Reasoning by Superposition:

A Theoretical Perspective on Chain of Continuous Thought

Hanlin Zhu*, Shibo Hao*, Zhiting Hu, Jiantao Jiao, Stuart Russell, **Yuandong Tian**

Meta

University of California Berkeley

University of California San Diego

Center for Human-Compatible Artificial Intelligence

LLMs on reasoning tasks using CoT

- LLMs are powerful in many reasoning tasks, especially with chain-of-thought (CoT)

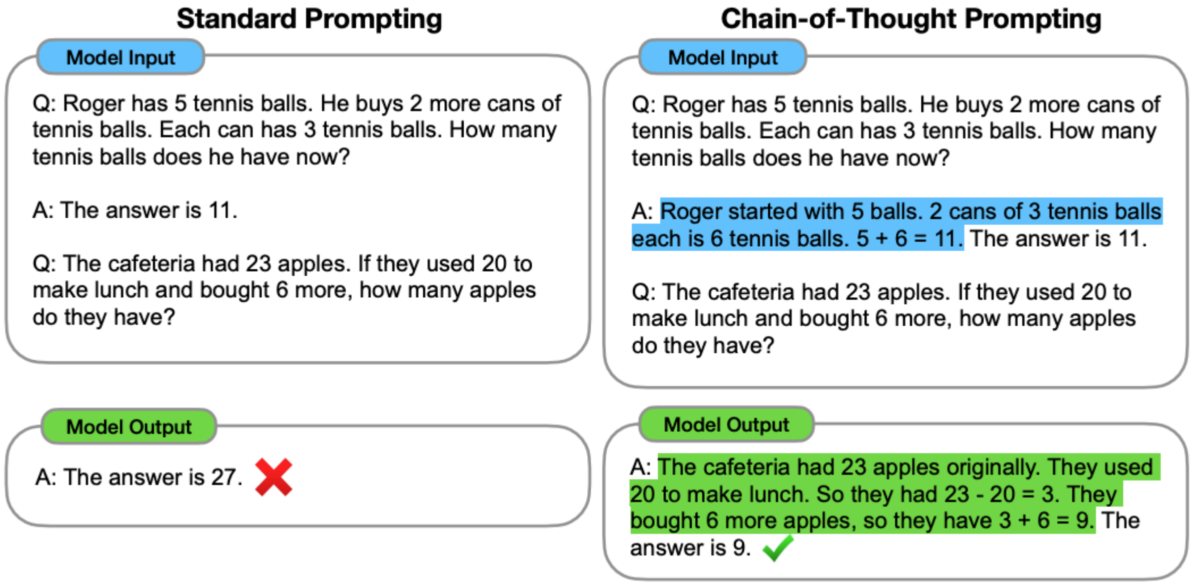


Figure credit to [1]

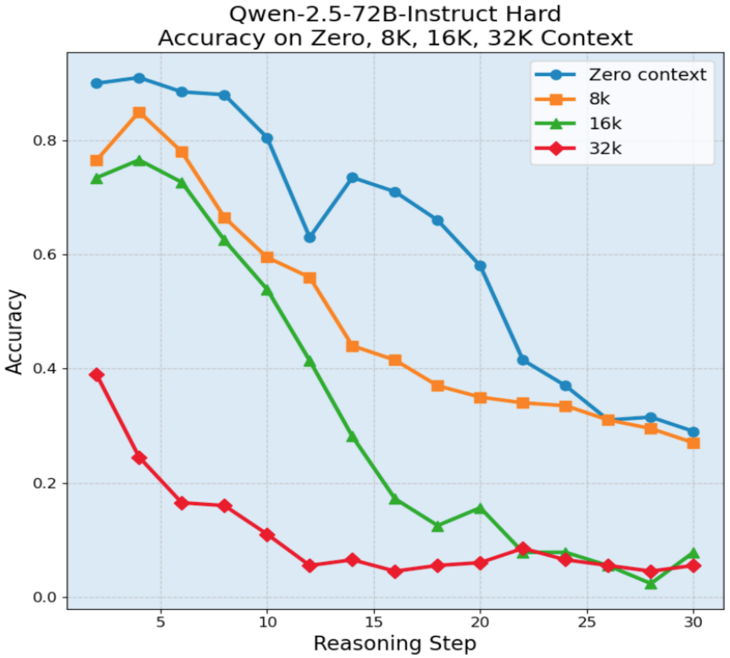


Figure credit to [2]

- LLMs still struggle with more complex reasoning tasks (e.g., longer reasoning steps)
- How to expand existing CoT methods to solve more complex problems?

[1] Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. "Chain-of-thought prompting elicits reasoning in large language models." NeurIPS'22
[2] Zhou, Yang, Hongyi Liu, Zhuoming Chen, Yuandong Tian, and Beidi Chen. "GSM-Infinite: How Do Your LLMs Behave over Infinitely Increasing Context Length and Reasoning Complexity?." *ICML'25*

Chain of continuous thought

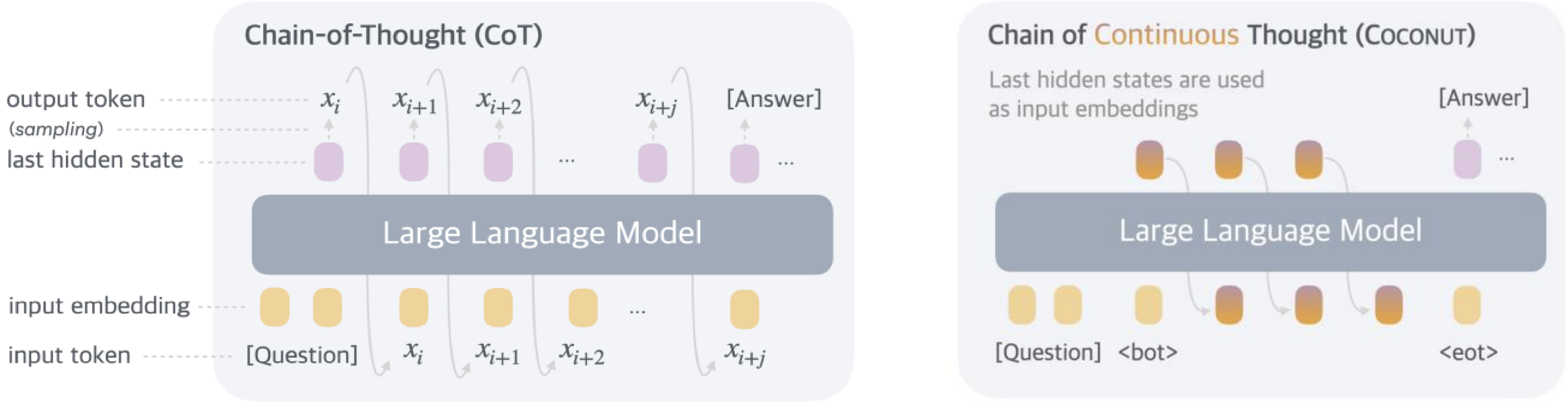


Figure credit to [1]

- Continuous CoT: directly uses the hidden state as the next input
- Outperforms discrete CoTs in various reasoning tasks
  - Especially problems with high branching factors/requires searching
- Lacks theoretical understanding of its power and mechanism

[1] Hao, Shibo, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. "Training large language models to reason in a continuous latent space." *COLM'25*

Main results

- *Construct* a 2-layer transformer with Continuous CoT that *solves* directed graph reachability using $O(n)$ steps (n: # of vertices)
  - The best known result for constant-depth transformers with discrete CoT requires $O(n^2)$ steps[1]

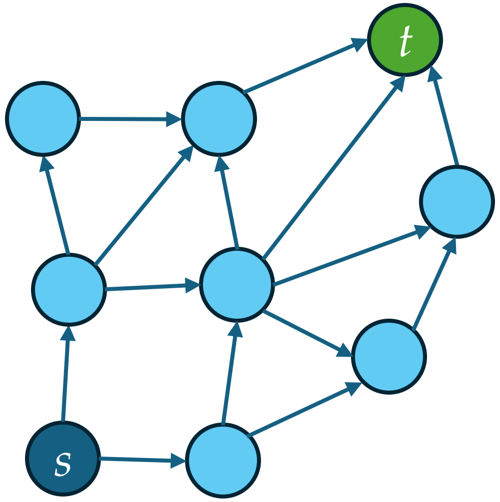
- **Insights:** Continuous thoughts maintain a "superposition" of explored vertices, performing a parallel BFS

- Empirical study is aligned with theoretical construction
  - Superposition representation **emerges** during training (no supervision)

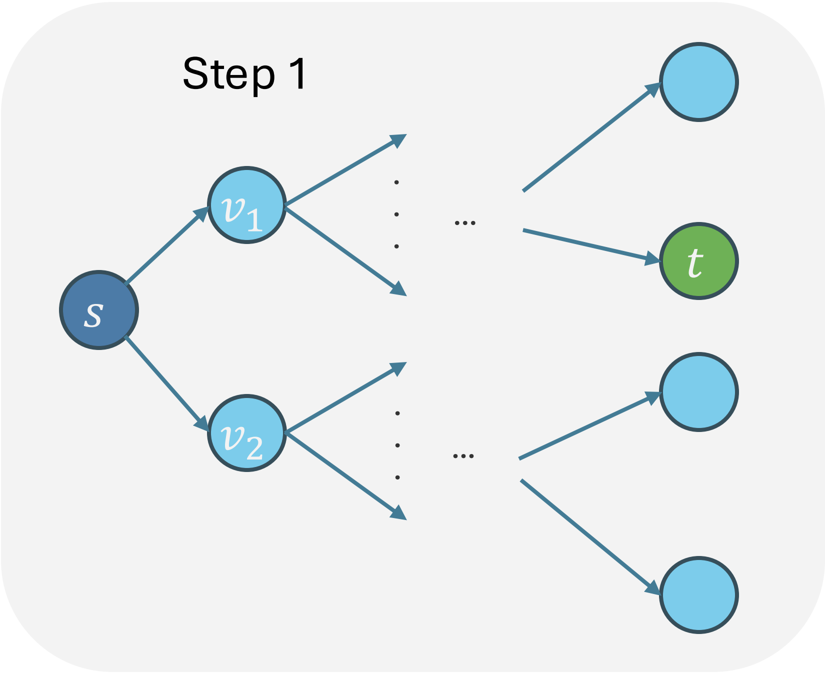
[1] Merrill, William, and Ashish Sabharwal. "The expressive power of transformers with chain of thought." *arXiv preprint arXiv:2310.07923* (2023).

Problem Definition: Graph reachability

- Given a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, decide whether a node $\boldsymbol{s}$ can reach $\boldsymbol{t}$
  - Many real-world reasoning problem can be abstracted as a graph (e.g., knowledge graph)
  - Many theoretical problems can be reduced to it (e.g., Turing machine halting problem)



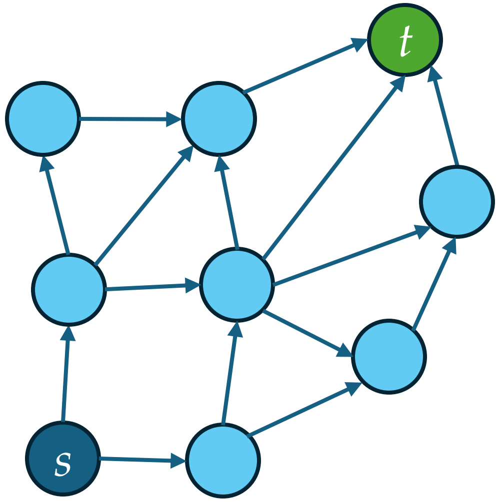
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$



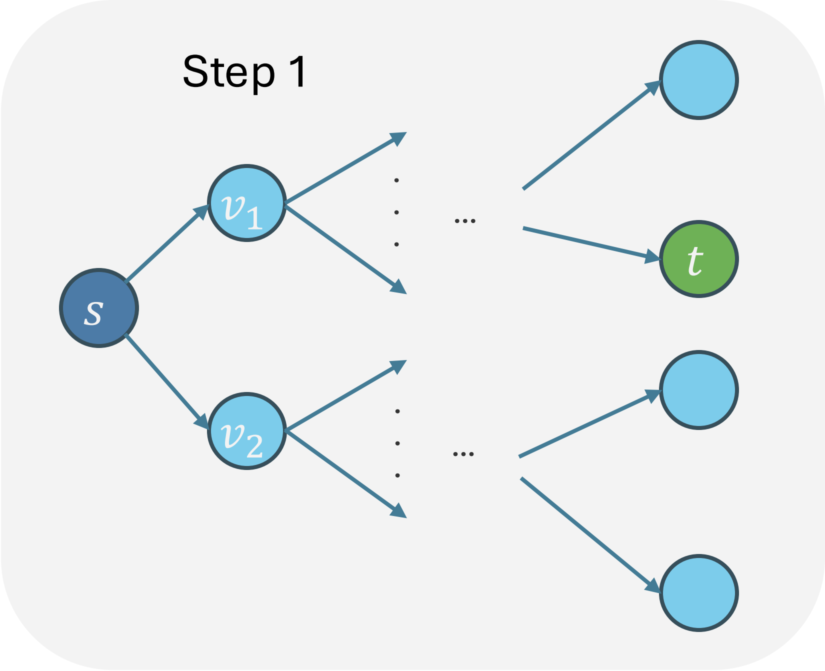
Search process

Problem Definition: Graph reachability

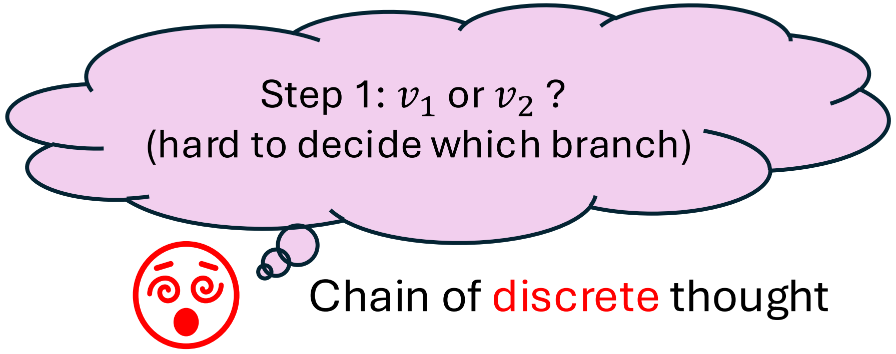
- Given a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, decide whether a node $\boldsymbol{s}$ can reach $\boldsymbol{t}$
  - Many real-world reasoning problem can be abstracted as a graph (e.g., knowledge graph)
  - Many theoretical problems can be reduced to it (e.g., Turing machine halting problem)



$\mathcal{G} = (\mathcal{V}, \mathcal{E})$

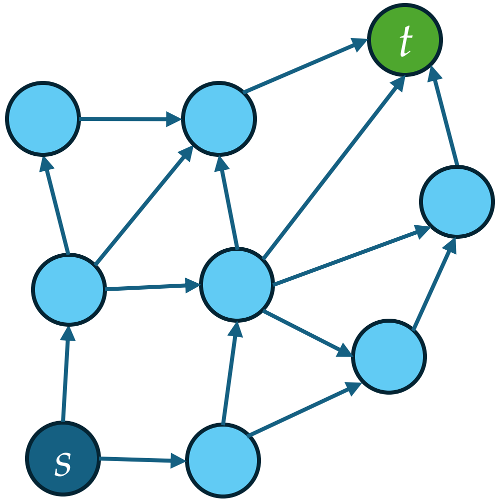


Search process

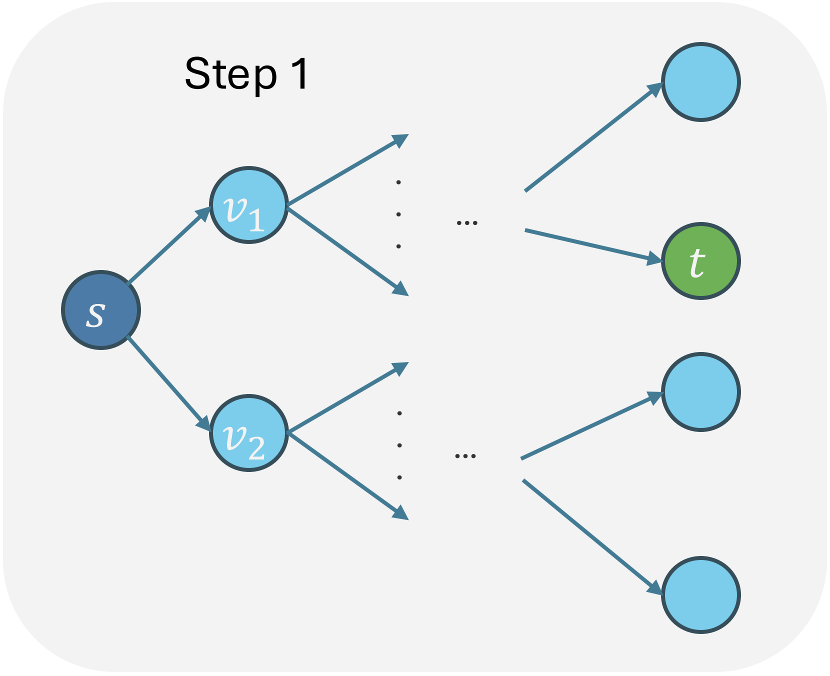


Problem Definition: Graph reachability

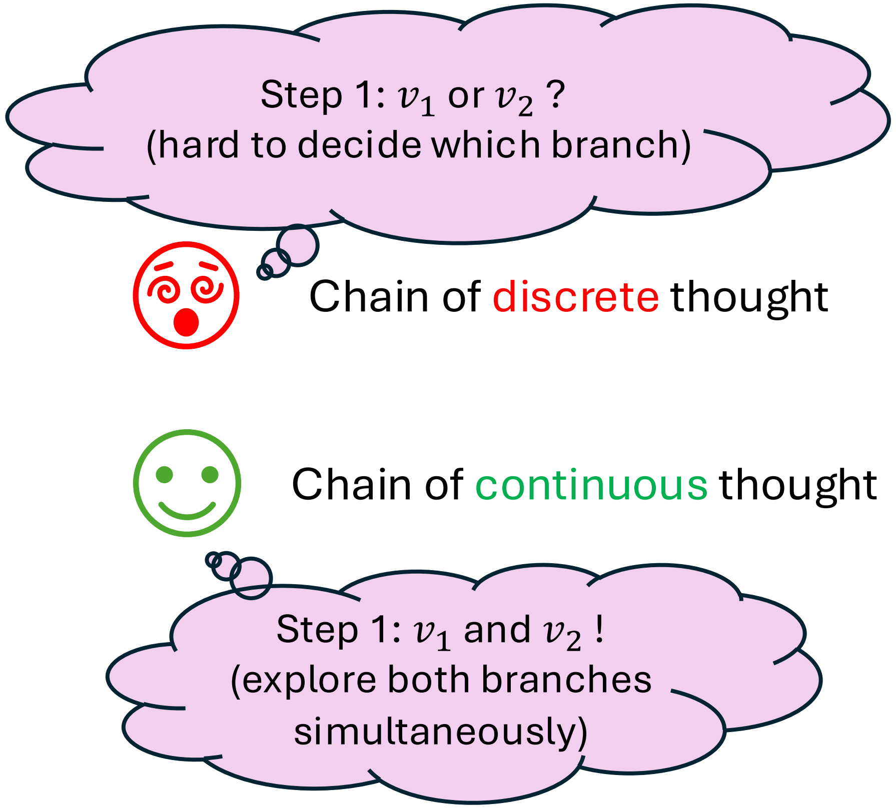
- Given a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, decide whether a node $\boldsymbol{s}$ can reach $\boldsymbol{t}$
  - Many real-world reasoning problem can be abstracted as a graph (e.g., knowledge graph)
  - Many theoretical problems can be reduced to it (e.g., Turing machine halting problem)



$\mathcal{G} = (\mathcal{V}, \mathcal{E})$

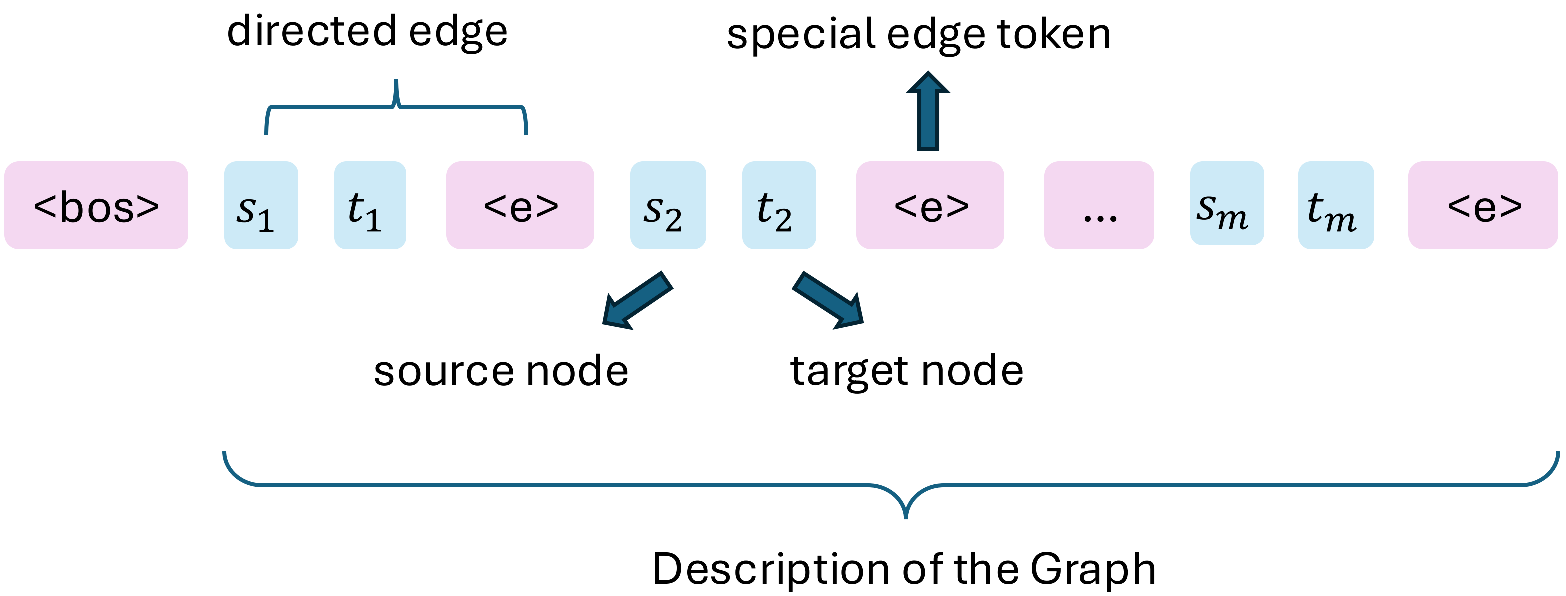


Search process



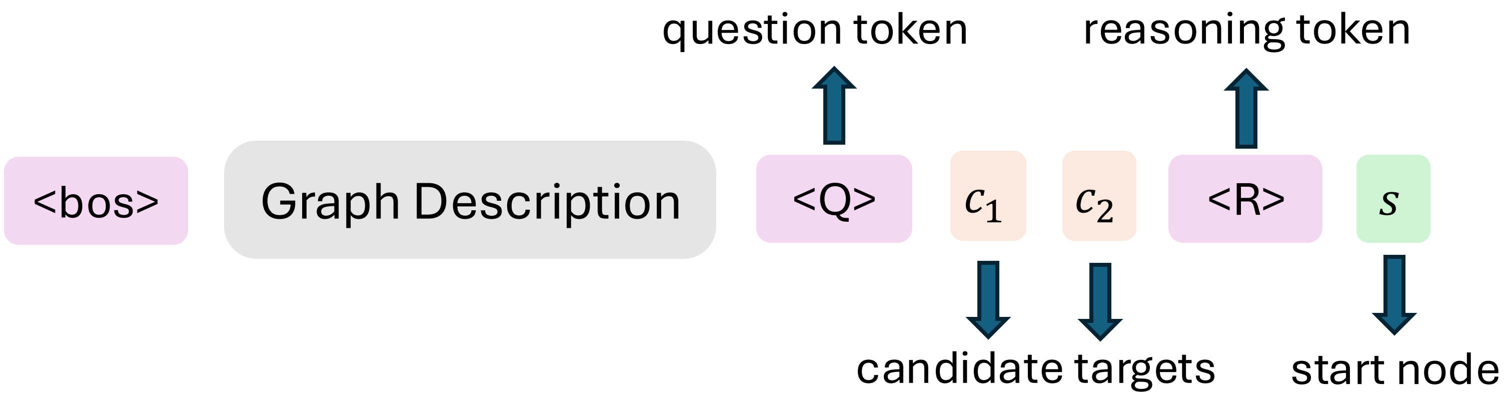
Prompt format

Given two candidate destination nodes, decide which one can be reached



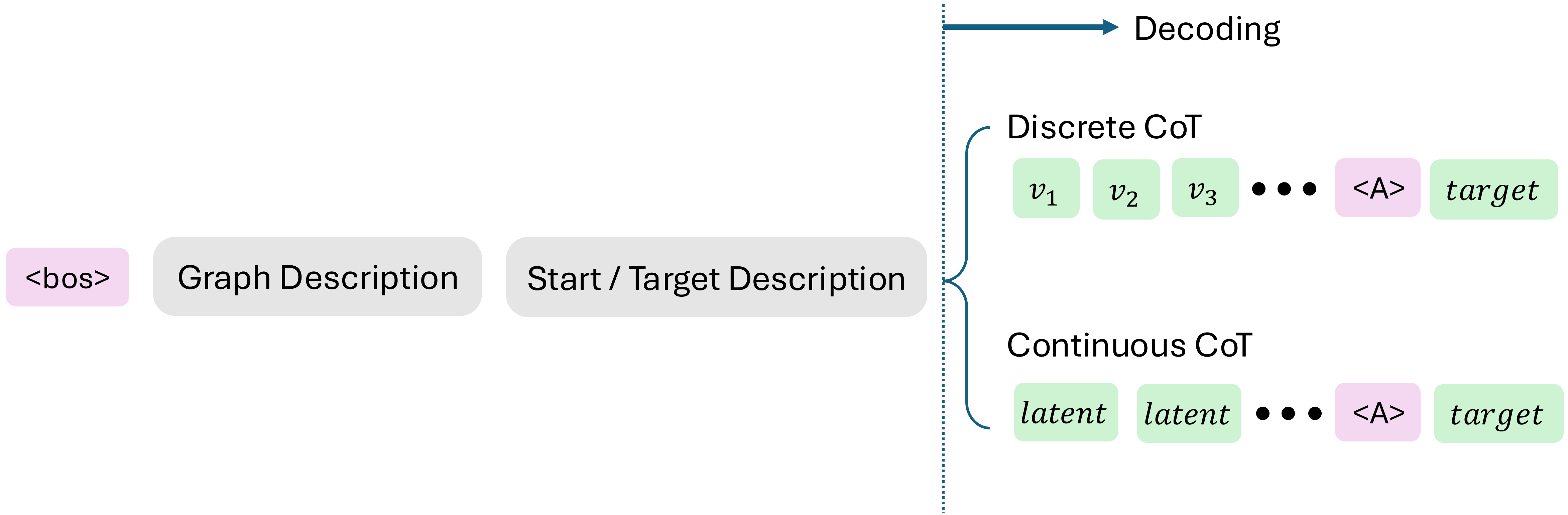
Prompt format

Given two candidate destination nodes, decide which one can be reached



Prompt format

Given two candidate destination nodes, decide which one can be reached



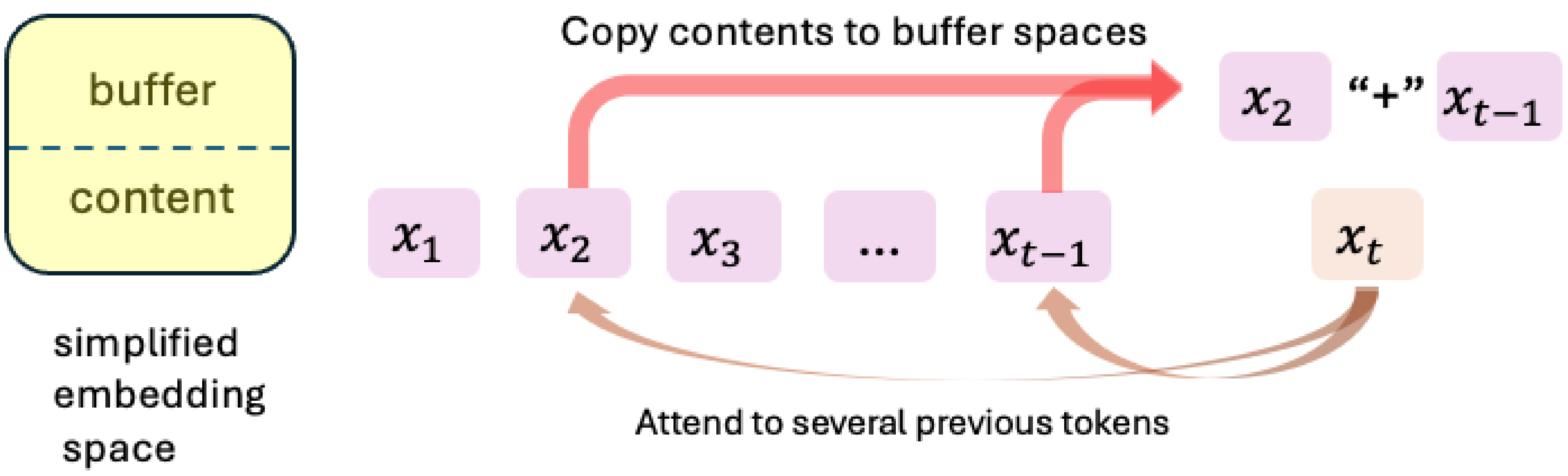
Main theorem

Theorem (informal)

For n-vertex directed graphs, a **2-layer** transformer with continuous CoT can solve reachability using $O(n)$ decoding steps with $O(n)$ embedding dimensions.

Secret Sauce: Superposition of the embeddings!

Mechanism in a single Attn-MLP block



Attention as an aggregator:

- Aggregate the information along the sequence axis.
- Form a superposition of concepts.

MLP as a filter:

- Filter out the involved embedding that are not strong enough

Mechanism in a single Attn-MLP block

$$h = \sum_{v \in \mathrm{Voc}} \lambda_v \vec{u}_v$$

MLP as a filter

$$h' = W_2 \sigma(W_1 h)$$
$$= U \sigma(U^T h)$$

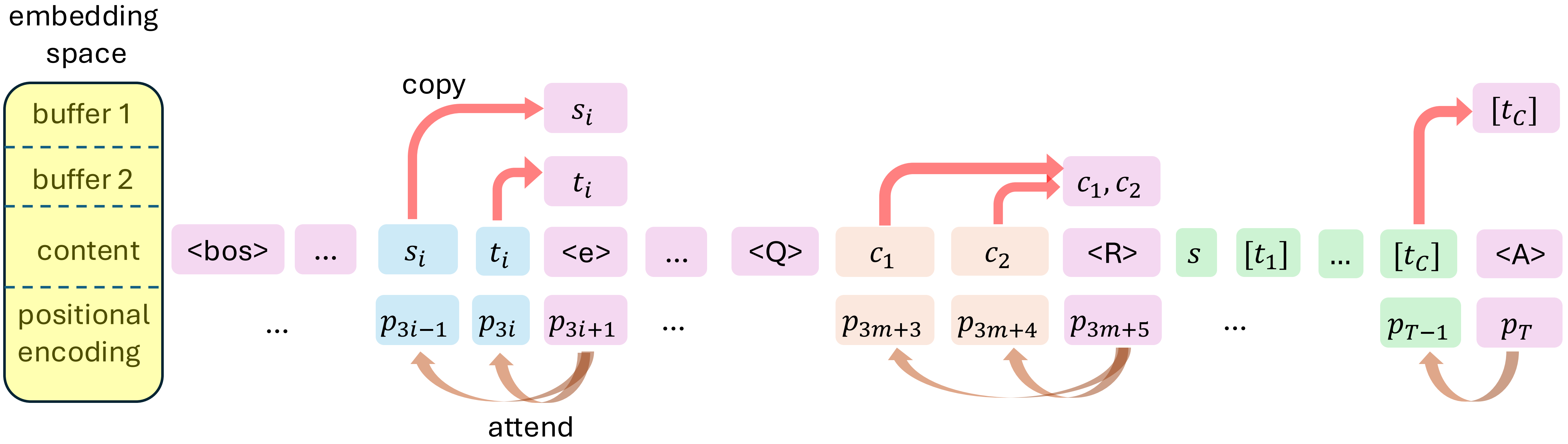
$$h' \propto \sum_{v \in \mathrm{Voc}} \mathbb{I}\{\lambda_v \geq \varepsilon\} \vec{u}_v$$

Eliminate noise

$U = [\vec{u}_1, \vec{u}_2, \ldots, \vec{u}_M]$: the embedding matrix

First-layer attention

Goal: collect all history information together into embedding space.

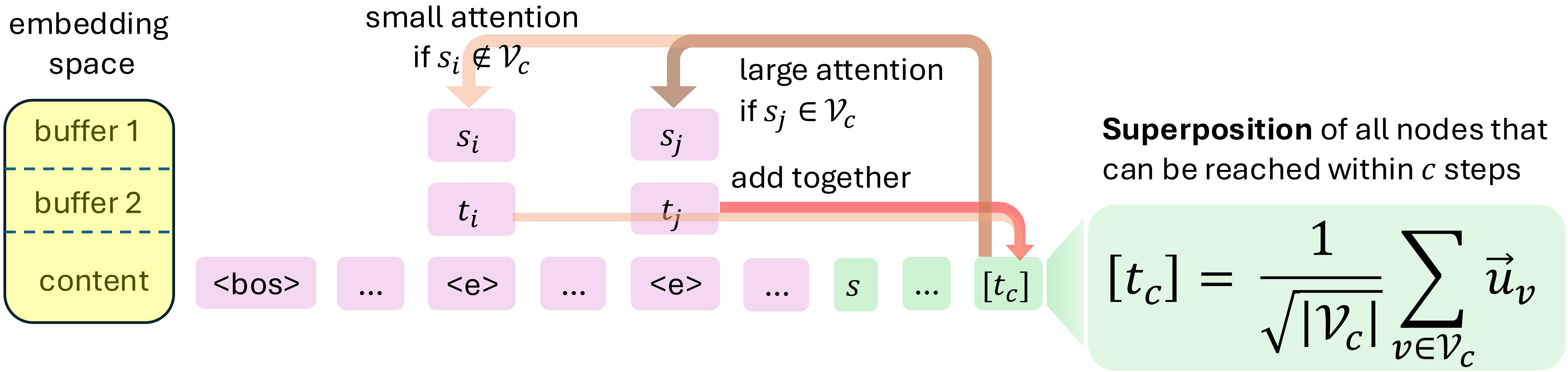


$[t_c]$ Continuous thought at step c

<A> Special answer token

FFN layers: removing low-attended embeddings

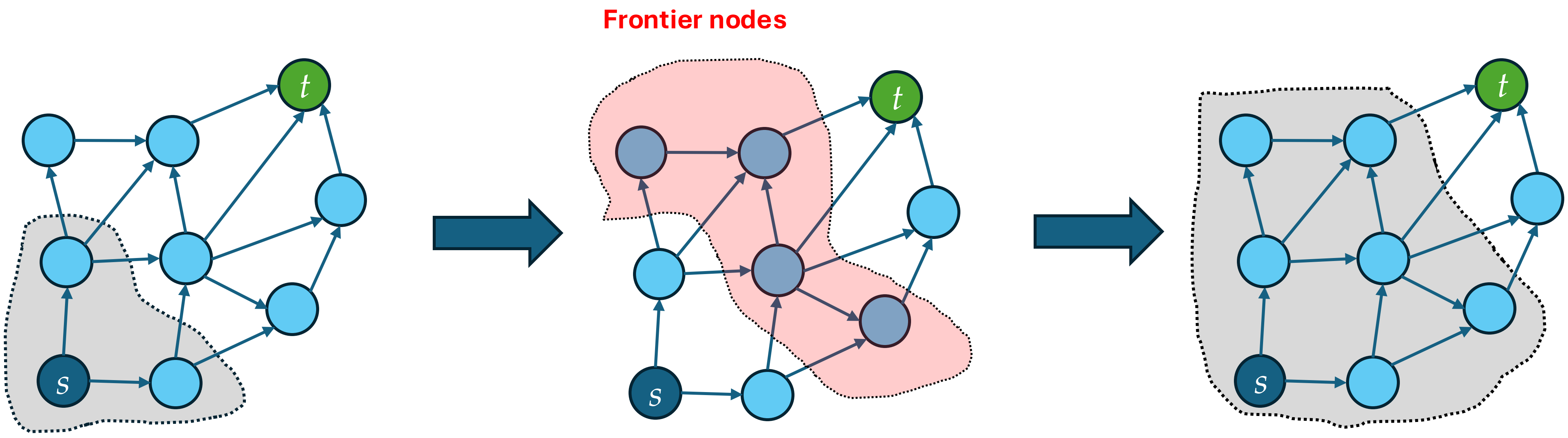
Second-layer attention



One-step expansion of $\mathcal{V}_c$

FFN layers: removing low-attended embeddings

Continuous CoT: Decoding as search



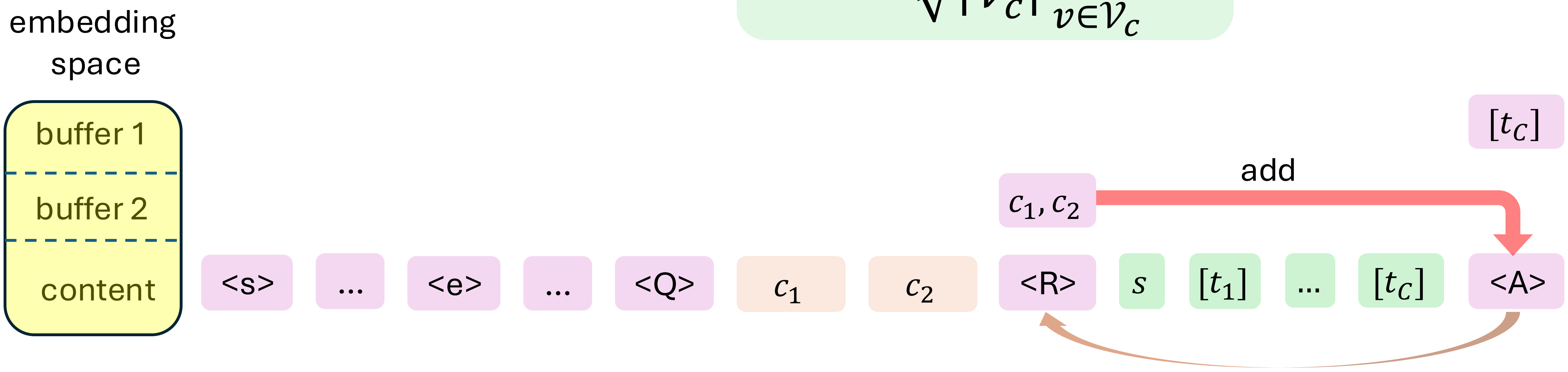
$$[t_1] = \frac{1}{\sqrt{|\mathcal{V}_1(s)|}} \sum_{v \in \mathcal{V}_1} \vec{u}_v$$

$$[t_2] = \frac{1}{\sqrt{|\mathcal{V}_2(s)|}} \sum_{v \in \mathcal{V}_2} \vec{u}_v$$

Autoregressive Decoding

$$[t_c] = \frac{1}{\sqrt{|\mathcal{V}_c|}} \sum_{v \in \mathcal{V}_c} \vec{u}_v$$

Reachable set from the starting point



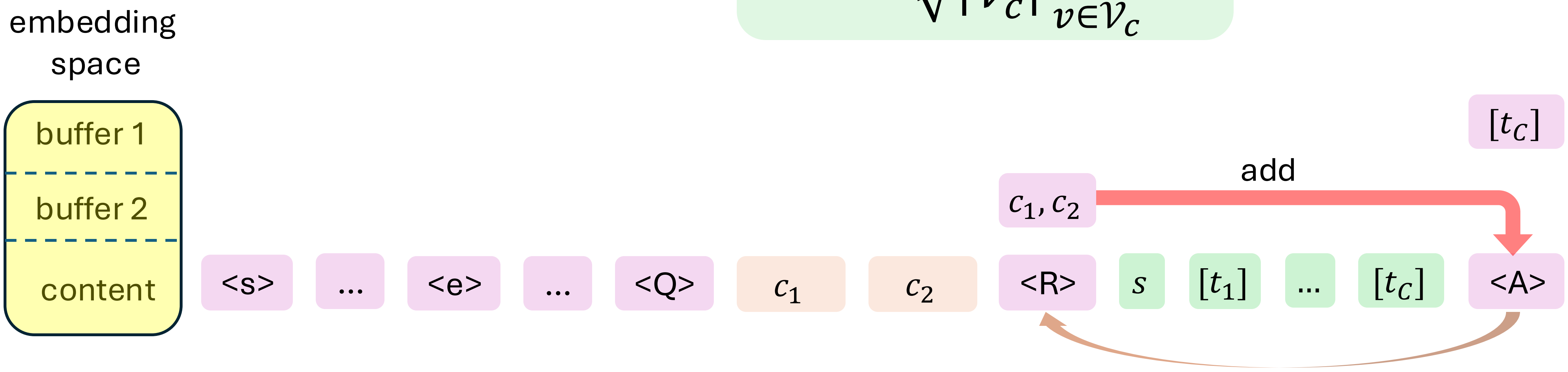
"Measure" $[t_C]$ using c_1 and c_2

The target c that overlaps with **reachable set** will be picked and returned

Autoregressive Decoding

$$[t_c] = \frac{1}{\sqrt{|\mathcal{V}_c|}} \sum_{v \in \mathcal{V}_c} \vec{u}_v$$

Reachable set from the starting point

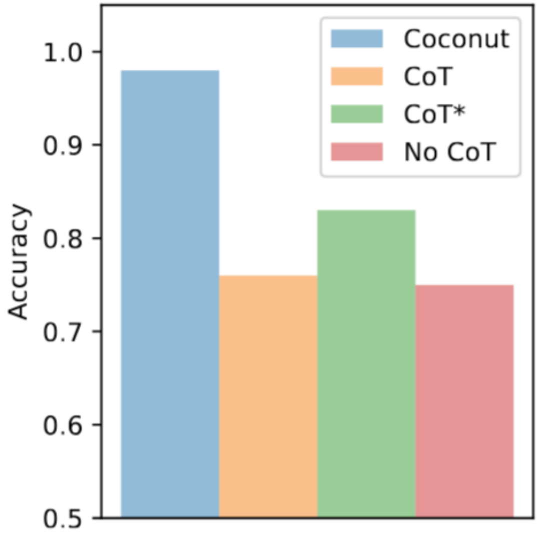


"Measure" $[t_C]$ using c_1 and c_2

The target c that overlaps with **reachable set** will be picked and returned

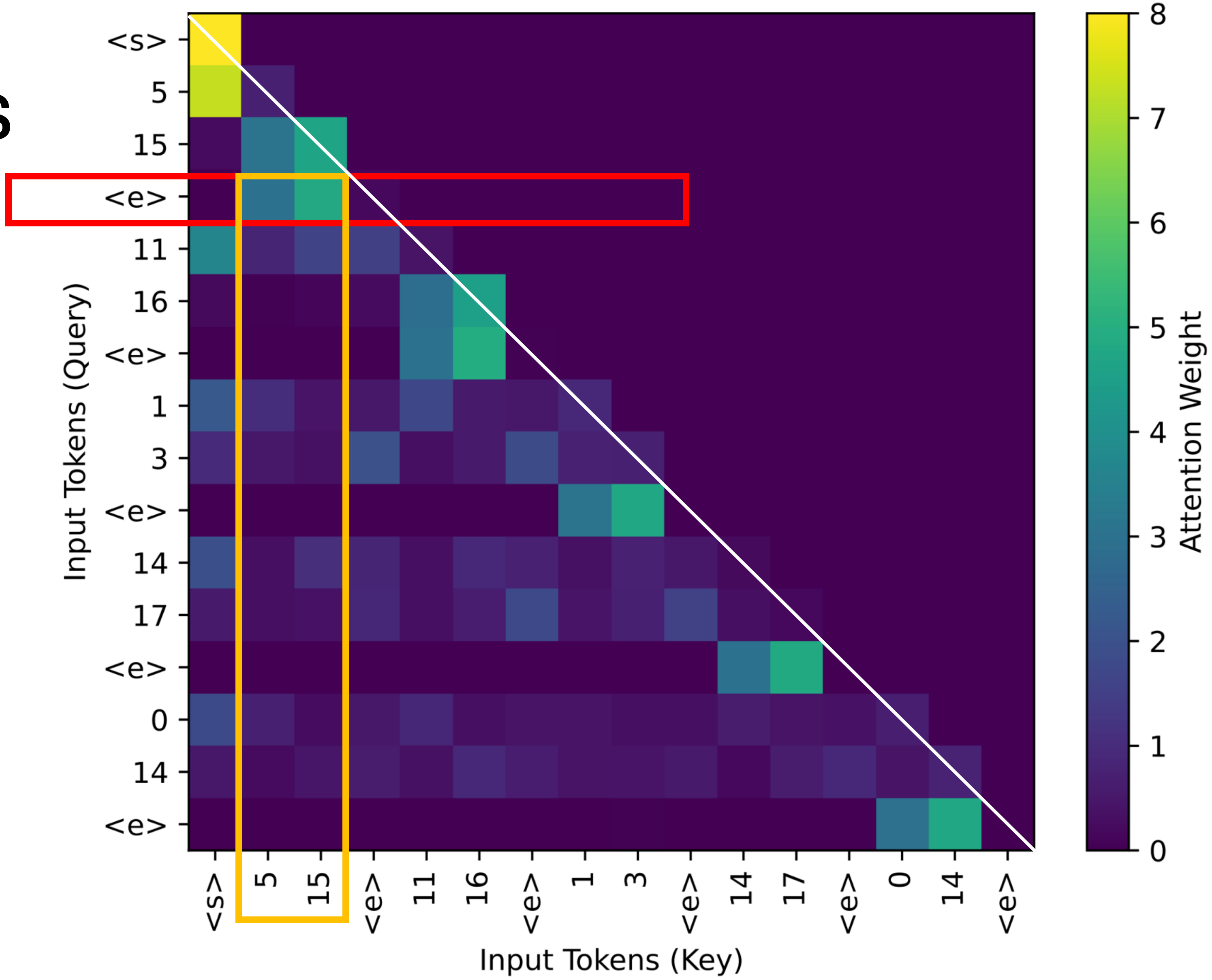
Comparison of continuous and discrete CoT

- Dataset: a subset of ProsQA[1], symbolic sequence, 3-4 steps
- Model: GPT2-style decoder
- Training: multi-stage training, stage i predicts i-th node in the optimal path using previous thoughts
- Overall results: 2-layer transformer with continuous CoT (Coconut) beats 12-layer transformer with discrete CoT (CoT*)

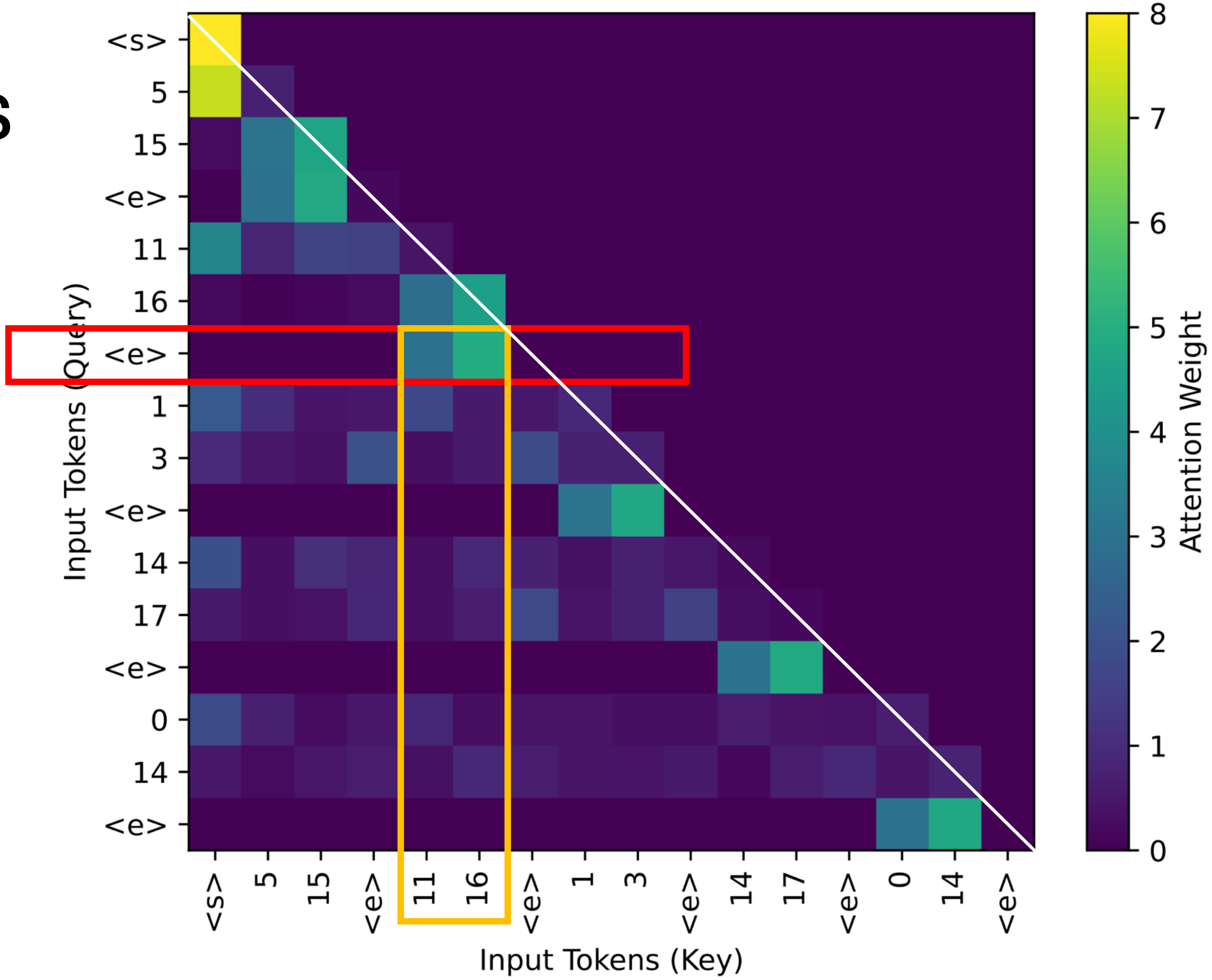


[1] Hao, Shibo, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. "Training large language models to reason in a continuous latent space." *arXiv preprint arXiv:2412.06769* (2024).

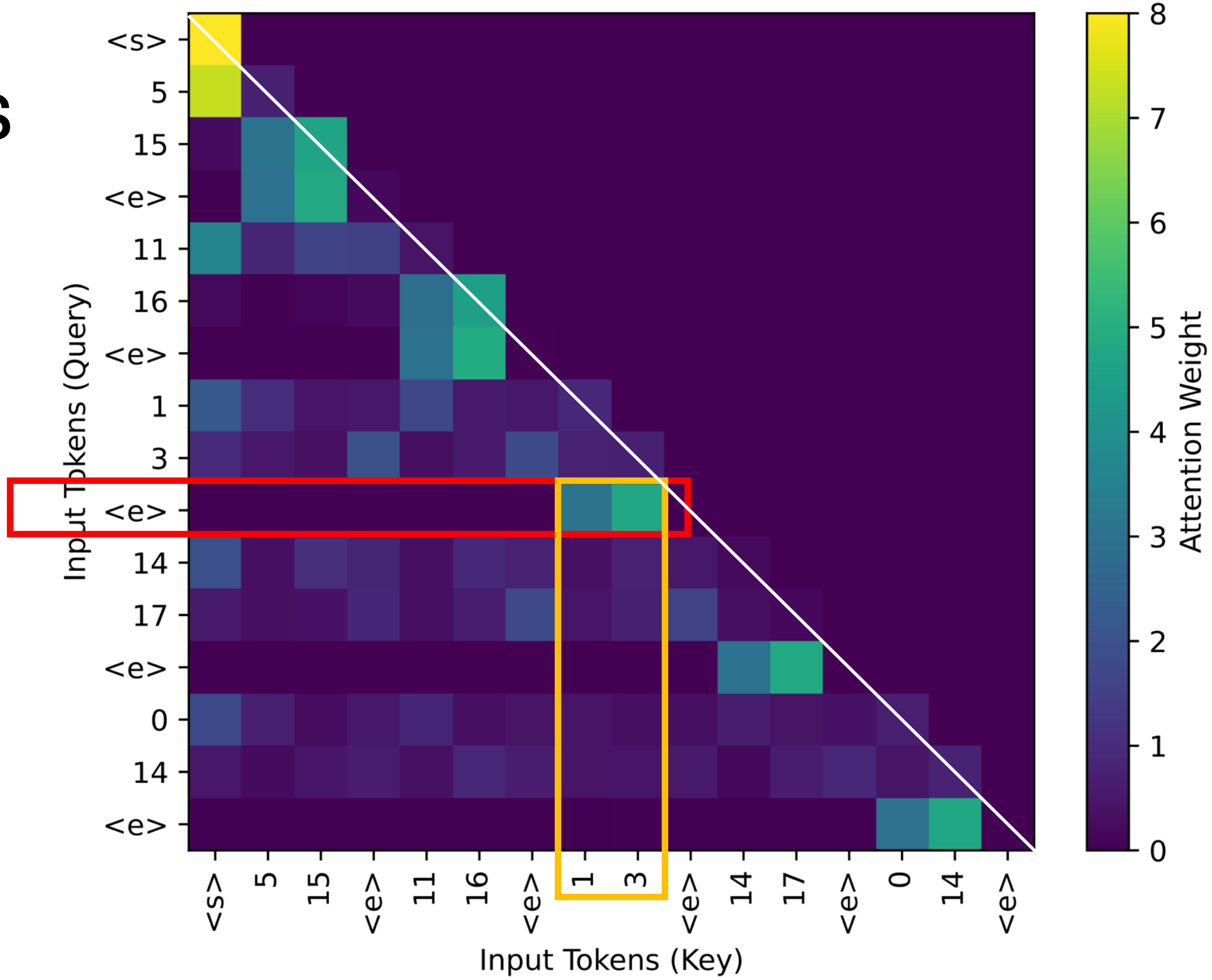
Layer 1
Attention Patterns



Layer 1
Attention Patterns



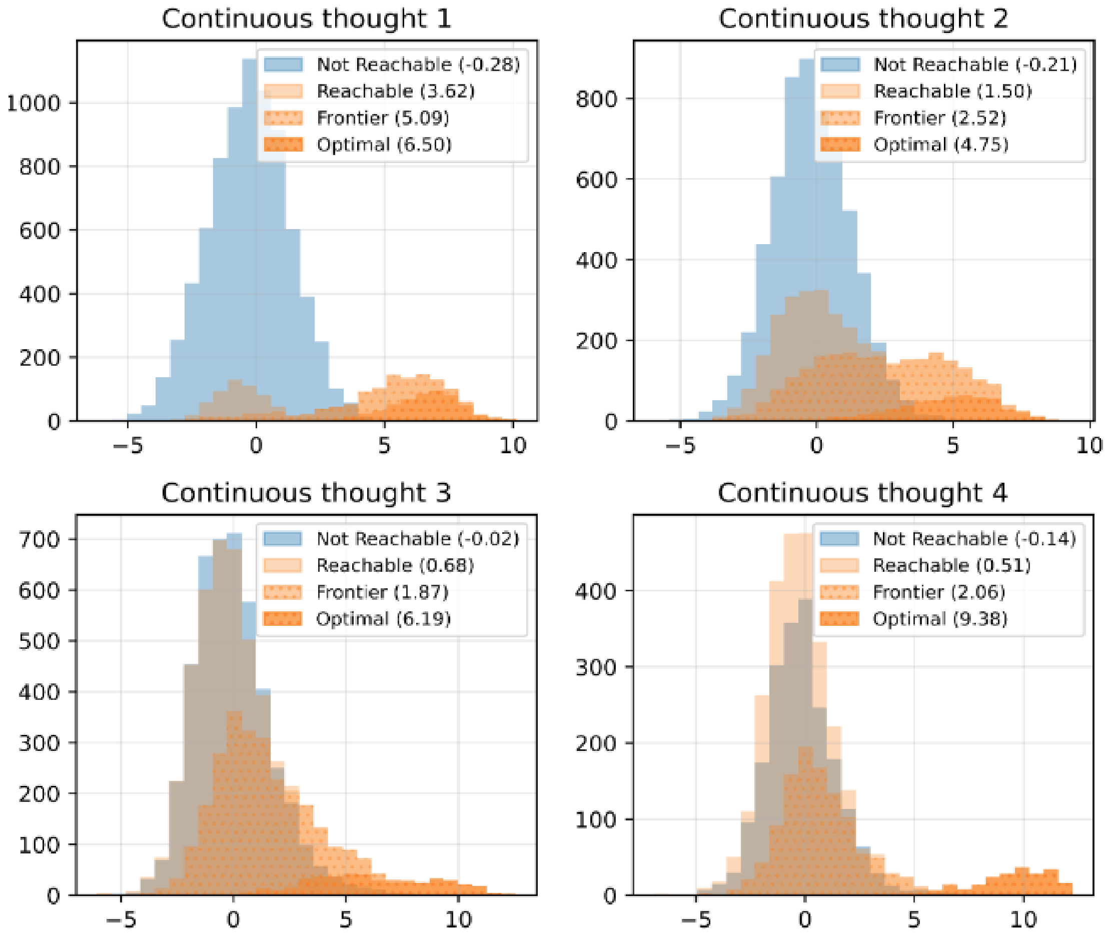
Layer 1
Attention Patterns



Superposition **emerges** during training

Inner products of the current thought and each node embedding

$$[t_c] = \frac{1}{\sqrt{|\mathcal{V}_c|}} \sum_{v \in \mathcal{V}_c} \vec{u}_v$$



COCONUT

Four Kinds of Nodes

- **Reachable node** (reachable from start node within i-th steps)
  - *Frontier node* (exactly i-th steps)
  - *Optimal node* (on the shortest path from the start node to the destination node)
- **Non-reachable node**

Coconut automatically **learns** to encode **frontier/optimal** nodes (**emerging!**)

Discussions

- Continuous thoughts can be powerful but hard to control
  - E.g., superposition states can be a subset of tokens (with different weights)
  - It can emerge even if the training data only contain single discrete traces
- Requires a deeper understanding if we want to use it reliably
  - Mechanism for more general tasks
  - How superposition emerges during training and how to control it

Thanks!