (8).

We now prove (9). Fix any $i \in \mathcal{K}_\delta$. By definition of $\mathcal{K}_\delta$,

$$D_{\text{KL}}(\mathsf{mux}(\mathbf{r}_i) \,\|\, f(\mathbf{x}_i)) \leq \delta.$$

Therefore, by definition of $\omega_t(\delta)$,

$$s_{t,i}(f(\mathbf{x}_i)) \geq s_{t,i}(\mathsf{mux}(\mathbf{r}_i)) - \omega_t(\delta).$$

By definition of Δ_t^{ref}, for every $b \in \mathcal{B}_t$,

$$s_{t,i}(\mathsf{mux}(\mathbf{r}_i)) \geq \xi_t(b) + \Delta_t^{\text{ref}}.$$

Combining the two displays gives

$$s_{t,i}(f(\mathbf{x}_i)) \geq \xi_t(b) + \Delta_t^{\text{ref}} - \omega_t(\delta), \qquad \forall b \in \mathcal{B}_t. \tag{10}$$

Choose $b_t^\star \in \mathcal{B}_t$ satisfying

$$\xi_t(b_t^\star) = \max_{b \in \mathcal{B}_t} \xi_t(b).$$

Applying (10) with $b = b_t^\star$ gives

$$s_{t,i}(f(\mathbf{x}_i)) \geq \xi_t(b_t^\star) + \Delta_t^{\text{ref}} - \omega_t(\delta).$$

Exponentiating both sides,

$$\exp(s_{t,i}(f(\mathbf{x}_i))) \geq \exp(\Delta_t^{\text{ref}} - \omega_t(\delta))\,\exp(\xi_t(b_t^\star)).$$

This holds for every $i \in \mathcal{K}_\delta$. Summing over $i \in \mathcal{K}_\delta$,

$$\sum_{i \in \mathcal{K}} \exp(s_{t,i}(f(\mathbf{x}_i))) \geq |\mathcal{K}_\delta|\,\exp(\Delta_t^{\text{ref}} - \omega_t(\delta))\,\exp(\xi_t(b_t^\star)).$$

On the other hand, by maximality of $\xi_t(b_t^\star)$,

$$\sum_{b \in \mathcal{B}_t} \exp(\xi_t(b)) \leq |\mathcal{B}_t|\,\exp(\xi_t(b_t^\star)).$$

Substituting these two bounds into (7) gives (9). The final claim follows directly from (10). □

Proposition 16 shows that the local objective controls how many positions stay inside a KL-ball around their aligned targets, while the sign and magnitude of $\Delta_t^{\text{ref}} - \omega_t(\delta)$ determine whether the answer-side preference survives inside that ball. The count bound becomes informative once $\delta > \mathcal{L}_{\text{local}}$, but the proposition itself does not assume any positivity condition on the margin.

To connect this statement back to the *discrete reasoning mode*, let $\bar{\ell}_t(v)$ denote the attention logit from token t to a discrete reasoning position v. For each aligned span $\mathbf{r}_i = (\mathbf{r}_i^1, \ldots, \mathbf{r}_i^{|\mathbf{r}_i|})$, define the corresponding span-level routing score by

$$\bar{s}_{t,i} := \log \sum_{u=1}^{|\mathbf{r}_i|} \exp(\bar{\ell}_t(\mathbf{r}_i^u)). \tag{11}$$

Thus, $\bar{s}_{t,i}$ is the log-sum-exp score assigned by token t to the entire aligned span $\mathbf{r}_i$ in the discrete reasoning mode. For each background position $b \in \mathcal{B}_t$, let $\bar{\xi}_t(b) \in \mathbb{R}$ denote its attention logit in the discrete reasoning mode. Now define the discrete routing margin

$$\Delta_t^{\text{disc}} := \min_{i \in \mathcal{K},\, b \in \mathcal{B}_t} \big(\bar{s}_{t,i} - \bar{\xi}_t(b)\big).$$

This is the formal version of the answer-side preference for aligned reasoning spans studied in prior routing analyses (Zhang et al., 2025b; Tutek et al., 2025). Also define the calibration gap

$$\Gamma_t := \max\left\{ \max_{i \in \mathcal{K}} \left| s_{t,i}(\mathsf{mux}(\mathbf{r}_i)) - \bar{s}_{t,i} \right|, \max_{b \in \mathcal{B}_t} \left| \xi_t(b) - \bar{\xi}_t(b) \right| \right\}.$$

Then, for every $i \in \mathcal{K}$ and $b \in \mathcal{B}_t$,

$$s_{t,i}(\mathsf{mux}(\mathbf{r}_i)) - \xi_t(b) \geq \bar{s}_{t,i} - \bar{\xi}_t(b) - 2\Gamma_t,$$

hence

$$\Delta_t^{\text{ref}} \geq \Delta_t^{\text{disc}} - 2\Gamma_t.$$

Substituting this into Proposition 16 yields

$$A_t(\mathbf{x}) \geq \frac{|\mathcal{K}_\delta| \, \exp(\Delta_t^{\text{disc}} - 2\Gamma_t - \omega_t(\delta))}{|\mathcal{K}_\delta| \, \exp(\Delta_t^{\text{disc}} - 2\Gamma_t - \omega_t(\delta)) + |\mathcal{B}_t|}.$$

This is the sense in which local distillation preserves answer-side routing: a routing preference present in the discrete trace transfers to the continuous mode provided the aligned targets retain a positive calibrated margin and the actual distilled tokens do not drift far enough to erase it.

We now contrast this with trajectory-level supervision alone. Let $\mathbf{h}_t^\star \in \mathbb{R}^d$ denote the answer-interface hidden state in the discrete reasoning mode at token t. Write the hidden state in the continuous reasoning mode as

$$\mathbf{h}_t(\mathbf{x}) = \sum_{i \in \mathcal{K}} a_{t,i}(\mathbf{x}) \, \mathbf{v}_{t,i} + \sum_{b \in \mathcal{B}_t} a_{t,b}(\mathbf{x}) \, \mathbf{v}_{t,b}, \tag{12}$$

where $a_{t,i}(\mathbf{x})$ and $a_{t,b}(\mathbf{x})$ are the attention weights at token t, and $\mathbf{v}_{t,i}, \mathbf{v}_{t,b} \in \mathbb{R}^d$ are the corresponding value vectors. Within this abstraction,

$$A_t(\mathbf{x}) = \sum_{i \in \mathcal{K}} a_{t,i}(\mathbf{x}).$$

Proposition 17 (Global distillation alone does not identify routing)**.** *Fix an answer-interface token t. Assume that there exist coefficients $(\lambda_b)_{b \in \mathcal{B}_t}$ with*

$$\lambda_b \geq 0, \qquad \sum_{b \in \mathcal{B}_t} \lambda_b = 1,$$

such that

$$\left\| \sum_{b \in \mathcal{B}_t} \lambda_b \mathbf{v}_{t,b} - \mathbf{h}_t^\star \right\|_2 \leq \varepsilon_0. \tag{13}$$

Then for every $\eta \in (0,1)$, there exists a choice of attention weights at token t such that

$$A_t(\mathbf{x}) = \eta$$

and

$$\|\mathbf{h}_t(\mathbf{x}) - \mathbf{h}_t^\star\|_2 \leq \varepsilon_0 + C_t \eta,$$

where

$$C_t = \left\| \sum_{b \in \mathcal{B}_t} \lambda_b \mathbf{v}_{t,b} \right\|_2 + \max_{i \in \mathcal{K}} \|\mathbf{v}_{t,i}\|_2.$$

Consequently, $\mathcal{L}_{\text{global}}$ alone does not imply any strictly positive lower bound on answer-side attention to previous latent reasoning tokens.

Proof. Define

$$\bar{\mathbf{h}}_t := \sum_{b \in \mathcal{B}_t} \lambda_b \mathbf{v}_{t,b}.$$

By (13),

$$\|\bar{\mathbf{h}}_t - \mathbf{h}_t^\star\|_2 \le \varepsilon_0.$$

Fix any $\eta \in (0,1)$. Choose the attention weights at token t by

$$a_{t,i}^{(\eta)}(\mathbf{x}) := \frac{\eta}{|\mathcal{K}|}, \qquad i \in \mathcal{K},$$

and

$$a_{t,b}^{(\eta)}(\mathbf{x}) := (1-\eta)\lambda_b, \qquad b \in \mathcal{B}_t.$$

These weights are nonnegative and satisfy

$$\sum_{i \in \mathcal{K}} a_{t,i}^{(\eta)}(\mathbf{x}) + \sum_{b \in \mathcal{B}_t} a_{t,b}^{(\eta)}(\mathbf{x}) = \eta + (1-\eta) \sum_{b \in \mathcal{B}_t} \lambda_b = 1.$$

Hence they define a valid attention distribution in the attention-mixture abstraction.

By construction, the total attention mass on previous latent reasoning tokens is exactly

$$A_t(\mathbf{x}) = \sum_{i \in \mathcal{K}} a_{t,i}^{(\eta)}(\mathbf{x}) = \eta.$$

Substituting the chosen weights into (12) gives

$$\mathbf{h}_t^{(\eta)}(\mathbf{x}) = \sum_{i \in \mathcal{K}} \frac{\eta}{|\mathcal{K}|}\, \mathbf{v}_{t,i} + \sum_{b \in \mathcal{B}_t} (1-\eta)\lambda_b\, \mathbf{v}_{t,b}.$$

Using the definition of $\bar{\mathbf{h}}_t$, we may rewrite this as

$$\mathbf{h}_t^{(\eta)}(\mathbf{x}) = (1-\eta)\bar{\mathbf{h}}_t + \frac{\eta}{|\mathcal{K}|} \sum_{i \in \mathcal{K}} \mathbf{v}_{t,i}.$$

Subtracting $\mathbf{h}_t^\star$ yields

$$\mathbf{h}_t^{(\eta)}(\mathbf{x}) - \mathbf{h}_t^\star = (\bar{\mathbf{h}}_t - \mathbf{h}_t^\star) - \eta\, \bar{\mathbf{h}}_t + \frac{\eta}{|\mathcal{K}|} \sum_{i \in \mathcal{K}} \mathbf{v}_{t,i}.$$

Taking norms and applying the triangle inequality,

$$\|\mathbf{h}_t^{(\eta)}(\mathbf{x}) - \mathbf{h}_t^\star\|_2 \le \|\bar{\mathbf{h}}_t - \mathbf{h}_t^\star\|_2 + \eta \|\bar{\mathbf{h}}_t\|_2 + \left\| \frac{\eta}{|\mathcal{K}|} \sum_{i \in \mathcal{K}} \mathbf{v}_{t,i} \right\|_2.$$

For the last term,

$$\left\| \frac{\eta}{|\mathcal{K}|} \sum_{i \in \mathcal{K}} \mathbf{v}_{t,i} \right\|_2 \le \frac{\eta}{|\mathcal{K}|} \sum_{i \in \mathcal{K}} \|\mathbf{v}_{t,i}\|_2 \le \eta \max_{i \in \mathcal{K}} \|\mathbf{v}_{t,i}\|_2.$$

Combining the preceding two displays with $\|\bar{\mathbf{h}}_t - \mathbf{h}_t^\star\|_2 \le \varepsilon_0$, we obtain

$$\|\mathbf{h}_t^{(\eta)}(\mathbf{x}) - \mathbf{h}_t^\star\|_2 \le \varepsilon_0 + \eta \left(\|\bar{\mathbf{h}}_t\|_2 + \max_{i \in \mathcal{K}} \|\mathbf{v}_{t,i}\|_2 \right).$$

By the definition of C_t, this is

$$\|\mathbf{h}_t^{(\eta)}(\mathbf{x}) - \mathbf{h}_t^\star\|_2 \le \varepsilon_0 + C_t \eta,$$

which proves the claim. □

Proposition 17 says that matching the answer-interface hidden state does not identify the routing pattern. The same observation applies to $\mathcal{L}_{\mathrm{CE}}$: if two routing patterns induce the same answer-interface hidden state, then they incur the same answer loss.

Propositions 16 and 17 separate what the local and global objectives control. Local distillation yields a tokenwise lower bound on answer-side attention exactly when the aligned targets retain a positive answer-side margin relative to their local score drift. By contrast, global supervision alone does not provide any analogous guarantee. The endpoint hidden state can be matched while the attention mass on previous latent reasoning tokens is made arbitrarily small.

D Benchmark details

D.1 MNNS task

The Minimum Non-Negative Sum (MNNS) task (Gozeten et al., 2026) takes as input H positive integers $a_1, \ldots, a_H$ and asks for the minimum value of $\sum_{k=1}^{H} \sigma_k a_k \geq 0$ over all sign assignments $\sigma_k \in \{-1, +1\}$. This is equivalent to finding the partition of $\{a_1, \ldots, a_H\}$ into two subsets with minimal non-negative difference, a variant of the subset-sum problem (Karp, 2009).

Graph construction. We define a layered directed graph $G = (\mathcal{N}, E)$ with

$$\mathcal{N} = \{(k, z) : k \in \{0, \ldots, H\},\ z \text{ is a partial sum reachable at depth } k\},$$

and edges $((k, z), (k{+}1, z + a_{k+1})),\ ((k, z), (k{+}1, z - a_{k+1})) \in E$. The source is $s = (0, 0)$. At depth k, the BFS frontier F_k contains all partial-sum states discovered for the first time, and the discovered set U_k accumulates all states seen up to depth k. The answer is the minimum non-negative z such that $(H, z) \in U_H$.

Data and vocabulary. For $H{=}4$ digits drawn from $\{1, \ldots, 9\}$, the vocabulary consists of integers in $[-S, S]$ (with S chosen so all reachable partial sums are covered) plus special tokens. The input is formatted as $\langle \textsc{bos} \rangle\ a_1\ a_2\ \ldots\ a_H\ \rightarrow$, and the output is the optimal sum value followed by $\langle \textsc{eos} \rangle$. Permutations of the same integer multiset are assigned to the same data split (80%/20% train/val) to prevent data leakage.

Architecture. We use a 2-layer, 2-head GPT-2 model with embedding dimension $d{=}32$, trained from scratch with AdamW (Loshchilov and Hutter, 2019) (learning rate 10^{-4}, no weight decay). Each of the $H{-}1$ intermediate steps corresponds to one latent token; the final discrete token produces the answer.

D.2 Game of 24 task

The Game of 24 (Yao et al., 2023) is a classic arithmetic puzzle: given a set of numbers, the goal is to combine them using arithmetic operations to reach the target value 24. We formulate a sequential variant that naturally maps to a layered reachability problem.

Task formulation. Given C cards with values drawn from $\{1, \ldots, D\}$ and an operator set $\mathcal{O}$, the model must determine whether the target value 24 is reachable by processing the cards strictly left to right. Starting with the first card as accumulator, at each step $k \in \{1, \ldots, C{-}1\}$ one applies an operation $\circ_k \in \mathcal{O}$ to produce $A_k = A_{k-1} \circ_k d_{k+1}$, where d_{k+1} is the $(k{+}1)$-th card. The answer is $y = \mathbf{1}[24 \in A_{C-1}]$, where A_{C-1} is the set of accumulated values reachable after all C cards over all operation sequences.

Graph construction. This defines a layered directed graph $G = (\mathcal{N}, E)$. The node set at depth k consists of all intermediate values reachable after incorporating $k{+}1$ cards:

$$\mathcal{N}_k = \{v : v \text{ is an accumulated value reachable at step } k\}.$$

Edges connect each node $v \in \mathcal{N}_k$ to nodes $\{v \circ d_{k+2} : \circ \in \mathcal{O}\} \cap \mathcal{N}_{k+1}$, and the source is $s = d_1$. At each step, the BFS frontier F_k records the set of newly discovered accumulated values, so the uniform multiplexed target $\mathsf{mux}(\mathbf{r}_k)$ is the distribution over the current frontier.

Table 8 LoRA adapter configuration (shared across all models).

Hyperparameter	Value
LoRA rank r	128
LoRA alpha	32
LoRA dropout	0.1

Table 9 Training hyperparameters for MUX. Method-specific parameters are listed in the top block; standard optimization settings are in the bottom block.

	GPT-2		LLaMA 3.2 1B		LLaMA 3.2 3B	LLaMA 3.1 8B
Hyperparameter	Aug	NL	Aug	NL	Aug	Aug
Method-specific						
Continuous tokens K	6	6	6	6	6	6
Weighting function	sin.	sin.	geo.	sin.	geo.	geo.
Decay rate ρ (geo.)	—	—	0.9	—	0.9	0.9
Positional scale λ (sin.)	1.0	1.0	—	1.0	—	—
Temperature τ	1.0	1.0	1.0	1.0	1.0	1.0
Chunking strategy	rand.	rand.	rand.	rand.	rand.	rand.
Local loss weight β	1.0	1.0	1.0	1.0	1.0	1.0
Global distill. weight γ	1.0	1.0	20.0	20.0	20.0	20.0
Answer loss weight	1.0	1.0	1.0	1.0	1.0	1.0
Ref. answer loss weight	1.0	1.0	1.0	1.0	1.0	1.0
Projection dim	768	768	2048	2048	3072	4096
Layer-wise std norm	✓	✓	✓	✓	✓	✓
Optimization						
Optimizer	AdamW					
LR scheduler	cosine					
Warmup ratio	0.03					
Effective batch size	128					
Learning rate	3e-3	3e-3	8e-4	8e-4	3e-4	1e-4
Weight decay	0.01	0.01	0.1	0.1	0.1	0.1
Gradient clipping	1.0	1.0	2.0	2.0	2.0	2.0
Epochs	40	40	10	10	8	6

Configuration. We use C=5 cards, digit range $\{1, \ldots, 5\}$, and operator set $\mathcal{O} = \{+, -, \times\}$. The dataset is balanced (50% reachable, 50% unreachable). We assign all permutations of the same card multiset to the same split. Because order affects the left-to-right process, this split prevents memorization of card multisets while still evaluating order-sensitive reasoning. Training uses 3 random seeds.

Architecture. We use the same 2-layer, 2-head GPT-2 model with d=32 as for MNNS. Each of the $C-1 = 4$ fold steps corresponds to one latent token; the final discrete token produces the YES/NO answer.

E Method and training details

E.1 Implementation details

Tables 8 and 9 summarize the hyperparameters used for all MUX experiments. We follow the same experimental protocol as CODI (Shen et al., 2025). All models were trained using a single H100 GPU with 96 GB of VRAM. Experiments with GPT-2 and LLaMA 1B took around 24 hours, while the experiments with larger backbones (LLaMA 3B/8B) ran for 2–3 days to complete.

MUX* (K=24) configuration. The parallel-decoding variant MUX* reported in Table 1 uses K=24 latent tokens generated via T=3 Jacobi iterations. We use uniform token chunking. Geometric positional weighting is applied with decay 0.9. The local loss weight is β=1.0 and the global distillation weight is γ=20.0. Optimization settings match the LLaMA 3.2 1B column of Table 9.

E.2 Details of probing for positional weighting

In Section 5.3, to study the role of positional weighting, we trained an MLP probe on discrete reasoning spans $\mathbf{r}_i = (r_i^1, \ldots, r_i^{S_i})$. For each non-empty span $\mathbf{r}_i$, we construct the same multiplexed target as in (2).

The probe takes $\mathsf{mux}(\mathbf{r}_i)$ as input and predicts the original span $\mathbf{r}_i$. We use a 5-layer MLP with hidden sizes $1024, 512, 256, 512, 1024$ and GELU (Hendrycks and Gimpel, 2016) activations, without input normalization. We set the maximum sequence length to 128, strip the delimiters « and » from each extracted step, and train for 20 epochs with batch size 128 and learning rate 10^{-3}. The data are extracted from GSM8K-AUG and split into 901,661 training spans and 100,185 evaluation spans using a 0.1 test split. For geometric weighting, we use $\rho = 0.9$. For sinusoidal weighting, we use $\tau = 1$. For rotary weighting, we use base = 1000.

E.3 Details of span-level alignments

The main text only assumes an order-preserving alignment between the M discrete reasoning spans and the K latent-token slots. Let $(\tilde{\mathbf{r}}_1, \ldots, \tilde{\mathbf{r}}_M)$ denote the aligned spans used for local supervision, where each non-empty $\tilde{\mathbf{r}}_i$ is a contiguous block of the original reasoning spans and the original order is preserved. When $M \le K$, all alignment variants reduce to the same prefix assignment: $\tilde{\mathbf{r}}_i = \mathbf{r}_i$ for $i \le M$, and the remaining $K - M$ slots are left empty. The differences arise only in the overfull regime $M > K$, which we summarize below.

No chunking. Assign one span to each latent token until one of the two sequences ends. Equivalently, $\tilde{\mathbf{r}}_i = \mathbf{r}_i$ for $i \le \min(M, K)$. If $M > K$, the remaining $M - K$ spans are discarded, so this is lossless only when $M \le K$.

Deterministic chunking. When $M > K$, partition the M reasoning spans into K contiguous groups with roughly equal sizes. Writing $M = qK + r$ with $0 \le r < K$, the first $K - r$ groups have size q and the last r groups have size $q + 1$. Equivalently, group sizes differ by at most one, with extra spans assigned to later groups.

Random chunking. When $M > K$, sample $K - 1$ cut points uniformly without replacement from $\{1, \ldots, M - 1\}$, sort them, and use the induced intervals to form K positive contiguous groups. This yields a random monotone partition of the M spans into K chunks. This is the default variant used in our main experiments. Randomness is resampled during training, so the model sees multiple valid local segmentations of the same reasoning trace without changing the answer target.

F Limitations and broader impact

Limitations. Our losslessness guarantees are stated under exact arithmetic. In finite precision, very long spans or large vocabularies may approach the separation boundary analyzed in Appendix C.2, though the analysis confirms that losslessness is preserved in practical regimes (standard span lengths and float32 precision). Our empirical evaluation focuses on mathematical reasoning and parallel search tasks. This is the established evaluation setting adopted by prior latent reasoning methods (Shen et al., 2025; Wei et al., 2026; Kuzina et al., 2026; Gozeten et al., 2026). Extending MUX to broader reasoning domains such as multi-hop question answering, code generation, and open-ended planning is a natural next step.

Broader impact. Compressing reasoning into fewer latent tokens can reduce inference cost. A concern with latent reasoning is reduced interpretability. Users cannot easily audit intermediate steps (Kuzina et al., 2026). MUX partially addresses this, since each latent token can be decoded into human-readable content through the LM head. Still, decoded tokens are approximate, not verbatim reasoning, so users should not treat them as ground truth. More broadly, more efficient reasoning inherits the risks of the underlying models, which can produce incorrect, biased, or overconfident outputs.