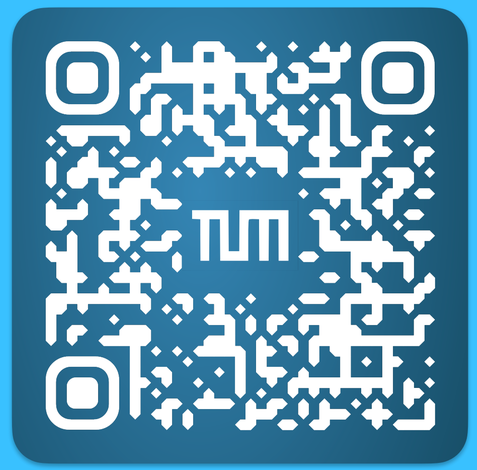


Cognitive Theory for Deliberate Reasoning: **Synthesizing Working Memory Architectures for LLMs**

Alexander Blatzheim
Master Thesis in Information Systems
Supervised by Tobias Eder

Research Group Social Computing @ Technical University of Munich
September 2025

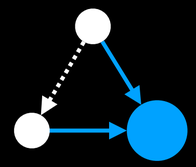
The Problem:

LLMs lack regulation of
expanding reasoning context

The Problem: LLMs Lack Regulation of Expanding Reasoning Context

↳ Approaches generate a new thought in the context of…

…**all** prior thoughts



Expansion leads to more retrieval errors, **risking collapse** *(cf. Liu et Al., 2023)*

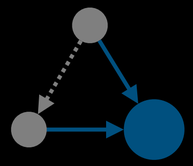
↳ Reasoning LLMs

or…

The Problem: LLMs Lack Regulation of Expanding Reasoning Context

↳ Approaches generate a new thought in the context of…

…**all** prior thoughts

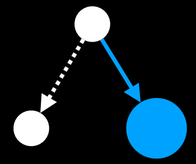


Expansion leads to more retrieval errors, **risking collapse** *(cf. Liu et Al., 2023)*

↳ Reasoning LLMs

or…

…**only** direct parent thoughts



Limited insight re-use

↳ **Graph of Thought**
(GoT; Besta et. al., 2024)

- static prompting graph
- limitation contradicts stated motivation

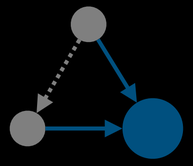
Humans only recall ~7 “thoughts” at once *(Miller, 1956)*

The Solution: **Working Memory** (WM)

The Solution: Working Memory (WM)

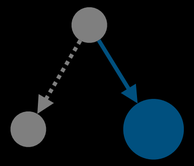
↳ Generate a new thought in the context of…

…**all** prior thoughts



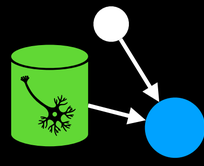
Expansion leads to more retrieval errors, **risking collapse** *(cf. Liu et al., 2023)*

…**only** direct parent thoughts



Limited insight re-use

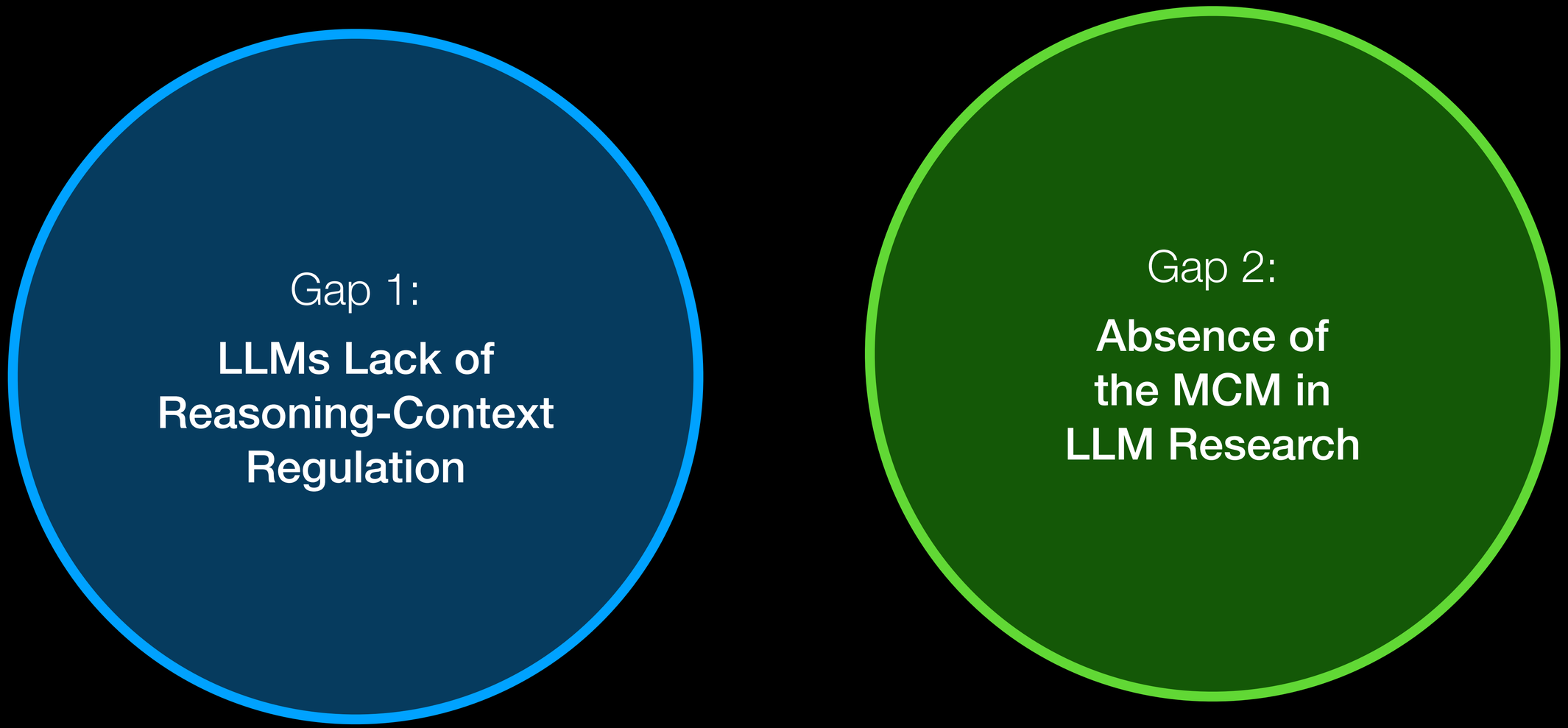
…a curated **Working Memory**



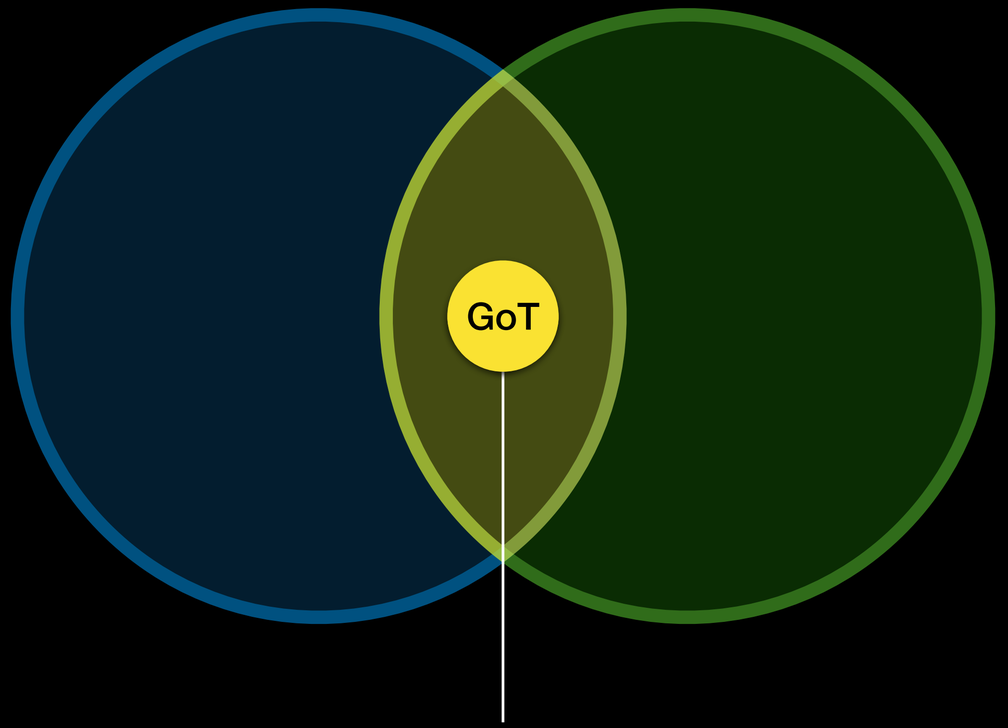
- Limited capacity
- Active control and curation of context

Dominant model in cognitive science (the *Multi-Component Model; MCM*) remains **entirely absent** from LLM research

Research Niche at the Intersection of a Dual Gap



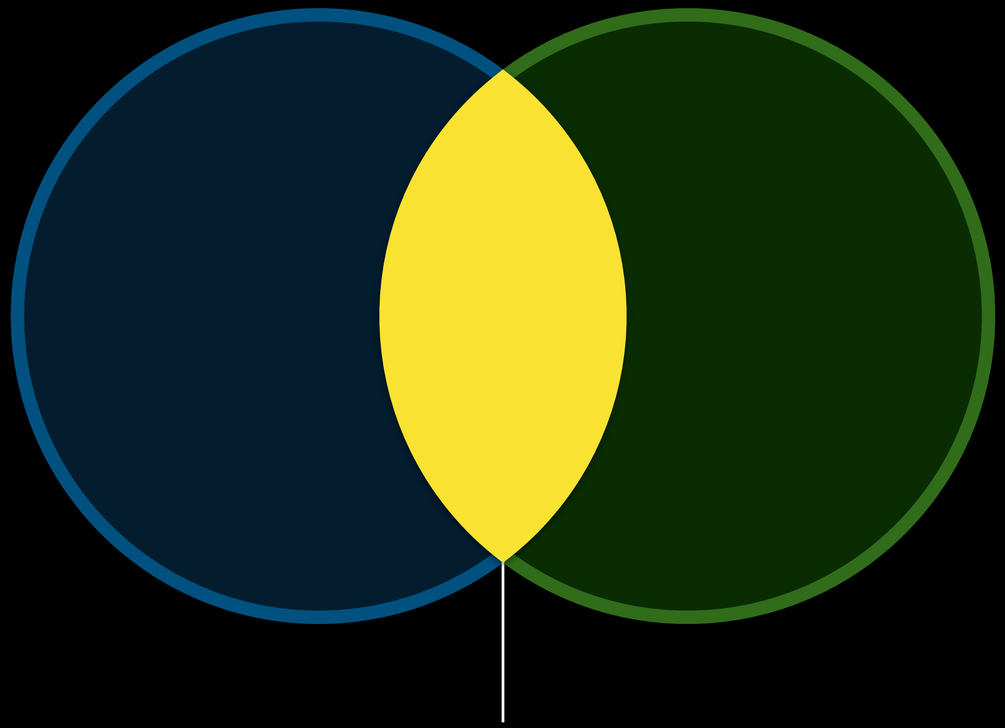
Research Niche at the Intersection of a Dual Gap



Research Question:

“How might a working memory mechanism for LLMs be designed to selectively retrieve relevant material from an arbitrarily large reasoning graph, when generating a new thought in GoT?”

Research Niche at the Intersection of a Dual Gap



Design Question:

“How might the Multi-Component Model of WM be transferred architecturally to the domain of LLM research?”

PART 1

Theoretical Foundation

Model-Agnostic Properties of Working Memory

Part I: Theoretical Foundation

Holistic frameworks of WM are conceptual tools deliberately leaving room for discussion.
Yet, consensus across models is that WM:

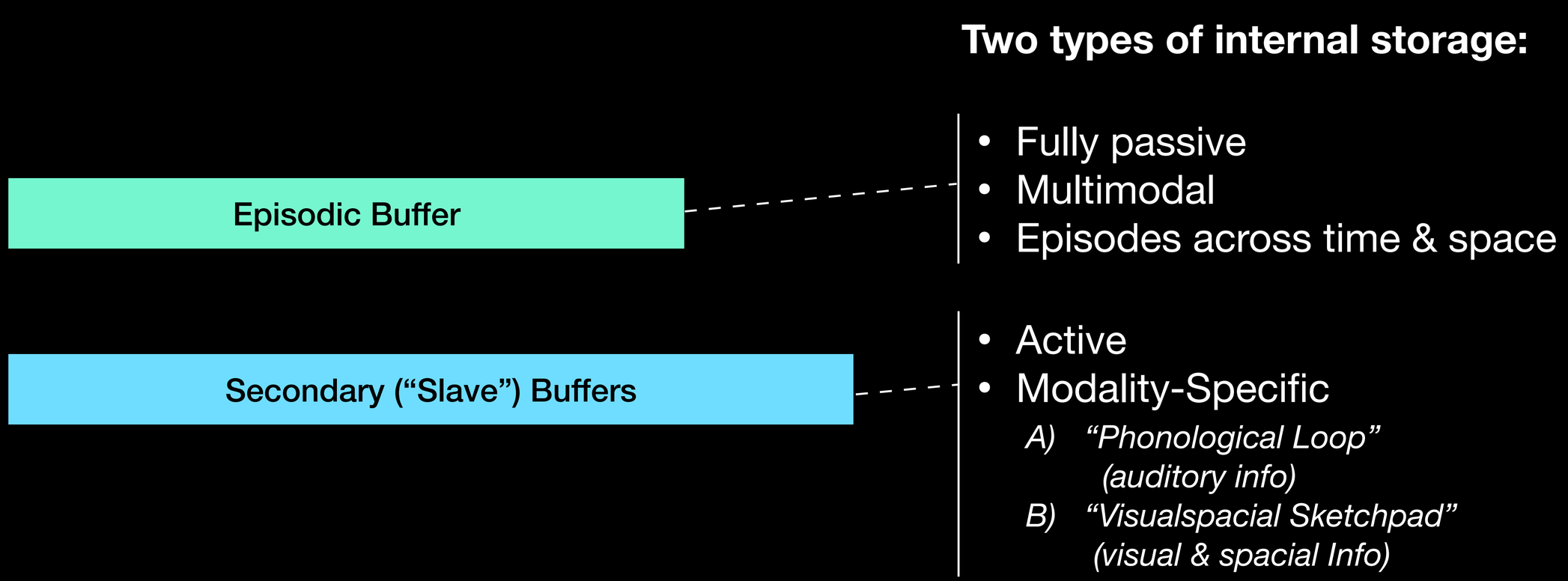
- is a limited capacity store
  ↳ Forgetting — Views: Decay mechanism, resource limits, interference
  ↳ Information not attended to fades rapidly
- is tightly coupled with long-term memory (LTM) — details largely unclear
- houses active processing functionality

where representations in memory are:

- actively manipulated
- hierarchically structured (into “chunks”)
- Varying in capacity demand

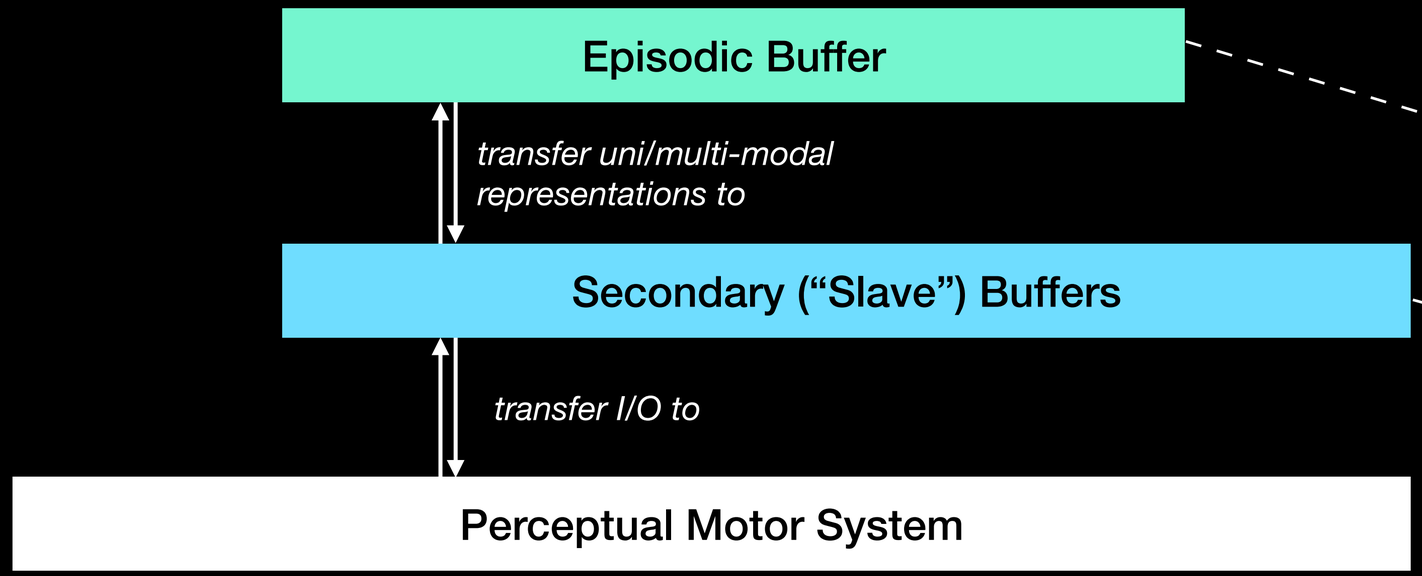
Baddeley (2000), Hitch et. Al. (2025) — See Thesis, Subsection 3.3.1.1

The Multi-Component Model of WM (Our Interpretation, Simplified)



Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

The Multi-Component Model of WM (Our Interpretation, Simplified)



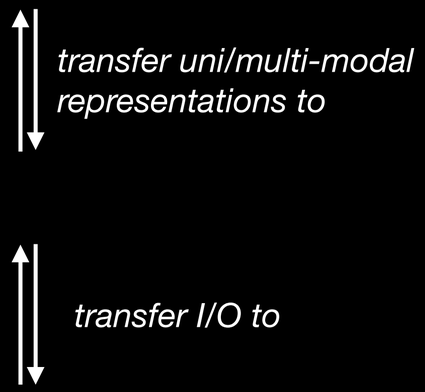
Two types of internal storage:

- Fully passive
- Multimodal
- Episodes across time & space

- Active
- Modality-Specific
- I/O buffers to the Episodic Buffer

Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

The Multi-Component Model of WM (Our Interpretation, Simplified)

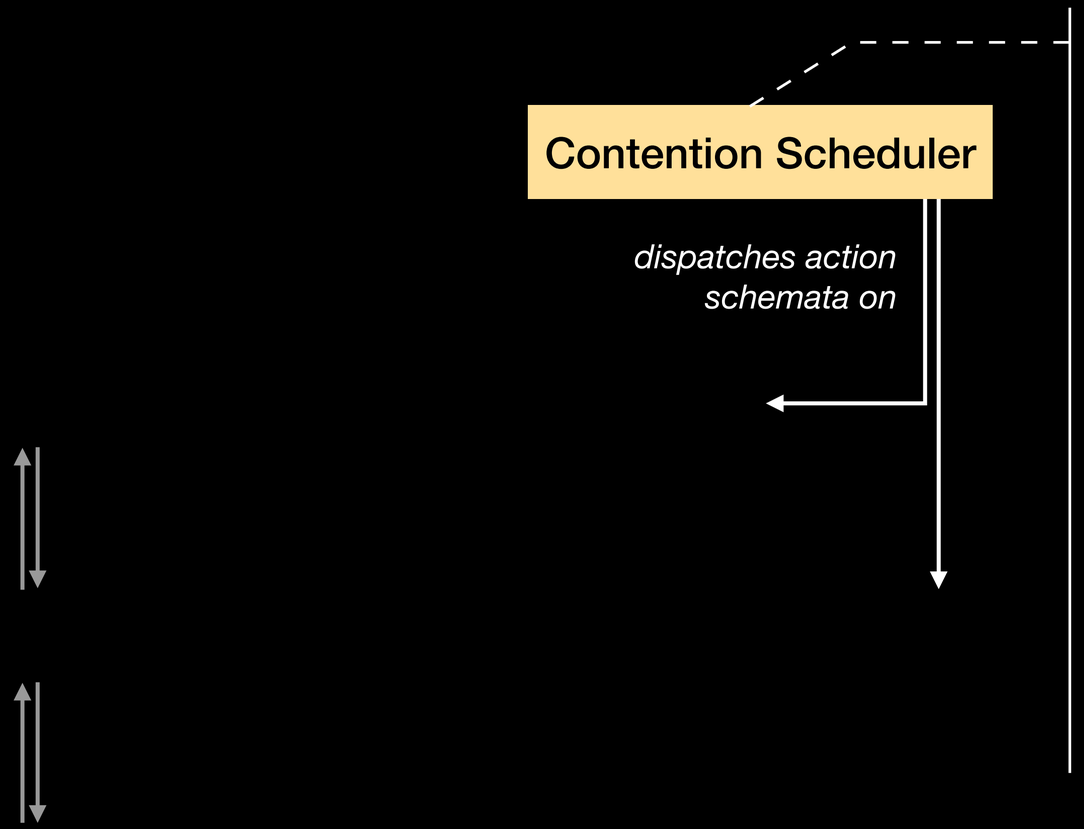


Any data transfer or manipulation occurs via **Action Schemata***:*

- Pre-learned “transactions”
- Hierarchically structured
- Become scheduable iff:
  activation value > activation threshold

Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

The Multi-Component Model of WM (Our Interpretation, Simplified)



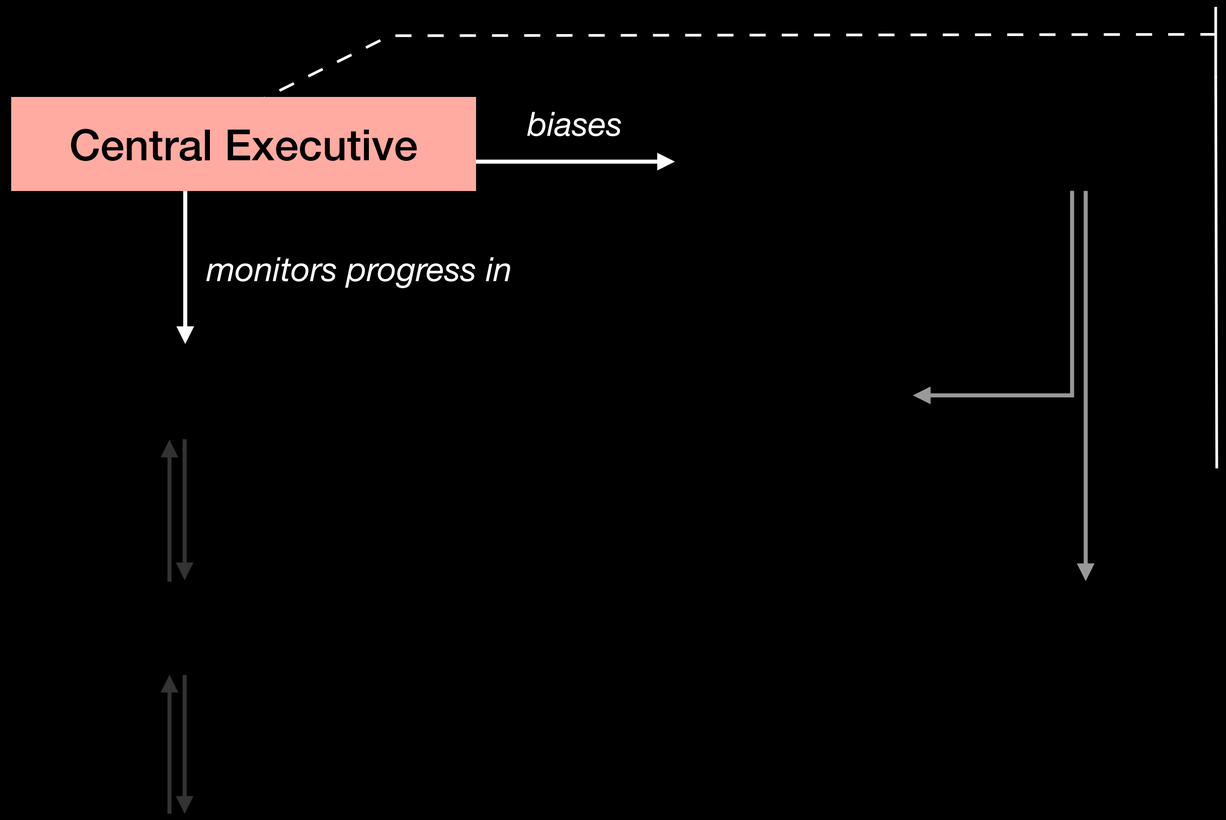
↳ **Automatically** dispatched by Contention Scheduler **iff no conflict arises**, e.g. when:

- Multiple schemas active
- No schema active

Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

The Multi-Component Model of WM (Our Interpretation, Simplified)

Contribution I: The MCM in SE Terms | Part I: Theoretical Foundation

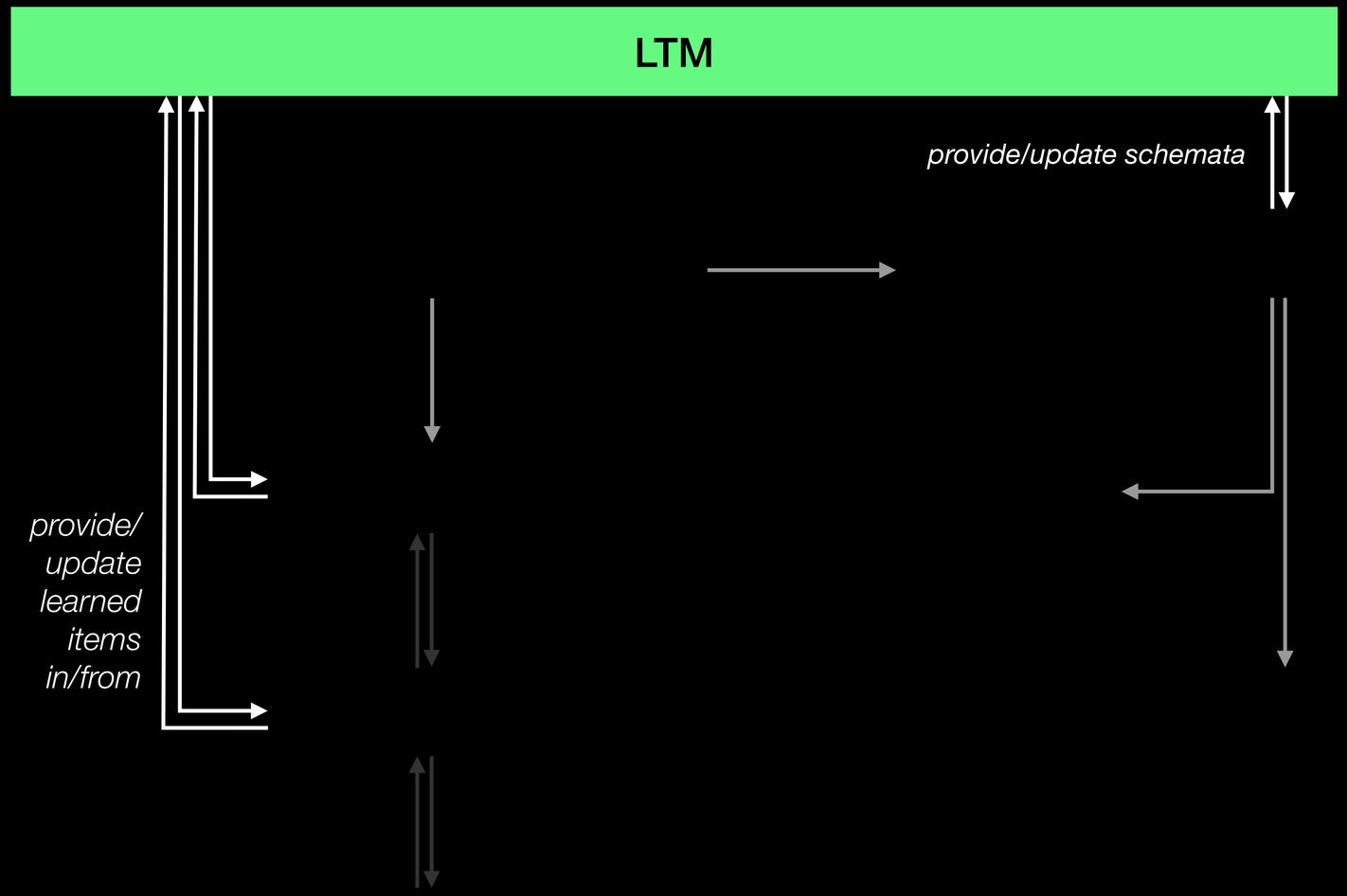


- Stateless process supervisor
- Monitors task progress in Episodic Buffer
- **Resolves conflicts** by **biasing** schemata activation values

Note:
The original MCM subsumes the Contention Scheduler into the Central Executive, citing it as key inspiration. We make it explicit.

Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

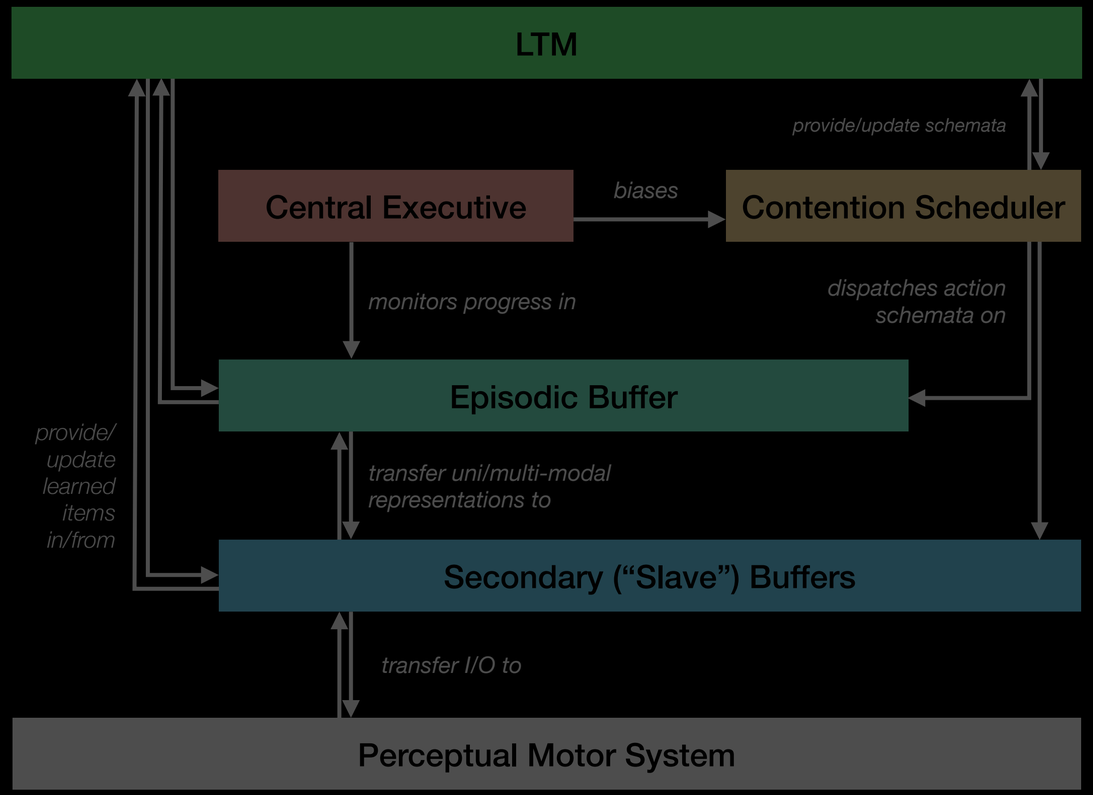
The Multi-Component Model of WM (Our Interpretation, Simplified)



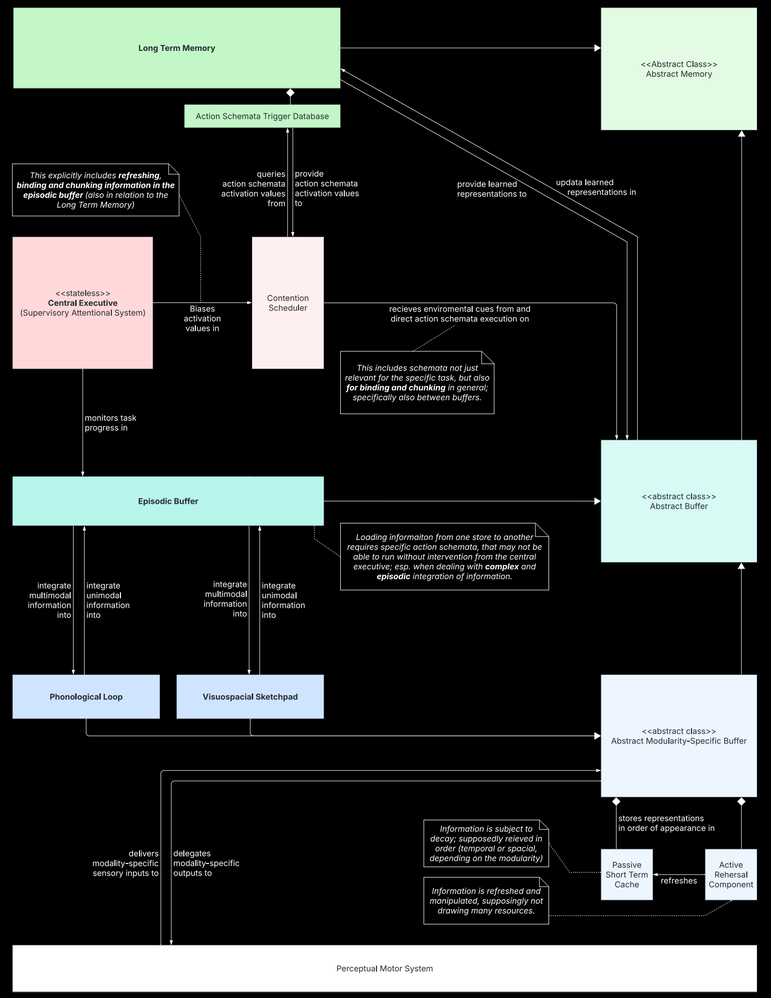
Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

The Multi-Component Model of WM (Our Interpretation, Simplified)

Contribution I: The MCM in SE Terms | Part I: Theoretical Foundation



<



Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

A WM Core Architecture for LLMs: Guiding Principles (Summarized)

1. **Adaptable reasoning structure; independent of reasoning environment** (e.g. GoT)
   - ↳ Arbitrary thought granularity — from sentences to whole paragraphs
   - ↳ Arbitrary thought representation — text, latent, etc.
   - ↳ Arbitrary reasoning granularity — entire sub-graphs or single CoTs *(Wei et. Al, 2023)*

2. Modular, extendible, & minimal viable design

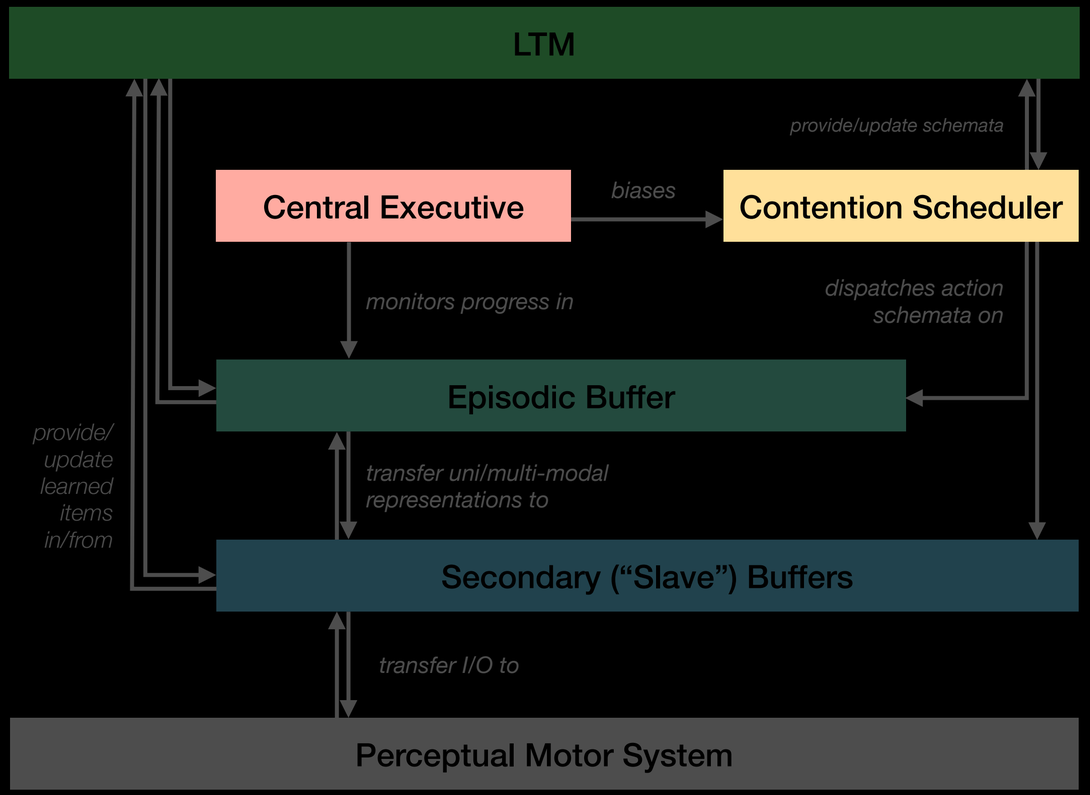
3. Maximal task-agnosticism

4. Out of scope: **Dynamic planning** of reasoning structures (not possible in GoT)

See Thesis, Subsection 4.1.

Synthesis I: Deciding which MCM Components to Model

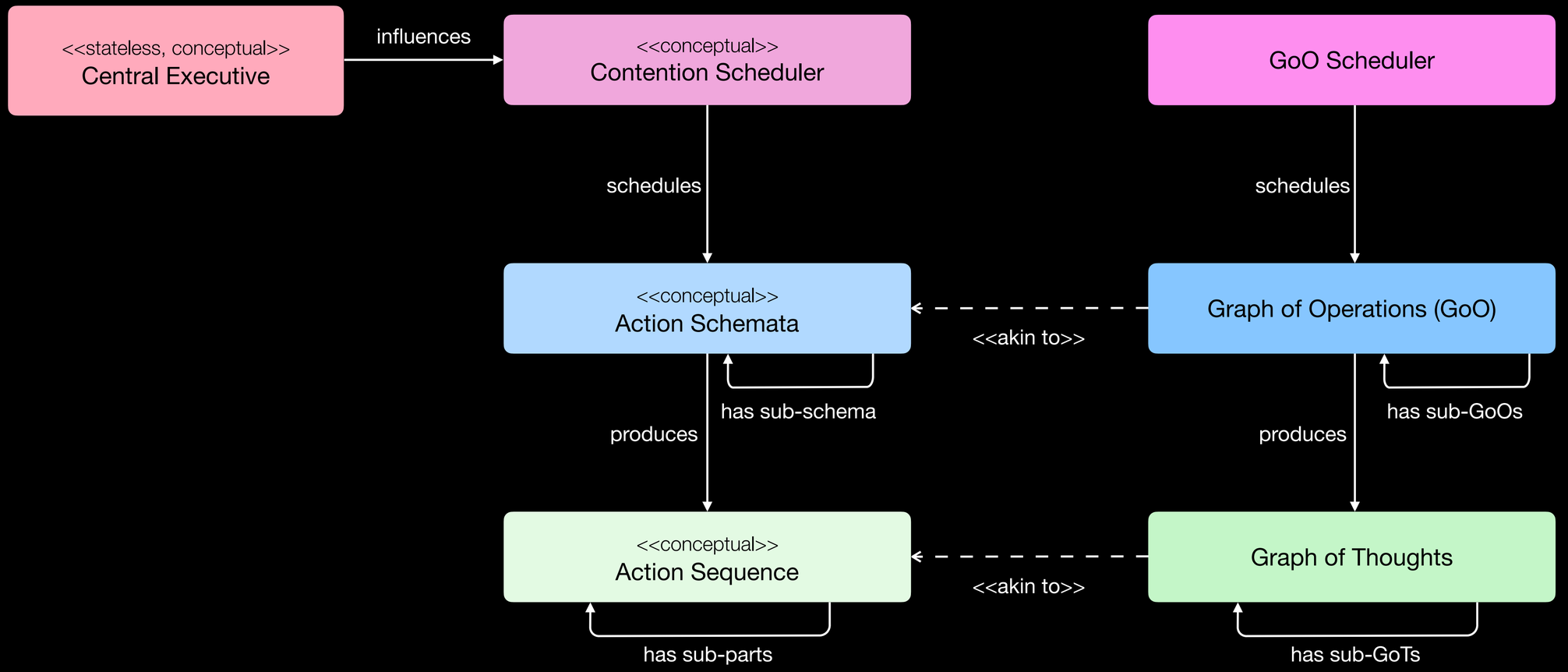
Contribution II: A WM Core Architecture for LLMs | Part I: Theoretical Foundation



Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

Synthesis I: Deciding which MCM Components to Model

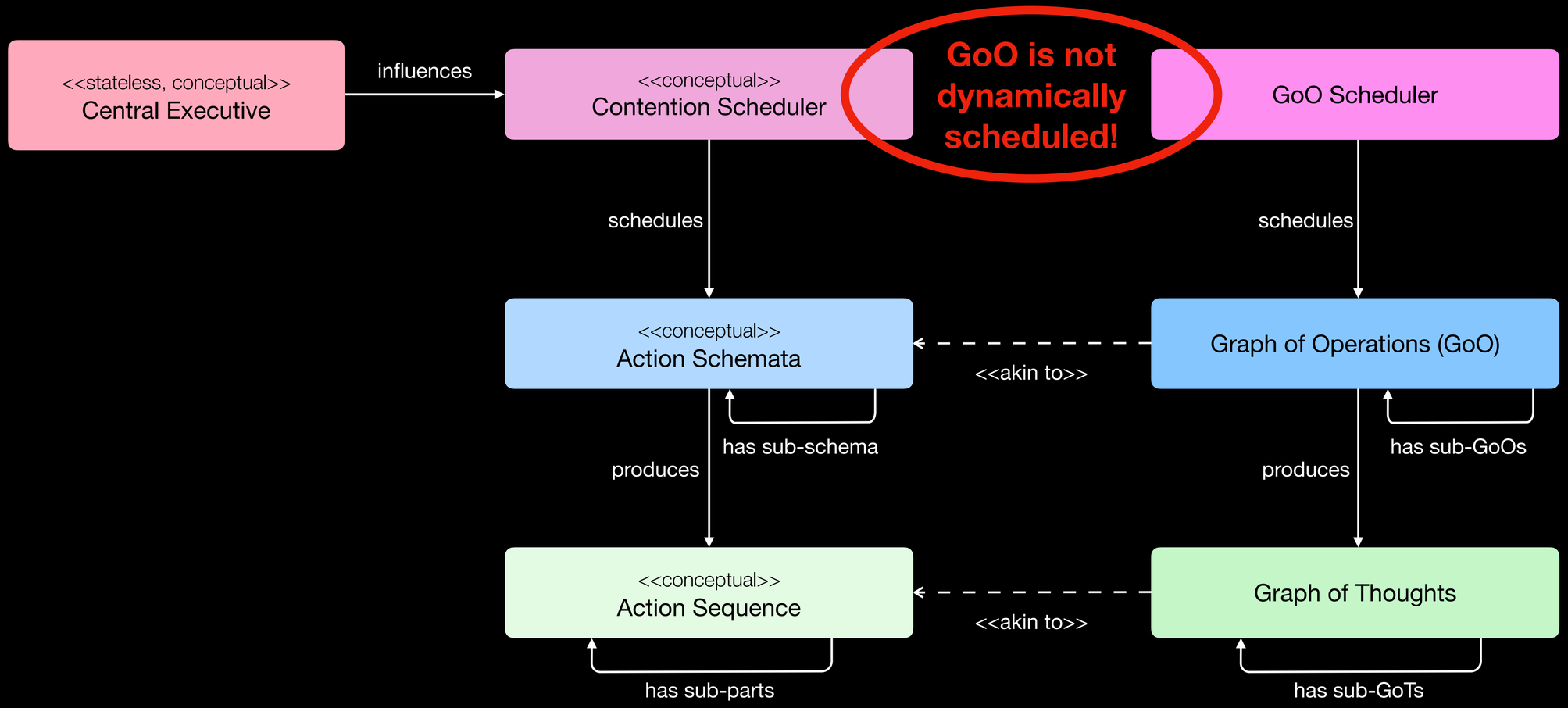
Contribution II: A WM Core Architecture for LLMs | Part I: Theoretical Foundation



See Thesis, Subsection 4.2.1.

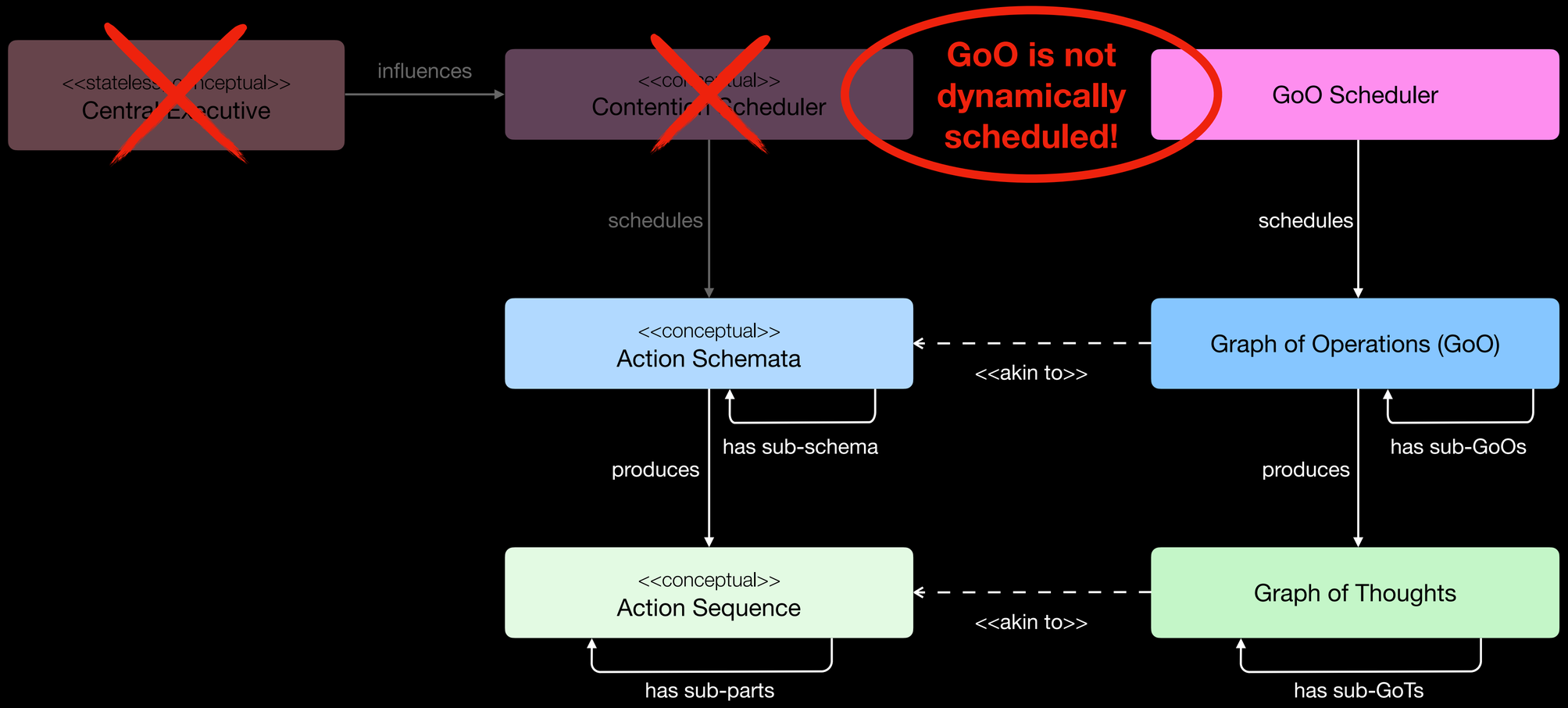
Synthesis I: Deciding which MCM Components to Model

Contribution II: A WM Core Architecture for LLMs | Part I: Theoretical Foundation



See Thesis, Subsection 4.2.1.

Synthesis I: Deciding which MCM Components to Model



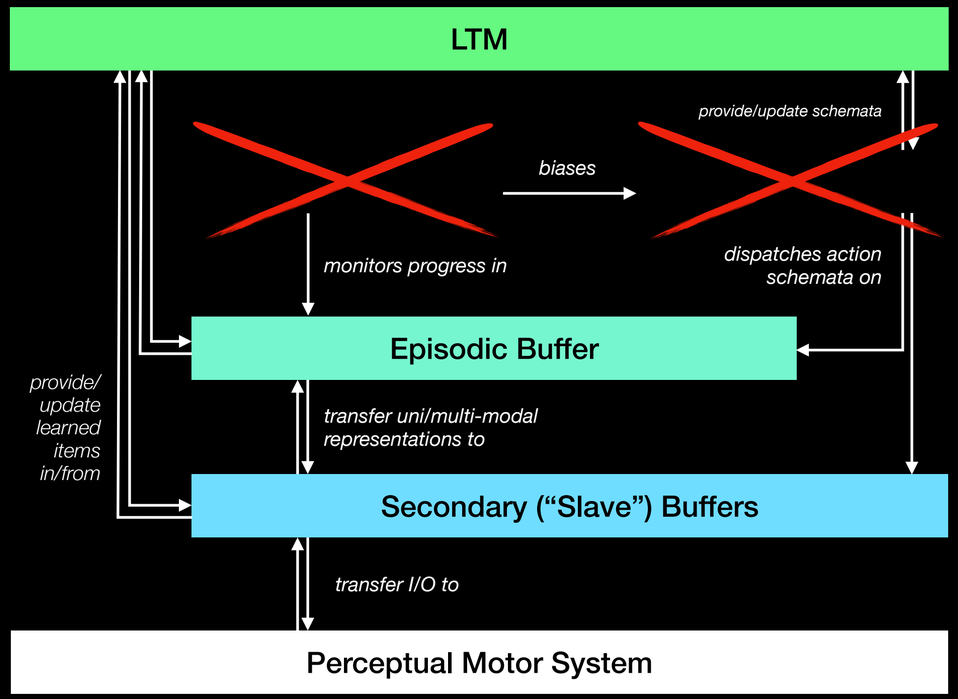
See Thesis, Subsection 4.2.1.

Synthesis I: Deciding which MCM Components to Model

→ No planning of reasoning structure (GoO)

→ No dedicated Contention Scheduler, nor Central Executive

→ **Fixed transactions** (Action Schemata)



Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

See Thesis, Subsection 4.2.1.

Synthesis I: Deciding which MCM Components to Model

Contribution II: A WM Core Architecture for LLMs | Part I: Theoretical Foundation

Support abstraction / learning

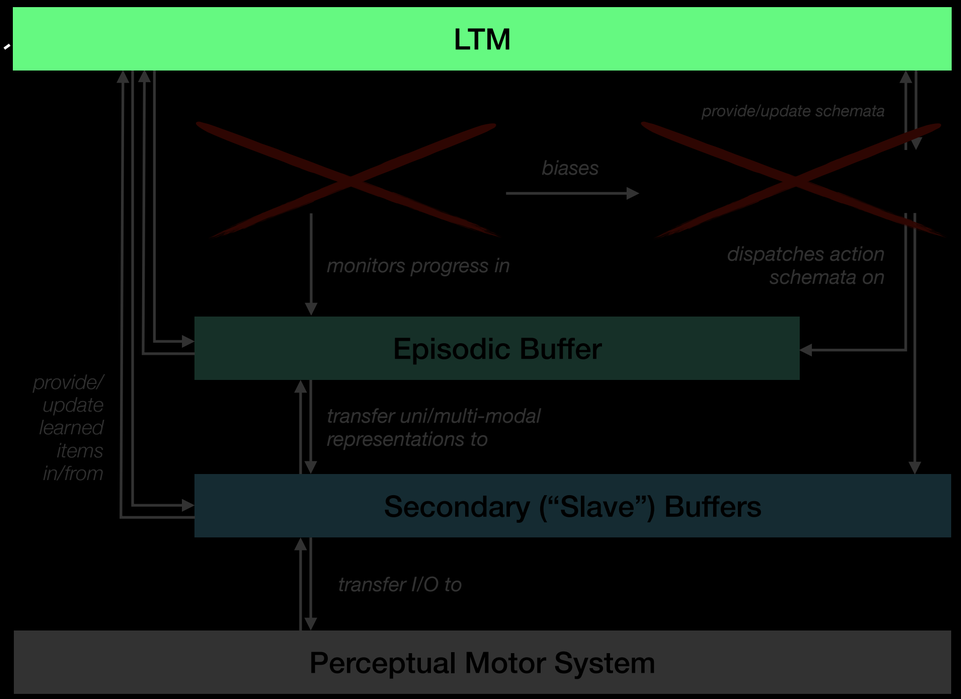
(Cf. Bugress & Hitch, 2005)

Single WM interaction can enable updates

(Cf. Bugress & Hitch, 2005)

Storage of task-items & Action Schemata

(Our Interpretation)



Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

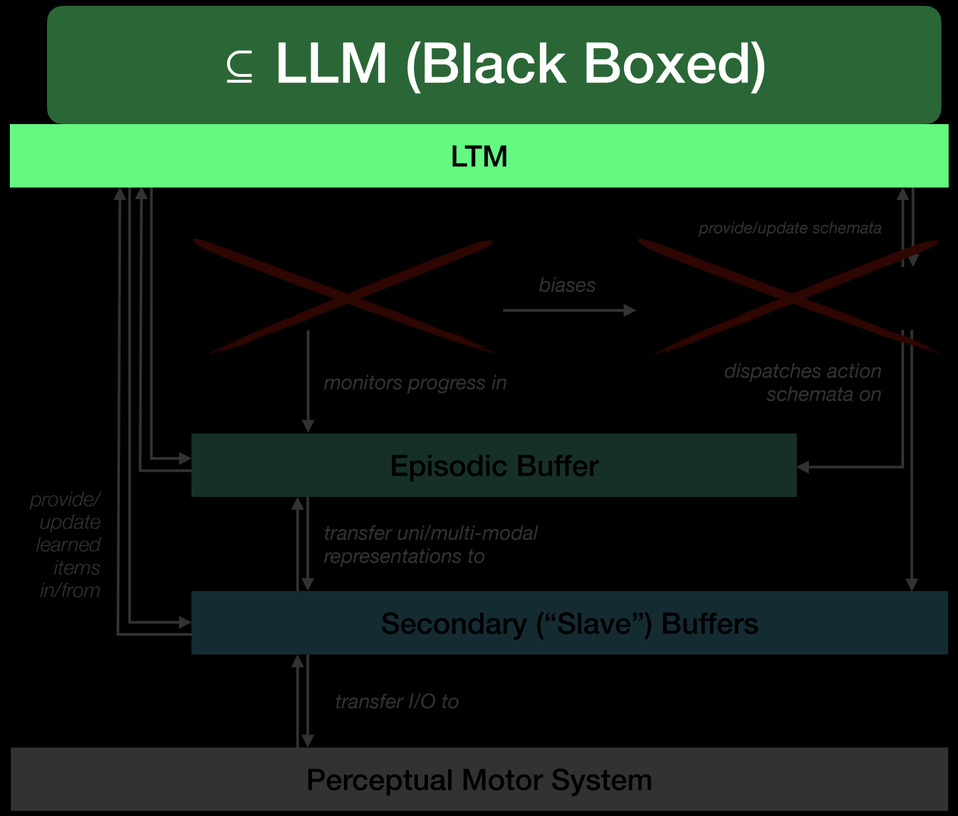
See Thesis, Subsection 4.2.1.

Synthesis I: Deciding which MCM Components to Model

Contribution II: A WM Core Architecture for LLMs | Part I: Theoretical Foundation

	LLM Weights	LTM Module	Trad. Storage
Support abstraction / learning *(Cf. Bugress & Hitch, 2005)*	✓	(✓)	✗
Single WM interaction can enable updates *(Cf. Bugress & Hitch, 2005)*	✓	(✓)	✗
Storage of task-items & Action Schemata *(Our Interpretation)*	✗ Lack persistence *(Mallen et al., 2023)*	(✓)	✗

See Thesis, Subsection 4.2.1.



Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

Synthesis I: Deciding which MCM Components to Model

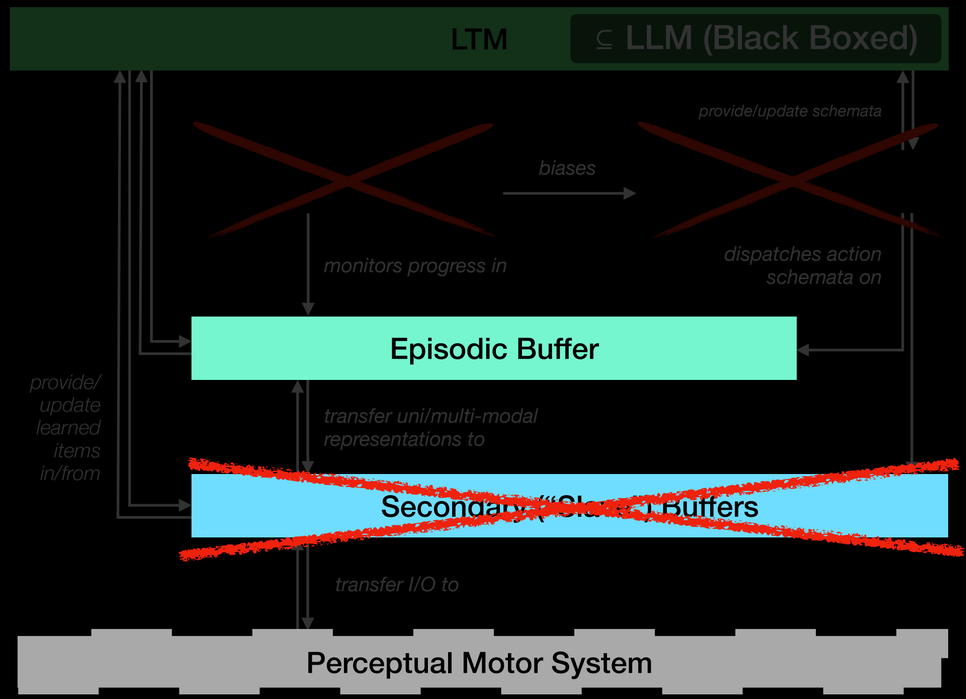
Unimodal setting (Text Only)

↳ **Modality-specific buffers out of scope**

But: Dedicated Pre- & Post-Processing in transactions

WM client is the reasoning environment (e.g. GoT)

↳ Not part of WM itself; requires abstract interface



Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

See Thesis, Subsection 4.2.1.

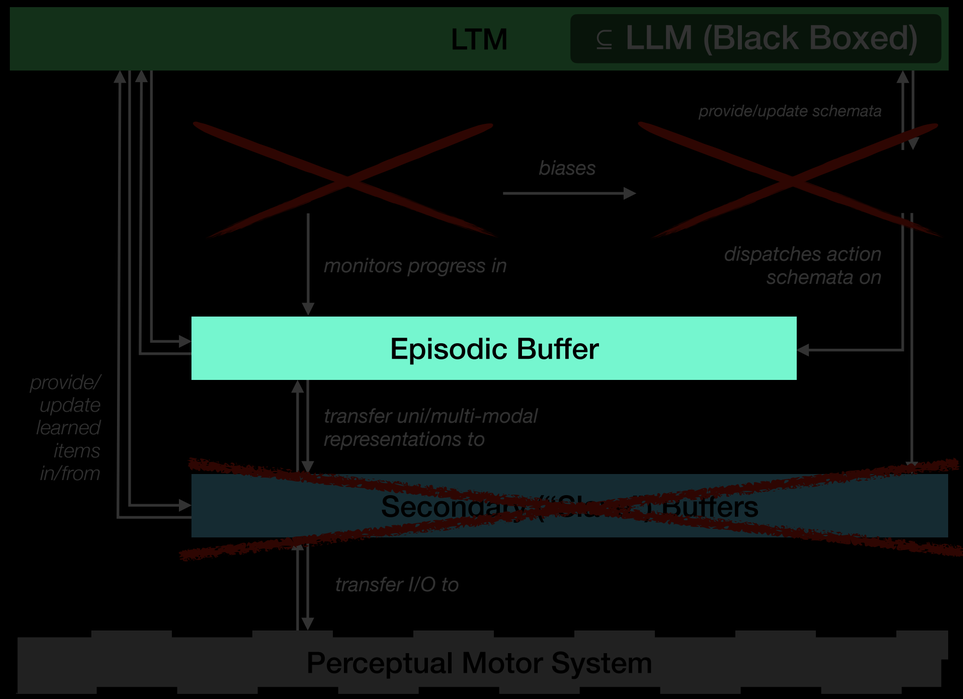
Synthesis I: Deciding which MCM Components to Model

Full focus on the **Episodic Buffer,** characterized by

- Passivity
- Episodicness across “time and space”

And more generally:

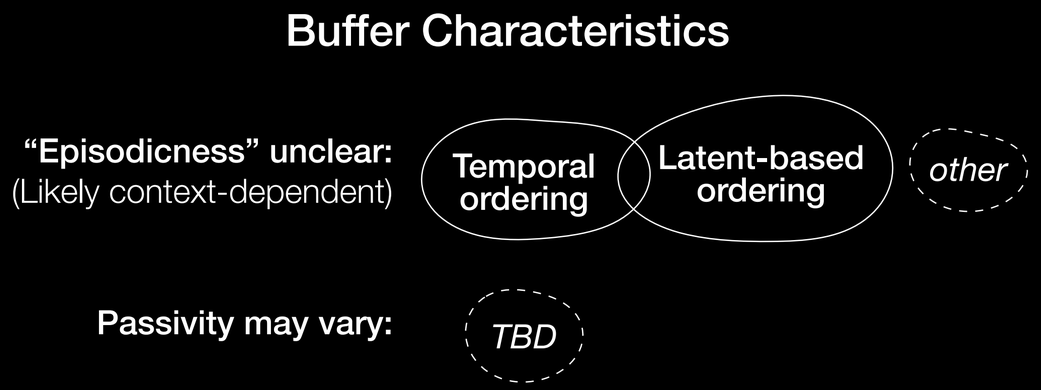
- Varying definitions of **capacity**
  ↳ A notion of **forgetting**
- A **mechanism for chunking** (abstracting/grouping)



Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986) — See Thesis, Subsection 3.3.1.3

See Thesis, Subsection 4.2.1.

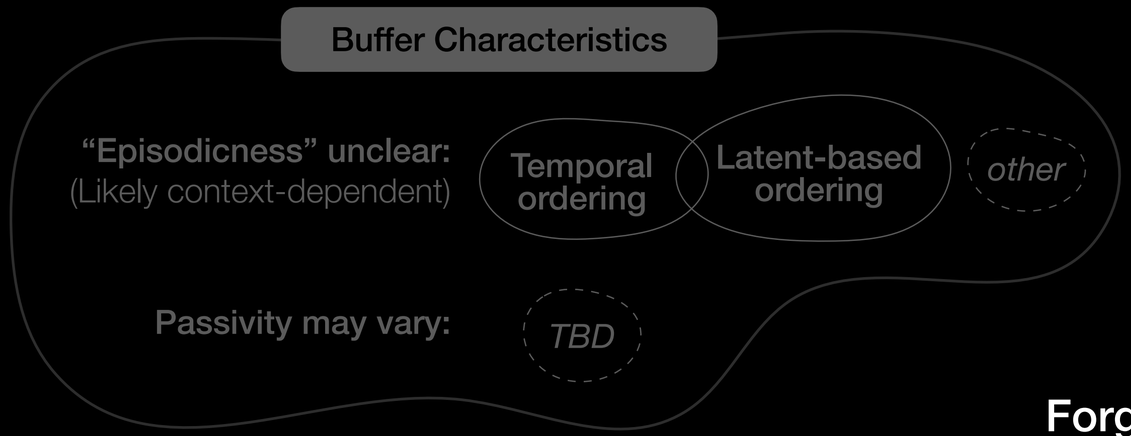
Synthesis II: Mapping Cognitive Concepts to Related Work on LLMs



See Thesis, Subsection 4.2.2.

Synthesis II: Mapping Cognitive Concepts to Related Work on LLMs

Contribution II: A WM Core Architecture for LLMs | Part I: Theoretical Foundation



Forgetting Mechanisms

Interference Based: *Out of scope*

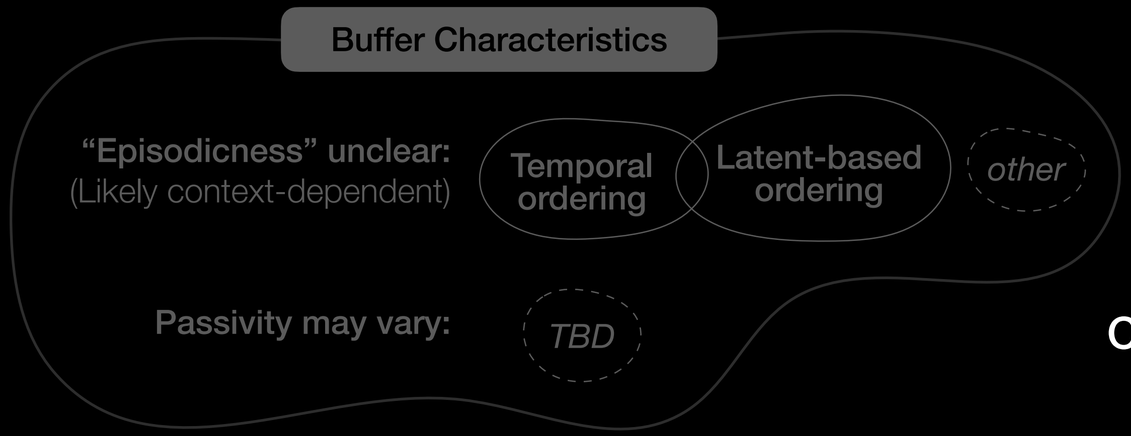
Decay-Based: **Exp. decaying survival scores** *(e.g. see Park et al., 2023)*

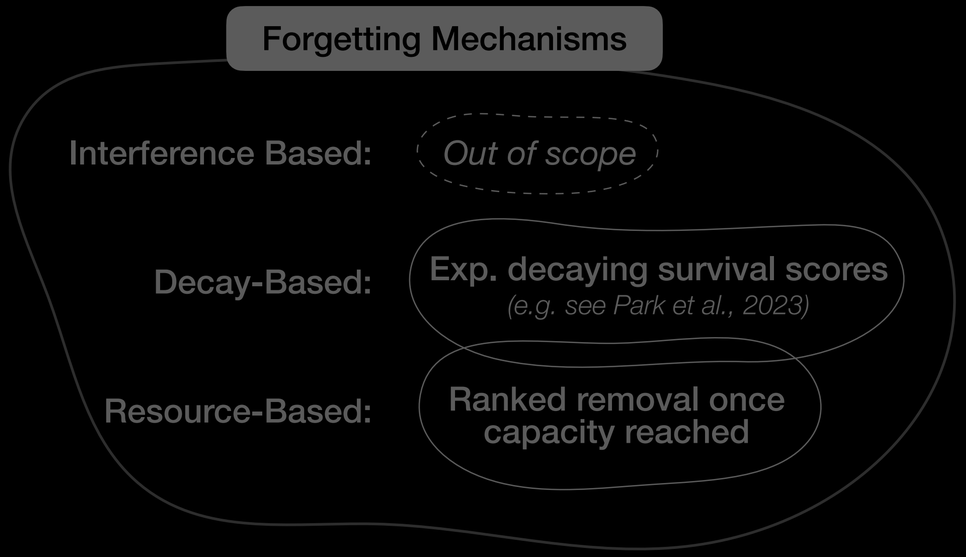
Resource-Based: **Ranked removal once capacity reached**

See Thesis, Subsection 4.2.2.

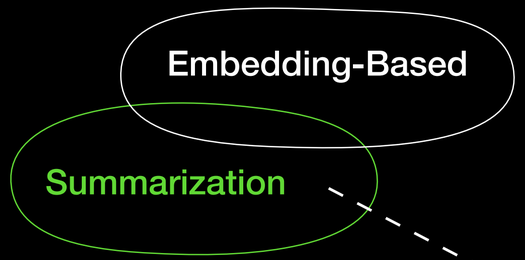
Synthesis II: Mapping Cognitive Concepts to Related Work on LLMs

Contribution II: A WM Core Architecture for LLMs | Part I: Theoretical Foundation





Chunking Mechanisms



As in ***Hi-Agent*** **(Hu et al., 2024):**

- Agent formulates sub-goal g
- Appends new *(action, observation)* pairs to context until g completed
- Summarizes (action, observation) pairs leading up to g in context

See Thesis, Subsection 4.2.2.

Synthesis II: Mapping Cognitive Concepts to Related Work on LLMs

Contribution II: A WM Core Architecture for LLMs | Part I: Theoretical Foundation

Buffer Characteristics

"Episodicness" unclear:
(Likely context-dependent)

- Temporal ordering
- Latent-based ordering
- *other*

Passivity may vary:

- *TBD*

Forgetting Mechanisms

Interference Based:

- *Out of scope*

Decay-Based:

- Exp. decaying survival scores
  (e.g. see Park et al., 2023)

Resource-Based:

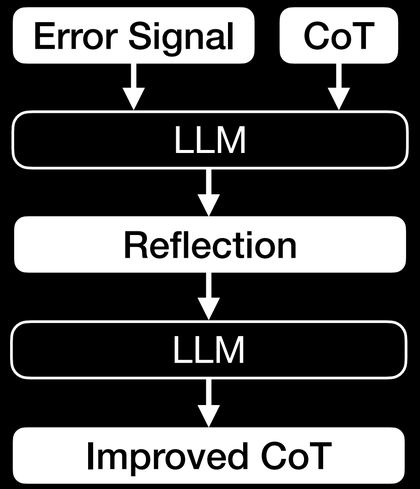
- Ranked removal once capacity reached

Chunking Mechanisms

- Embedding-Based
- Summarization
- Self-Reflection

High level process:

Cf. e.g. Madaan et al. (2023), Shinn et al. (2023), Renze and Guven (2024)



Error signals are generally
- Textual or numerical
- External or internal

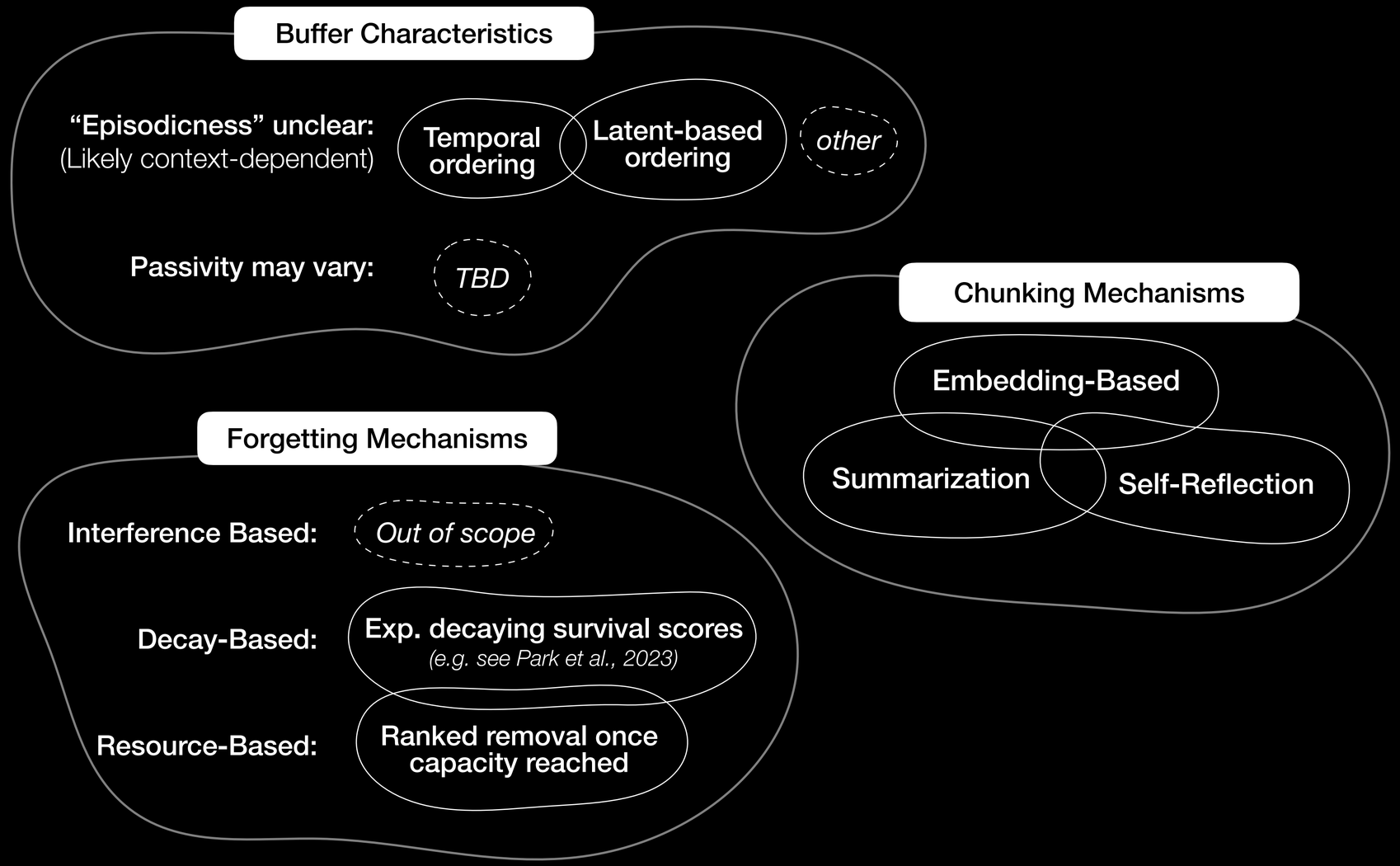
LLMs' ability to identify own errors remains inconclusive
(Renze & Guven, 2024)

↳ Working assumption:
Works reasonably well

See Thesis, Subsection 4.2.2.

Synthesis II: Mapping Cognitive Concepts to Related Work on LLMs

Contribution II: A WM Core Architecture for LLMs | Part I: Theoretical Foundation



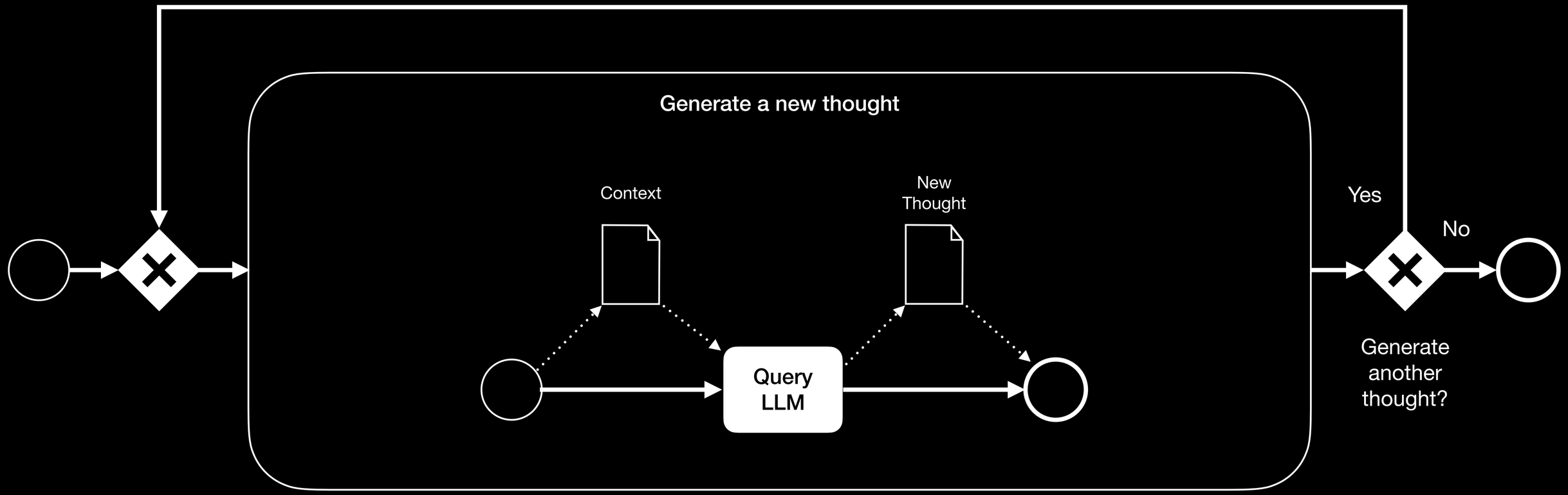
See Thesis, Subsection 4.2.2.

PART 2

Implementation

Abstract Reasoning Process without WM Interaction

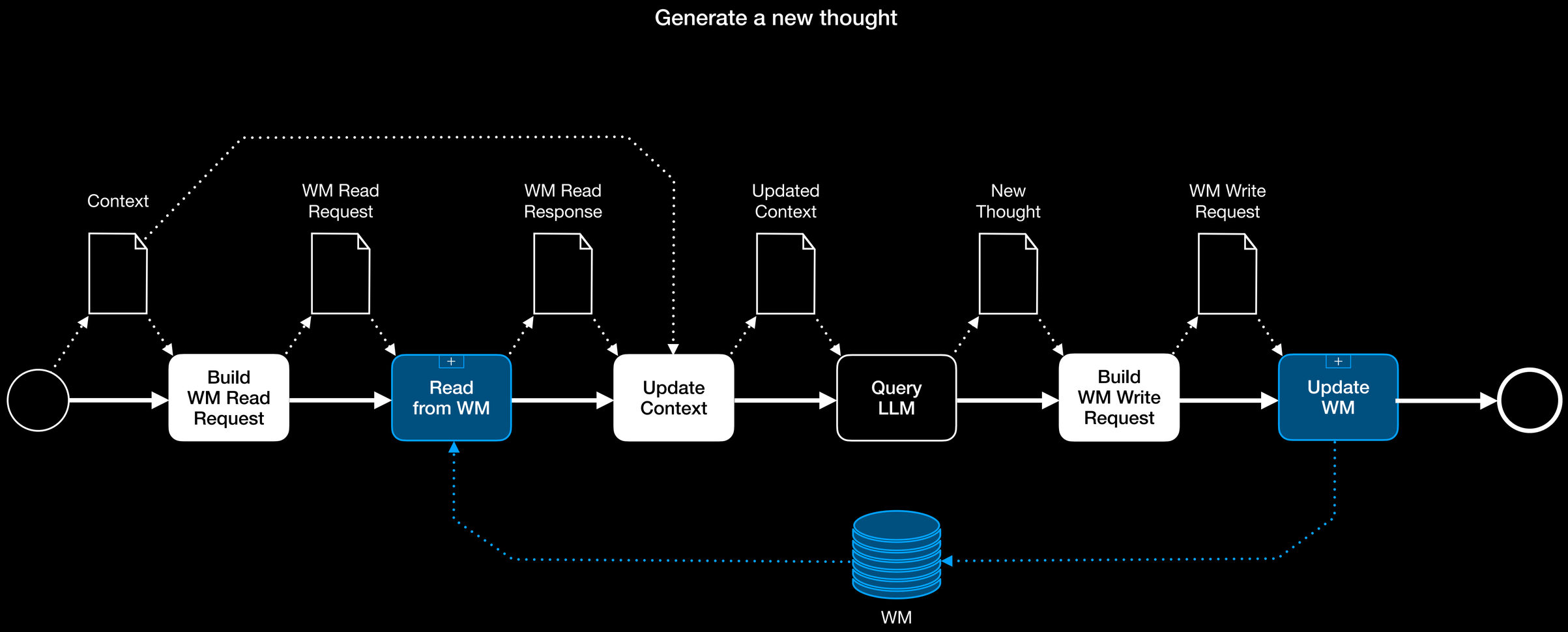
Contribution II: A WM Core Architecture for LLMs | Part II: Implementation



See Thesis, Subsection 4.3.2.

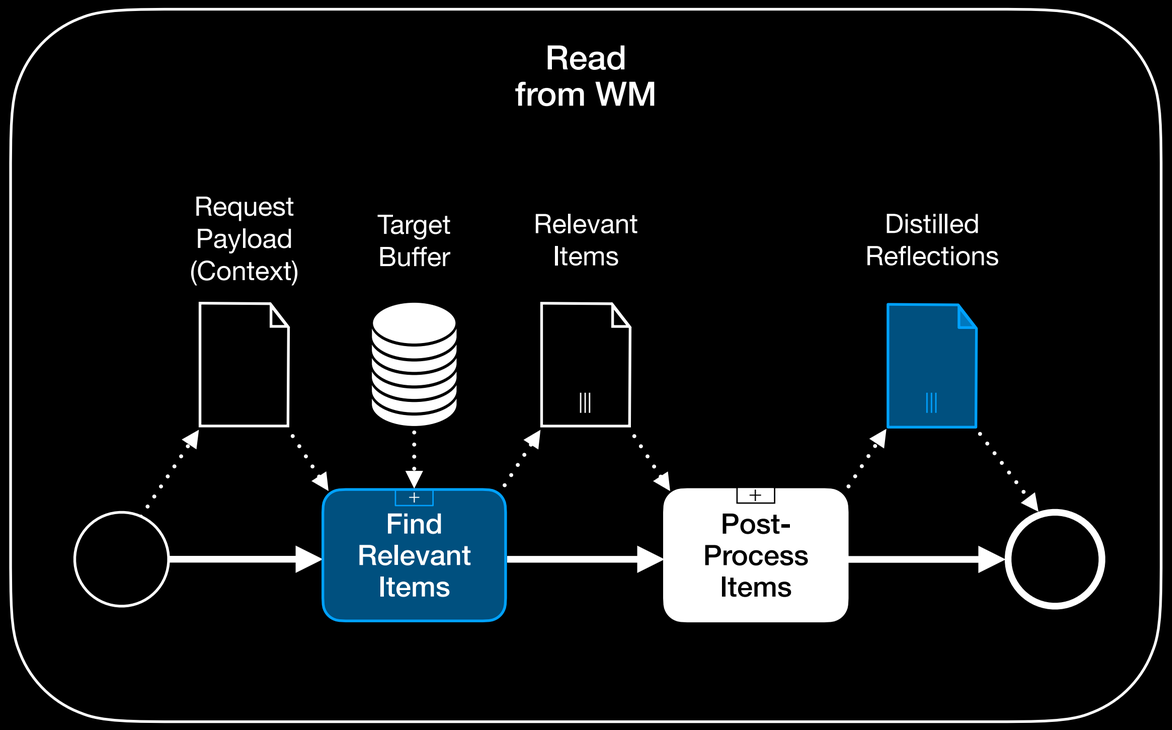
Abstract Reasoning Process with WM Interaction (Simplified)

Contribution II: A WM Core Architecture for LLMs | Part II: Implementation



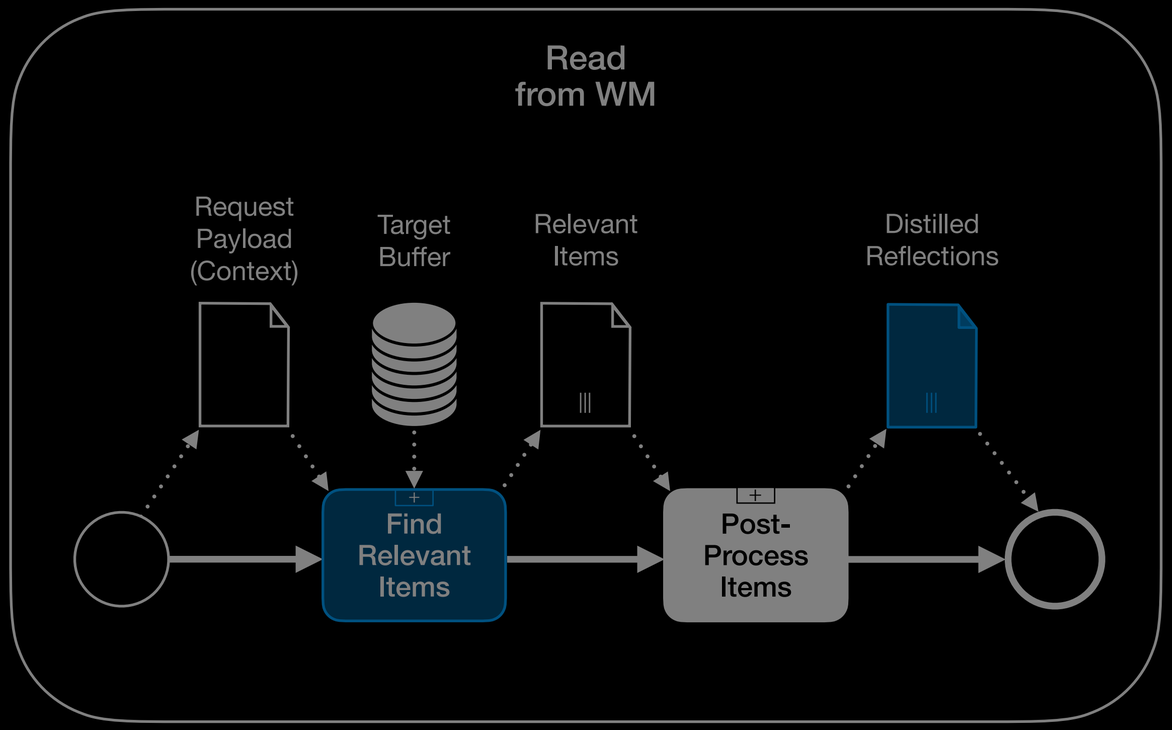
See Thesis, Subsection 4.3.2.

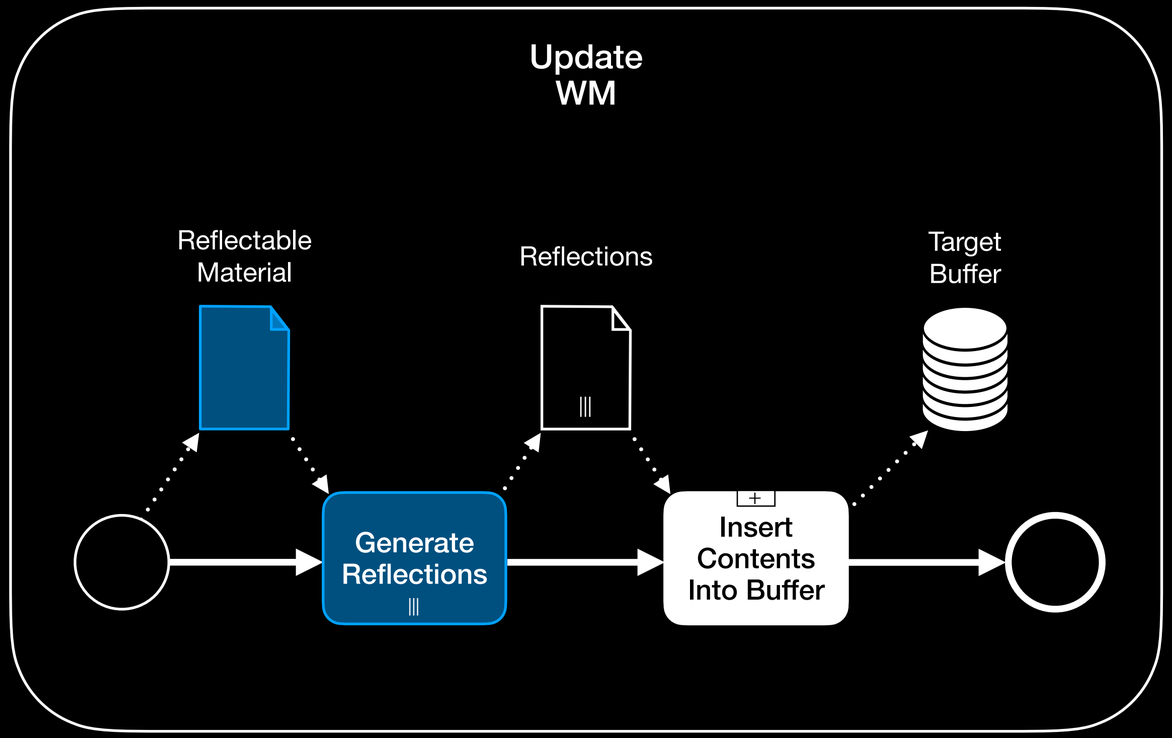
Reflection-Based WM Read and Write Subprocesses (Simplified)



See Thesis, Subsection 4.3.2.

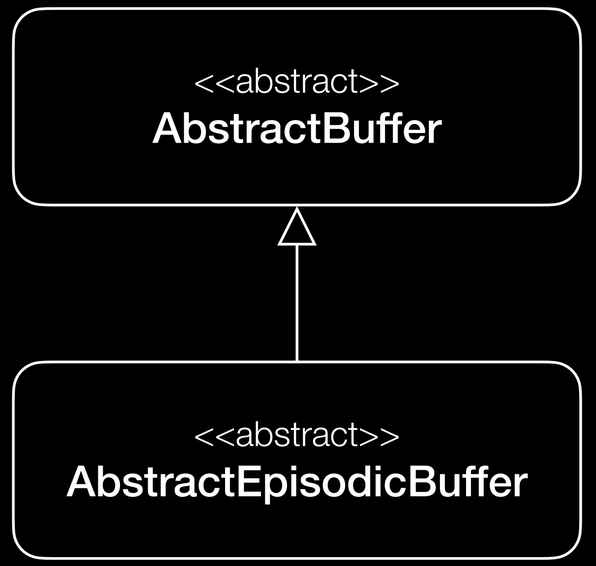
Reflection-Based WM Read and Write Subprocesses (Simplified)





See Thesis, Subsection 4.3.2.

A Two Tiered Buffer Hierarchy



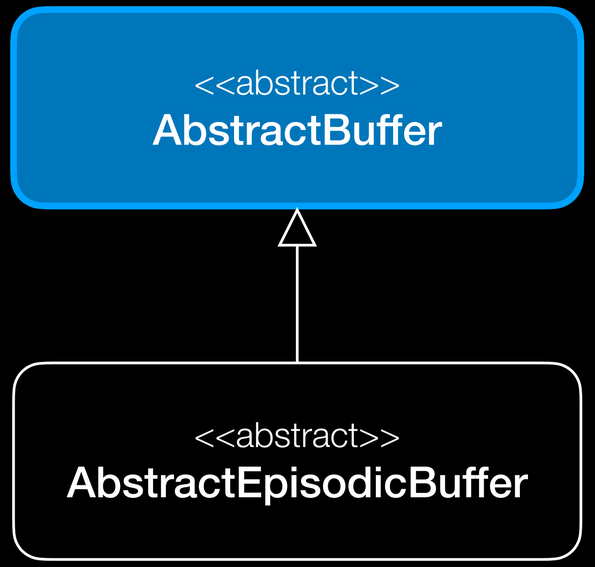
A Two Tiered Buffer Hierarchy: AbstractBuffer

1. **Owns the item space**
   - ↳ Declares expected *item* and *request* type to accept
   - ↳ Decides how items relate to *another* and a *request*
     (via abstract functions)

2. **Enforces a capacity limit** (various def. possible)
   - ↳ Items define their capacity
   - ↳ Buffer provides functions to check current and maximal capacity
   - ↳ Buffer rejects CRUD operations if capacity would be exceeded

3. **Strictly passive** (per LSP*): **Only CRUD operations**
   - ↳ Delegates forgetting operations to environment (transactions)

4. Requires a notion of local time



*LSP: Liskov Substitution Principle

See Thesis, Subsection 4.3.3.

A Two Tiered Buffer Hierarchy: AbstractBuffer

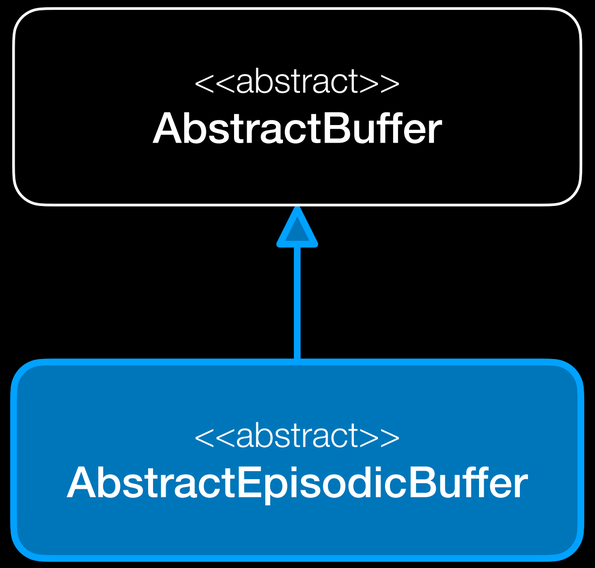
1. Operates on “Chunked” Items

↳ Stores a forest of items where

- ***Abstract* nodes** serve as cluster labels:
  ↳ Assumes information loss w.r.t. children
- ***Concrete* nodes** bear payload (e.g. a reflection);
  may also be an aggregate of other nodes payload’s
  ↳ Assumes no information loss w.r.t. children (are representative)

↳ Def. *Chunking*: Building an item tree via

- Merging of concrete nodes
- Assigning abstract parents

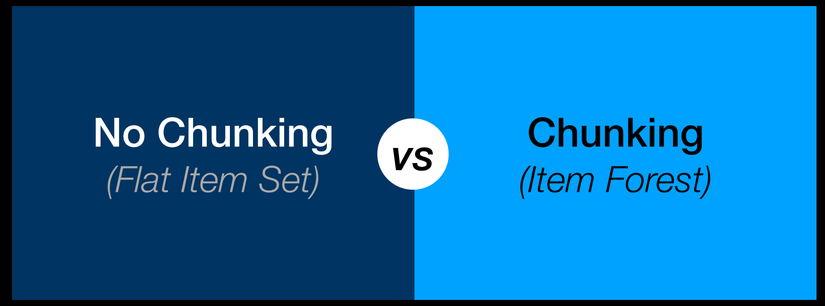


2. Defines an implicit notion of episodic coherence via

- Abstract grouping functions
- Abstract matching functions, for *part-whole* and *equivalence* between nodes
  (Extensions of AbstractBuffers matching function)

See Thesis, Subsection 4.3.4.

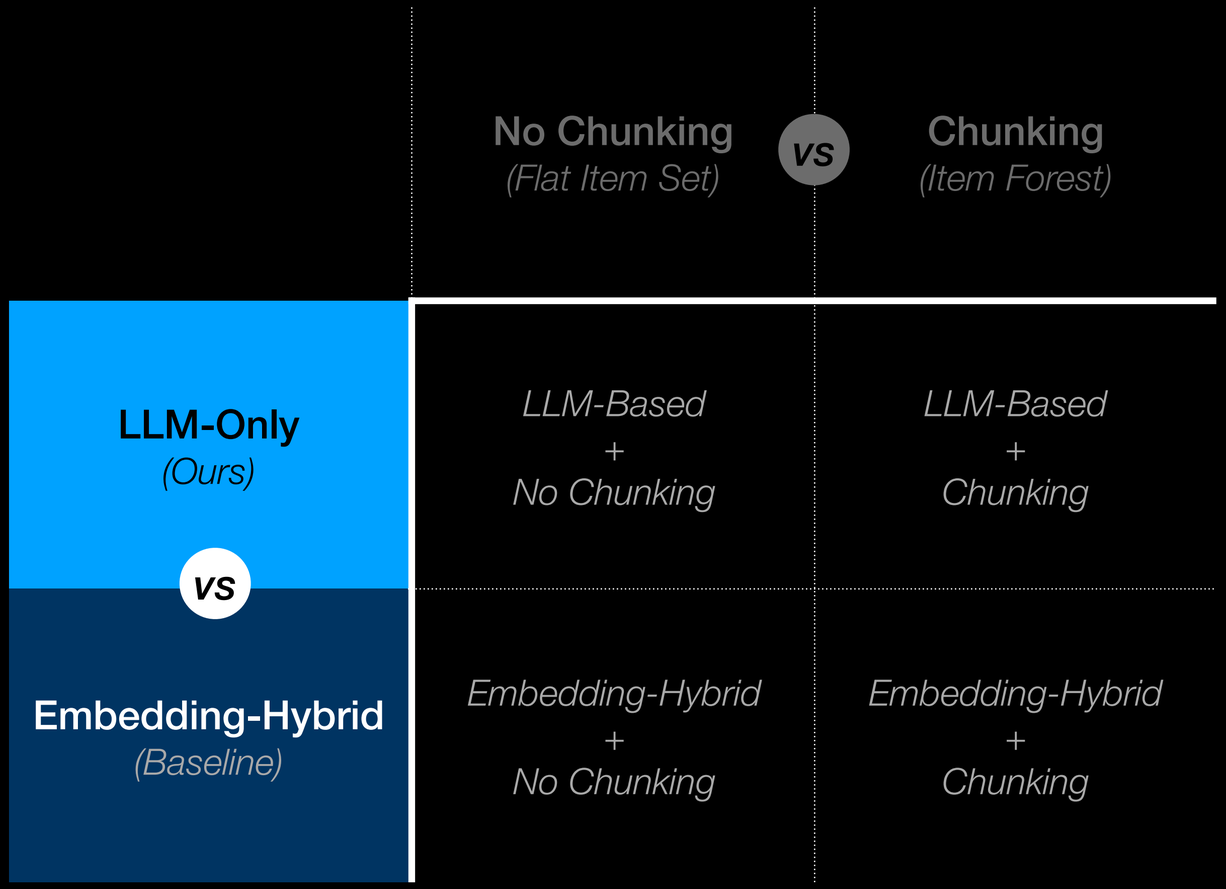
Implement WM Variants: Design Matrix



See Thesis, Subsection 4.4.1.

Implement WM Variants: Design Matrix

Contribution III: Four Concrete WM Variants | Part II: Implementation



See Thesis, Subsection 4.4.1.

Implement WM Variants: The Underlying Hypotheses

Contribution III: Four Concrete WM Variants | Part II: Implementation

Hypothesis I

GoT + WM > GoT

Hypothesis II

Chunking > No Chunking

Hypothesis III

LLM-Only > Embedding-Hybrid

(Given limited capacity)

See Thesis, Subsection 4.4.1.

Implement WM Variants: The Embedding-Hybrid Backbone

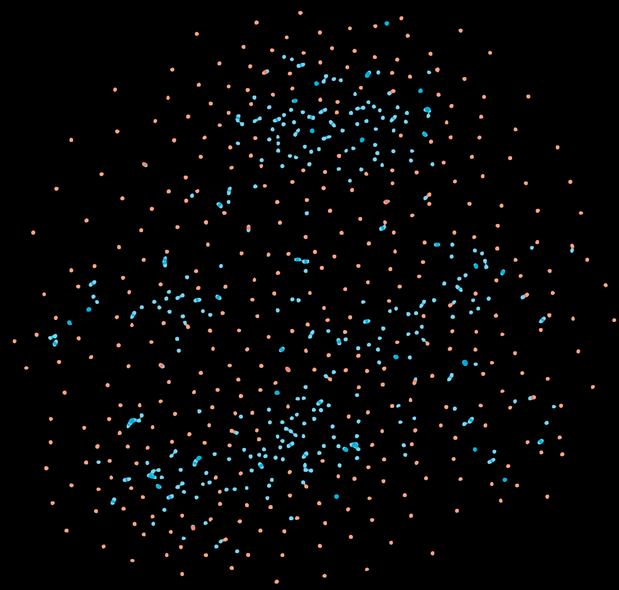
Contribution III: Four Concrete WM Variants | Part II: Implementation

	No Chunking *(Flat Item Set)*	vs	Chunking *(Item Forest)*
LLM-Only *(Ours)*	*LLM-Based + No Chunking*		*LLM-Based + Chunking*
vs			
Embedding-Hybrid *(Baseline)*	*Embedding-Hybrid + No Chunking*		*Embedding-Hybrid + Chunking*

Based on ***A-MEM (Xu et al., 2025):***

- S.O.T.A. memory for agentic setups
- Reverses the idea of contrastive learning:
  - ↳ Neighborhoods are verified by LLM
  - ↳ Verified neighbors receive similar tags
  - ↳ Similar items move closer in space

T-SNE Visualization of Memory Embeddings (adapted from Xu et al., 2025):
A-MEM (blue) vs Base Memory (red)

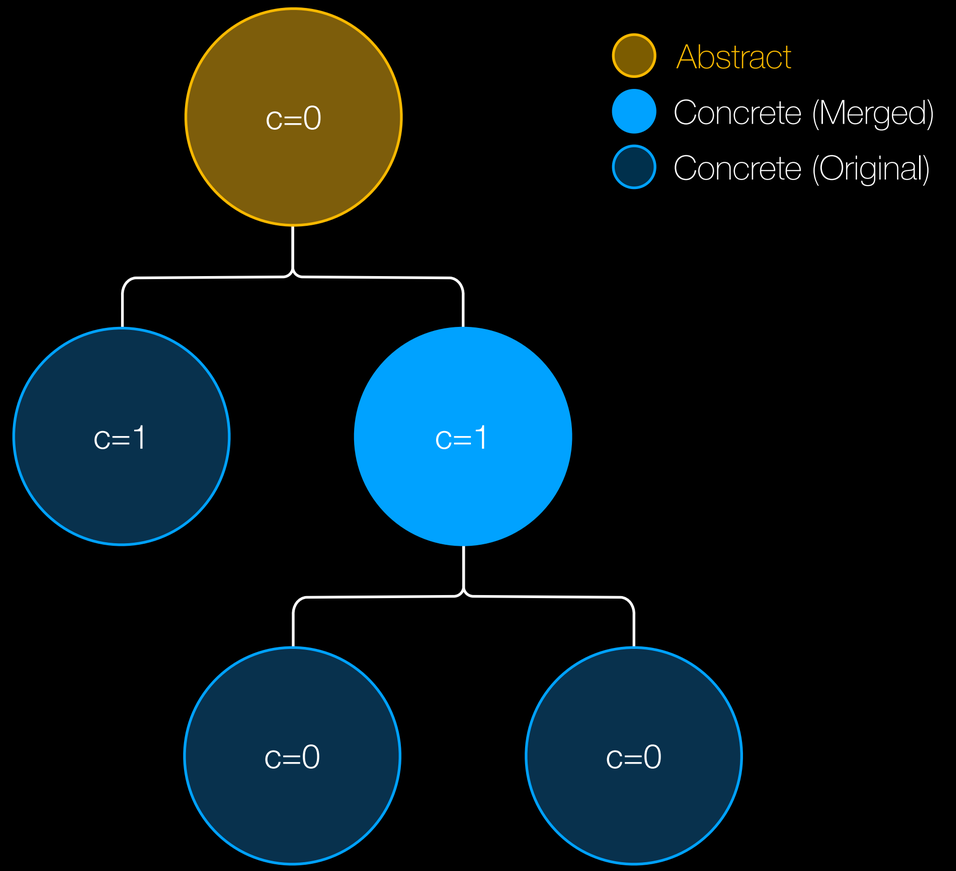


See Thesis, Subsection 4.4.1.

Implement WM Variants: Further Constraints

1. Shallow, Text-Only Chunks

- At most one abstract node per tree
- Abstract nodes hold a textual cluster label
- Concrete nodes hold reflections as payload which are merged when deemed equivalent (via LLM)

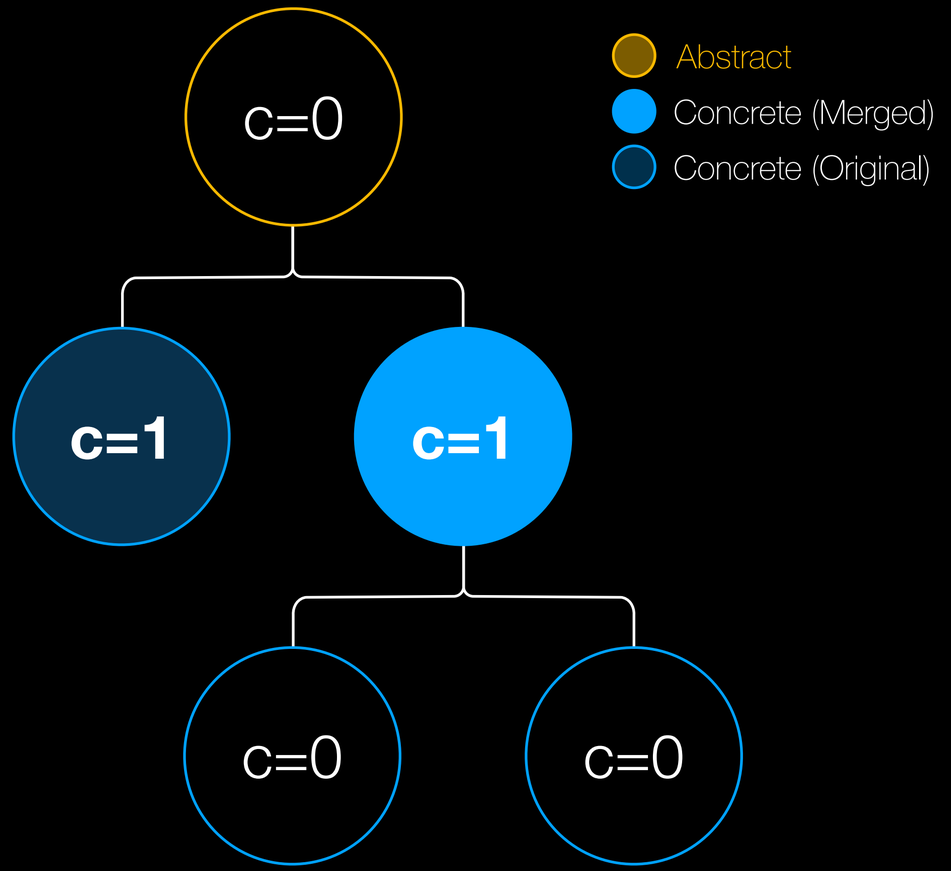


See Thesis, Subsection 4.4.

Implement WM Variants: Further Constraints

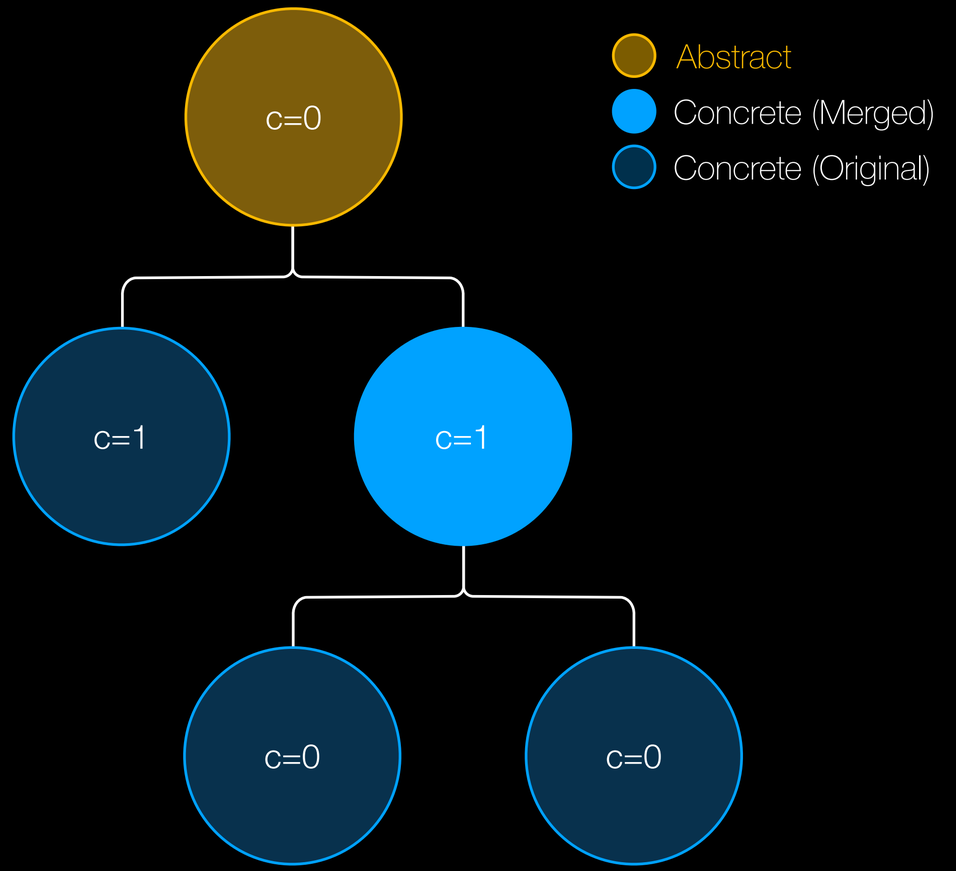
2. Forgetting: Resource + Decay

- Resource-based capacity (c) threshold where only items in the concrete front contribute
- Exponential decay of survival scores based on item access & repetition + LLM-based usability scoring



See Thesis, Subsection 4.4.

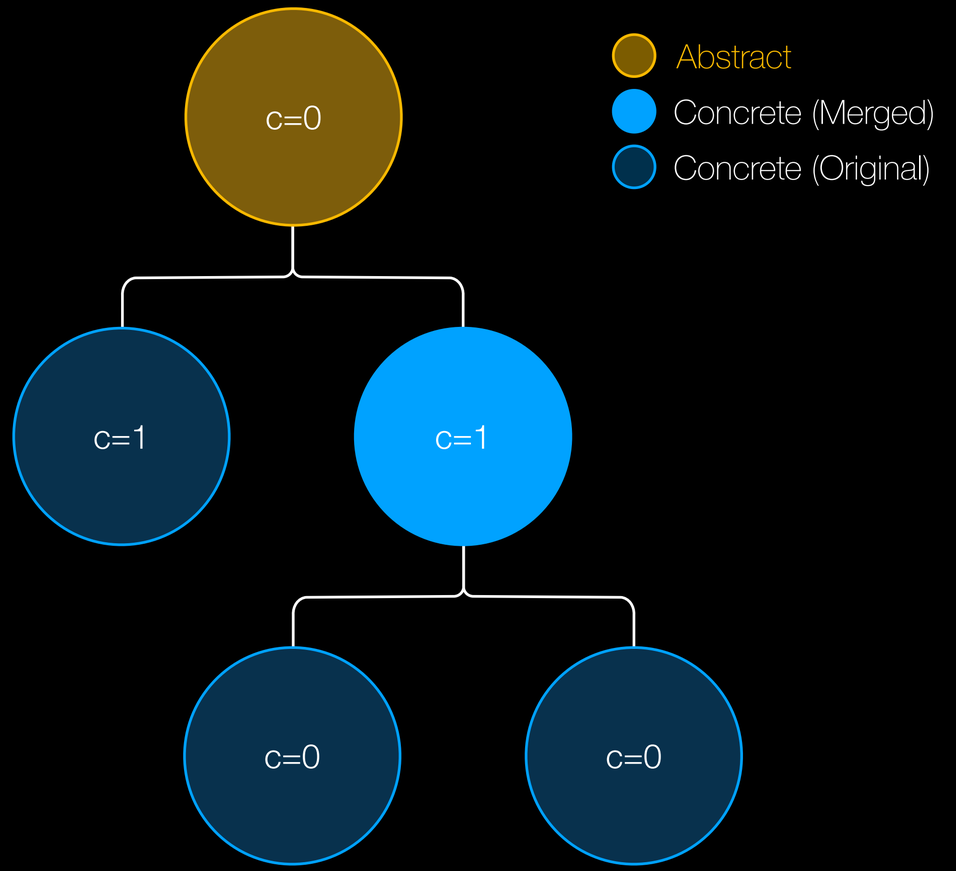
Implement WM Variants: Further Constraints



3. **No reorganization of forest**
   - ↳ Insertion > New trees (→ Prone to over-aggregation)

See Thesis, Subsection 4.4.

Implement WM Variants: Further Constraints



4. **Reflection generation limited to a single thought**

See Thesis, Subsection 4.4.

PART 3

Main Findings

Experimental Setup (Simplified)

- **Tasks: Sorting & Set-Intersection** — adapted from GoT *(Besta et. al., 2024)*
  - On a list/set of integers (size=32)
  - Eval metric: Accuracy

- **5 competing setups:**
  - ↳ No WM (Vanilla GoT); GoT+ each of our four WM variants (resp.)
    - ↳ Each assessed across I/O, CoT, ToT, GoT prompting
      (cf. Brown et al. 2020; Wei et al., 2023; Yao et al., 2023; Besta et. al., 2024, resp.)
      - ↳ For each: 20 samples x redraw of 3 → 60 datapoints in total

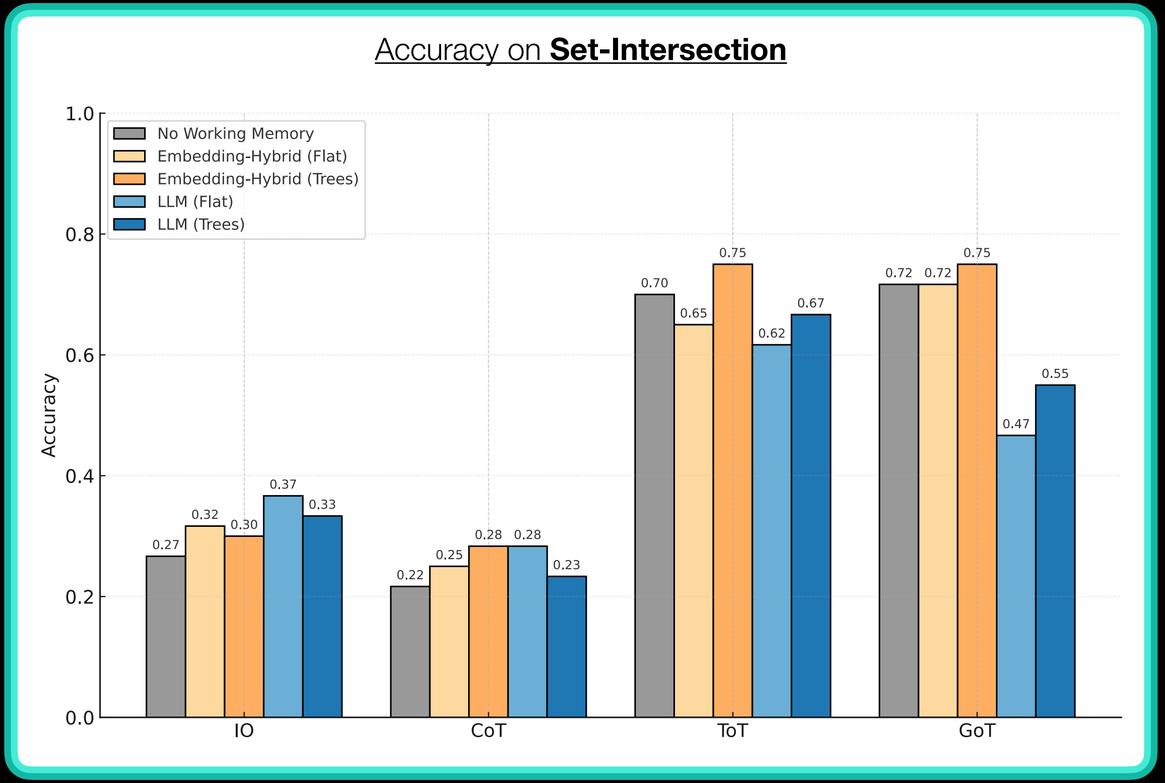
- **WM capacity = 7**
  (mean number of retrieved items in a trial run with capacity = ∞)

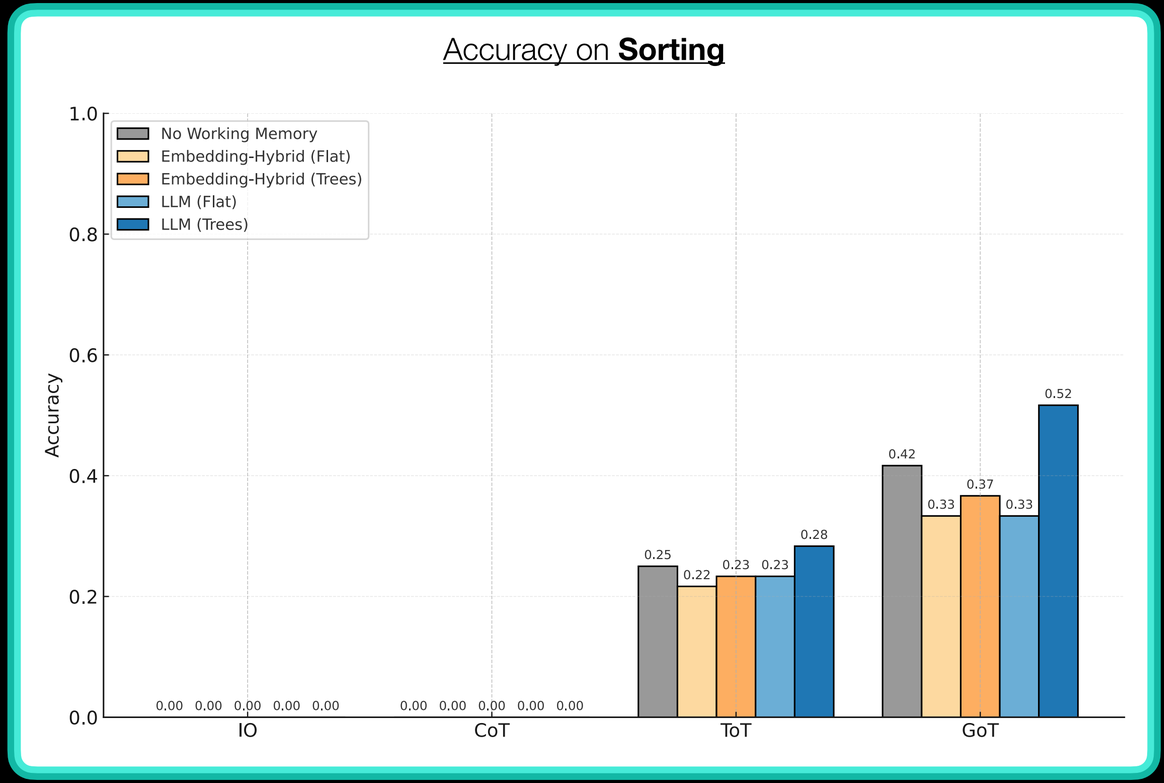
- **Model:** gpt-4o-mini-2024-07-18
  (Documented at https://platform.openai.com/docs/models/gpt-4o-mini)

See Thesis, Subsection 5.3.1.

Assessment of Hypothesis I: GoT + WM > GoT?

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings

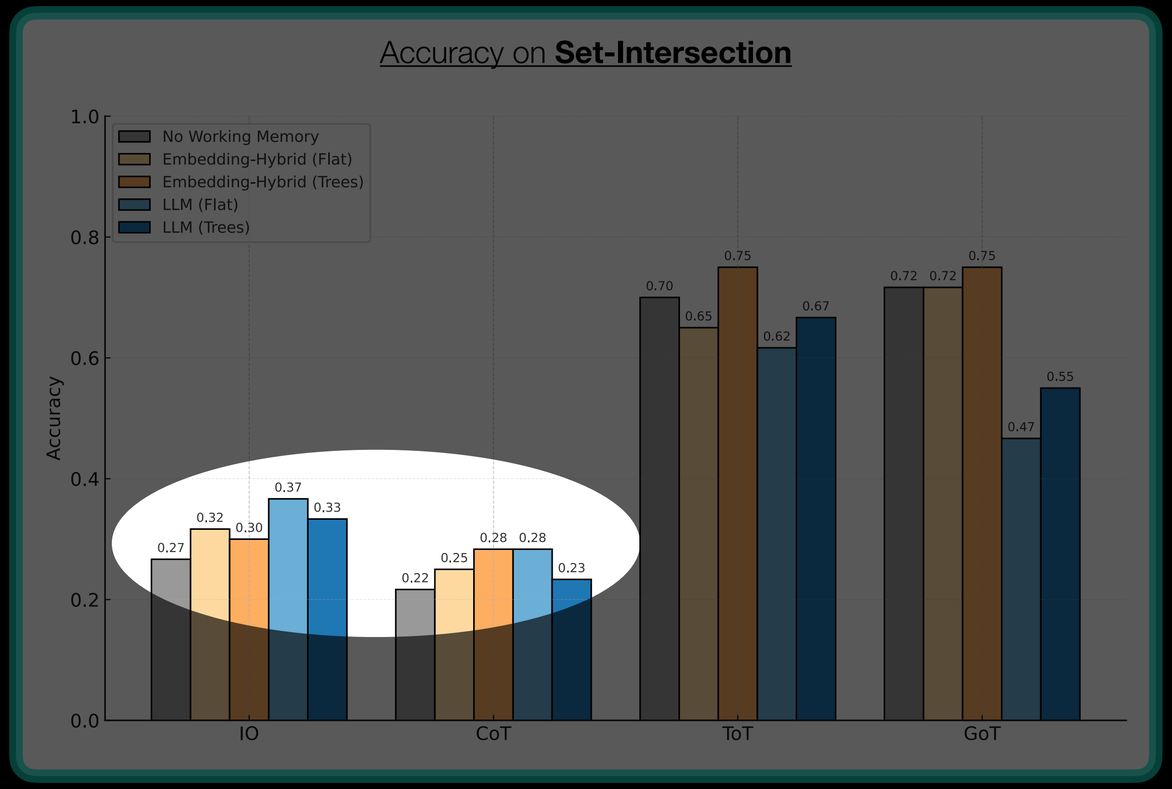


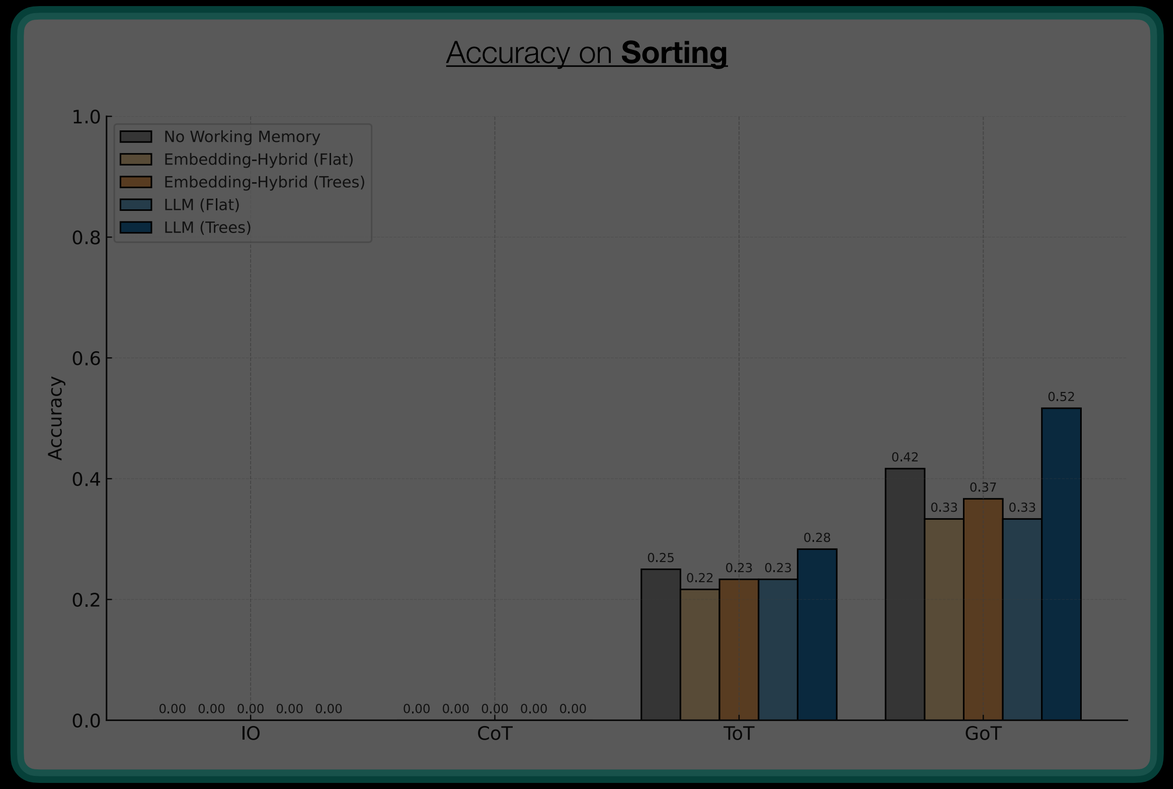


See Thesis, Subsection 5.3.2.

Assessment of Hypothesis I: GoT + WM > GoT?

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings



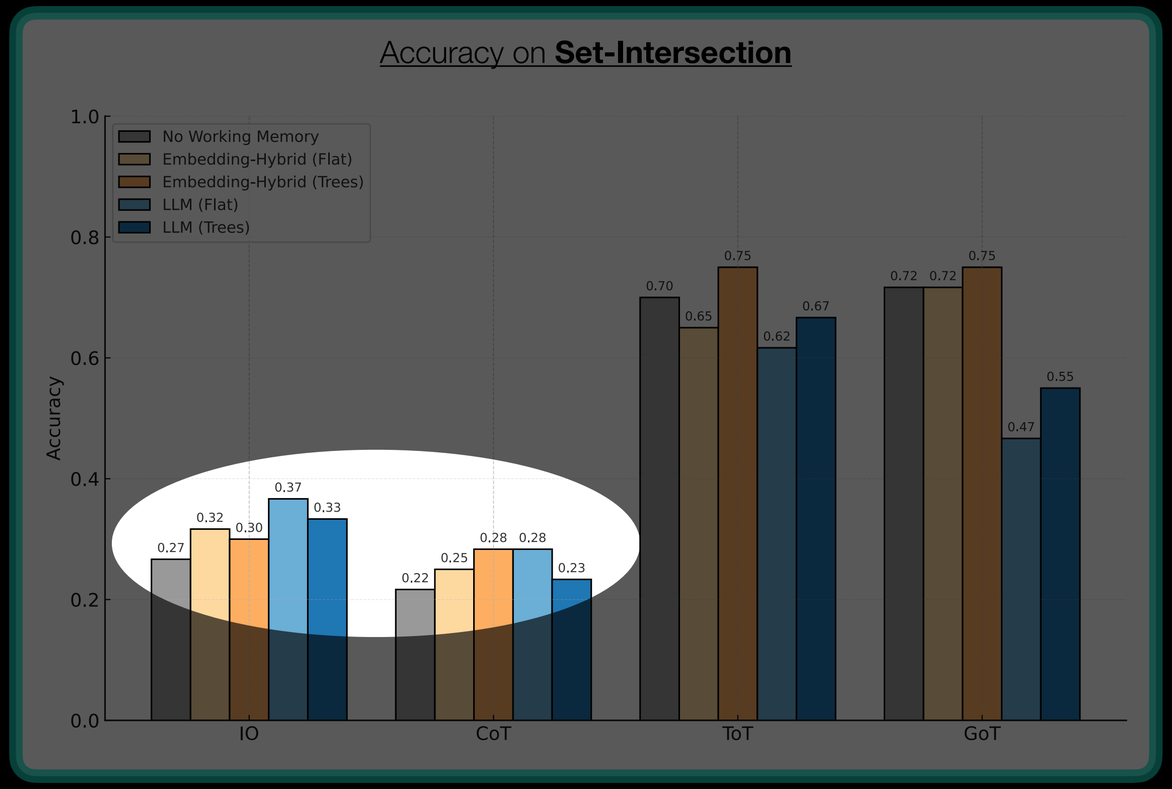


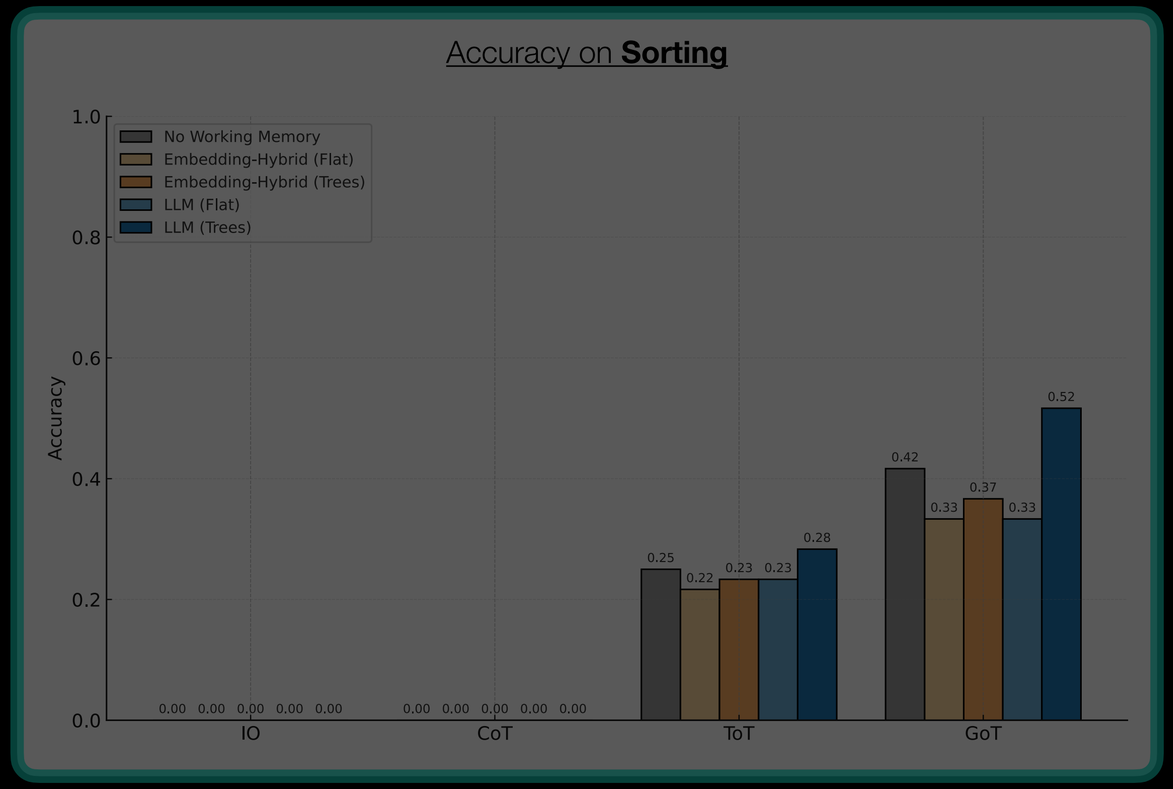
↳ Impossible:
I/O and CoT cannot benefit from WM!

See Thesis, Subsection 5.3.2.

Assessment of Hypothesis I: GoT + WM > GoT?

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings



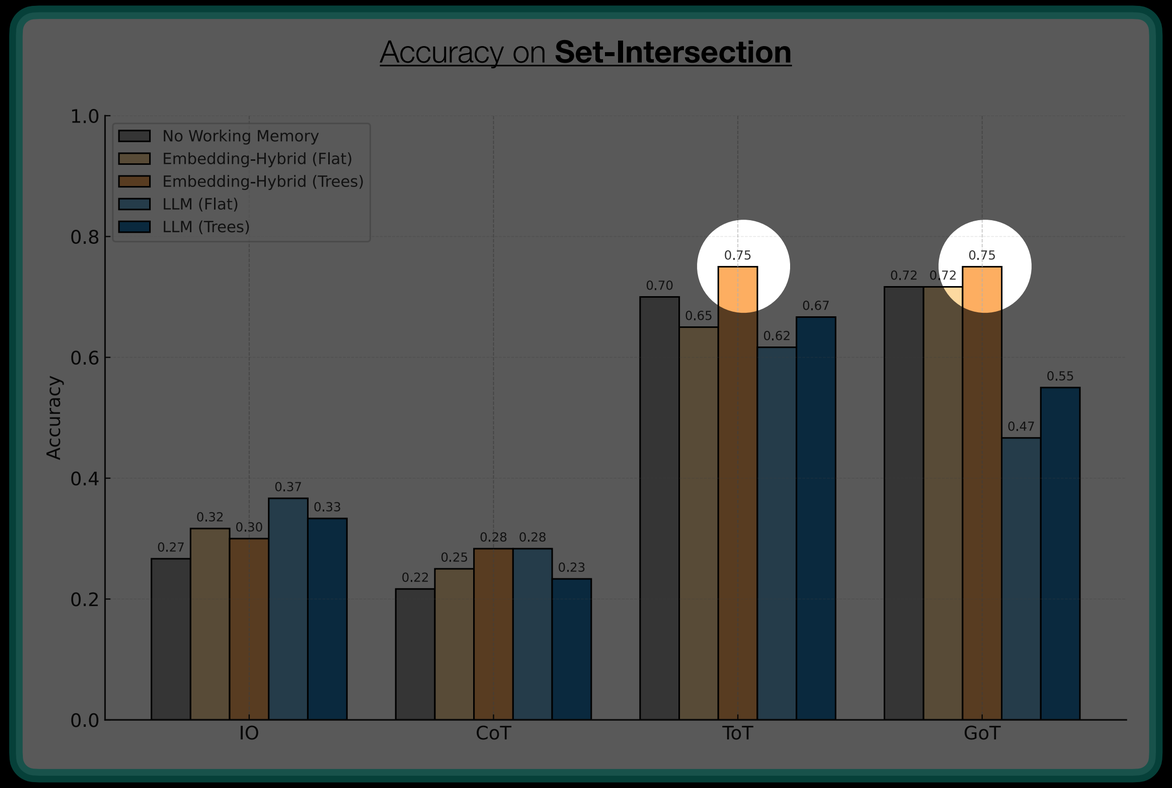


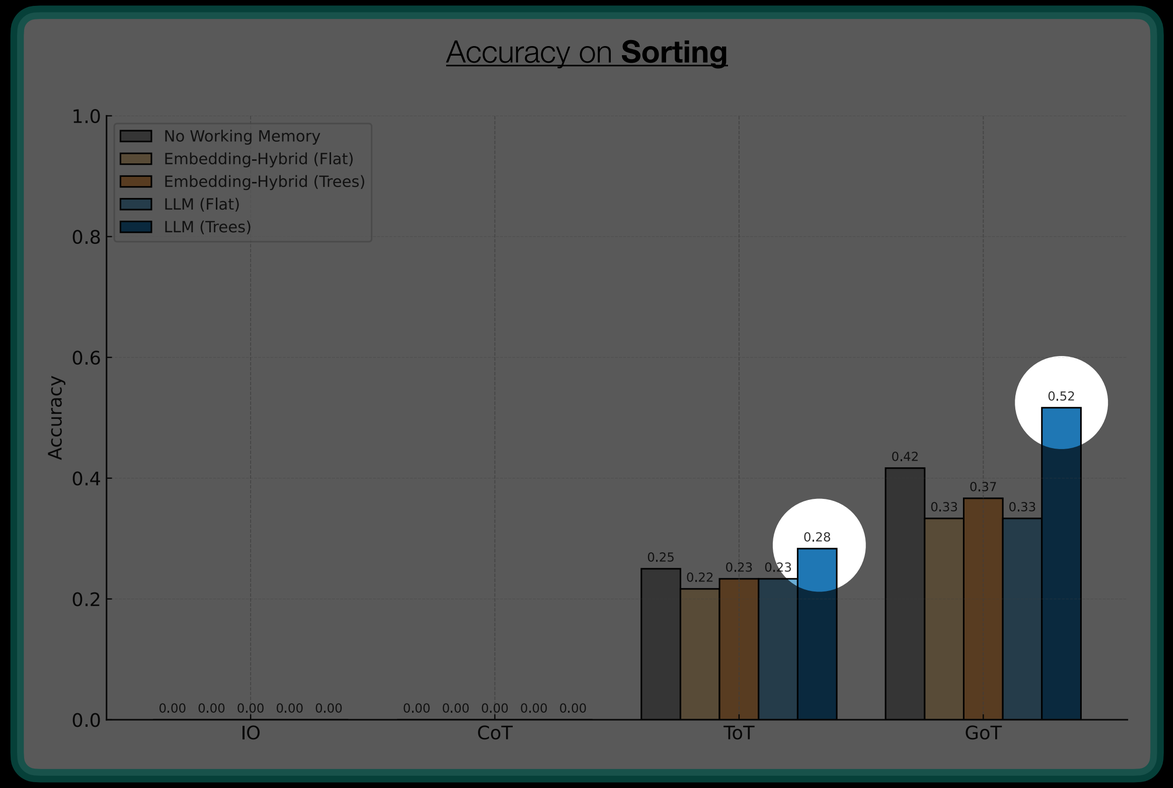
1. I/O and CoT reveal significant instability

See Thesis, Subsection 5.3.2.

Assessment of Hypothesis I: GoT + WM > GoT?

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings



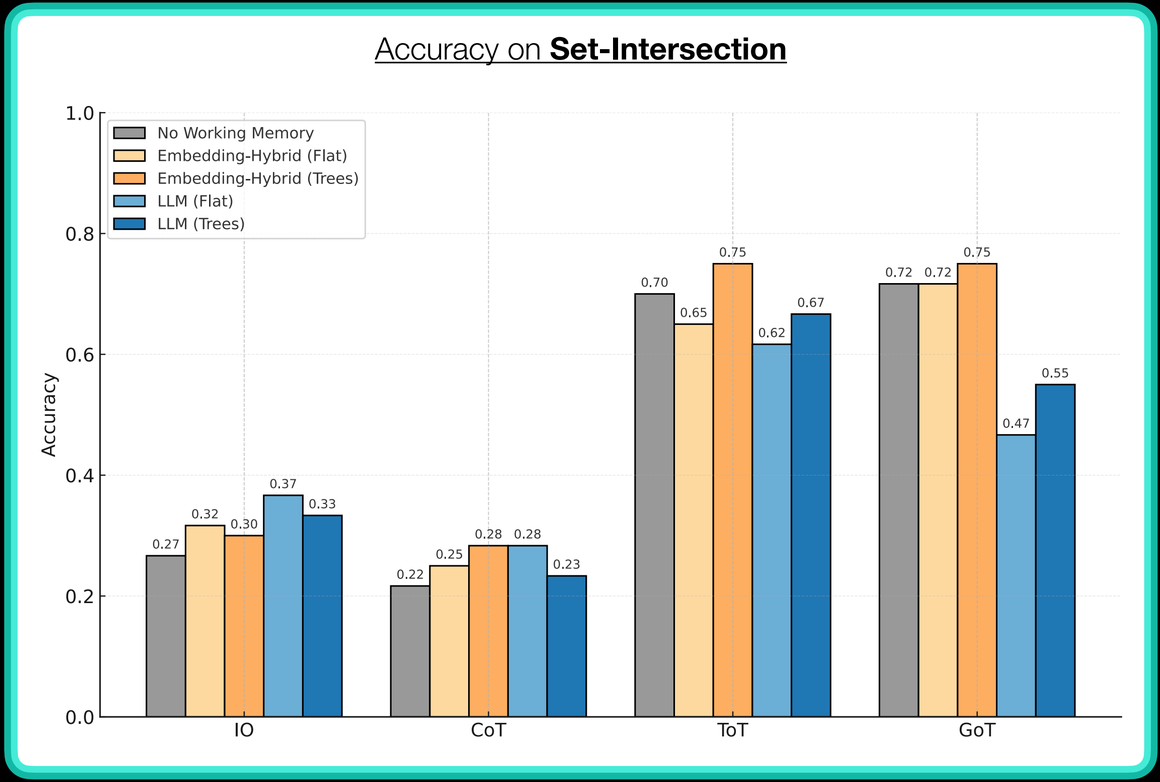


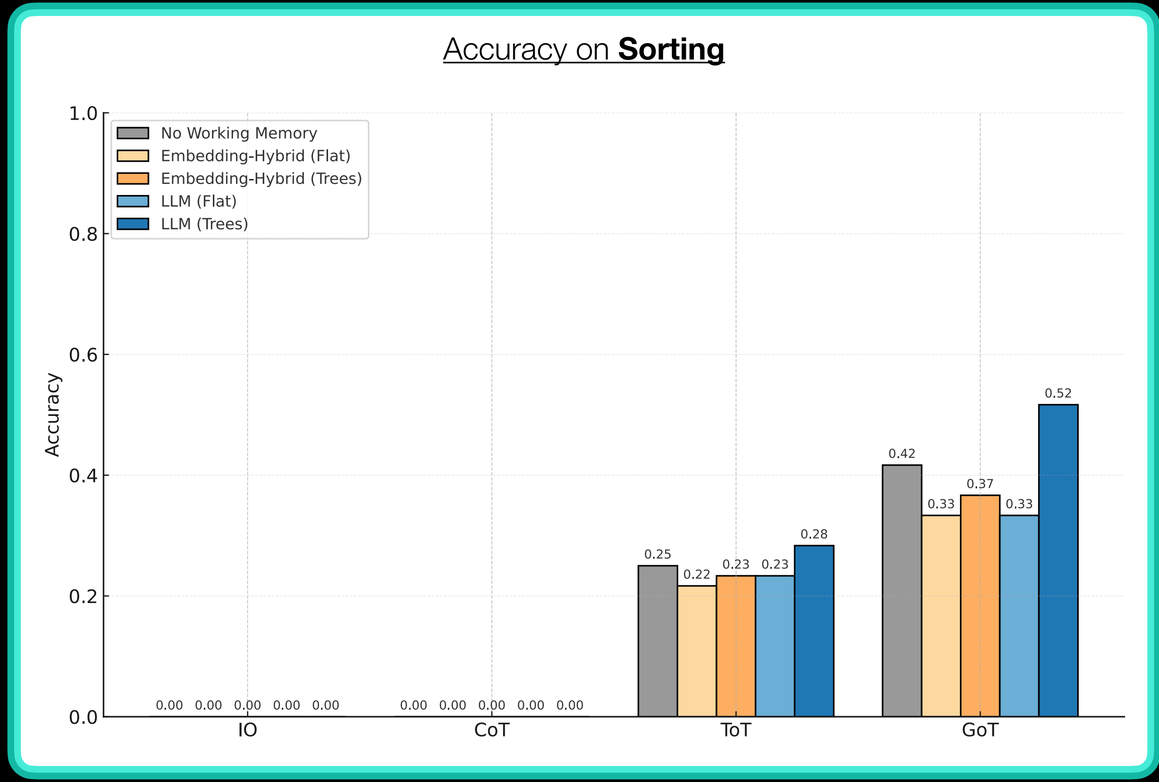
2. Intuition: Chunking variants might be superior

See Thesis, Subsection 5.3.2.

Assessment of Hypothesis I: GoT + WM > GoT?

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings



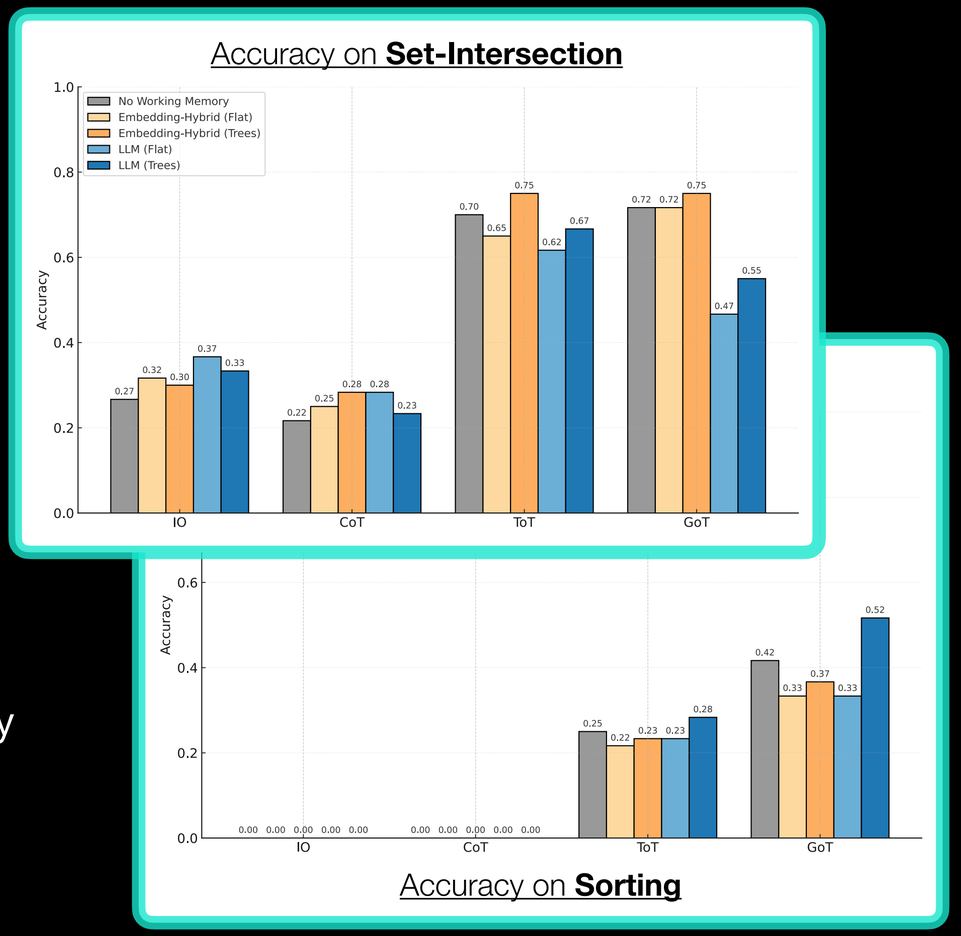


3. Overall: No clear winner

See Thesis, Subsection 5.3.2.

Assessment of Hypothesis I: GoT + WM > GoT?

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings



↳ **Verdict: Instability renders results inconclusive**

Need more compute, deeper variance analysis, or more stable tasks

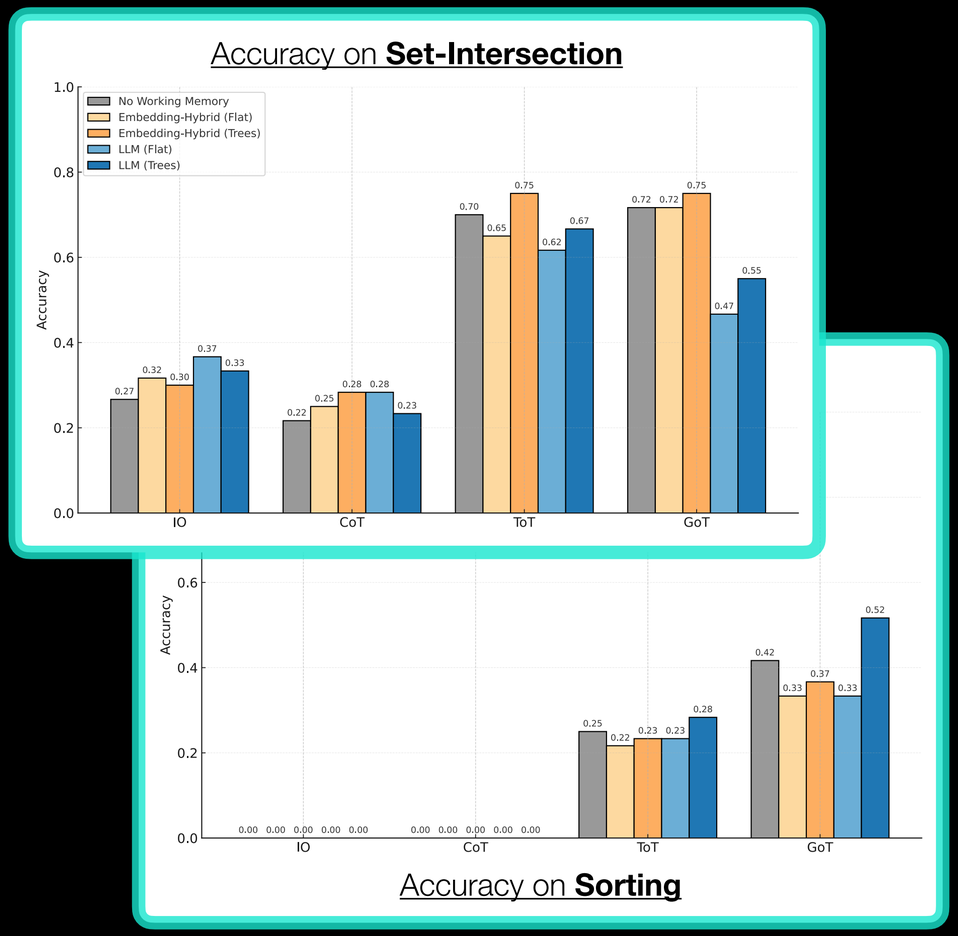
This is our main limitation!

- Task lack nat. language understanding (NLU), semantic diversity
- Lowering temperature —> Higher costs —> Not a fix!

See Thesis, Subsection 5.3.2. and 6.3.

Assessment of Hypothesis II: Chunking > No Chunking?

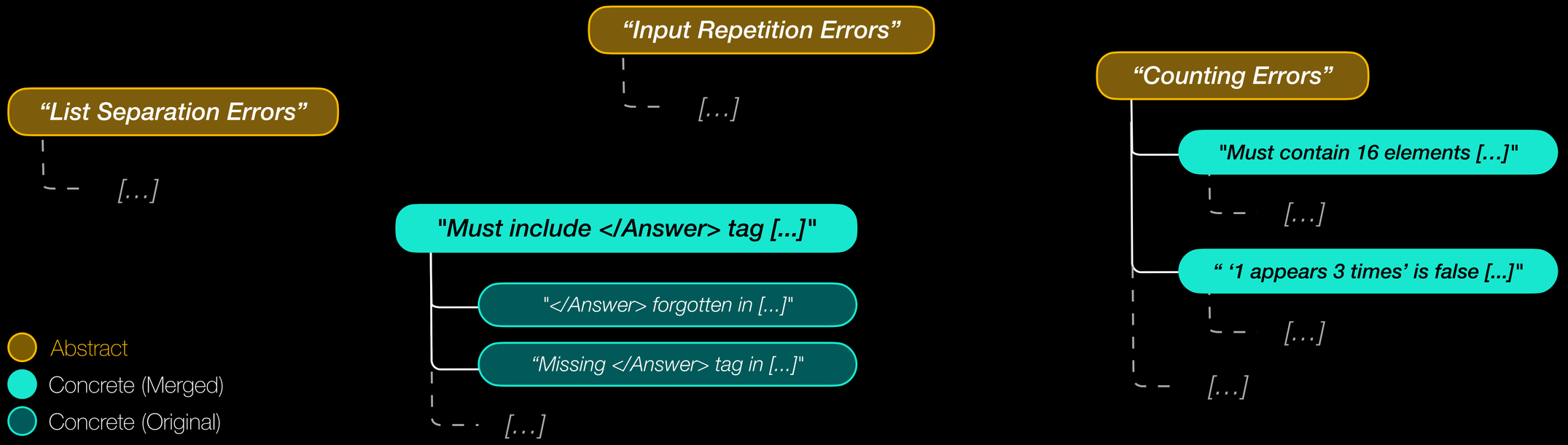
- **Set aside due to instability**



See Thesis, Subsection 5.3.2.

Assessment of Hypothesis II: Chunking > No Chunking?

- Tentative support: **Distinct error sources e**ven in mundane tasks
  ↳ Motivates organized memory **beyond an unordered set** (but not necessarily a forest)

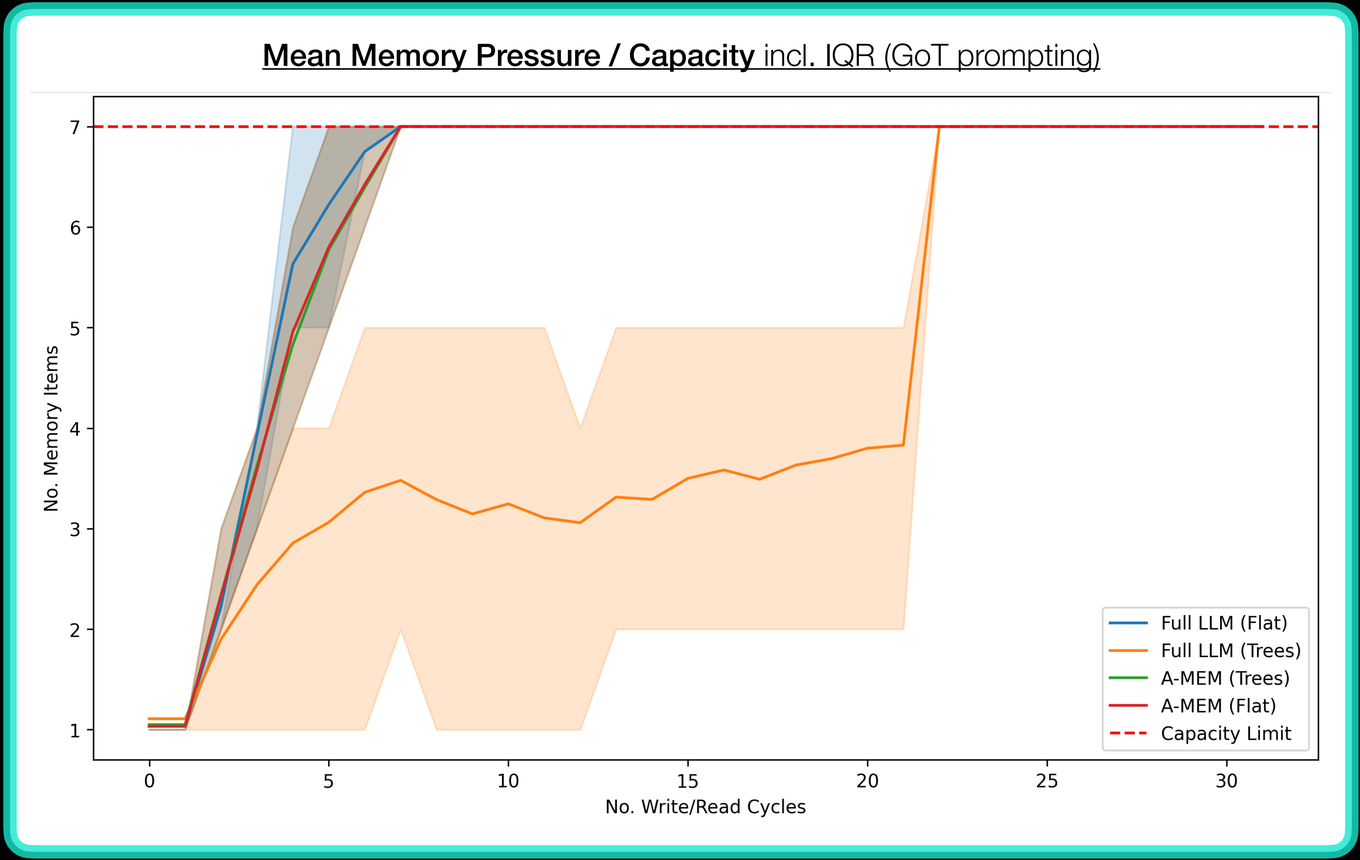


See Thesis, Subsection 5.3.2. and 5.3.3.

Assessment of Hypothesis III: LLM-Only > Embedding-Hybrid

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings

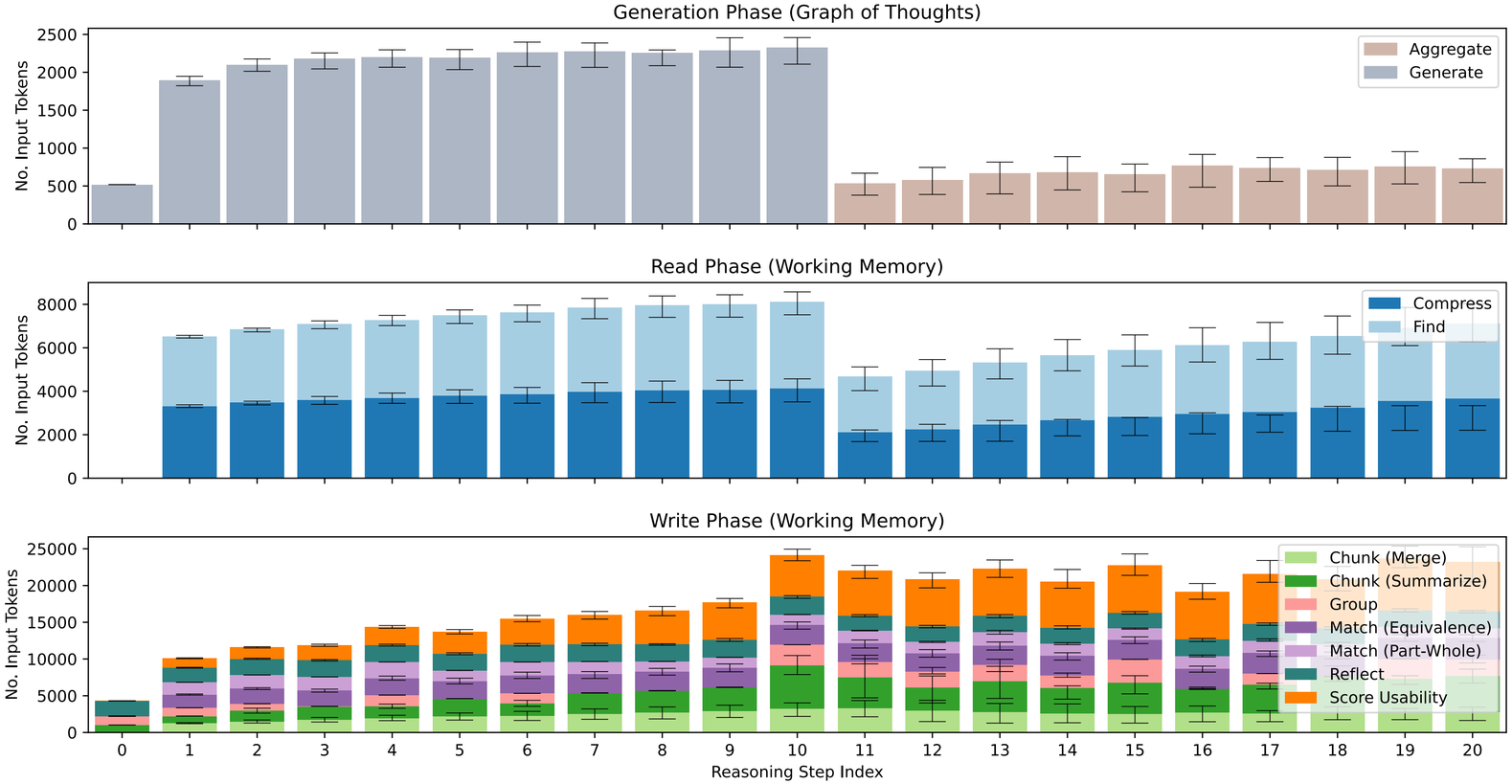
1. LLM-Only appears **more flexible** in…
   - …chunking (building item trees)
   - …discerning item-**relationships**
     (from qual. eval; not depicted here)



See Thesis, Subsection 5.3.2.

Assessment of Hypothesis III: LLM-Only > Embedding-Hybrid

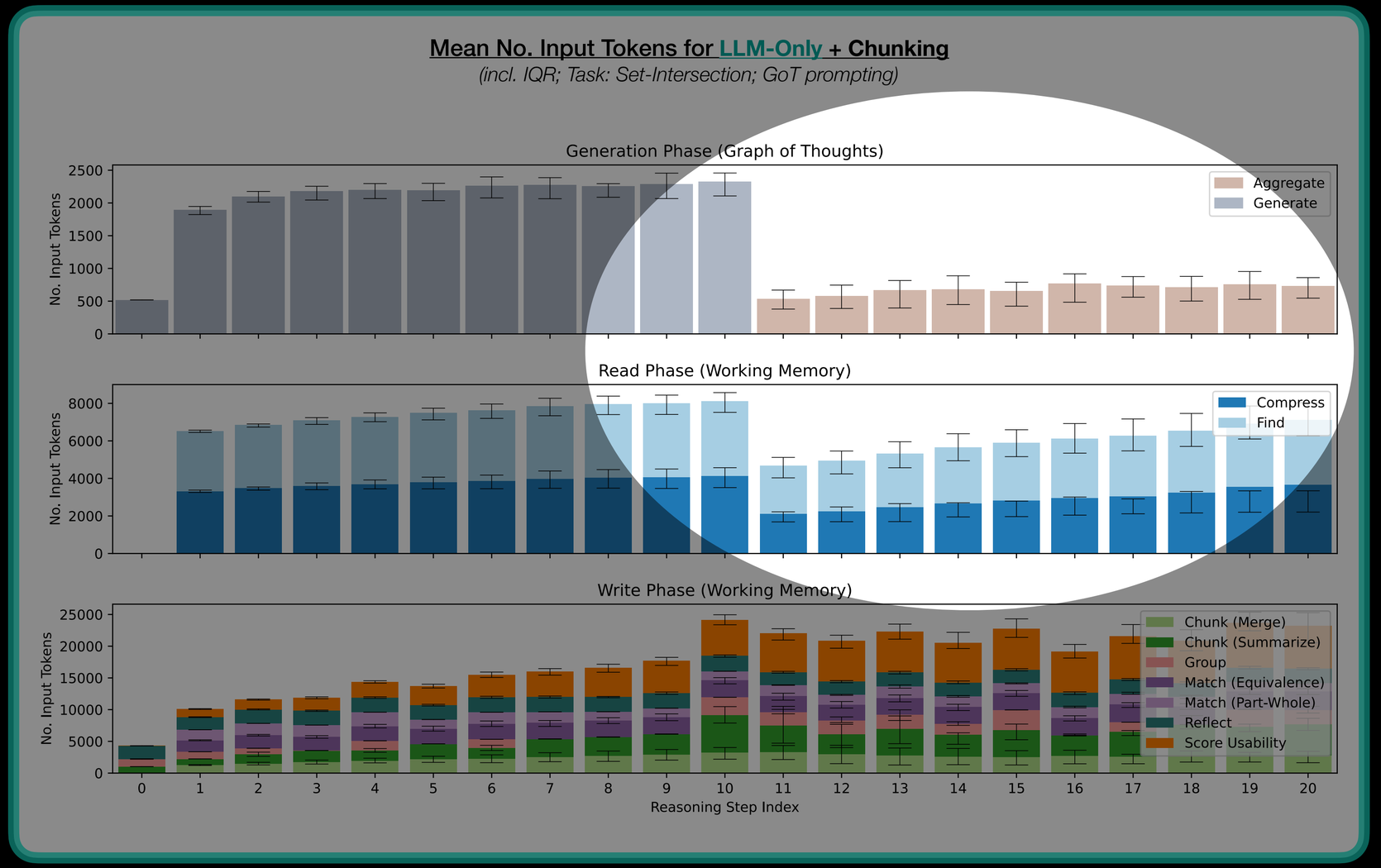
Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings



See Thesis, Subsection 5.3.4.

Assessment of Hypothesis III: LLM-Only > Embedding-Hybrid

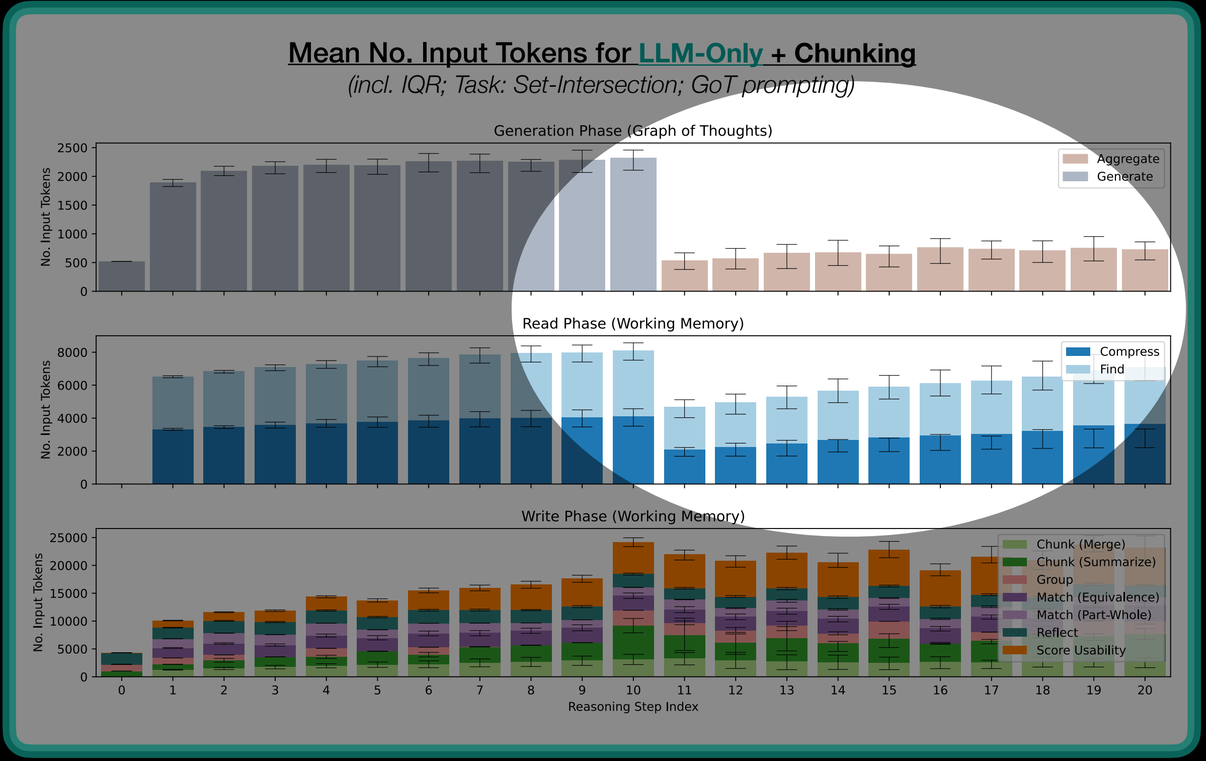
Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings

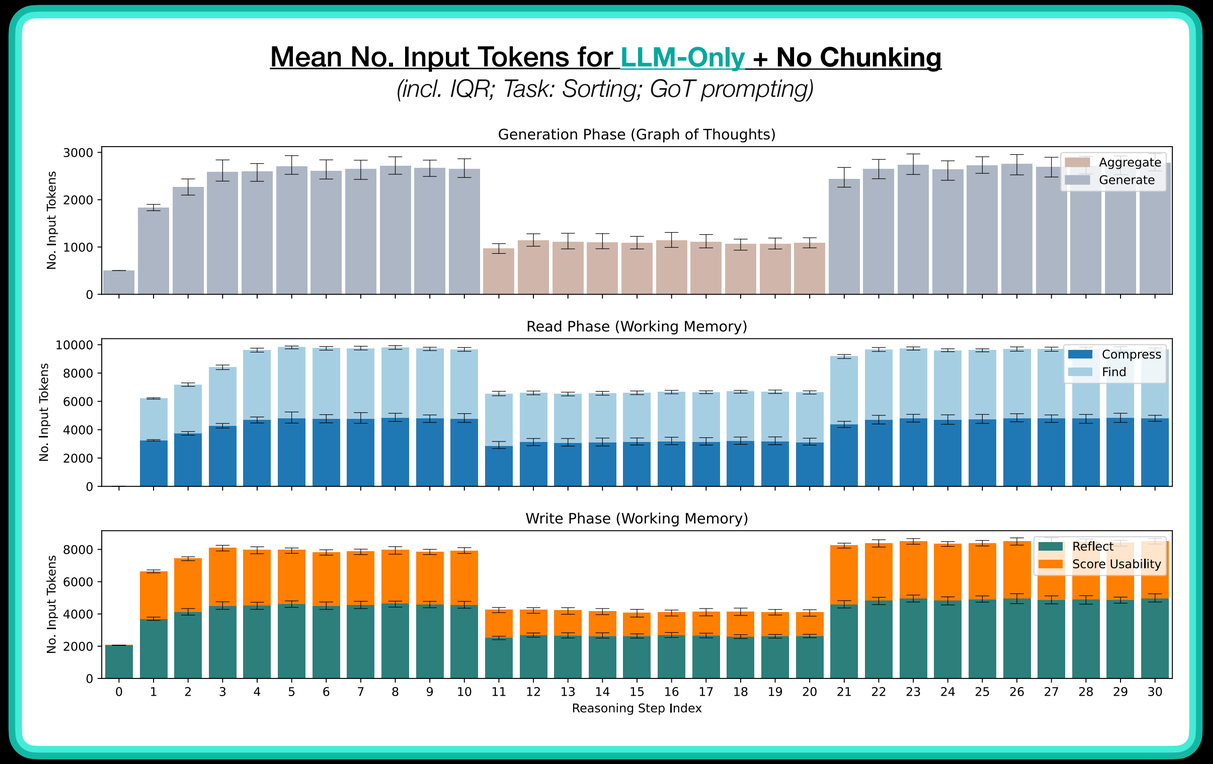


See Thesis, Subsection 5.3.4.

Assessment of Hypothesis III: LLM-Only > Embedding-Hybrid

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings

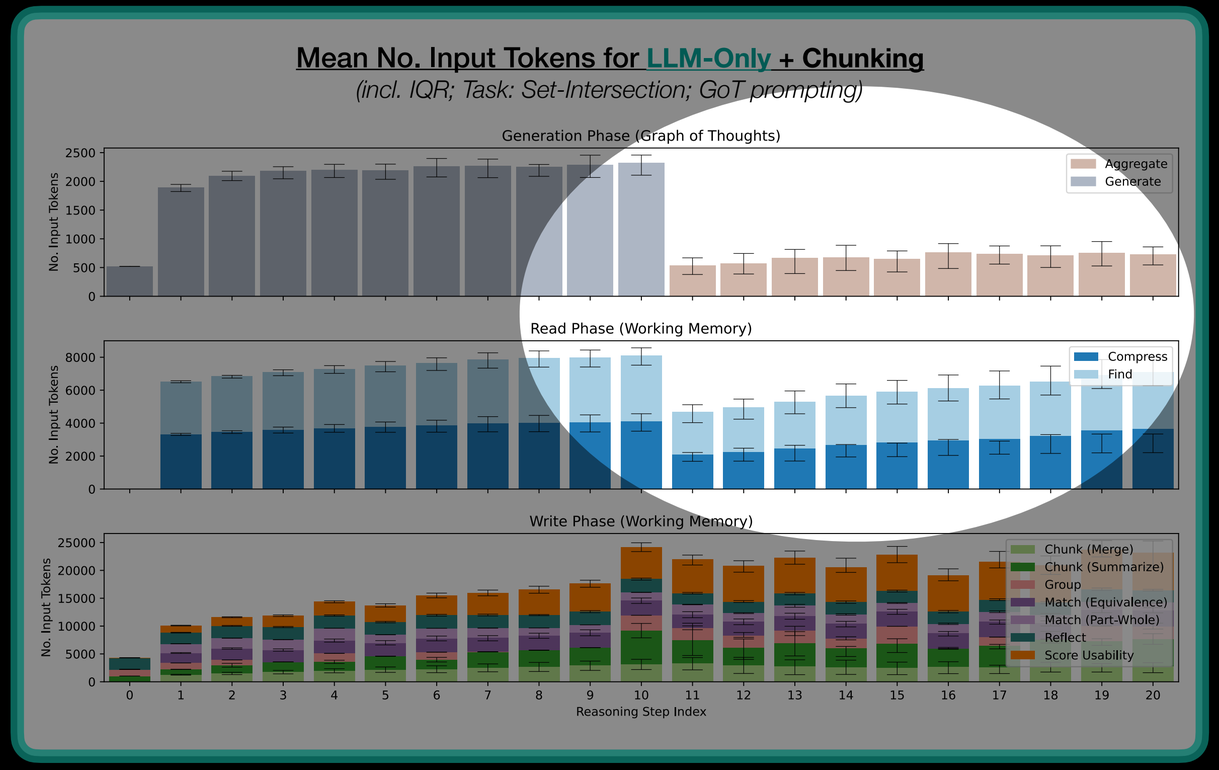


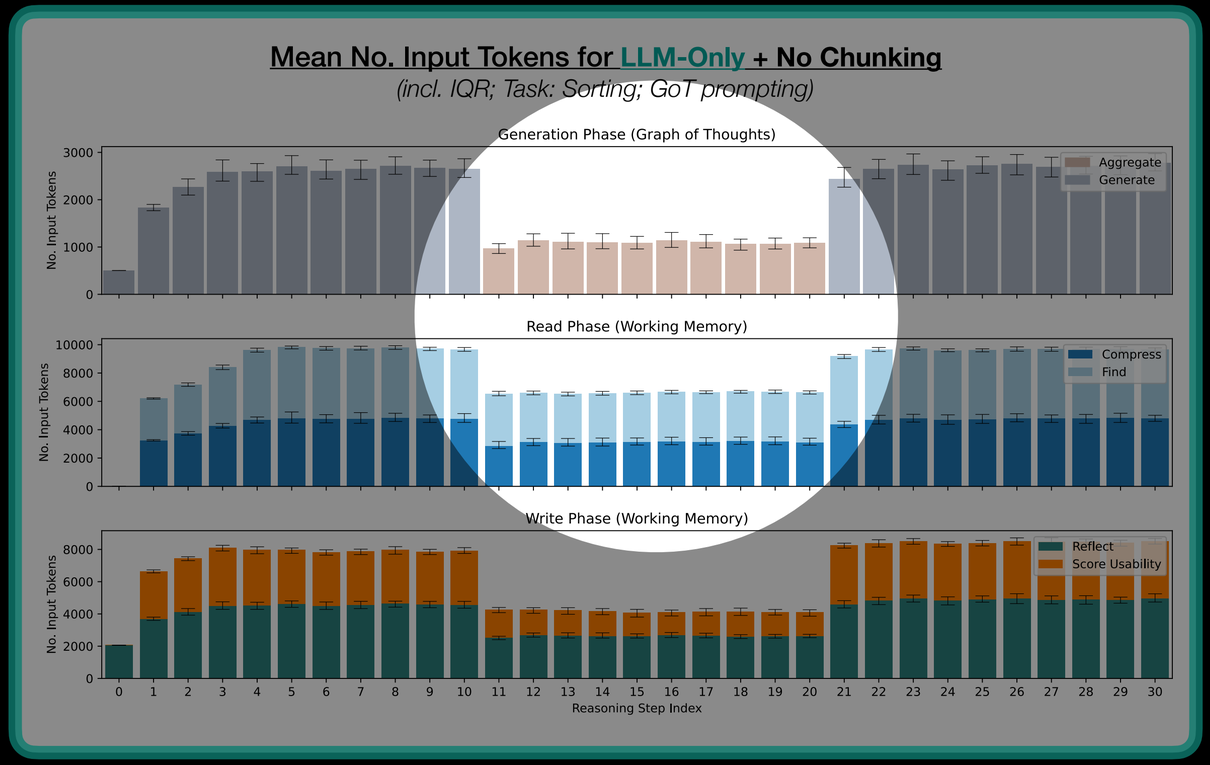


See Thesis, Subsection 5.3.4.

Assessment of Hypothesis III: LLM-Only > Embedding-Hybrid

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings





2. **LLM-Only variants appear responsive to reasoning context**

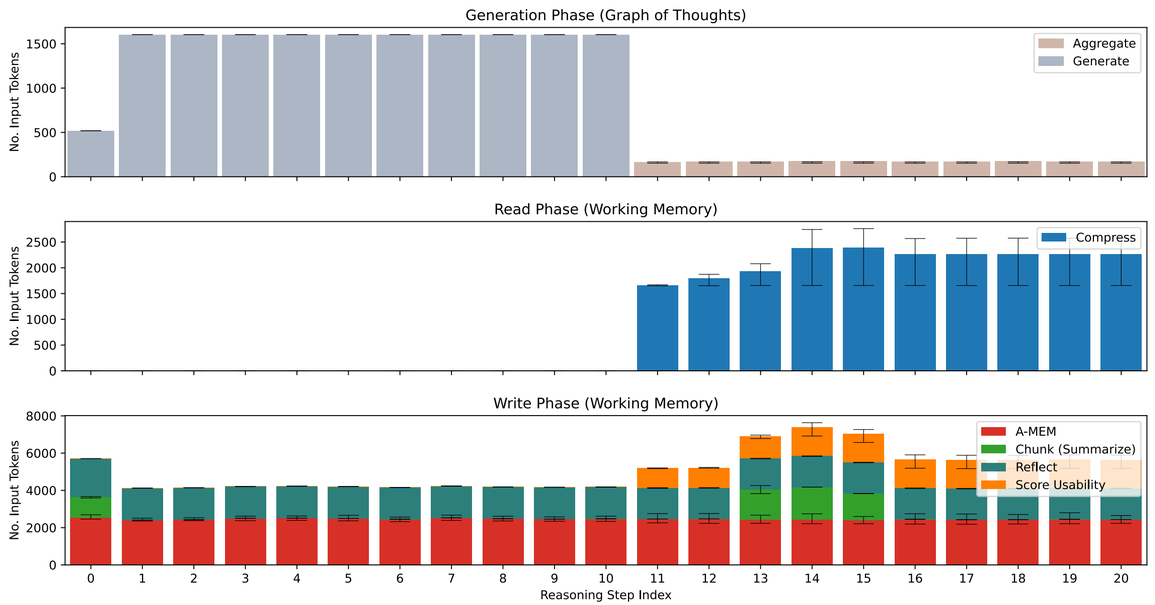
See Thesis, Subsection 5.3.4.

Assessment of Hypothesis III: LLM-Only > Embedding-Hybrid

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings

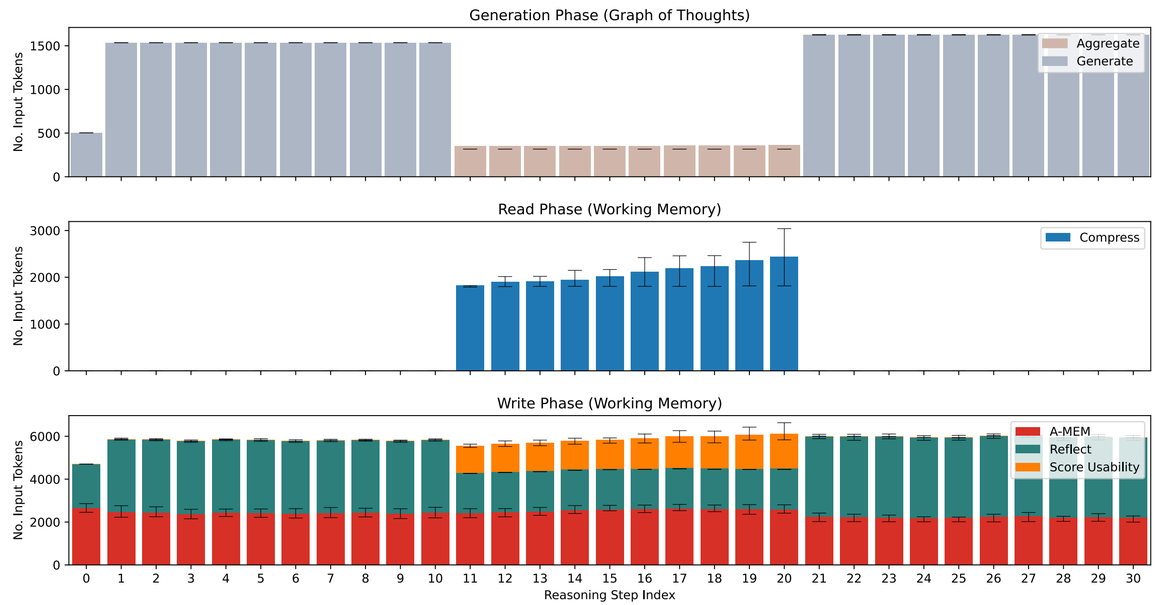
Mean No. Input Tokens for Embedding-Hybrid + Chunking

(incl. IQR; Task: Set-Intersection; GoT prompting)



Mean No. Input Tokens for Embedding-Hybrid + No Chunking

(incl. IQR; Task: Sorting; GoT prompting)



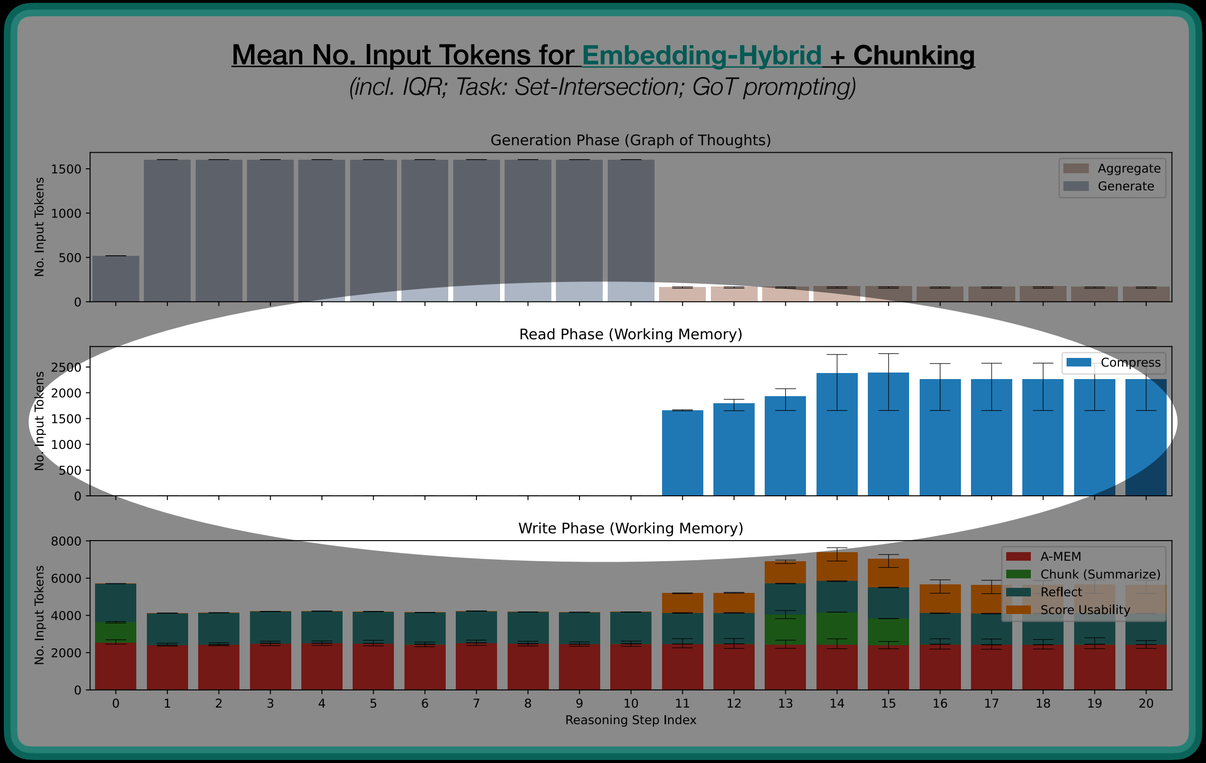
3. Embedding-Hybrids: Limited retrieval activity
↳ Less Flexible

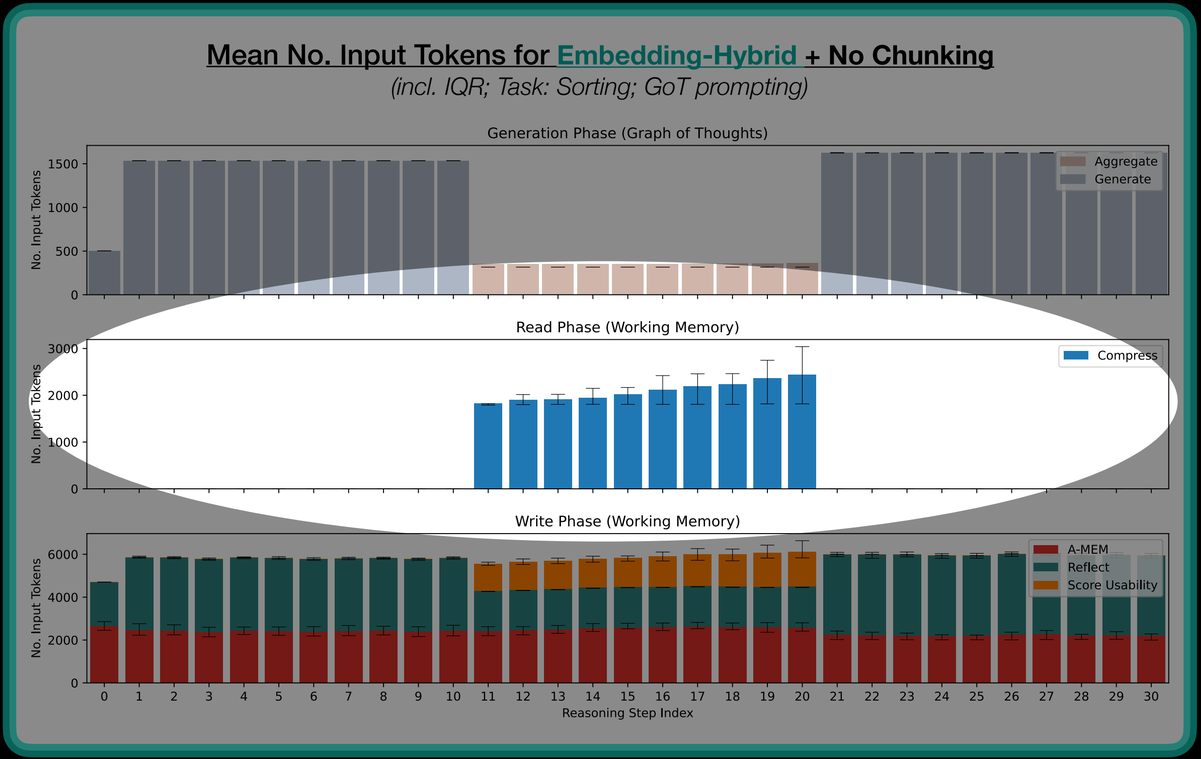
(in line with Zhou et al., 2025)

See Thesis, Subsection 5.3.4.

Assessment of Hypothesis III: LLM-Only > Embedding-Hybrid

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings



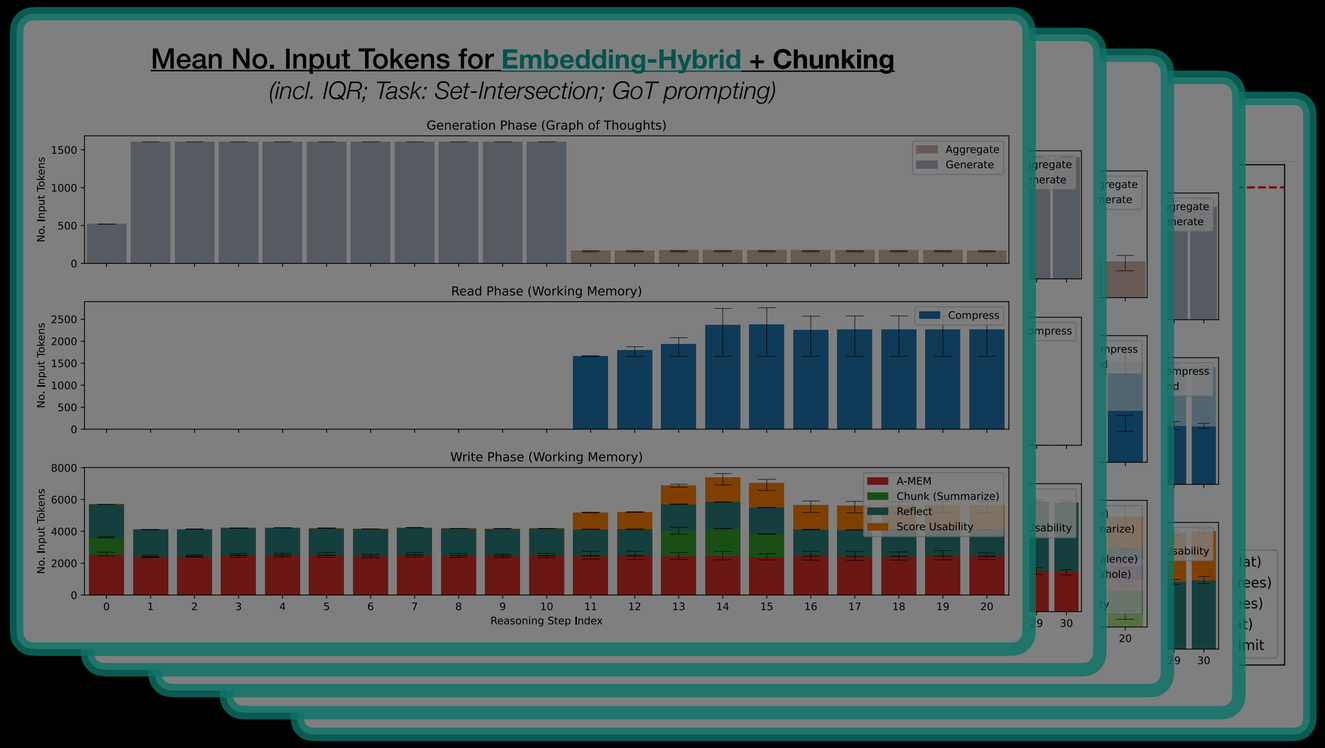


3. **Embedding-Hybrids: Limited retrieval activity**
   ↳ Less Flexible
   (in line with Zhou et al., 2025)

See Thesis, Subsection 5.3.4.

Assessment of Hypothesis III: LLM-Only > Embedding-Hybrid

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings



Limitation: No Robust Embedding-Based Baseline

- Dynamic similarity-threshold based on A-MEM's internals
- Not fine-tuned at all

See Thesis, Subsection 5.3.4. and 6.2.1.3.

Overall Token Consumption

Total no. input tokens (in millions) per operation **across all runs**
(GoT prompting, n = 120; AM = A-MEM, F = No Chunking, T = Chunking).

Operation	Overall	Flat	Trees	AM (F)	LLM (F)	AM (T)	LLM (T)
App (GoT; Besta et al. (2024b))							
Aggregate	2.9	1.4	1.5	0.3	1.1	0.3	1.1
Generate	15.0	7.4	7.5	2.9	4.5	2.9	4.6
Total	*17.8*	*8.9*	*9.0*	*3.2*	*5.6*	*3.2*	*5.7*
Read (WM)							
Compress	29.3	12.2	17.1	0.3	11.9	~~0.3~~	~~16.8~~
Find	26.9	12.7	14.2	–	12.7	–	14.2
Total	*56.2*	*24.9*	*31.3*	*0.3*	*24.6*	*0.3*	*31.0*
Write (WM)							
A-MEM	26.8	10.5	16.3	10.5	–	16.3	–
Chunk (Mer.)	11.6	–	11.6	–	–	–	11.6
Chunk (Sum.)	8.6	–	8.6	–	–	1.2	7.4
Group	1.2	–	1.2	–	–	–	1.2
Match (Eq.)	11.6	–	11.6	–	–	–	11.6
Match (P.W.)	2.6	–	2.6	–	–	–	2.6
Reflect	35.9	17.9	18.0	7.7	10.2	7.7	10.3
Usability	22.8	7.1	15.8	0.2	6.9	0.2	15.6
Total	*121.2*	*35.5*	*85.7*	*18.4*	*17.0*	*25.5*	*60.2*

See Thesis, Subsection 5.3.4.

Note: “Compress" for Chunking variants is compromised due to a duplication bug discovered in hindsight.

Overall Token Consumption

1. All WM variants incur substantial token costs compared to vanilla GoT
   — **Expectedly!**
   ↳ Cost scales per n thoughts:
   - GoT: $\Theta(n)$
   - Reasoning LLMs: $\Theta(n^2 + n)$
     (Which we aspire towards)

Total no. input tokens (in millions) per operation **across all runs**
(GoT prompting, n = 120; AM = A-MEM, F = No Chunking, T = Chunking).

Operation	Overall	Flat	Trees	AM (F)	LLM (F)	AM (T)	LLM (T)
App (GoT; Besta et al. (2024b))							
Aggregate	2.9	1.4	1.5	0.3	1.1	0.3	1.1
Generate	15.0	7.4	7.5	2.9	4.5	2.9	4.6
Total	*17.8*	*8.9*	*9.0*	*3.2*	*5.6*	*3.2*	*5.7*
Read (WM)							
Compress	29.3	12.2	17.1	0.3	11.9	~~0.3~~	~~16.8~~
Find	26.9	12.7	14.2	–	12.7	–	14.2
Total	*56.2*	*24.9*	*31.3*	*0.3*	*24.6*	*0.3*	*31.0*
Write (WM)							
A-MEM	26.8	10.5	16.3	10.5	–	16.3	–
Chunk (Mer.)	11.6	–	11.6	–	–	–	11.6
Chunk (Sum.)	8.6	–	8.6	–	–	1.2	7.4
Group	1.2	–	1.2	–	–	–	1.2
Match (Eq.)	11.6	–	11.6	–	–	–	11.6
Match (P.W.)	2.6	–	2.6	–	–	–	2.6
Reflect	35.9	17.9	18.0	7.7	10.2	7.7	10.3
Usability	22.8	7.1	15.8	0.2	6.9	0.2	15.6
Total	*121.2*	*35.5*	*85.7*	*18.4*	*17.0*	*25.5*	*60.2*

See Thesis, Subsection 5.3.4.

Note: "Compress" for Chunking variants is compromised due to a duplication bug discovered in hindsight.

Overall Token Consumption

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings

Total no. input tokens (in millions) per operation **across all runs**
(GoT prompting, n = 120; AM = A-MEM, F = No Chunking, T = Chunking).

Operation	Overall	Flat	Trees	AM (F)	LLM (F)	AM (T)	LLM (T)
App (GoT; Besta et al. (2024b))							
Aggregate	2.9	1.4	1.5	0.3	1.1	0.3	1.1
Generate	15.0	7.4	7.5	2.9	4.5	2.9	4.6
Total	*17.8*	*8.9*	*9.0*	*3.2*	*5.6*	*3.2*	*5.7*
Read (WM)							
Compress	29.3	12.2	17.1	0.3	11.9	~~0.3~~	~~16.8~~
Find	26.9	12.7	14.2	–	12.7	–	14.2
Total	*56.2*	*24.9*	*31.3*	*0.3*	*24.6*	*0.3*	*31.0*
Write (WM)							
A-MEM	26.8	10.5	16.3	10.5	–	16.3	–
Chunk (Mer.)	11.6	–	11.6	–	–	–	11.6
Chunk (Sum.)	8.6	–	8.6	–	–	1.2	7.4
Group	1.2	–	1.2	–	–	–	1.2
Match (Eq.)	11.6	–	11.6	–	–	–	11.6
Match (P.W.)	2.6	–	2.6	–	–	–	2.6
Reflect	35.9	17.9	18.0	7.7	10.2	7.7	10.3
Usability	22.8	7.1	15.8	0.2	6.9	0.2	15.6
Total	*121.2*	*35.5*	*85.7*	*18.4*	*17.0*	*25.5*	*60.2*

2. Clear token-efficiency advantage for retrieval with Embedding-Hybrid variants *(though only partially active)*

See Thesis, Subsection 5.3.4.

Note: "Compress" for Chunking variants is compromised due to a duplication bug discovered in hindsight.

Final Takeaways

Final Takeaways

I

Carefully Consider a Hybrid WM Approach:
Costly but Flexible LLMs v.s. Efficient but Rigid Embeddings Methods

II

Investigate WM for Reasoning LLMs *(see Thesis, Subsection 7.1.2.)*

↳ **May lack active reasoning-context regulation → Need MCM's Central Executive**

↳ **May address symptoms like Lost-in-the-Middle, Overthinking, Error Propagation, etc.**

(cf. Liu et al. (2023), Sui et al. (2025), Gan et al. (2025) respectively)

III

Consider WM Across the Implementation Stack: *(see Thesis, Subsection 7.1.1.)*
Let Agents Profit from WM by Embracing Active and Non-Linear Memory

IV

Recognize Cognitive WM Models Potential for LLMs

Overview of Limitations

- Core Architecture: No inherent notion of episode order
- Setup: No backtracking in GoT (generates thoughts in BFS order) → Limits accumulation in WM

Implemented WM variants:

- Limited reflection setup:
  ↳ No task-specific scoring scheme *(as suggested by Madaan et. al., 2023)*
  ↳ Minimal context (only a single thought)
  ↳ Minimal error signal captured (not entire CoT, only erroneous step)
- Greedy, shallow, and non-reorganizing writing
- Not fully optimized (e.g. no. write request)

Deferred experiments:

- Effects of different memory capacities on performance
- Ablation on forgetting scores
- Cross-trial memory reuse
- …

See Thesis, Chapter 6.

Bibliography

Baddeley, A. D. (2000). The episodic buffer: A new component of working memory?
Trends in Cognitive Sciences, 4(11), 417–423. https://doi.org/10.1016/S1364-6613(00)01538-2

Baddeley, A. D. (2010). Working memory: Thought and action.
Oxford University Press. https://doi.org/10.1093/acprof:oso/9780198528012.001.0001

Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Gianinazzi, L., Gajda, J., Lehmann, T., Niewiadomski, H., Nyczyk, P., & Hoefler, T. (2024).
Graph of thoughts: Solving elaborate problems with large language models.
Proceedings of the AAAI Conference on Artificial Intelligence, 38(16), 17682–17690. https://doi.org/10.1609/aaai.v38i16.29720

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R.., . . . Amodei, D. (2020). Language models are few-shot learners.
https://arxiv.org/abs/2005.14165

Gan, Z., Liao, Y., & Liu, Y. (2025). Rethinking external slow-thinking: From snowball errors to probability of correct reasoning.
https://arxiv.org/abs/2501.15602

Hitch, G. J., Allen, R. J., & Baddeley, A. D. (2025). The multicomponent model of working memory fifty years on.
Trends in Cognitive Sciences, 29(5), 389–403. https://doi.org/10.1016/j.tics.2025.04.003

Hu, M., Chen, T., Chen, Q., Mu, Y., Shao, W., & Luo, P. (2024). Hiagent: Hierarchical working memory management for solving long-horizon agent tasks with large language model.
https://arxiv.org/abs/2408.09559

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., & Liang, P. (2023). Lost in the middle: How language models use long contexts.
https://arxiv.org/abs/2307.03172

Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., Gupta, S., Majumder, B. P., Hermann, K., Welleck, S., Yazdanbakhsh, A., & Clark, P. (2023).
Self-refine: Iterative refinement with self-feedback.
https://arxiv.org/abs/2303.17651

Mallen, A., Asai, A., Zhong, V., Das, R., Khashabi, D., & Hajishirzi, H. (2023). When not to trust language models: Investigating effectiveness of parametric and non-parametric memories.
https://arxiv.org/abs/2212.10511

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information.
Psychological Review, 63(2), 81–97. https://doi.org/10.1037/ h0043158

Murdock, B. B. (1962). The serial position effect of free recall.
Journal of Experimental Psychology, 64(5), 482–488. https://doi.org/10.1037/h0045106

Norman, D. A., & Shallice, T. (1986). Attention to action: Willed and automatic control of behavior.
In R. J. Davidson, G. E. Schwartz, & D. Shapiro (Eds.), Consciousness and self-regulation: Advances in research and theory, vol. 4 (pp. 1–18). Plenum Press. https://doi.org/10.1007/978-1-4613-2177-6_1

Park, J. S., O'Brien, J., Cai, C. J., Morris, M. R., Liang, P., & Bernstein, M. S. (2023). Generative agents: Interactive simulacra of human behavior.
Proceedings of the 36th annual acm symposium on user interface software and technology, 1–22.

Renze, M., & Guven, E. (2024). Self-reflection in llm agents: Effects on problem-solving performance.
arXiv preprint arXiv:2405.06682.

Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., & Yao, S. (2023). Reflexion: Language agents with verbal reinforcement learning.
Advances in Neural Information Processing Systems, 36, 8634–8652.

Stroop, J. R. (1935). Studies of interference in serial verbal reactions.
Journal of Experimental Psychology, 18(6), 643–662. https://doi.org/10.1037/h0054651

Sui, Y., Chuang, Y.-N., Wang, G., Zhang, J., Zhang, T., Yuan, J., Liu, H., Wen, A., Zhong, S., Chen, H., & Hu, X. (2025). Stop overthinking: A survey on efficient reasoning for large language models.
https://arxiv.org/abs/2503.16419

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). Chain-of-thought prompting elicits reasoning in large language models.
https: //arxiv.org/abs/2201.11903

Xu, W., Mei, K., Gao, H., Tan, J., Liang, Z., & Zhang, Y. (2025). A-mem: Agentic memory for llm agents.
https://arxiv.org/abs/2502.12110

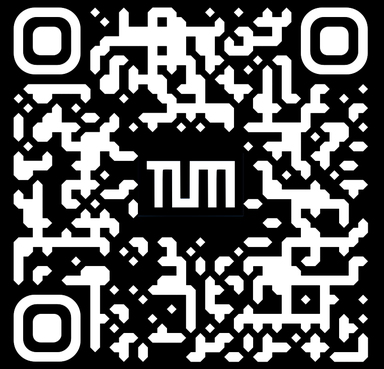
Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., & Narasimhan, K. (2023b). Tree of thoughts: Deliberate problem solving with large language models.
https://arxiv.org/abs/2305.10601

Zhou, X., He, J., Zhou, W., Chen, H., Tang, Z., Zhao, H., Tong, X., Li, G., Chen, Y., Zhou, J., et al. (2025). A survey of llm x data.
arXiv preprint arXiv:2505.18458.

Cognitive Theory for Deliberate Reasoning: **Synthesizing Working Memory Architectures for LLMs**

Alexander Blatzheim
Master Thesis in Information Systems
Supervised by Tobias Eder

Research Group Social Computing @ Technical University of Munich
September 2025



Cognitive Theory for Deliberate Reasoning: **Synthesizing Working Memory Architectures for LLMs**

Alexander Blatzheim
Master Thesis in Information Systems
Supervised by Tobias Eder

Research Group Social Computing @ Technical University of Munich
September 2025

Backup Slides

Full MCM Interpretation Class Diagramm

Contribution I: The MCM in SE Terms | Part I: Theoretical Foundation

Note:
The original MCM subsumes the Contention Scheduler into the Central Executive, citing it as key inspiration. We make it explicit.

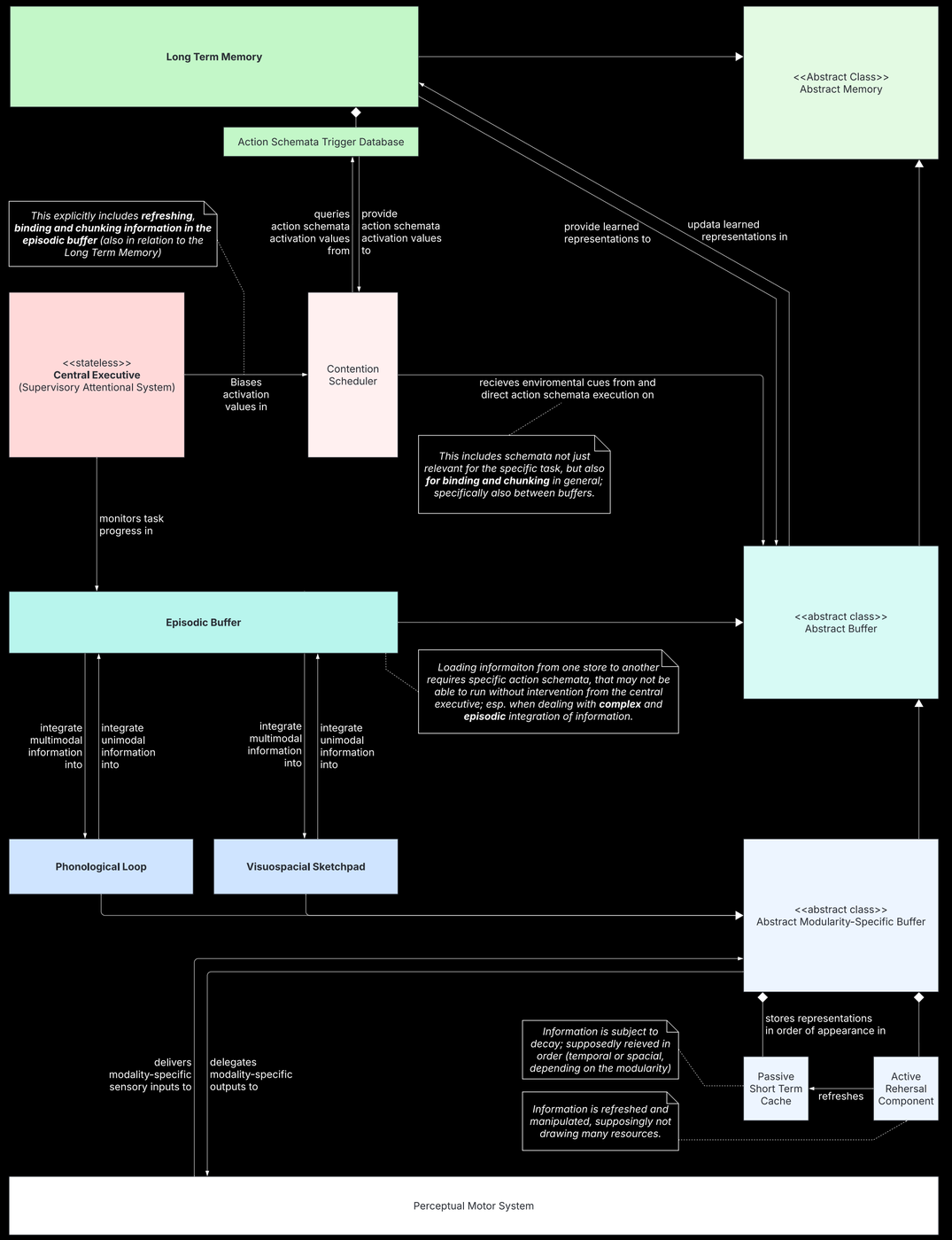
Classes not mentioned previously:

- Memory hierarchy
- Action Schemata Trigger Database:
  Delivers activation value for a given action schema + Environment

Limitations we see in our model:

- No memory item class for simplicity
  (Introducing one would require modeling action schemata as a form of memory item, which would make the class hierarchy more complicated).
  ↳ No schemata model, though plausible.
- Uncertain what constitutes “environmental cues” *(as in Norman & Shallice, 1986)*
  ↳ We assume these to be a subset of the states of all buffers.
- Deliberately refraining from specifying multiplicities
  (due to general uncertainty expressed by Hitch et al. (2025) towards how many more sub-memories or processors there may be).

Interpretation based on Baddeley (2000, 2010), Hitch et. Al. (2025), Norman & Shallice (1986)
See Thesis, Subsection 3.3.1.3



Item Survival Score Computation

- Items can record events (timestamps provided by a buffer), which can also be paired with other scores.
- We implement:
  - Two events: Item **access &** Item **repetition**
  - One subjective score: **Reflection Usability** (evaluated by the LLM)
- Formally, the survival contribution of an event or score recorded at time-step $t \in \mathbb{R}$ w.r.t. current timestamp $T \in \mathbb{R}$ is weighted by λ^{T-t}

  - An event's survival contribution is the decayed sum of timestamps, eg:

$$\text{AccessScore}(T) = \sum_{t_i \in \text{AccessHistory}} \lambda^{T-t_i}$$

  - a score's survival contribution is the decayed sum of all recorded timestamps t_i weighted by its respective score value s_i, eg:

$$\text{UsabilityScore}(T) = \sum_{(t_i, s_i) \in \text{UsabilityHistory}} s_i \cdot \lambda^{T-t_i}$$

  - Which, in our case, is combined into the final item-survival score as

$$\text{CombinedScore}(T) = (0.5 \cdot \text{AccessScore}(T) + \text{RepetitionScore}(T)) \cdot (1 + \text{UsabilityScore}(T))$$

See Thesis, Subsection 4.3.4.

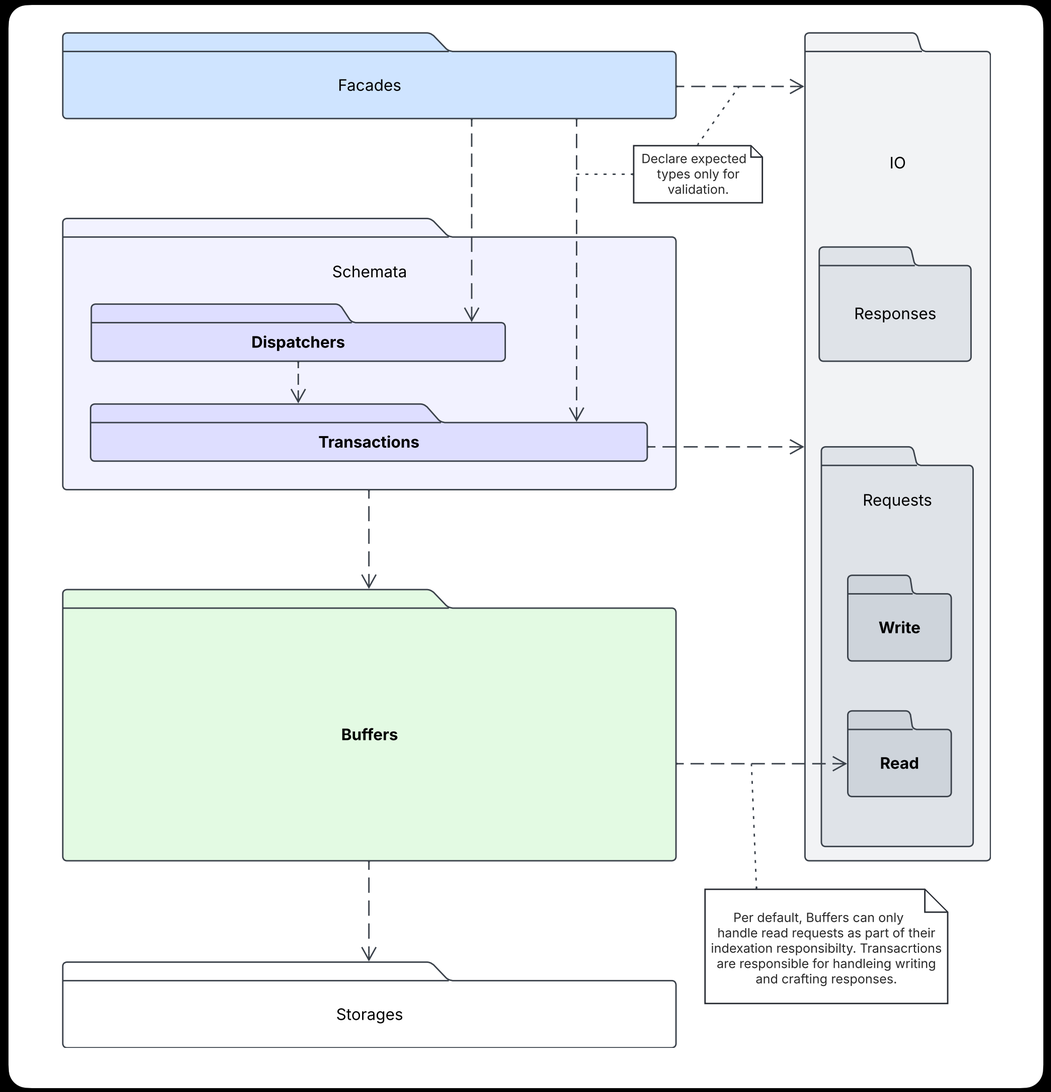
Core Architecture: Package Diagram

Hybrid 4-layer WM architecture with vertical control flow; I/O package ensures consistent communication across layers:

- **Facades** – specific client APIs delegating to dispatchers
- **Schemata** – unify dispatchers (schedulers) and typed transactions (action schemata).
- **Buffers** – typed memory components.
- **Storages** – storage backends.

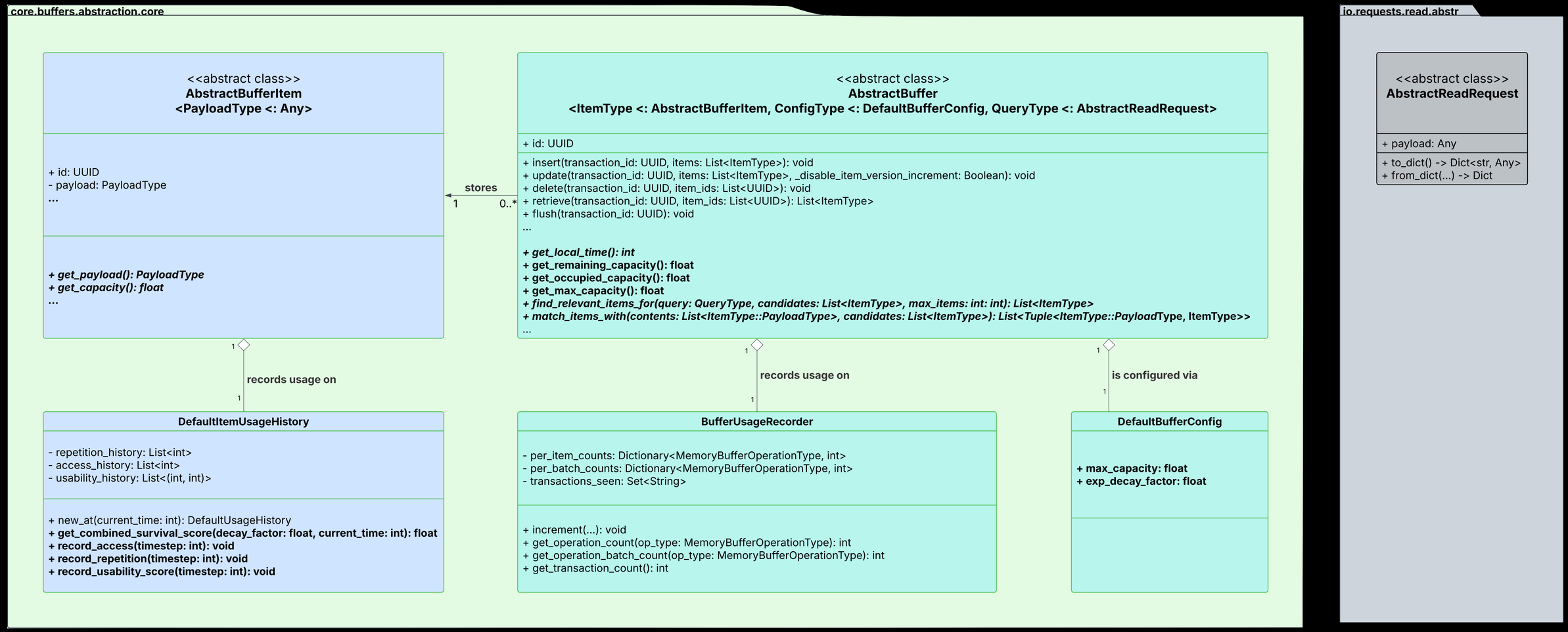
Implementation may still undergo minor structural refactoring to fully align with this package layout. Nonetheless, the core responsibilities of each layer are already clearly defined. Local utilities such as the SnapshotService for the buffer layer are omitted for simplicity.

See Thesis, Subsection 4.3.1.



Core Architecture: AbstractBuffer & AbstractBufferItem Class Diagram

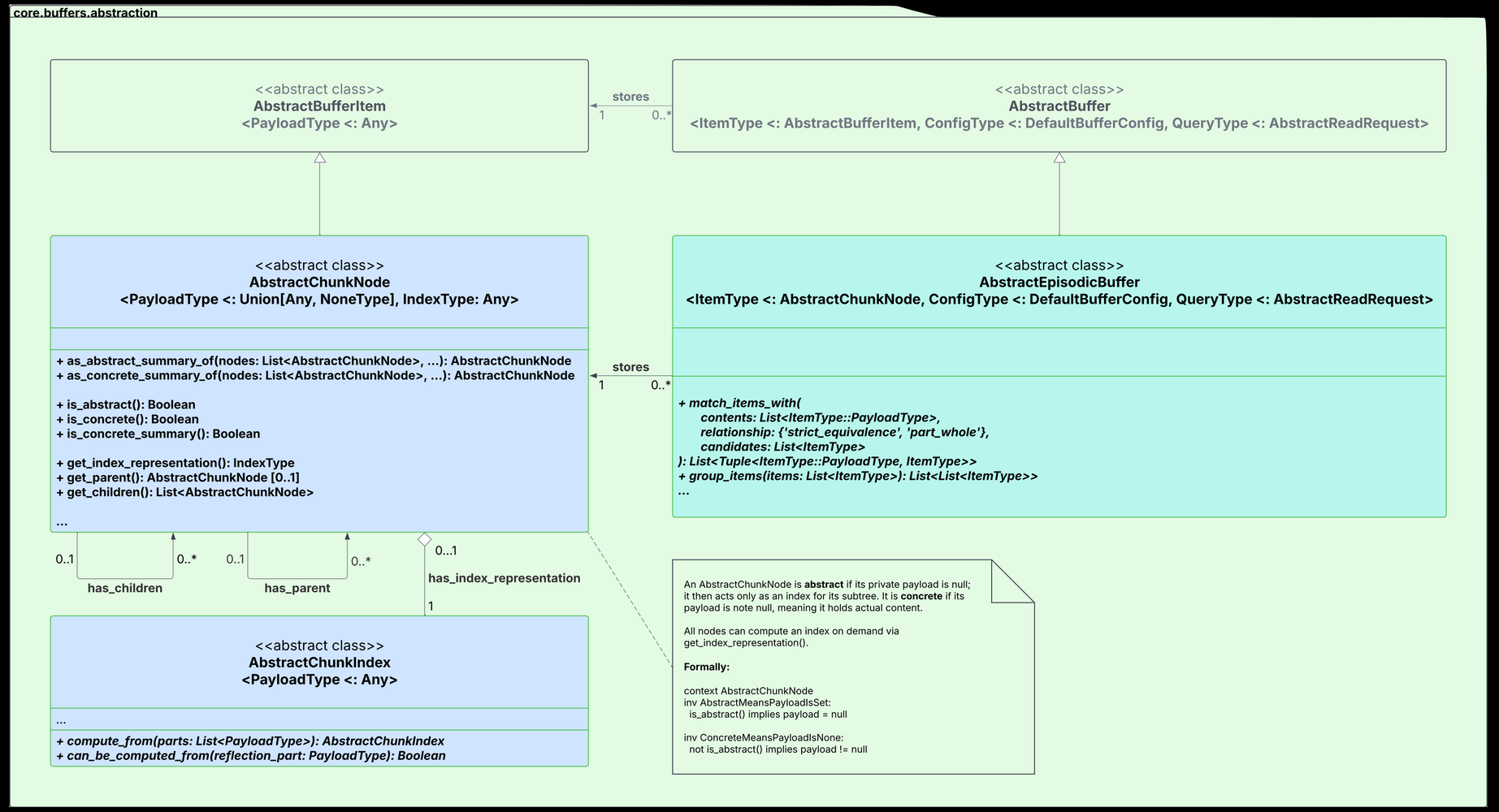
Contribution II: A WM Core Architecture for LLMs | Part II: Implementation



See Thesis, Subsection 4.3.3.
Implementation may still undergo minor structural refactoring to fully align with this definition.

Core Architecture: AbstractEpisodicBuffer & AbstractChunkNode Class Diagram

Contribution II: A WM Core Architecture for LLMs | Part II: Implementation



See Thesis, Subsection 4.3.4.
Implementation may still undergo minor structural refactoring to fully align with this definition.

Core Architecture: Notes on Extensibility

Extensibility within current design:

- Multiple buffer types with defensive verification & pre-/post- operation hooks
- Varying capacity declarations
- Multiple forgetting strategies
- Transaction typing & I/O verification (foundation for hierarchy)
- Flexible reasoning granularity (*AbstractRequest/Response*)
- Flexible reflection granularity (*AbstractReflectableMaterial*)

Deferred / Speculative Extensions:

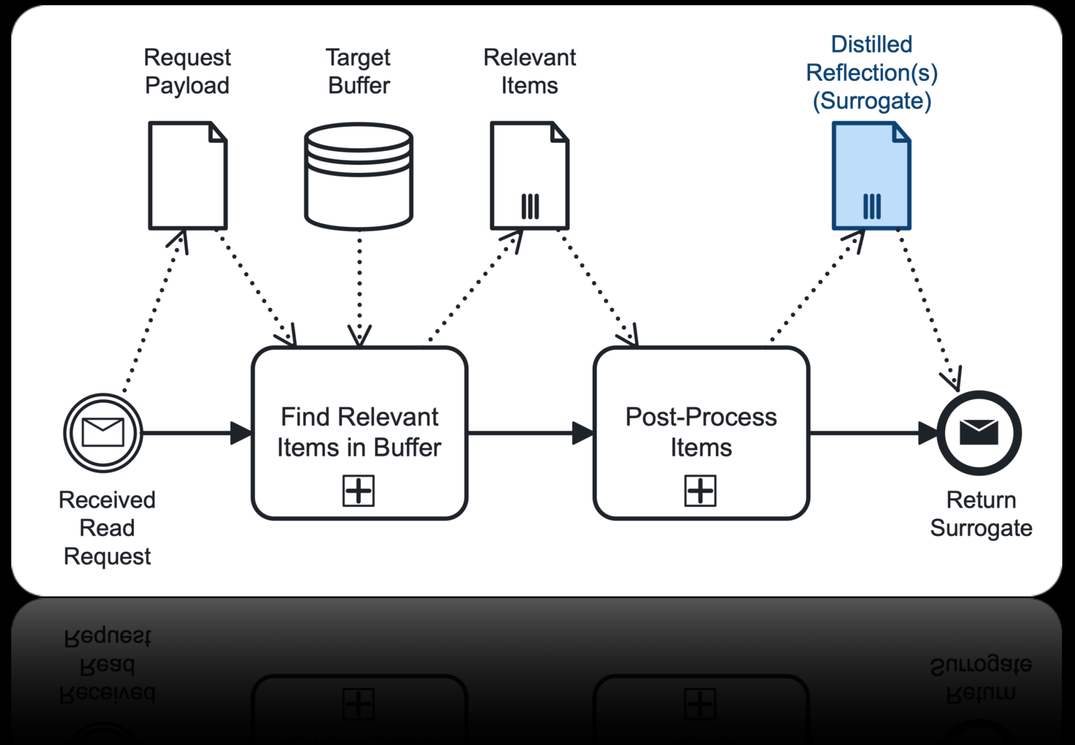
- Hierarchical transactions and contention scheduling (in *Schema* Package)
- Long-Term Memory (LTM) integration across the board:
  - **Facades** – LTM may augment read/write requests to improve retrieval & payload construction
  - **Schemata** – LTM may supply schemata & activation values for scheduling
  - **Buffers** – LTM may preload knowledge and consolidates stable WM reps.
  - **Storages** – passive; could feature specialized LTM storage

See Thesis, Subsection 4.3.5.

Implement WM Variants: The Reading Transactions

- Uses the find-operation of the buffer (variant-specific)
- Shared across all variants (reading from a forest subsumes reading from a flat item set)
- Two step process:

1. **Find:**
   Use buffer's find-op. to filter the set of items (nodes), given the thought-prompt as query
   (Filters all items, except children of aggregated summary nodes)
2. **Post-Process:**
   Compress/Re-Summarize the filtered set up to at most 8 reflections, in context of the query
   (All items contributing to a summary are recorded as accessed)



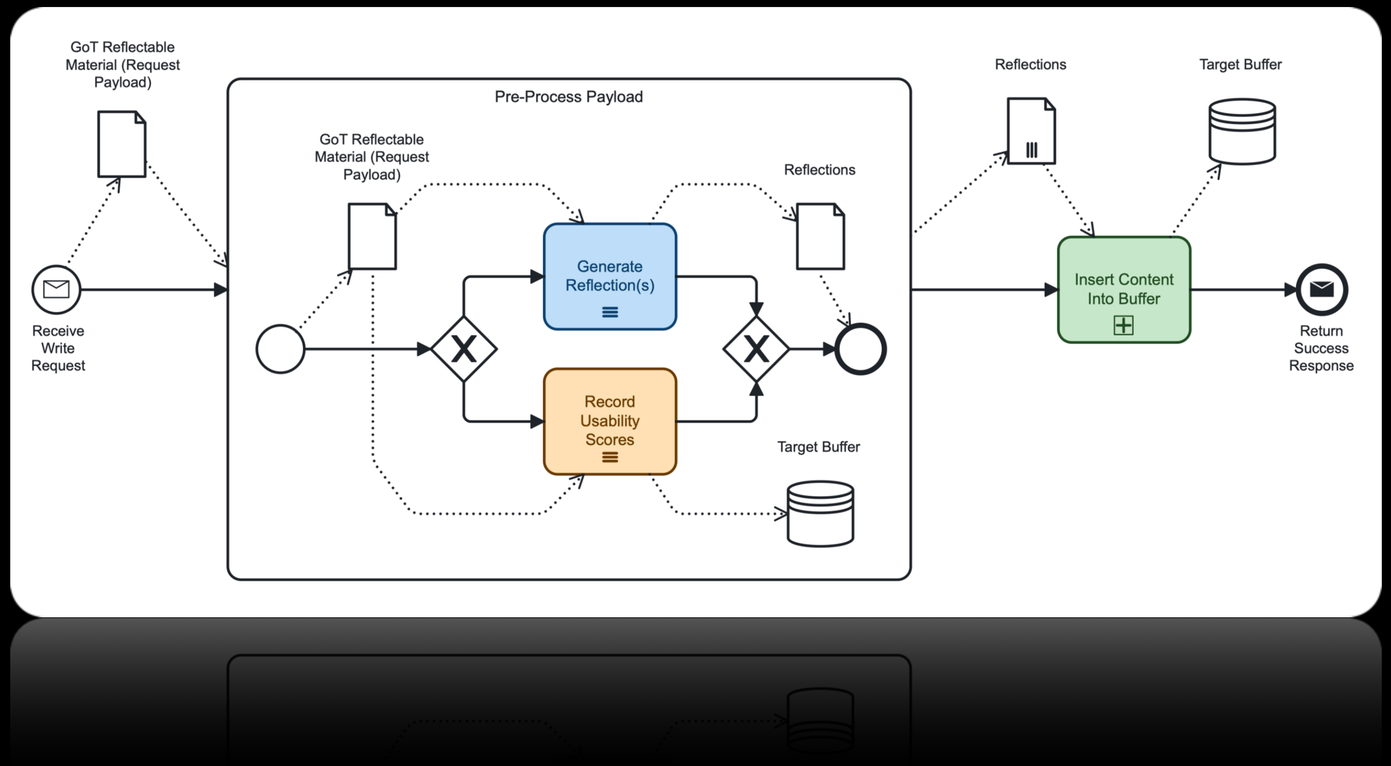
See Thesis, Subsection 4.4.5.

Implement WM Variants: The Writing Transactions (1/2)

- Uses the match & group operation of the buffer (variant-specific)
- Three step process:

1. **Record Usability**:
   If a reflection is present in the given thought, estimate its usability and record the score.
   (Shared by all variants)
2. **Generate reflections** for the newly given thought.
   (Shared by all variants)

(Non-BPMN-compliant dataflows go across subprocess boundary to avoid clutter).



3. **Insert** the reflections as new nodes in the forest (variant-specific)
   ↳ Forgetting: If committing changes exceeds the buffer capacity:
   Rank all existing items by their combined survival score and delete as many as needed
   (forgetting mechanism is shared by all variants)

See Thesis, Subsection 4.4.5.

Implement WM Variants: The Writing Transactions (2/2) — Insertion

- **Non-Chunking (Flat)** variants simply dump new items as concrete root nodes into the forest.
- **Chunking enabled variants** assumes a near-perfect clustering to be present (no reorganization), and prioritize insertion in existing trees over the formation of new ones:
  - ↳ Leading to the following relationship-assumptions: A new node n_r can have at most:
    - 1 part-whole relationship to an existing abstract node
    - 1 equivalence relationship to an existing concrete node (that has no concrete parent)
    - n group-relationships to concrete root nodes that are currently sole or disconnected members of their group
  - ↳ Insertion for a single node $\boldsymbol{n_r}$ can be summarized as follows:

 # First, search for equivalent nodes:

If: $\boldsymbol{n_r}$ equivalent to any concrete root node *(buffer.match_items(…))*
  Subsume $\boldsymbol{n_r}$ into the root by replacing with a summary node → Return

Else if: $\boldsymbol{n_r}$ has part-whole relationship to any abstract node $\boldsymbol{n_a}$ *(buffer.match_items(…))*

  If: $\boldsymbol{n_r}$ equivalent to any concrete child $\boldsymbol{n_c}$ of $\boldsymbol{n_a}$
    Subsume $\boldsymbol{n_r}$ into the $\boldsymbol{n_c}$ by replacing with a summary node → Return

  Else: *# No equivalent nodes found*
    Insert $\boldsymbol{n_r}$ as new child of $\boldsymbol{n_a}$ → Return

 # No equivalence or part-whole relationships detected → Build a new tree

Search for groupings among concrete root nodes *(buffer.group_items(…))*
  Any grouping >= 1 receives a new abstract parent node

If: $\boldsymbol{n_a}$ not assigned to any grouping:
  Insert $\boldsymbol{n_r}$ as new concrete root node

Event recordings throughout:

- Access events on node $\boldsymbol{n_i}$ every time a relationship between $\boldsymbol{n_r}$ and $\boldsymbol{n_i}$ is checked (propagated up to root, and down to subtree of $\boldsymbol{n_i}$)
- Repetition Events on node $\boldsymbol{n_i}$ every time an equivalence relationship between $\boldsymbol{n_r}$ and $\boldsymbol{n_i}$ is detected (propagated down to subtree of $\boldsymbol{n_i}$)

See Thesis, Subsection 4.4.5.

Variant-Specific Buffer Implementations

All buffers implement:

- **find_relevant_items_for(query, candidates, max_size) → ranked_items**
  Filters non-meaningful items in *candidates* and ranks the remaining according to the relevance to the given *query* (a thoughts prompt), optionally capped to *max_size.*
  - ↳ **LLM-Only variants**:
    Filter all given nodes individually in one inference call
  - ↳ **A-MEM-Variants:**
    Embed *query* as *q*; filter *c* ∈ *candidates if d(c, q) <= max_link_distance; rank by d(c, q).*

- **match_items(contents, relationship, candidates) → matches**
  Identifies two types of relationships (part-whole or equivalence) between *contents* (reflections) and *candidate* nodes. A reflection may match at most one candidate; which may be matched by multiple.
  - ↳ **LLM-Only variants:**
    n x n matching in one inference call, specific prompts per relationship.
  - ↳ **A-MEM-Variants:**
    n x (1 x n) filtering. For each *c* ∈ *content:* Filter *a* ∈ *candidates* if
    - ↳ Part-Whole relationship: *d(c, a) <= max_link_distance*
    - ↳ Equivalence relationship: *d(c, a) <= 0.05*

- **group_items(items) → groupings**
  Returns a clustering (set of groupings) of the given set of (concrete) nodes.
  - ↳ **LLM-Variants:**
    All items in one call.
  - ↳ **A-MEM-Variants:**
    Agglomerative clustering on all items with max_link_distance as cluster threshold.

A-MEM-based / Embedding-Hybrid Variants:

A) Operate on absolute cosine distance:

$$d(\mathbf{u}, \mathbf{v}) = 1 - \left| \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \, \|\mathbf{v}\|} \right|$$

B) obtain a flexible distance threshold, *max_link_distance*, the max. distance between any two verified neighbors in $\mathscr{E}$: $\max_{(i,j)\in\mathscr{E}} d(\mathbf{u}_i, \mathbf{u}_j)$, with a fallback to 0.3 if $\mathscr{E} = \varnothing$.

C) Any parent node's distance is rebalanced as the mean of its native similarity score and the average score of its children.

See Thesis, Subsection 4.4.6.

Implement WM Variants: Reflection Generation & Best Practices

Contribution III: Four Concrete WM Variants | Part II: Implementation

Reflection Format:

```
<Reflection Item>
    <Error Location>
        "If A is taller than B, then B is taller than A."
    </Error Location>
    <Error Explanation>
        This step reverses the comparison logic; the
        conclusion contradicts the premise.
    </Error Explanation>
    <Error Fixing Suggestion>
        Keep direction consistent: If A > B, then B < A.
    </Error Fixing Suggestion>
    <Takeaway>
        Maintain logical direction when chaining.
    </Takeaway>
    <Scores>
        <Repetition Score> 0.21 </Repetition Score>
        <Access Score> 0.37 </Access Score>
        <Usability Score> 0.64 </Usability Score>
        <Combined Score> 0.58 </Combined Score>
    </Scores>
</Reflection Item>
```

Reflection Generation:

- Limited to a single thought (may contain prior reflection).
- Multiple distinct reflections 1-shot (early on separation)
- Instruct LLM to combine multiple reflection strategies (e.g. keywords, advice, explanation, etc.)

Reflection Incorporation:

- Limited to a single thought (may contain prior reflection).
- Multiple distinct reflections in one shot (early on separation)
- Instruct LLM to combine multiple reflection strategies (e.g. keywords, advice, explanation, etc.)

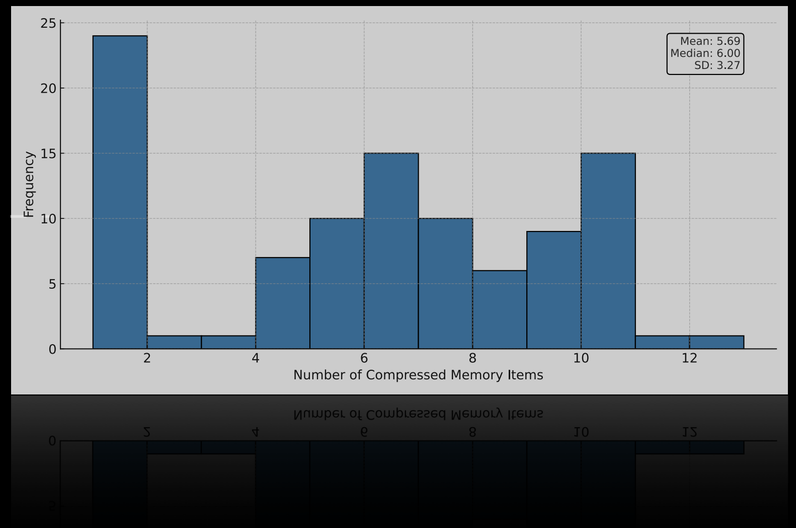
Best Practices from Related Work

- **Diverse Feedback** quality dimensions stabilize gains (Madaan et al., 2023)
  - ↳ We remain task-agnostic; don't add task-specific feedback scores → ***(Limitation ✗)***
    To counteract, we add survival scores (access/repetition/usability) as a proxy.
- **Actionable & Specific Feedback** — concrete fixing suggestion + error is key *(Madaan et al., 2023)*
  - ↳ We include both the <Error Location> and <Error Fixing Suggestion> → ✓
- **Combined feedback styles** yield strongest gains *(Renze & Guven, 2024)*
  - ↳ We encourage multiple variants, but leave the choice to the model → ✓
- **Include Prior Attempt(s) in the subsequent Generation** *(Shinn et al., 2023)*
  - ↳ We restrict reflection generation to the local thought;
    when incorporated, the reflected-upon thought is not included → ***(Limitation ✗)***
  - ↳ Noticed in hindsight that only the erroneous step,
    not the relevant (sub-)CoT is included most of times → ***(Limitation ✗)***
- **Multiple Iterations** improve quality with diminishing returns *(Madaan et al., 2023; Shinn et al., 2023).*
  - ↳ Every GoT thought is reflected on; prior reflections carry forward.
    Iteration depth is not fixed, but depends on the reasoning structure → ***(Partial Limitation)***
- **Reflecting More Often doesn't Hurt** — better reflect too often than too little (Madaan et al., 2023).
  - ↳ We apply reflections after every GoT thought → ✓
- **No. of Reflections to Include Unclear** – too many may risk retrieval noise (cf. Liu et al., 2023).
  - ↳ We summarize to max. 8 reflections at read-time before incorporation → ✓

See Thesis, Subsection 4.4.4.

Experimental Setup (With Hyperparameters)

- **Tasks: Sorting & Set-Intersection — adapted from GoT (Besta et. al., 2024)**
  - On a list/set of integers (size=32)
  - Eval metric: Accuracy
- 5 competing setups:
  ↳ No WM (Vanilla GoT); GoT+ each of our four WM variants (resp.)
  ↳ Each assessed across I/O, CoT, ToT, GoT prompting
  ↳ For each: 20 samples x redraw of 3 → 60 datapoints in total
- **WM capacity = 7**
  (mean number of retrieved items in a trial run with capacity = ∞)
- **Model: gpt-4o-mini-2024-07-18**
  (Documented at https://platform.openai.com/docs/models/gpt-4o-mini)
- **Hyperparameters:**
  - *LLM — Temperature = 0.8; max. gen. length = ∞*
  - *WM — Exponential forgetting factor: 0.995 (adapted from Park et al., 2023)*
  - *A-MEM:*
    - *Re-indexing threshold lowered from 100 to 10*
    - *Fallback distance threshold: 0.3*
    - *Embedding model: all-MiniLM-L6-v2 (a variant of the model proposed by Wang et al., 2020)*
      (https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2)



See Thesis, Subsection 5.3.1.

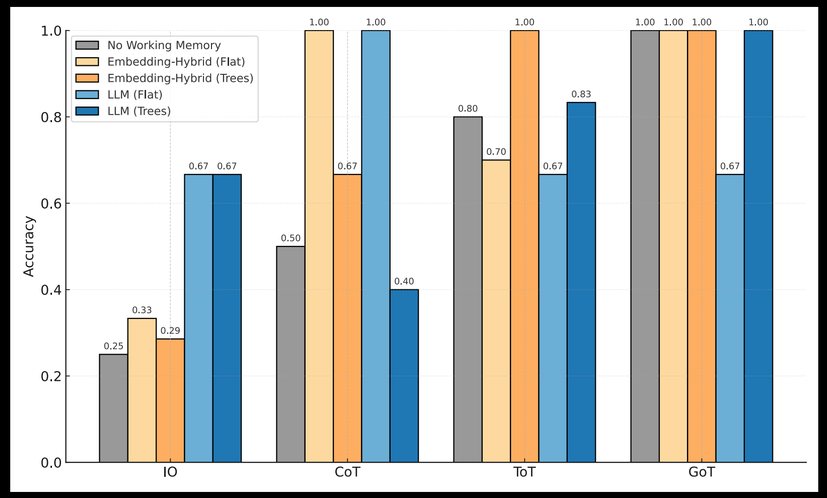
Assessment of Hypothesis I: Overall vs. Majority Vote Accuracy

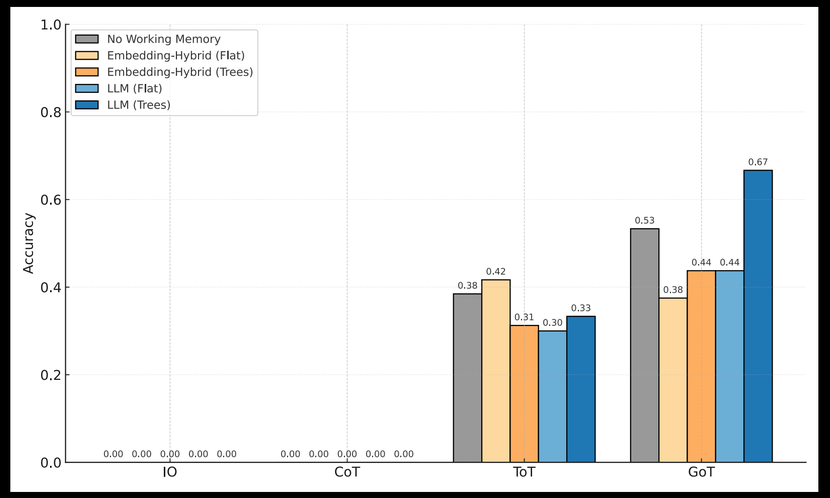
Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings

Set-Intersection

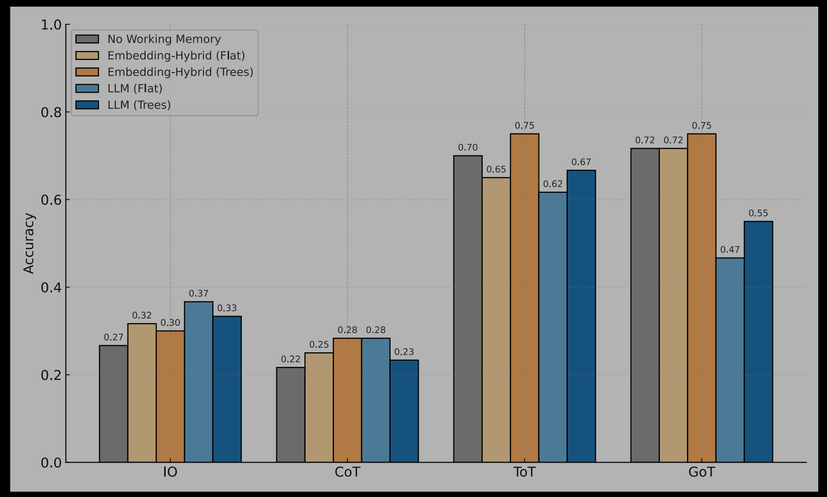
Sorting

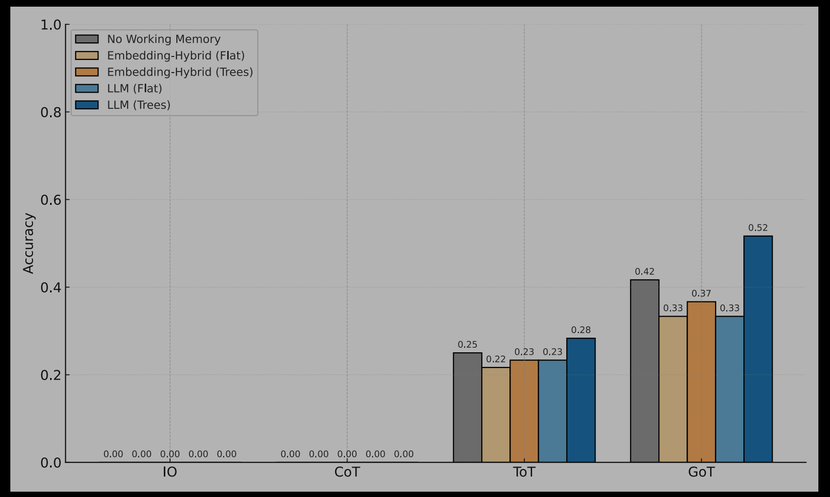
Accuracy of Majority Vote (3 votes each)





Accuracy across all votes (As before)

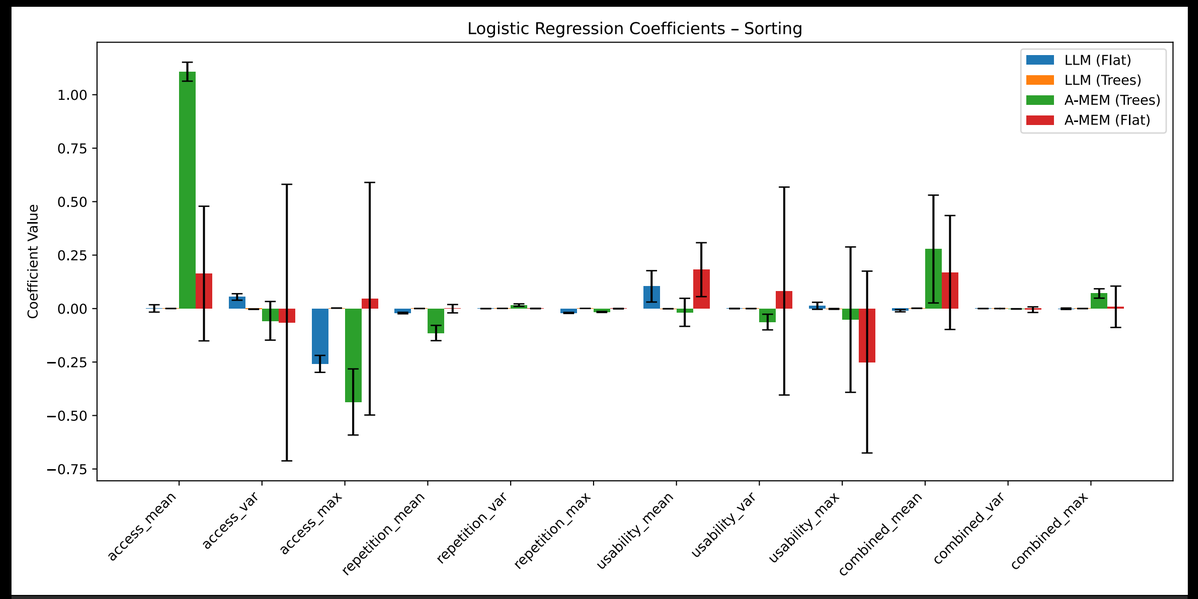


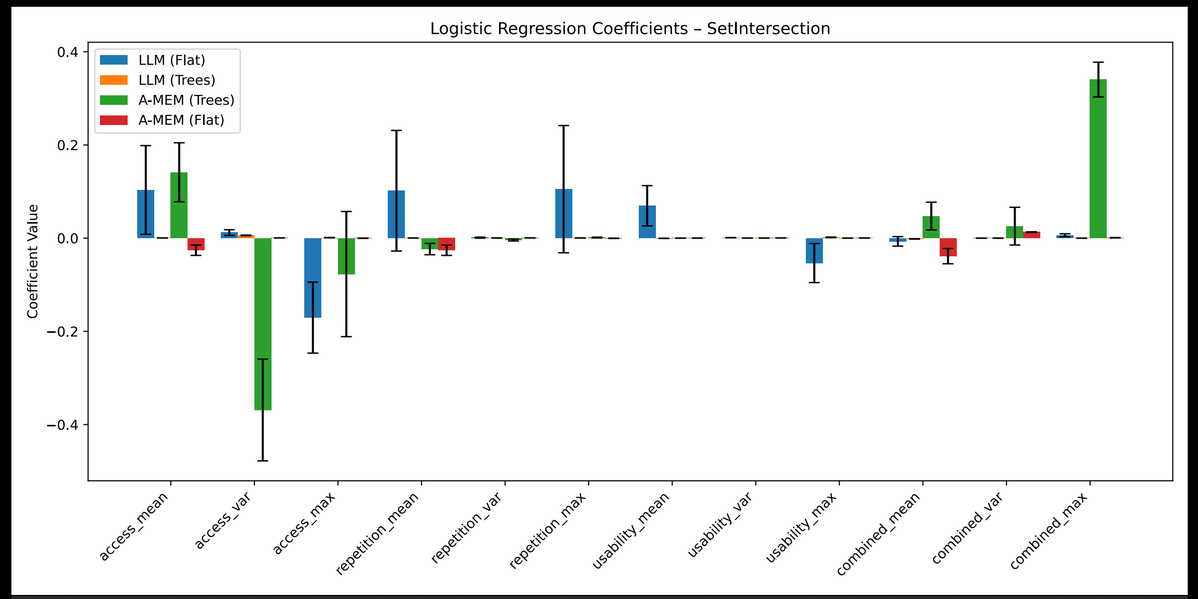


See Thesis, Subsection 5.3.2.

Minor Explorative Finding: Survival Scores as Predictor of Trial Success (1/2)

Qualitative Eval: Found **no systematic differences** in buffer contents between successful vs. unsuccessful trials
↳ Further explored: Logistic **regression** on survival scores (max/mean/std) to predict trial success



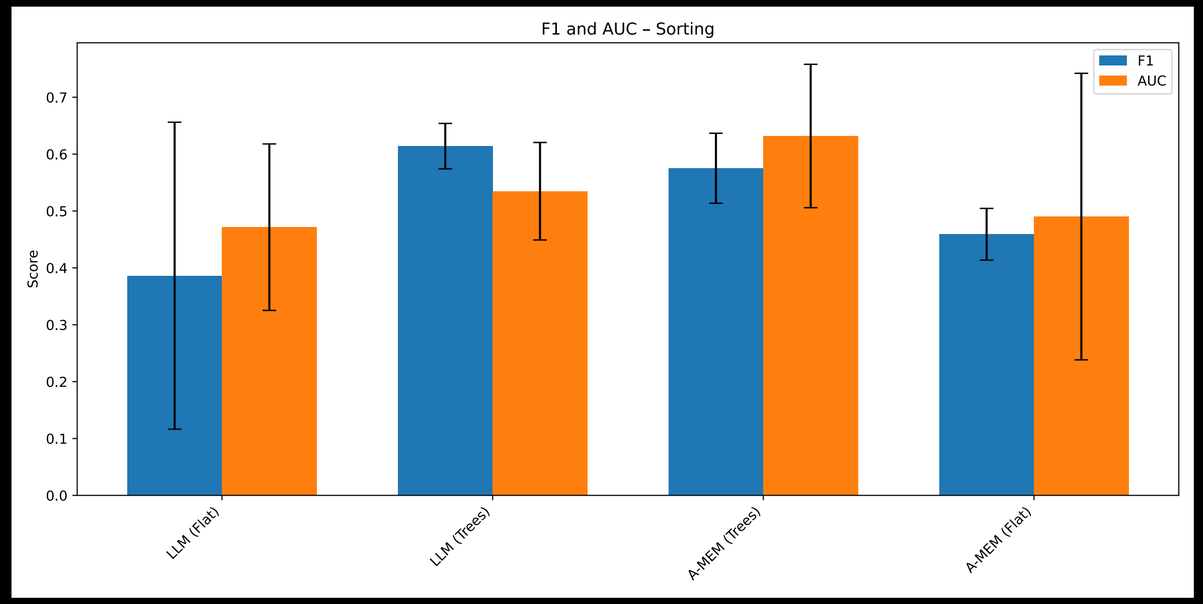


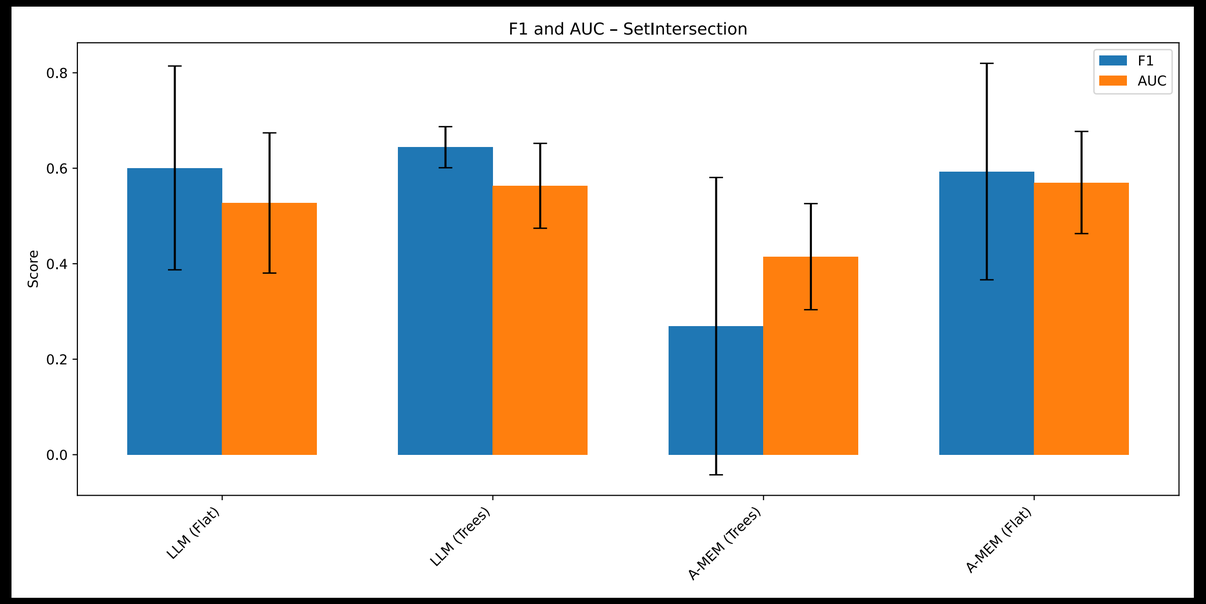
- Across variants: No consistent patterns
- **LLM-Tree**: higher coefficients across both tasks (but on different features; see green bars above)
  ↳ Suggests a **weak link** between survival-based accumulation and trial success in LLM-Tree
- Results remain **inconsistent & exploratory (no CI's or sig. tests)**

See Thesis, Subsection 5.3.3.4.

Minor Explorative Finding: Survival Scores as Predictor of Trial Success (2/2)

Contribution IV: WM Variants Evaluated within GoT | Part III: Main Findings





See Thesis, Subsection 5.3.3.4.

Further Discussion: Ideas for WM in Reasoning LLMs

Analogy: Proof-Writing

- **Humans:**
  WM balances forward planning (exploration) and backward curation (discarding dead ends, refining assumptions).
- **LLMs:**
  Externalize **all** intermediate steps → Condition final answer on "a notebook full of fragments" (noisy, uncurated).
  ↳ **What if distractions were processed to retain only informative insights, guiding the next step without distractions?**
  ↳ ***WM in LLMs*:** Act as a Central Executive, curating and planning to reduce cognitive load.

Some Challenges & Initial Ideas for WM Interventions

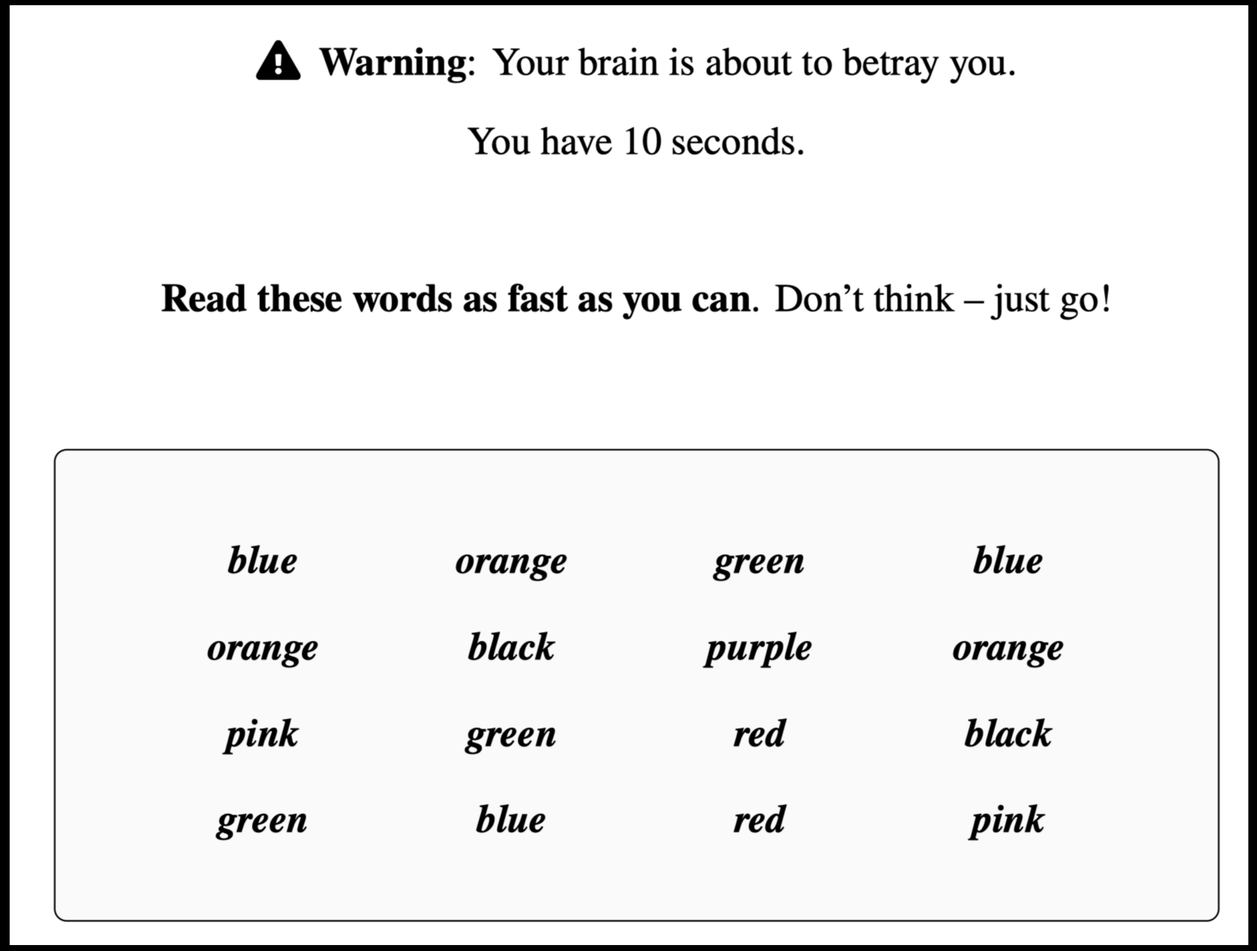
1. **Long-range retrieval accuracy**
   - Problem: lost-in-the-middle effect (Liu et al., 2023).
   - WM: Curate a smaller, focused context window; suppress irrelevant traces.
2. **Overthinking**
   - Problem: excessive, redundant reasoning steps (Sui et al., 2025).
   - WM: Detect stagnating loops or diminishing progress; dynamically constrain reasoning depth; identify completion points.
3. **Error propagation**
   - Problem: Flawed premises snowball through reasoning (Gan et al., 2025).
   - WM: Curate focused context to enable deliberate planning and exploration that challenges prior assumptions; suppress distractions from prior (potentially erroneous) steps; trigger backtracking when needed.

Possible First Step:
Probe whether attention mechanisms alone suffice for context regulation (e.g. in a proof-writing scenario)
→ Test with reasoning chains mixing relevant vs. distracting content.

See Thesis, Subsection 7.1.2.

Additional Figures not Included so Far (1/14):

Demonstration of the Stroop Effect (Stroop, 1935; 1/2)



See Thesis, Subsection 2.1.1.

Additional Figures not Included so Far (2/14):

Demonstration of the Stroop Effect (Stroop, 1935; 2/2)

Now again – but this time, don't read the words.

Say only the *color* they are displayed in.

Ready? Go!

blue	*orange*	*green*	*blue*
orange	*black*	*purple*	*orange*
pink	*green*	*red*	*black*
pink	*purple*	*purple*	*red*

See Thesis, Subsection 2.1.1.

Additional Figures not Included so Far (3/14):

Demonstration of priming

Read the following words quickly:

`fork knife plate dinner napkin eat`

Now, what is the first word that comes to mind when you see this?

`S _ _ P`

Did you say **SOUP**?

But if a different group sees this sequence:

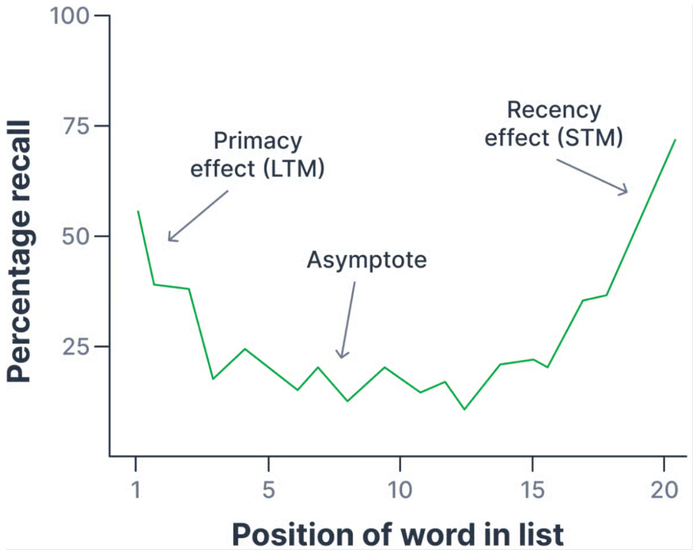
`shower towel clean brush bathe wash`

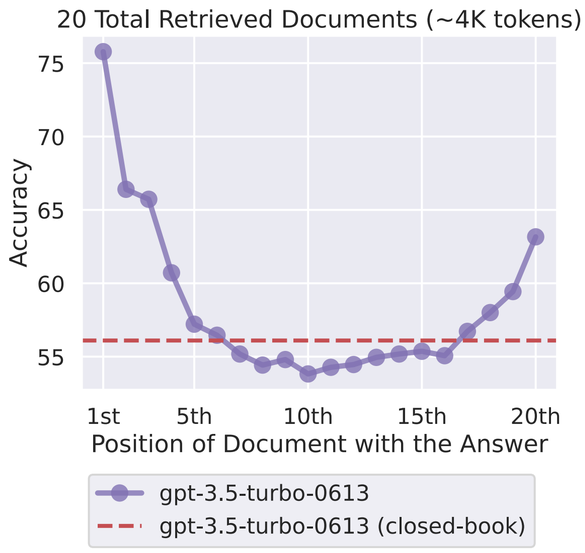
they tend to complete `S _ _ P` as **SOAP**.

See Thesis, Subsection 2.1.1.

Additional Figures not Included so Far (4/14):

Two parallel U-shaped recall patterns in human and artificial cognition.





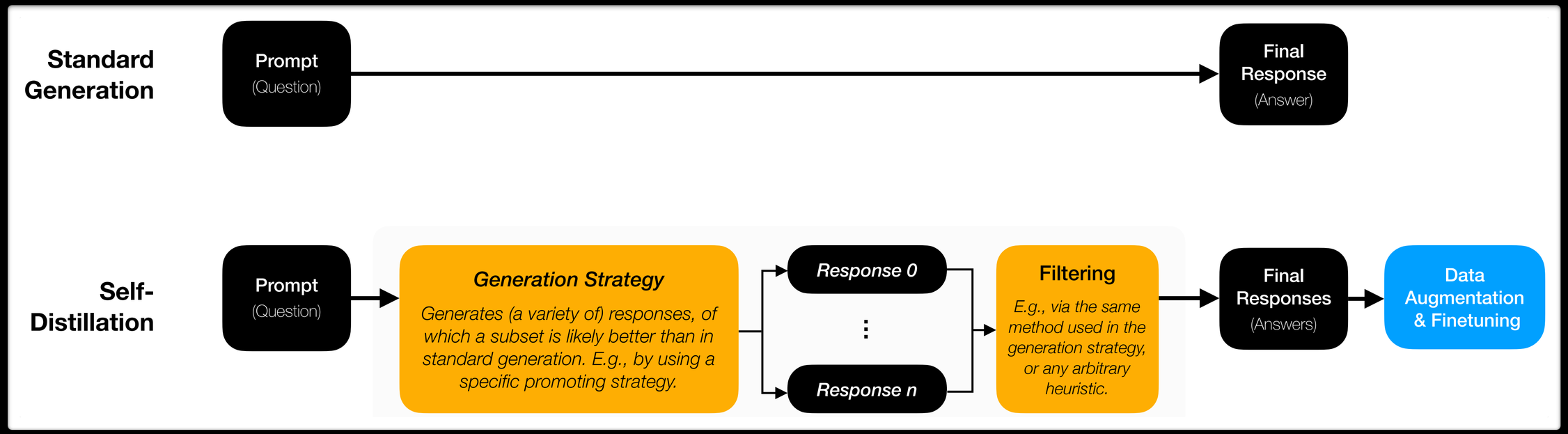
Primacy-recency effect in humans. Recall follows a U-curve, with higher accuracy for early and late items (Murdock, 1962) (LTM: Long term memory; STM: Short term memory, alias Working Memory)
Graphic adapted from: https://www.pipedrive.com/en/blog/quotes-for-email-signature.

Lost-in-the-middle effect in LLMs: Performance drops for information placed in the middle of the input context.
Adapted from Liu et al. (2023).

See Thesis, Subsection 2.1.3.

Additional Figures not Included so Far (5/14):

Conceptualization of Self-Distillation



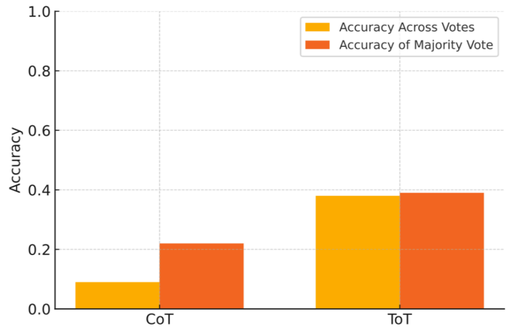
See Thesis, Subsection 3.1.1.2.

Additional Figures not Included so Far (6/14):

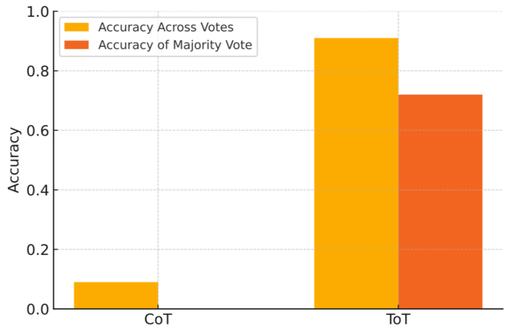
Experimental Results from Fine-tuning a LLM for Self-Distillation (1/2): Data Quality & Empirical Trends

Figure 20

Accuracy comparison of CoT and ToT during the data labeling step of Self-Distillation. We report both the average accuracy across individual generations and the final accuracy of the majority-voted label. Each method is applied to two tasks, where for each, the total number of samples labeled is $n = 80$, with majority voting size 32, yielding $80 \cdot 32 = 2.560$ individual generations per task.



(a) ***Sorting task.*** *The ToT method outperforms CoT across both metrics. However, the gap between majority-voted accuracy and accuracy across votes is marginal for ToT, but large for CoT.*

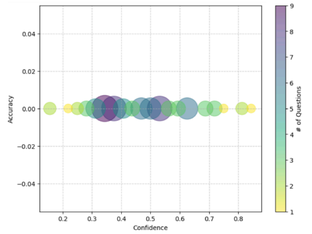


(b) ***Set-Intersection task.*** *ToT drastically improves accuracy compared to CoT. Interestingly, majority voting significantly lowers accuracy in both cases. For CoT, to* 0.0.

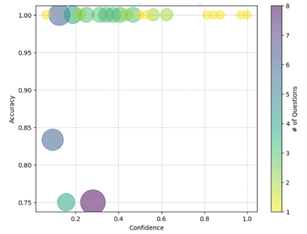
- ToT produces better labels then CoT
- Majority-Voting reduces label quality for Set-Intersection

Figure 22

Accuracy versus confidence on generated labels of the Set-Intersection training-set. Accuracy reflects the proportion of correct answers at each confidence level. Confidence denotes the fraction of generation attempts that produced the majority-voted label. For voting rounds resulting in a tie, an answer was chosen at random.



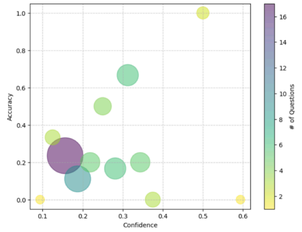
(a) *Accuracy versus confidence on Set-Intersection, with CoT. 0/80 voting rounds resulted in a tie.*



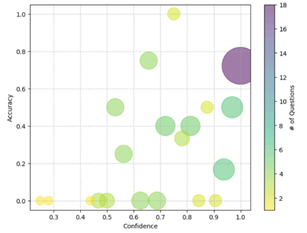
(b) *Accuracy versus confidence on Set-Intersection, with ToT. 18/80 voting rounds resulted in a tie.*

Figure 23

Accuracy versus confidence on generated labels of the Sorting training-set. Accuracy reflects the proportion of correct answers at each confidence level. Confidence denotes the fraction of generation attempts that produced the majority-voted label. For voting rounds resulting in a tie, an answer was chosen at random.



(a) *Accuracy versus confidence on Sorting, with CoT. 18/80 voting rounds resulted in a tie.*



(b) *Accuracy versus confidence on Sorting, with ToT. 0/80 voting rounds resulted in a tie.*

- Tasks seem not as well calibrated; rather undercondifent
- Weak correlation between confidence & accuracy
- Inconsistencies not pursued further

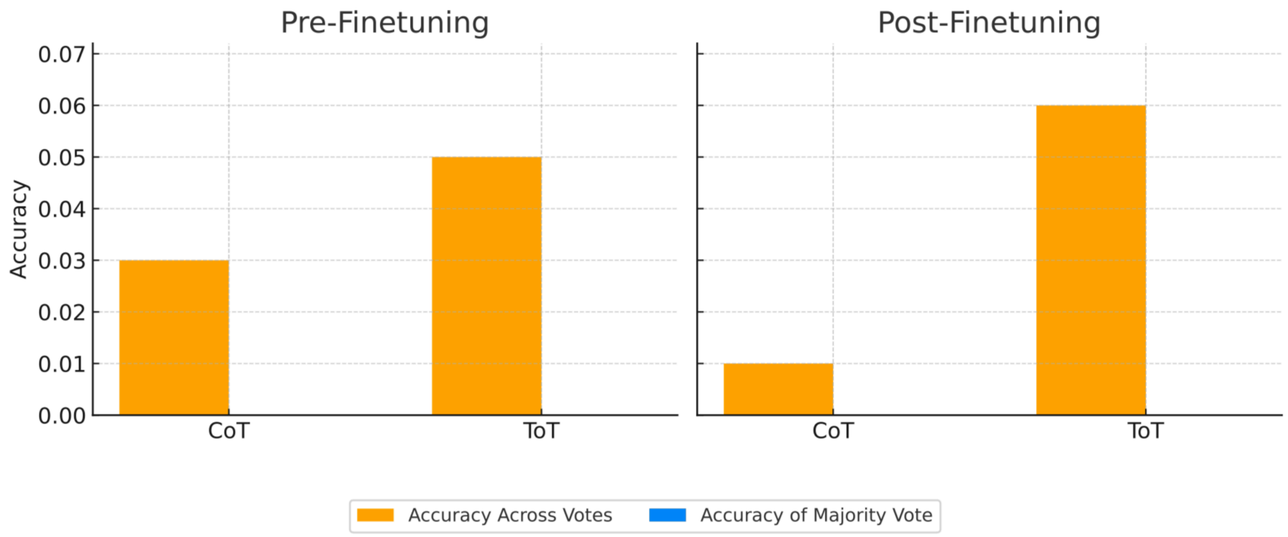
See Thesis, Subsection 5.2.2.1.

Additional Figures not Included so Far (7/14):

Experimental Results from Fine-tuning a LLM for Self-Distillation (2/2): Model Performance

Figure 24

Accuracy comparison before and after fine-tuning on the Set-Intersection task using standard prompting. "CoT" and "ToT" refer to reasoning strategies used during label generation (not inference). We report both average per-generation accuracy (Accuracy Across Votes) and majority-vote accuracy (Accuracy of Majority Vote), using 5 votes per data point (20 points total). ***All majority-vote results yielded*** 0.0% ***accuracy****. The vertical axis is capped to emphasize fine-grained differences.*



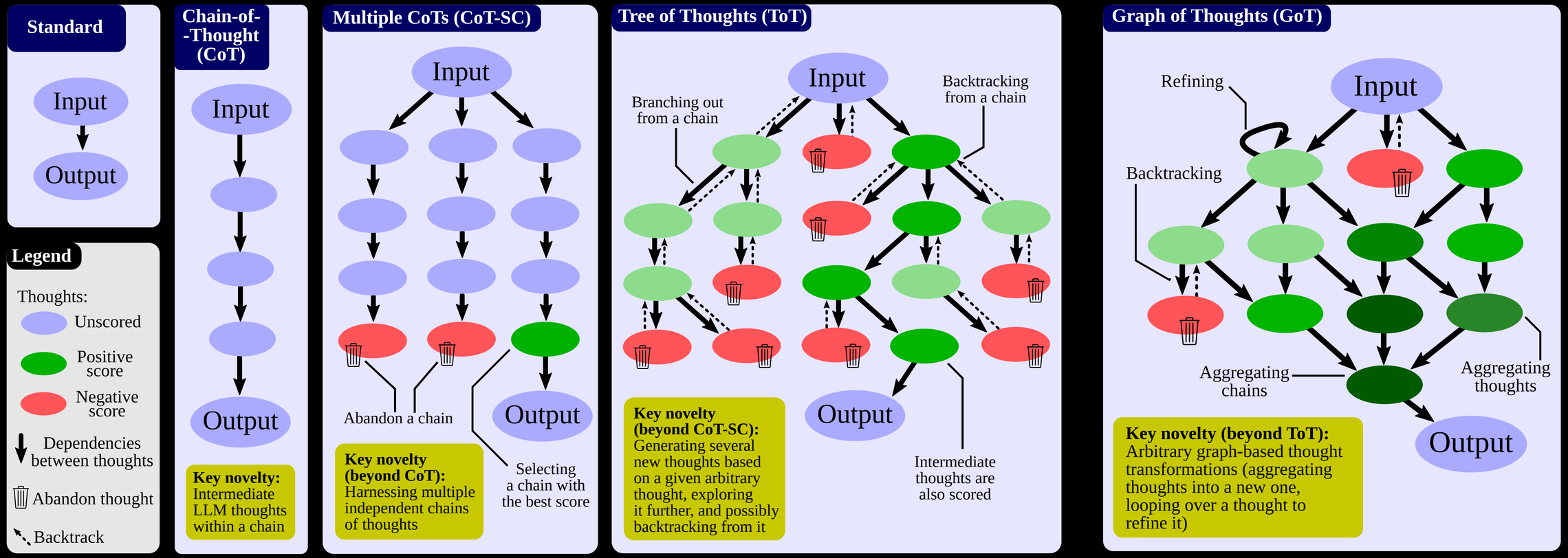
- ToT leads to superior training data over CoT, translating to model performance
- Evaluation regime with I/O prompting relatively unstable

See Thesis, Subsection 3.1.1.2.

Additional Figures not Included so Far (8/14):

Evolution of Reasoning Paradigms / Static Prompting Methods

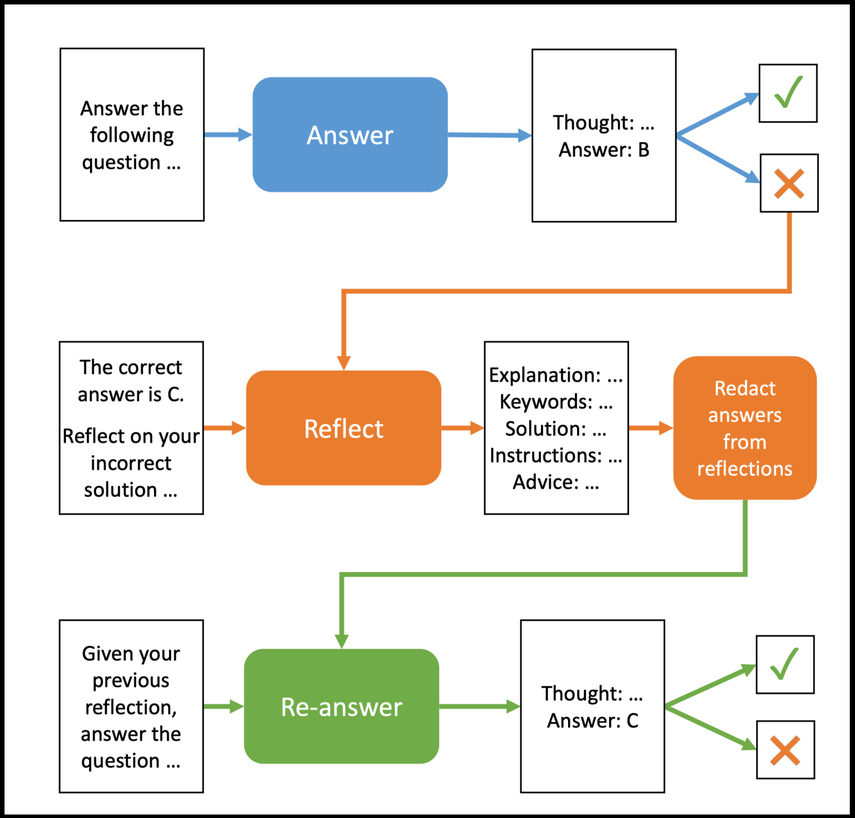
(Adapted from Besta et. Al, 2024)



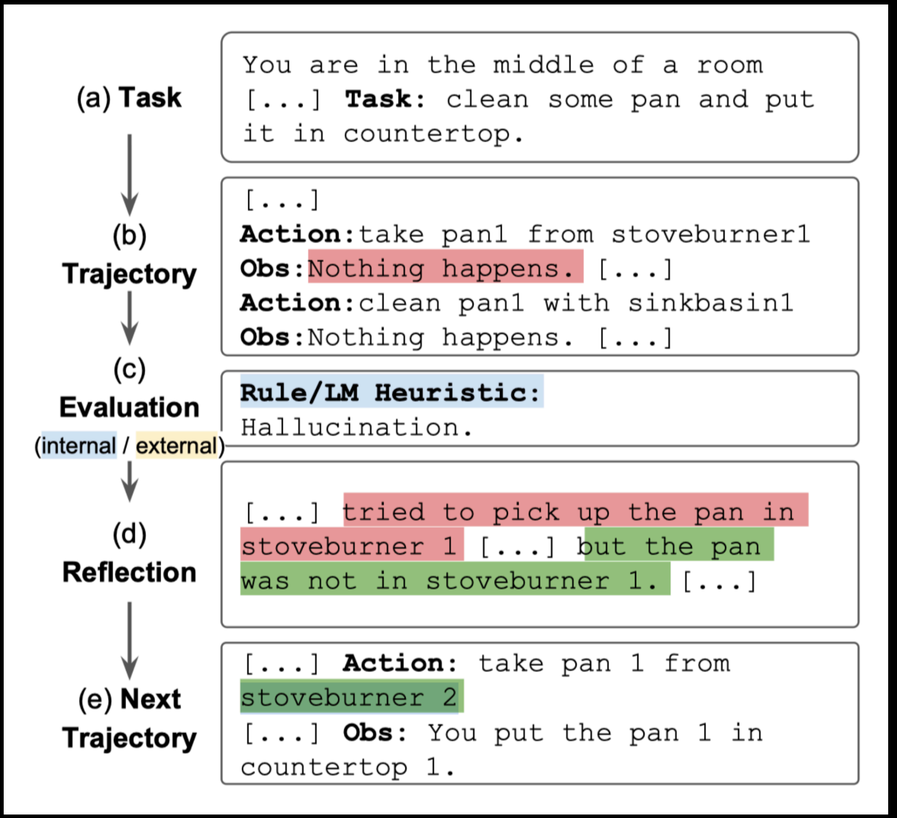
See Thesis, Subsection 3.2.1.

Additional Figures not Included so Far (9/14):

Reflection Examples form Related Work



A Self-Reflection setup in which the correct answer is used as error signal, but later redacted from the produced reflection (not depicted in the figure). Adapted from Renze and Guven (2024).

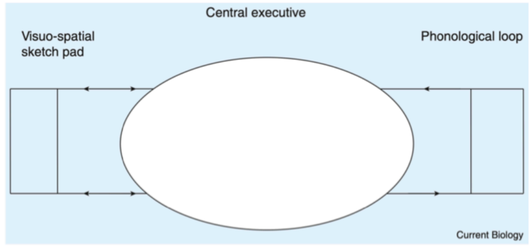


A Self-Reflection setup with variable error signal source, on the tasks of "decision making". Adapted from Shinn et al. (2023);

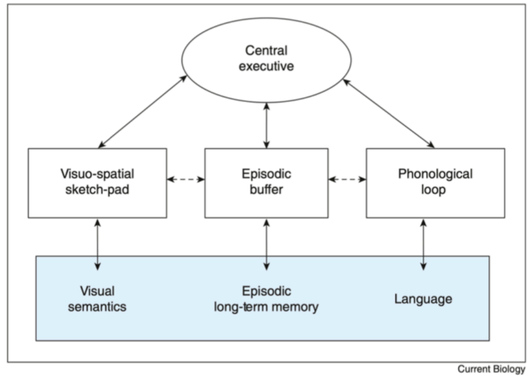
See Thesis, Subsection 3.3.2.2.

Additional Figures not Included so Far (10/14):

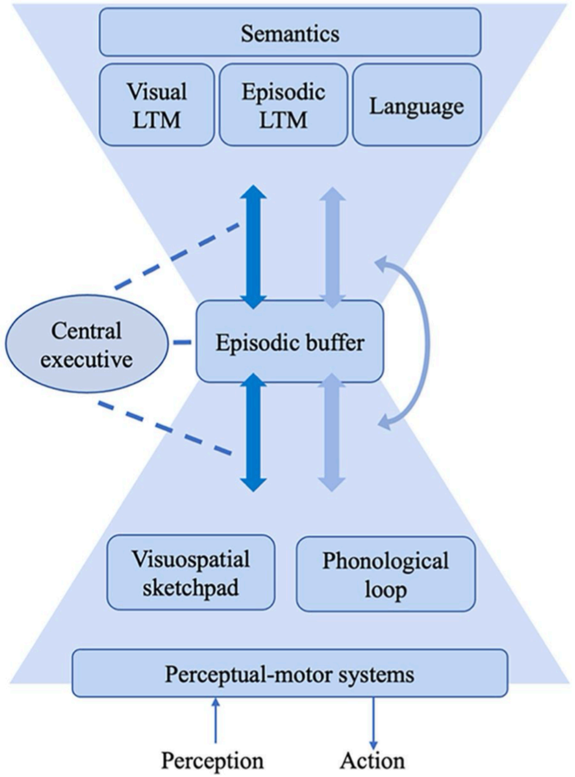
Evolution of the Multi-Component Model (MCM) of Working Memory.



(a) *MCM (1974). Adapted from Baddeley and Hitch (1974): "The earlier concept of the short-term memory has been elaborated to include an attentional controller and two modality based temporary stores. The components are assumed to interact, and to be linked to both perception and long-term memory."*



(b) *MCM (2000). Adapted from (Baddeley, 2000): "Includes links to long-term memory and a fourth component, the episodic buffer that is accessible to conscious awareness."*



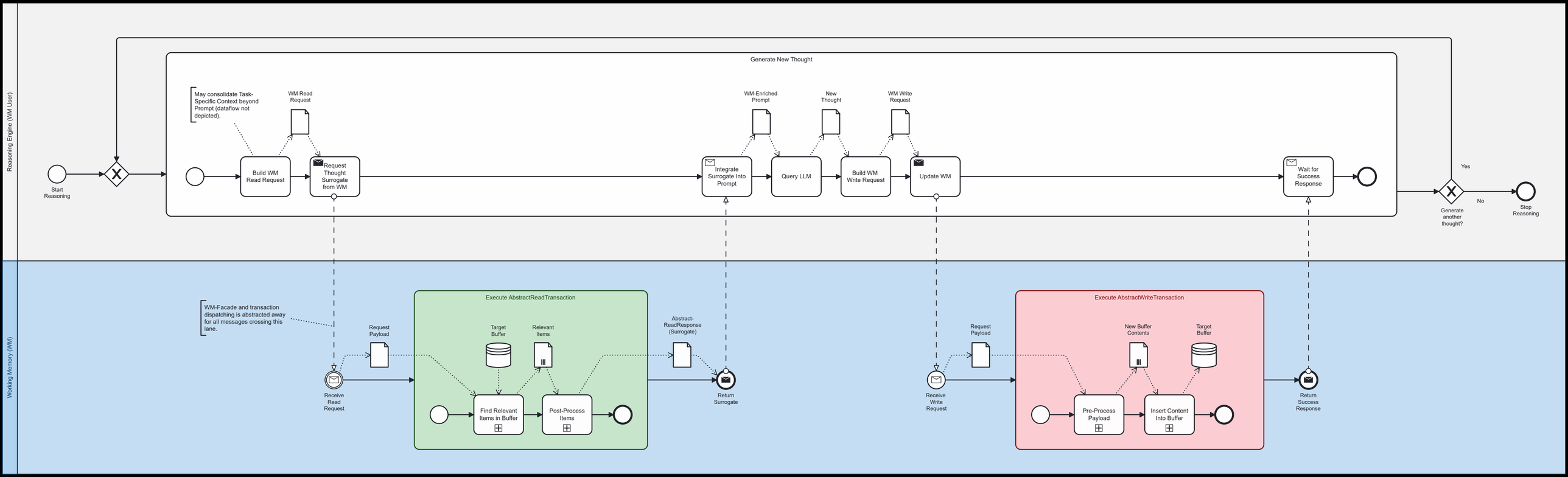
(c) *MCM (2025). Adapted from (Hitch et al., 2025): "A multicomponent view of the cognitive system with working memory located between semantic long-term memory and perceptuo-motor subsystems with the episodic buffer as the central hub. The central executive is shown as a separate resource that can be deployed in a variety of ways to interact with many parts of the system. Light and dark arrows indicate implicit and explicit processes, respectively."*

See Thesis, Subsection 3.3.1.2.

Additional Figures not Included so Far (11/14):

Full High Level WM Process Diagramm

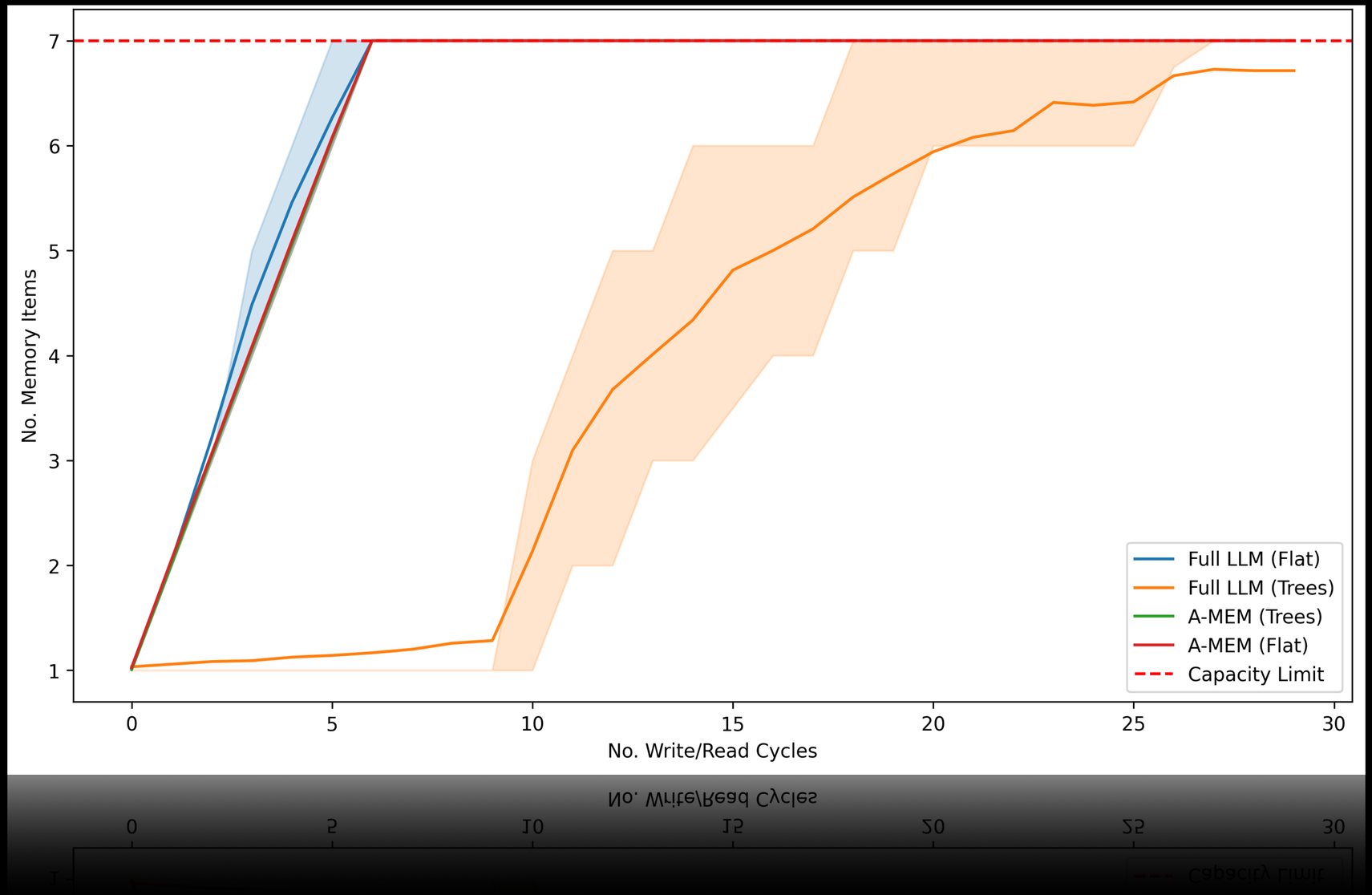
(Non-BPMN-compliant dataflows go across subprocess boundary to avoid clutter).



See Thesis, Subsection 4.3.2.

Additional Figures not Included so Far (12/14):

Memory Pressure / Capacity for the ToT Prompting Method



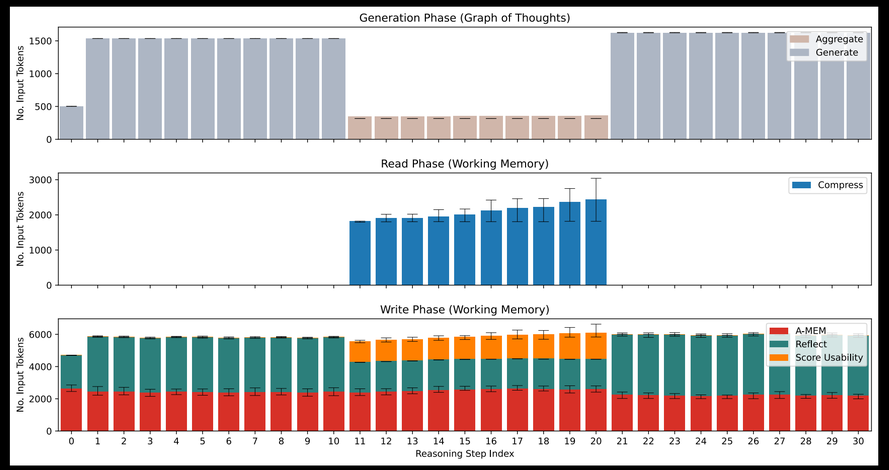
See Thesis, Subsection 5.3.2.

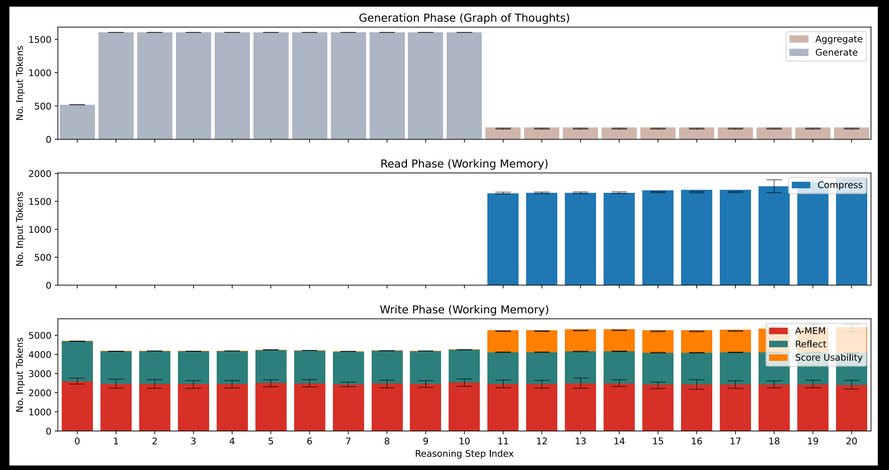
Token Usage Histograms on GoT Prompting Method — Embedding-Hybrid Variants

Sorting

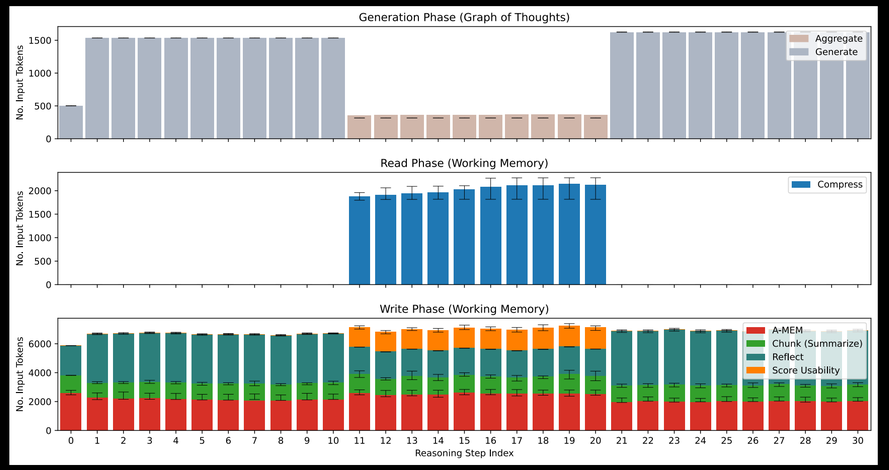
Set-Intersection

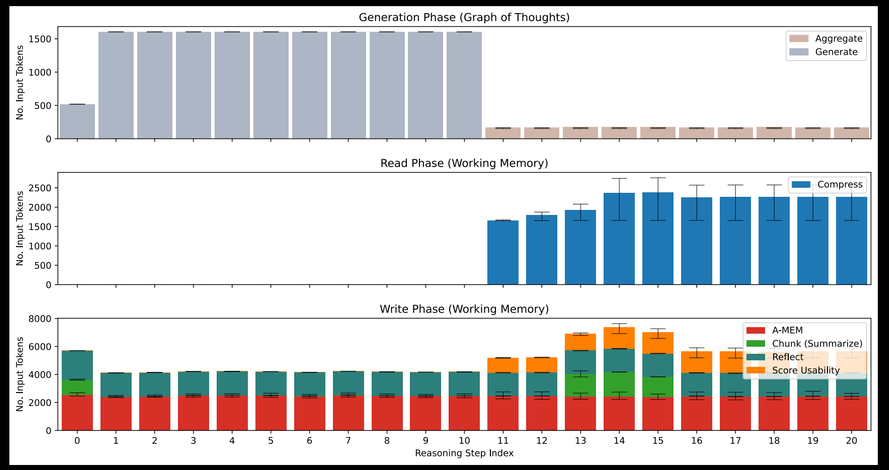
Without Chunking





With Chunking





See Thesis, Subsection 5.3.4.

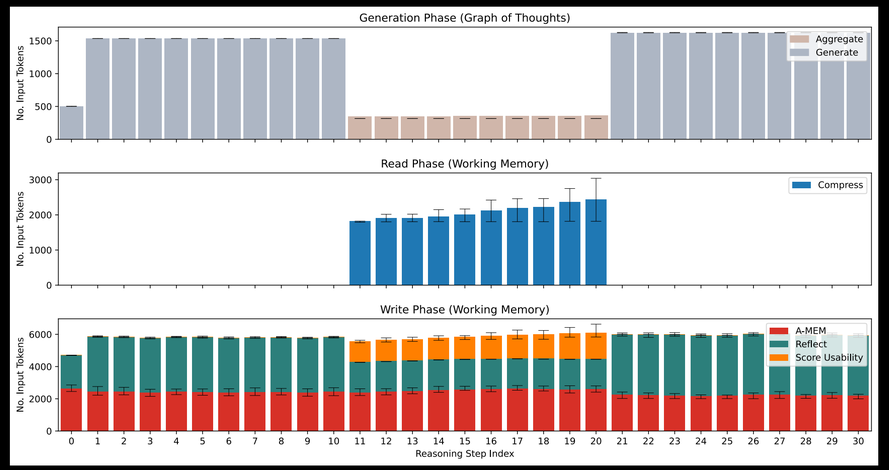
Additional Figures not Included so Far (13/14):

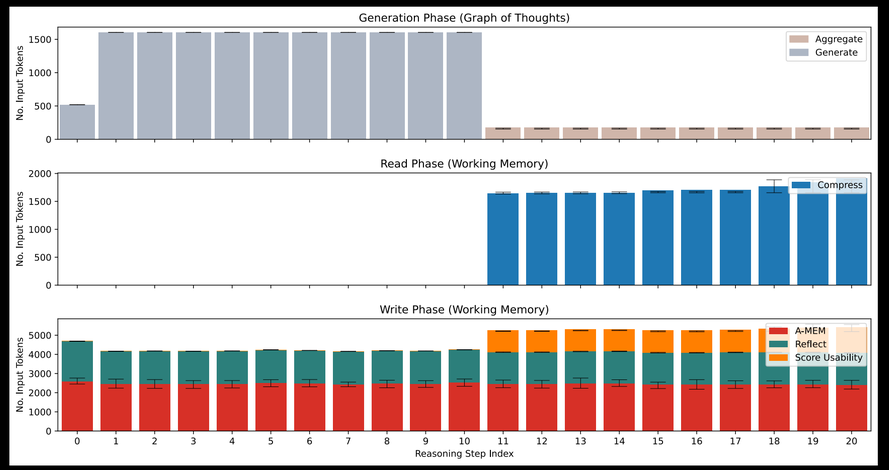
Token Usage Histograms on GoT Prompting Method — Embedding-Hybrid Variants

Sorting

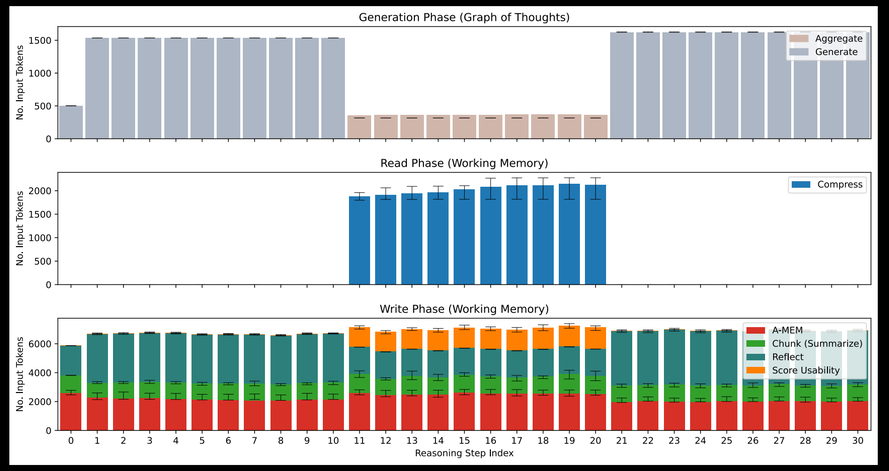
Set-Intersection

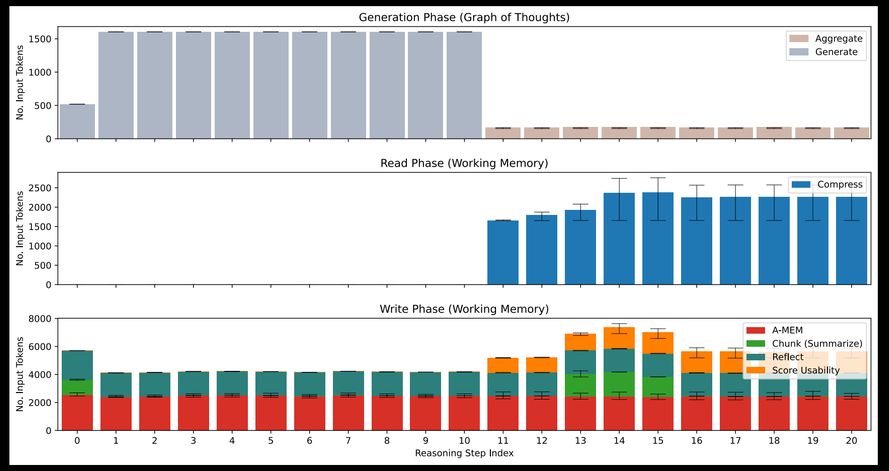
Without Chunking





With Chunking





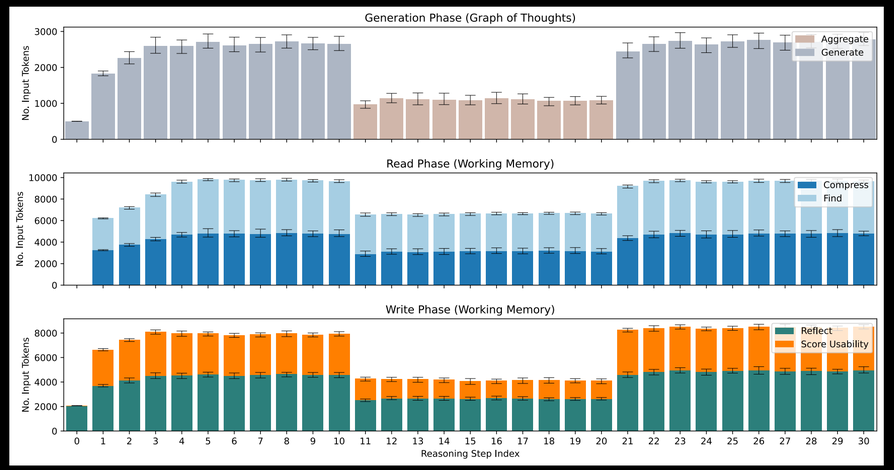
See Thesis, Subsection 5.3.4.

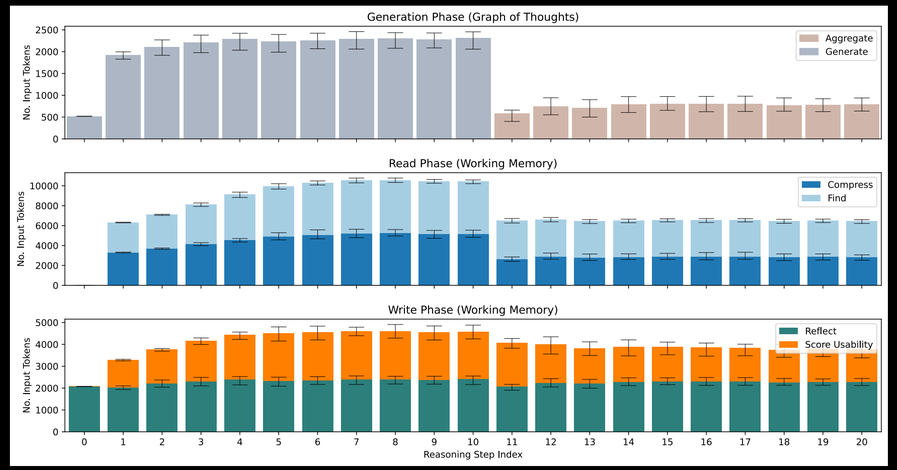
Token Usage Histograms on GoT Prompting Method — LLM-Only Variants

Sorting

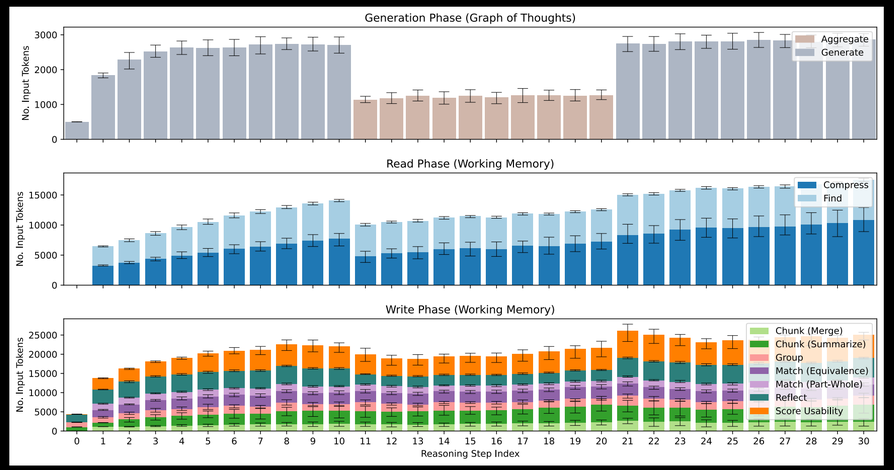
Set-Intersection

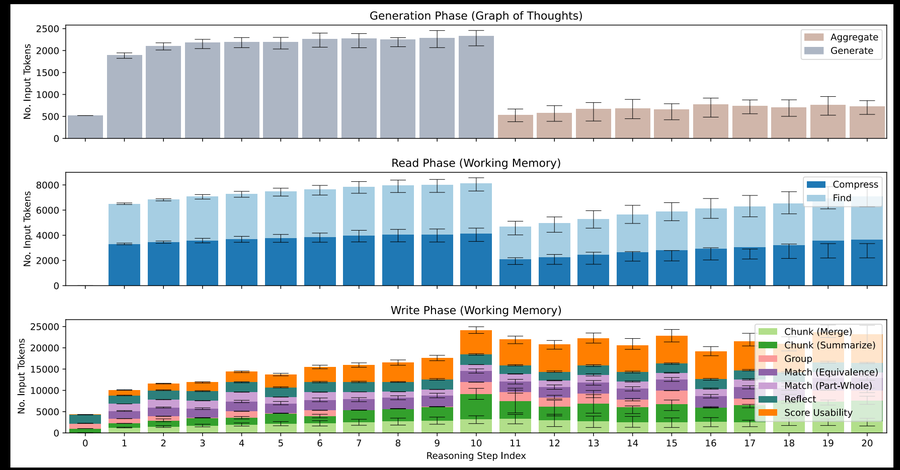
Without Chunking





With Chunking





See Thesis, Subsection 5.3.4.

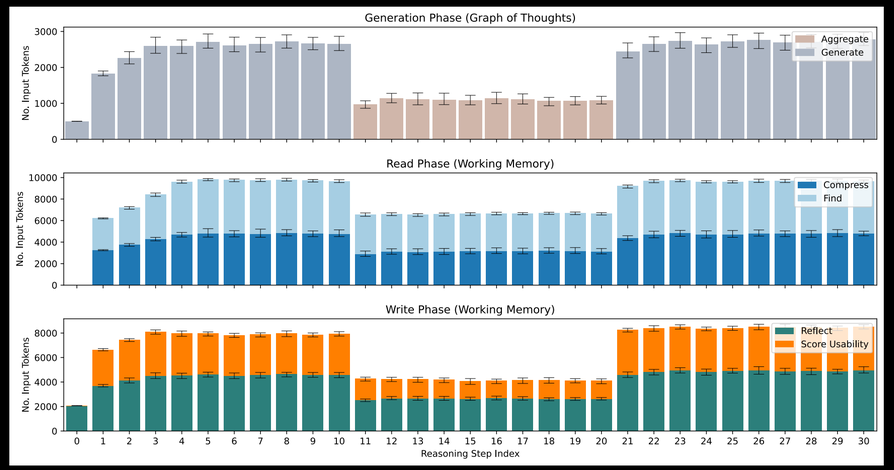
Additional Figures not Included so Far (14/14):

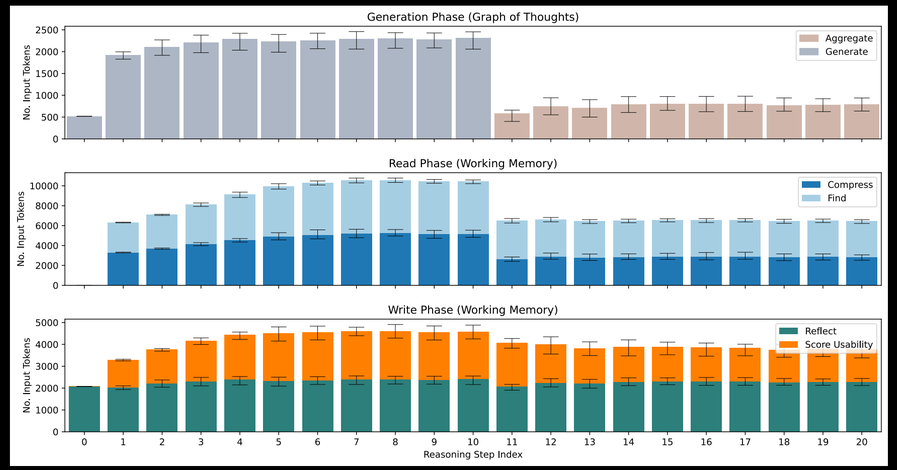
Token Usage Histograms on GoT Prompting Method — LLM-Only Variants

Sorting

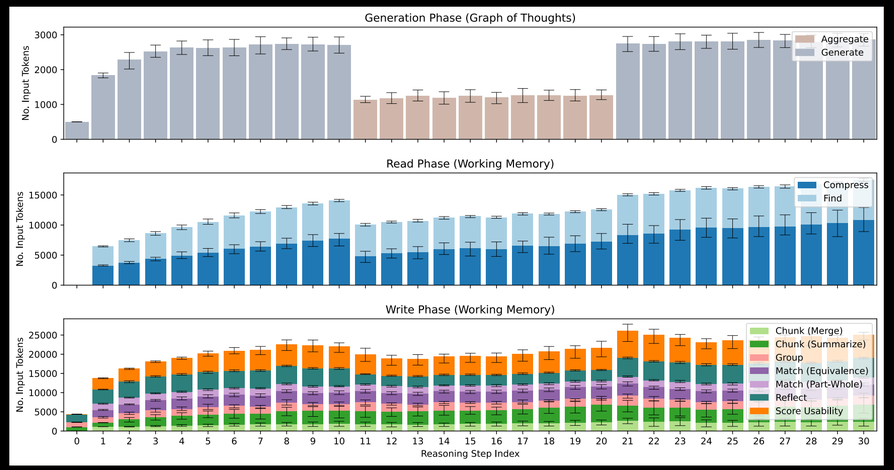
Set-Intersection

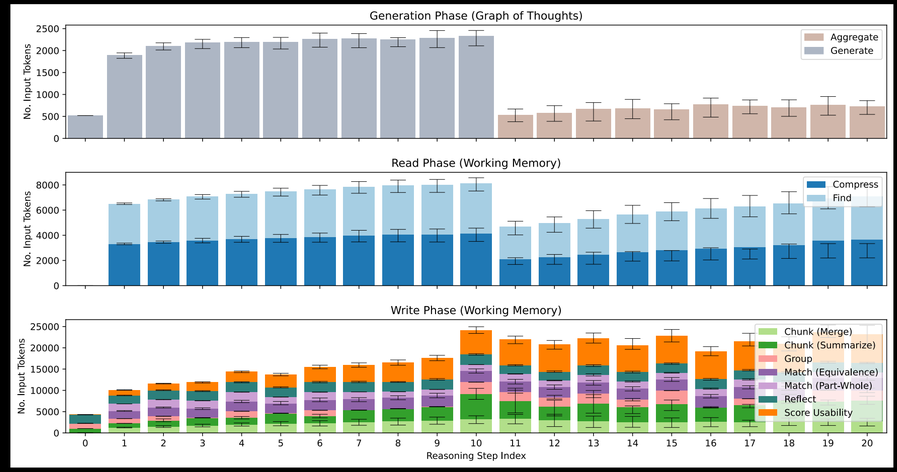
Without Chunking





With Chunking





See Thesis, Subsection 5.3.4.