Hidden Hops: When the Corpus Decides How Many Retrieval Steps a Question Needs

Tien An Vu **Lin Tian** **Marian-Andrei Rizoiu**
Behavioral Data Science Lab, University of Technology Sydney, Australia
TienAn.Vu@student.uts.edu.au
lin.tian-3@uts.edu.au Marian-Andrei.Rizoiu@uts.edu.au

Abstract

How many retrieval steps does a question need? The answer depends on what information is available as well as on the question, not just the question alone. For example, a question may ask which sport a player plays, this seems like a single hop question. However, if the database has only information about which position the player holds and which sport uses that position, answering then requires an extra step that a plan based on the question alone would miss. We call this a *hidden hop*. We compare three query procedures while keeping the retriever and reader fixed within each dataset: a single query, a fixed plan of sub-questions, and an adaptive plan that splits a failed sub-question using a fact found in its retrieved evidence. On FC-MH, the fixed plan gains 37.5 SubEM points over a single query, and adaptive plan adds 6.5 points. On MuSiQue, the fixed plan gains 4.4 F1 points, but adaptive plan gives no overall gain. Rewriting MuSiQue paragraphs as facts makes splits more frequent without a reliable F1 gain. We release our code at https://github.com/anvu2211/hidden-hop.

1 Introduction

How many hops does the question *"what sport does Christian Abbiati play?"* need? A decomposer or human would leave it as one hop. Suppose, however, that the corpus being searched never says what sport Abbiati plays. It says that he plays the position of cornerback, and elsewhere it says which sport cornerbacks play. The question now needs two hops, and nothing in its wording could have said so. The second hop is in the corpus. This is the situation of a system whose store holds one relation per record, such as an agent memory of extracted facts (Packer et al., 2023; Chhikara et al., 2025; Rasmussen et al., 2025). In our sample of MuSiQue questions, whose evidence is stored as paragraphs, we found this pattern rarely (§6.3).

Most work in multi-hop question answering treats the hop count as fixed by the question. Benchmarks compose each question from single-hop questions and record the count as an annotation (Yang et al., 2018; Trivedi et al., 2022), and retrieval-augmented systems inherit the assumption, whether they plan before retrieving (Zhou et al., 2023) or interleave retrieval with reasoning (Trivedi et al., 2023). The Abbiati example says the count depends on the question *and the corpus*, and the corpus half shows itself only after a retrieval has come back without the answer. We call the missing step a **hidden hop**. A plan written from the question alone is one step short of it, and neither a deeper pool nor a rewritten query closes the gap, because the fact the plan asks for is not there under any phrasing. The pool that lacks the sport can, however, contain the position, and that is where a repair can start (§3).

Two empirical questions follow: (1) how much of the gain attributed to decomposition comes from the corpus rather than from the question?; and (2) does a plan need to change once evidence arrives? Results from previous work cannot settle either, because they come from different evidence conditions, whether candidate paragraphs supplied with each question, a full Wikipedia dump or a purpose-built pool of facts, and changing the condition can move a number further than changing the method does (§6.5). We address both questions in this paper.

We fix one retriever and one reader per dataset. SINGLE retrieves once for the whole question. FIXED decomposes the question once and answers each hop in turn. ADAPTIVE is FIXED with one change: when the reader cannot answer a hop from that hop's own pool, the hop is split in two around a bridge fact found in the pool. The difference $\Delta_1 = \text{FIXED} - \text{SINGLE}$ measures what decomposition is worth, and $\Delta_2 = \text{ADAPTIVE} - \text{FIXED}$ what revising the plan is worth. We run all three on

FC-MH, which records one relation per line, and on MuSiQue, whose records are paragraphs that state several relations each. We also test MuSiQue questions against an atomised fact store to examine whether record granularity alone explains the difference.

Our contributions. We define the hidden hop, a corpus-induced step invisible to a planner that sees only the question, and introduce a refusal-triggered procedure that repairs it from retrieved evidence. Holding retrieval and reading fixed, we measure decomposition and repair as separate increments on FC-MH and MuSiQue. Decomposition helps on both datasets, but its gain does not grow with annotated hop count. Adaptive repair raises the mean score on FC-MH with the development reader, varies across other readers, and has no reliable overall gain on MuSiQue even after its paragraphs are rewritten as facts. We found that changing only the index more than doubles the single-query score on FC-MH, so the increments can be seen only with the retrieval side held fixed. Moreover, applied to fact-checking, we found a small and uncertain gain from supplying a verdict with question-answer pairs to a fixed set of sub-questions, while recursive repair added no measurable gain. Lastly, we release manual annotations for all 100 FC-MH questions.

2 Related Work

Work on multi-hop retrieval-augmented generation changes one of two things. The *retrieval side* is how the corpus is stored and how a query is scored against it. The *query side* is what is asked of the retriever. We hold the first fixed and vary the second. We review both, then the studies that compare query procedures over a fixed retriever.

The retrieval side. Much work improves what a single query can reach. Some of it changes the stored unit, to propositions (Chen et al., 2024) or atomic facts (Li et al., 2025); some builds structure over the corpus, as trees (Sarthi et al., 2024) or graphs (Gutiérrez et al., 2024, 2025; Wang and Han, 2025); and some retrieves passage chains directly (Xiong et al., 2020; Zhang et al., 2024; Erker et al., 2026). Related agent-memory systems organise persistent context through virtual context management, extracted memories, or temporal knowledge graphs (Packer et al., 2023; Chhikara et al., 2025; Rasmussen et al., 2025). Each of these moves the number a query-side method obtains, which is the confound we design around.

The query side. Plan-then-execute methods write sub-questions before seeing any evidence (Zhou et al., 2023; Khot et al., 2023; Zhang et al., 2023; Cao et al., 2023; Ammann et al., 2025); iterative methods choose each query from the evidence so far (Press et al., 2023; Trivedi et al., 2023; Yao et al., 2023; Jin et al., 2025); and a third group retrieves or decomposes only on a signal, such as the question's predicted complexity (Jeong et al., 2024), the model's own generation (Jiang et al., 2023; Su et al., 2024; Asai et al., 2024), or the sufficiency of what was retrieved (Li et al., 2026; Jiao et al., 2026). Closest to our adaptive plan are methods that decompose further when a step fails: ADaPT (Prasad et al., 2024), RA-ISF (Liu et al., 2024) and, concurrently with this work, Hi-Q (Kim et al., 2026), which splits a query into a dependency-ordered pair whenever the reader abstains. A critical difference over Hi-Q is that it refines the whole question along the dependencies the query expresses, whereas our procedure repairs one hop of an existing plan around a bridge fact in that hop's own pool. A hidden hop requires exactly this, as the hop is already atomic with respect to the question.

Controlled comparisons. Query procedures have been compared over a fixed retriever before, within method papers (Trivedi et al., 2023; Jeong et al., 2024; Jiang et al., 2023) and in studies that isolate decomposition (Shaikh, 2026; Ammann et al., 2025; Hu et al., 2025; Maharjan et al., 2026); Hi-Q ablates one-shot decomposition against its refinement (Kim et al., 2026). Comparisons *across* papers remain confounded by retrievers, readers and budgets (Ji et al., 2026). We add a failure mode that depends on what the corpus records, measured as two nested increments with everything else held constant, on one dataset where the failure is common and one where it is rare.

3 The Hidden Hop

3.1 Setup

A corpus C is a set of records, each a one-line fact or a paragraph. A retriever R maps a query s and a depth k to a pool $R(s, k) \subseteq C$ of k records. A reader M is a language model prompted with a query and a pool. For a question Q with gold answer y, a *query procedure* π is a program that

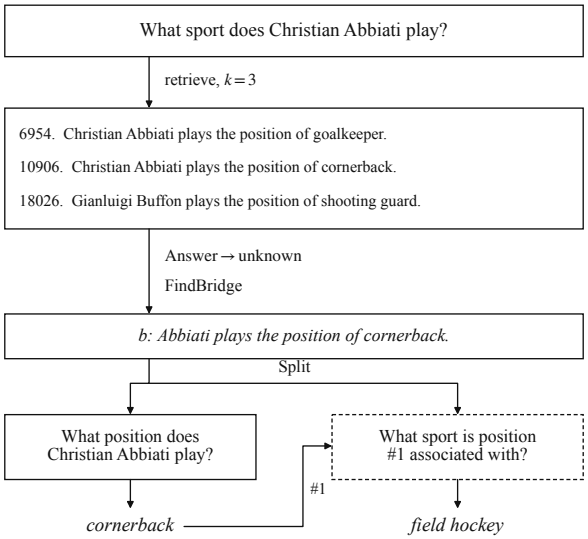

Figure 1: SOLVE on the first hop of the running example on FC-MH at k=3. The pool records Abbiati's position but no sport, so the reader refuses and the split routes the hop through the position. The dashed box marks the hidden hop.

calls R and M and returns a prediction $\hat{y}_\pi(Q)$; its cost is the number of calls it makes to M. With C, R, M and a scoring function σ held fixed, two procedures are compared over the same question set D by the paired difference

$$\Delta(\pi, \pi') = \frac{1}{|D|} \sum_{Q \in D} \big[\sigma(\hat{y}_{\pi'}(Q), y) - \sigma(\hat{y}_\pi(Q), y)\big].$$

3.2 Definition

Let q be a query put to R, which may be a whole question or one hop of a plan, and suppose it asks for relation r of entity e. A *hidden hop* exists for the pair (q, C) when C does not record $r(e)$, but there is an entity e' such that C records some relation linking e to e' and also records $r(e')$. Answering q over C then takes two retrievals even though q names one relation. We call the record linking e to e' the *bridge fact*, and e' the *bridge entity*. Because $r(e)$ is absent from C, no depth k and no rephrasing of q retrieves it; what a pool for q can contain is the bridge fact.

Table 1 gives the example from the introduction as it stands in FC-MH, a corpus of one-line facts each prefixed with a serial number, where the newer of two conflicting facts, marked by the higher serial, is the one that counts and the older is stale (§5.1). Of its 18,336 facts, exactly two mention Abbiati, and both give a playing position rather than a sport. Here r is *sport*, e is Abbiati, and the bridge fact is

Hop	Sub-question	Older fact	Newer fact
1	What position does Abbiati play?	goalkeeper (6954)	**cornerback** (10906)
2*	What sport is cornerback a position in?	Am. football (3070)	**field hockey** (7164)
3	Where did field hockey originate?	England (1923)	**Philippines** (7774)

Table 1: The evidence chain for the running example in FC-MH. Each relation appears in an older and a newer fact (serial numbers in parentheses), and the newer one counts, so following the newer facts gives the gold answer, *Philippines*. A decomposer writes hops 1 and 3 only; hop 2 (*) is the hidden hop.

"Christian Abbiati plays the position of cornerback". A decomposer reads two hops in the question, and that is a fair reading of the question; over this corpus it is one hop short. Figure 1 shows the repair that §4.1 describes.

This holds for any planner that sees only the question: a human annotator writing a gold decomposition from the question would have no reason to write more than two hops.

3.3 Hypothesis and the Two Increments

Hidden hops depend on the relations stored in a corpus. We initially expected record granularity to affect how often a repair finds a useful bridge: a one-line fact states one relation, while a paragraph can state several. This predicts different split rates and repair gains on FC-MH and MuSiQue. The datasets also differ in construction, so their comparison cannot isolate record granularity. We examine that explanation further by rewriting MuSiQue paragraphs as facts (§6.6).

To test both predictions we compare three query procedures (§4) that differ only in what they ask of R and M: a single query π_{single}, a fixed plan π_{fixed}, and an adaptive plan π_{adapt} that can revise a hop after seeing its pool. We report two increments, each a paired difference $\Delta(\pi, \pi')$ as defined in §3.1. The first, $\Delta_1 = \text{FIXED} - \text{SINGLE}$, asks whether decomposing helps at all. The second, $\Delta_2 = \text{ADAPTIVE} - \text{FIXED}$, asks whether the plan needs to be revisable once evidence arrives, which is where a hidden hop, if present, shows itself. §6 reports them.

Algorithm 1 The adaptive plan. DECOMPOSE, ANSWER, COMMIT, FINDBRIDGE and SPLIT are calls to the language model M, each with its own prompt (Appendix A).

Require: question Q, retriever R, depth k, cap $d_{\max}$

1: $[s_1, \ldots, s_n] \leftarrow$ DECOMPOSE(Q)
2: **return** SOLVESEQ$([s_1, \ldots, s_n],\ 0)$

3: **function** SOLVESEQ$([s_1, \ldots, s_n],\ d)$
4: **for** $j = 1, \ldots, n$ **do**
5: fill placeholders in s_j from $a_1, \ldots, a_{j-1}$
6: $a_j \leftarrow$ SOLVE$(s_j,\ d)$
7: **end for**
8: **return** a_n
9: **end function**

10: **function** SOLVE$(s,\ d)$
11: $P \leftarrow R(s, k)$
12: **if** $d \geq d_{\max}$ **then return** COMMIT(s, P)
13: $a \leftarrow$ ANSWER(s, P)
14: **if** $a \neq$ UNKNOWN **then return** a
15: $b \leftarrow$ FINDBRIDGE(s, P)
16: **if** $b =$ NONE **then return** COMMIT(s, P)
17: $(s_1, s_2) \leftarrow$ SPLIT(s, b)
18: **if** split unusable **then return** COMMIT(s, P)
19: **return** SOLVESEQ$([s_1, s_2],\ d{+}1)$
20: **end function**

4 Query Procedures

Algorithm 1 gives the adaptive plan. The other two procedures are special cases of it, so we describe the algorithm once and then say what each procedure keeps. Five of the functions in the algorithm are single calls to the language model M, each with its own prompt, given in full in Appendix A. DECOMPOSE writes a question as an ordered list of sub-questions. ANSWER answers a sub-question from a pool or returns UNKNOWN. COMMIT answers from a pool without the option to refuse. FINDBRIDGE names a bridge fact in a pool, and SPLIT rewrites a sub-question around one. The retriever R and the depth k are those of §5.

4.1 The Adaptive Plan

Line 1 decomposes the question into sub-questions $s_1, \ldots, s_n$, where s_j may contain the placeholder #i to refer to the answer of an earlier sub-question s_i ($i < j$). SOLVESEQ (lines 3–9) then works through them in order: it fills each placeholder from the earlier answers, solves the sub-question, and returns the last answer as the prediction.

SOLVE (lines 10–20) handles one sub-question. It retrieves a pool $P = R(s, k)$ (line 11) and asks the reader to answer from it (line 13). The reader may answer or return UNKNOWN. An answer ends the call (line 14). A refusal is the trigger of the adaptive plan, and it starts a repair that uses only the pool already retrieved. FINDBRIDGE (line 15) names a fact in P about an entity that has to be resolved before s can be, or returns NONE; this is the bridge fact of §3.2. SPLIT (line 17) rewrites s as two sub-questions, the first answerable from the bridge fact and the second referring to the first's answer by #1. The two re-enter SOLVESEQ one level deeper (line 19), so a sub-question that is still unanswerable can be split again, up to the cap $d_{\max} = 2$ (line 12). Whenever the repair cannot proceed, because the cap is reached, no bridge is found, or the split is unusable (lines 12, 16 and 18), COMMIT gives the best answer the pool supports. A split is unusable if it cannot be parsed, if its second sub-question does not refer to the first, or if its first sub-question only restates s. Figure 1 traces SOLVE on the first hop of the running example, and Table 13 gives three full runs.

Note that the trigger is a refusal by the reader on a pool it has already been given. No retrieval score or reader confidence enters into it, and nothing is judged about the question on its own. Two consequences follow for §6.2. The procedure can only help where a refusal is followed by a usable bridge, and it can only hurt where the reader refuses a sub-question that was answerable.

4.2 The Three Procedures

SINGLE skips line 1. It retrieves one pool for the whole question and calls COMMIT(Single) on it, at a cost of one retrieval and one reader call. FIXED keeps line 1 and SOLVESEQ, but replaces lines 12–19 of SOLVE with a single call to COMMIT(Fixed), so every sub-question is answered from its own pool without the option to refuse, at a cost of one decomposer call and one reader call per sub-question. ADAPTIVE is Algorithm 1 as written. Its cost varies by question: a sub-question answered at line 14 costs one reader call, and one that is split costs three, for ANSWER, FINDBRIDGE and SPLIT, plus the cost of its two replacements. The three procedures share the same retriever, decomposer and

reader.

5 Experimental Setup

5.1 Datasets

We use two datasets, FC-MH and MuSiQue. They differ in what a record is, a one-line fact or a paragraph, and in what retrieval has to do: on FC-MH a query is scored against all 18,336 facts, whereas on MuSiQue it ranks the twenty paragraphs supplied with the question.

FC-MH is the multi-hop conflict split of Fact-Consolidation in MemoryAgentBench (Hu et al., 2026), at its 262K-token setting: 18,336 one-line facts and 100 questions. The corpus is built to contradict itself. Every fact carries a serial number, a later fact supersedes an earlier one with the same subject and relation, and a system is expected to prefer the newer (Table 1). Its entities are counterfactual by construction, so an answer that is true of the real world is usually wrong here, which limits what a reader can gain from parametric knowledge. A prediction is scored by SubEM, which counts it correct if the gold string appears in it.

MuSiQue (Trivedi et al., 2022) contributes all 2,417 questions of its answerable development split. Each question ships with up to twenty candidate paragraphs and an annotated hop count of two to four. A prediction is scored by answer F1 against the gold answer and its aliases.

5.2 Retrieval and Reading

Within a dataset, the three procedures share one retriever, one reader and one set of prompts, and differ only in what they ask (§4).

Retriever. On FC-MH we index one fact per record, embed each with `text-embedding-3-large`, and score by cosine similarity. We do not chunk: a 512-token chunk of this corpus would bundle around thirty unrelated facts and dilute the embedding. As facts go stale and newer ones win, the score carries a recency term. For a query s and a fact f with serial number n_f,

$$\text{score}(s, f) = \text{sim}(s, f) \cdot \exp\big(-\lambda \cdot \text{age}(f)\big),$$
$$\text{age}(f) = (n_{\max} - n_f) \,/\, n_{\max},$$

where $n_{\max}$ is the largest serial in the corpus and $\lambda = 0.5$, so the oldest fact keeps about 61% of its similarity. The pool is shown to the reader in serial order. On MuSiQue there is no index: the retriever is the `roberta-large` cross-encoder paragraph selector released with the dataset (Trivedi et al., 2022), which ranks each question's own candidates.

Reader. Every call in the main comparison uses `gpt-4o-mini` (snapshot `2024-07-18`) at temperature 0 with a fixed seed, and every prompt instructs the model not to answer from world knowledge.[1] The decomposer is the same model. §6.4 repeats the FC-MH comparison with three further readers.

Runs and tests. Each procedure is run at least twice per dataset (Appendix C.5 gives every run), and scores are the mean ± standard deviation over runs, with each question's score averaged over runs before an increment is computed. The 95% CI of an increment is a paired bootstrap over questions. On FC-MH, where SubEM is binary, p is from an exact McNemar test on the least favourable pair of runs; on MuSiQue, whose F1 is not binary, p is from a paired sign-flip permutation test. We also report the mean number of reader calls per question (Calls/q) and the percentage of questions on which a rewrite or a split was executed (Action %).

Depth. Every query a procedure issues, whether the whole question or a sub-question, is retrieved at the same depth k. Depth is the one parameter we do not hold fixed across procedures, as it is a property of the procedure rather than of the retrieval side: a single query has to cover the whole evidence chain in one pool, while a sub-question asks about one relation, for which a tight pool can be enough. We ran every procedure at $k \in \{3, 5, 10\}$ on both datasets and report each at its best depth, which is k=10 for SINGLE and FIXED and k=5 for ADAPTIVE on MuSiQue, and k=10 for SINGLE and k=3 for FIXED and ADAPTIVE on FC-MH (Appendix C.2).

6 Results and Discussion

Tables 2 and 3 compare the three procedures on FC-MH and MuSiQue. We examine decomposition, adaptive repair, hidden-hop annotations, reader variation, and the atomised MuSiQue below.

6.1 Effect of Decomposition (Δ_1)

Looking at Tables 2 and 3, both plans FIXED and ADAPTIVE with decomposition outperform the sin-

[1] Output is capped at 20 tokens for an answer, 96 for FINDBRIDGE, 128 for SPLIT and 256 for the decomposer.

Reader	Procedure	k	SubEM	Δ	95% CI	p	Calls/q	Action %
gpt-4o-mini*	SINGLE	10	11.5 ±0.7	—	—	—	1.0	—
	FIXED	3	49.0 ±1.4	+37.5	[26.5, 48.5]	$< 10^{-7}$	3.3	—
	FIXED + rewrite	3	48.5 ±0.7	−0.5	—	—	6.2	46.5
	ADAPTIVE	3	**55.5** ±1.7	+6.5	[0.8, 12.5]	0.58	5.5	25.5
gpt-5.6-luna	SINGLE	10	13.5 ±0.7	—	—	—	1.0	—
	FIXED	10	54.5 ±2.1	+41.0	[29.5, 52.0]	$< 10^{-7}$	3.7	—
	ADAPTIVE	5	**58.0**	+3.5	[−2.0, 9.5]	0.75	5.4	18.0
Qwen3.5-35B-A3B	SINGLE	3	9.0 ±0.0	—	—	—	1.0	—
	FIXED	3	**49.5** ±2.1	+40.5	[29.0, 52.0]	$< 10^{-7}$	3.5	—
	ADAPTIVE	3	48.0	−1.5	[−6.5, 3.0]	1.00	5.0	9.0
Gemma-4-26B-A4B-it	SINGLE	5	10.0 ±0.0	—	—	—	1.0	—
	FIXED	5	54.5 ±0.7	+44.5	[33.0, 55.5]	$< 10^{-8}$	3.5	—
	ADAPTIVE	5	**56.0**	+1.5	[0.0, 4.0]	1.00	4.5	5.0

Table 2: HIDDEN HOP on FC-MH over all 100 questions for four readers, each procedure at its best depth. Δ is relative to SINGLE for FIXED, and to FIXED for FIXED + rewrite and ADAPTIVE. Bare scores are single runs, and boldface marks the best procedure for each reader. * marks the reader the prompts were developed with.

Procedure	k	F1	Δ	95% CI	p	Calls/q	Action %
SINGLE	10	52.99 ±0.11	—	—	—	1.0	—
FIXED	10	**57.40** ±0.23	+4.41	[2.47, 6.30]	$< 10^{-3}$	3.6	—
ADAPTIVE	5	57.13 ±0.26	−0.28	[−1.53, 0.97]	0.66	5.3	3.2

Table 3: HIDDEN HOP on the full MuSiQue development split of 2,417 questions with `gpt-4o-mini`, each procedure at its best depth. Δ is relative to the preceding procedure, and boldface marks the best procedure.

gle query SINGLE on both datasets.

FC-MH. FIXED substantially outperforms SINGLE (49.0 against 11.5 SubEM). The failure that decomposition repairs is in the retrieved pool.

MuSiQue. The increment is +4.4 F1 (57.4 against 53.0). This is a much smaller margin, but it is measured over all 2,417 questions rather than a sample of 100. Note that MuSiQue has only twenty candidate paragraphs per question, so a single query at k=10 already sees half of them, and decomposition has less to add. Table 4 breaks the result down by annotated hop count. SINGLE is flat across hop counts (52.9, 53.3 and 52.6 F1). The gain from decomposition, however, falls with depth: +10.5 F1 on 2-hop questions, +1.7 on 3-hop questions and −9.3 on 4-hop questions. This suggests that, over a small candidate set, errors accumulated across longer plans outweigh the limited retrieval benefit of additional hops.

6.2 Effect of Adaptivity (Δ_2)

§3.3 predicts that splits should be far more common over atomic facts than over paragraphs, however, Δ_2 is not necessary needed to follow it (Appendix C.3).

	2-hop	3-hop	4-hop	All
n	1,252	760	405	2,417
SINGLE	52.9	53.3	**52.6**	53.0
FIXED	63.4	**55.0**	43.3	**57.4**
ADAPTIVE	**63.5**	53.0	45.1	57.1
Δ_1	+10.5	+1.7	−9.3	+4.4
Δ_2	+0.1	−2.1	+1.8	−0.3
Δ_2 split fired	−11.3	+5.9	+13.6	+0.5
Δ_2 split not fired	+0.4	−2.3	+1.3	−0.3
Split %	2.6	3.4	4.7	3.2
Refusal %	39.3	61.6	82.5	53.5

Table 4: MuSiQue F1 by annotated hop count. Refusal % and Split % are the percentage of questions with at least one refusal and at least one executed split. Boldface marks the best procedure in each column.

Split rate. On FC-MH a split was executed on 25.5% of questions on average, against 3.2% on MuSiQue, which is the order §3.3 predicts. Note that on MuSiQue the reader refuses far more often than it splits: it returned UNKNOWN on at least one hop of 53.5% of questions (Table 4), but a bridge fact was found and a usable split written on 3.2%. The refusal rate doubles with the annotated hop count, from 39.3% on 2-hop questions to 82.5% on 4-hop questions, while the split rate barely moves (2.6% to 4.7%), so on paragraphs a refusal is common and a usable bridge is rare.

Gain from adaptivity. On FC-MH, ADAPTIVE improves over FIXED by 6.5 SubEM, for 2.2 extra reader calls per question. On MuSiQue, however, the score moves by −0.28 F1 for 1.7 extra calls, which is indistinguishable from zero. In this original two-dataset comparison, the larger gain

coincides with the higher split rate. The atomised MuSiQue experiment in §6.6 shows that frequent splits alone do not ensure a gain. The overall zero on MuSiQue hides a pattern by depth. We condition each ADAPTIVE run on the questions where a split was executed (66, 51 and 38 instances for 2, 3 and 4 hops) and compare with the run-averaged FIXED score (Table 4, last two Δ_2 rows). A split is worth $+13.6$ F1 on 4-hop questions and $+5.9$ on 3-hop questions, but costs 11.3 on 2-hop questions, and the two cancel over the full development set, suggesting that a split pays off where the question is deep enough for a hop to be missing and costs where it replaces an answerable hop with two.

Depth and rewriting. A gain from splitting a refused hop could also come from a shortfall in retrieval or in the phrasing of the hop, and we test both on FC-MH. Raising the depth of FIXED from k=3 to k=10 lowers SubEM from 49.0 to 42.0, and that of ADAPTIVE from 55.5 to 46.2 (Appendix C.2), suggesting that a larger pool adds distractors and not the missing fact. If the hop were badly phrased, would rewriting it recover it? The FIXED + rewrite row of Table 2, which replaces the split with a rewrite of the refused hop and retrieves again. It performs no better than FIXED, whereas the split improves on FIXED by 6.5 points over the same questions, so a refused hop is not a formulation.

6.3 Hidden-Hop Annotation and the Trigger

We annotated all 100 FC-MH questions and a 100-question sample of MuSiQue for hidden hops, details in Appendix B. 17 of the 100 FC-MH questions contain a hidden hop, while it is only 1 of the 100 for MuSiQue.

Table 5 compares the annotation with the split decisions of ADAPTIVE at k=3. As the trigger varies between runs, we pool over four runs, which gives 400 question–run decisions. A split was executed on 102 of them, 31 on an annotated hidden-hop question, so the precision of the trigger is 30.4%; it fired on 31 of the 68 hidden-hop opportunities, a recall of 45.6%. The low precision does not mean that the other splits are wasted. Where a split fired on a hidden hop, ADAPTIVE scores 32.3 SubEM against 6.5 for FIXED, and where it fired on a hop that is not hidden (n=71), 38.0 against 14.1; where no split fired, the two procedures are within half a point. The split thus repairs more than the hidden hop: a hop that the decomposer wrote too broadly, or whose pool missed the fact, also becomes answerable once it is split. Note that the annotation also flags 35 of the 100 released targets as not following the corpus chain under the most-recent-fact rule, ten of them on hidden-hop questions (Appendix B).

6.4 Effect of the Reader

The prompts are tuned on `gpt-4o-mini` and we use them on different readers. We present in Table 2 the FC-MH comparison with three further readers, each at its best depth. Looking at the Δ column, Δ_1 is large and consistent across readers ($+37.5$ to $+44.5$ SubEM), suggesting that the gain from decomposition is a property of the corpus rather than of the reader. Δ_2 is far smaller and no longer uniform in sign. The split rate varies: from 25.5% of questions for `gpt-4o-mini`, to 5% for Gemma. A split can only change the answer on a question where it ran, so the split rate bounds the gain from splitting, and the two readers that split most often are the two with the largest Δ_2.

6.5 Effect of the Evidence Condition

Table 6 lists the results of previous work on FC-MH, each under its own reader and evidence condition. Looking at the table, the nine systems with the same reader over a chunked index perform poorly and alike (2 to 7 SubEM), while a fixed plan over a fact-level index scores 27 with the same reader and 41 with `gpt-4o`. The evidence condition thus moves the number by more than any of these methods does within a condition. Our own runs show the same contrast with everything else held fixed: SINGLE scores 5.0 over 512-token chunks (not shown in the table) and 11.5 over single facts (Table 2), before any decomposition is applied. Note that the systems built for stores of extracted facts, MemGPT, Mem0 and Zep, are among the chunked-index rows, at 2 to 3 SubEM.

6.6 Atomised MuSiQue

We ran a follow-up on 600 MuSiQue development questions, rewriting their candidate paragraphs as mostly single-relation facts in a shared store with FC-MH distractor facts. All procedures used the same cached question plans. Answering and repair used prompts adapted from FC-MH (Appendix C.3).

A split ran on 17.0% to 23.8% of questions across retrieval depths k=5, 10 and 35, compared

with 3.2% in the original MuSiQue paragraph setting. Yet Δ_2 ranged only from -0.46 to $+0.31$ F1, with no significant gain at any tested depth (Table 8). Splits were more frequent in this fact-pool setting, but did not reproduce the FC-MH repair gain. This experiment does not isolate the properties of datasets.

Together with the original hidden-hop annotations ($17/100$ on FC-MH and $1/100$ on MuSiQue), this suggests that repair helps when the evidence requires an intermediate relation missing from the question's natural plan. FC-MH deliberately contains such routes; the MuSiQue paths we examined usually follow the question's natural plan.

7 Case Study: Fact Verification on EX-FEVER

We apply the three procedures to claim verification to see what carries over when the task is a verdict on a claim rather than an answer to a question.

Procedures over claims. We use the fixed 1,000-claim mini-test subset of EX-FEVER (Ma et al., 2024), whose claims are labelled SUPPORT, REFUTE or NOT ENOUGH INFO. We treat the declarative claim as the object to be resolved. The shared decomposer rewrites it into an ordered list of answerable sub-questions whose answers jointly determine whether the claim is supported. For example, a claim that relates a person, a work and a location becomes separate questions about the person–work and work–location relations. With the ordered list of sub-questions, each hop retrieves and answers as in Algorithm 1, and ADAPTIVE splits a hop on a refusal. The one change is at the end: every procedure closes with the same verdict call, which labels the claim from five claim documents and working notes of the form sub-question $\rightarrow$ answer. The retriever is reciprocal rank fusion of BM25 (Robertson and Zaragoza, 2009) and a dense index over the full corpus.[2] The retriever is shared by all three procedures, and the reader and prompts are those of §5. For reference, the trained systems of Ma et al. (2024) report 54.7 to 73.3 accuracy on the full 6,099-claim test set.

[2] `bm25s` uses k_1=1.5 and b=0.75 over title and text; `bge-large-en-v1.5` uses its query instruction. Each retriever contributes its top $\max(4k, 20)$ documents, which are fused with reciprocal-rank-fusion constant 60. The original claim is retrieved at k=5 and each sub-question at k=3. The final verdict receives the five original-claim documents together with successful top-level question–answer notes.

Decomposition and trigger. Over two runs, FIXED improves over SINGLE by 1.75 accuracy points, a small gain that the test does not confirm. As the verdict call is identical across procedures, whatever gain there is can only come from the question–answer notes. The trigger fires: a split ran on 6.55% of claims, twice the rate on MuSiQue. This is perhaps not surprising, as a claim about a named entity is verified against an encyclopedia, where the relation a sub-question asks for is often stored under another entity's article.

Recursive repair. At least one split is executed on 71 and 60 claims in the two runs. ADAPTIVE differs from FIXED by only +0.15 points for 10.7 reader calls per claim against 9.2. Across the 131 claim–run decisions on which a split fired, the verdict changes ten times against the matched FIXED run: five changes correct FIXED and three break a correct verdict. On the question-answering datasets a split helps by replacing a wrong intermediate answer before it enters the next hop, where it decides the route the plan takes. Here the answer enters the verdict as one note among the documents, and a corrected note changes the label only when the verdict happened to hinge on it.

8 Conclusion

We found that the number of retrieval steps a multi-hop question needs depends on both corpus and question. Decomposition improved over a single query on both datasets. Adaptive repair raised the mean score on FC-MH, although its gain varied across readers. It gave no reliable overall gain on MuSiQue, as we also showed by hand annotation that hidden hops are common on FC-MH and rare over MuSiQue, and that neither a deeper pool nor a rewritten query recovered what the split did. With that difference in nature of dataset in mind, in the atomised MuSiQue setting, there was no overall score gain even with more frequent split. On fact verification the split fired but changed little.

In future work we will search over several bridge facts instead of one. We will also experiment on other multi-hop QA datasets and more realistic settings, then evaluate how ADAPTIVE and other agentic search methods handle them.

Limitations

Readers and prompts. The MuSiQue results use `gpt-4o-mini` and one set of prompts. On FC-MH, three further readers give similarly large Δ_1, but their Δ_2 ranges from -1.5 to $+3.5$ (§6.4). Their SINGLE and FIXED cells have two runs, while their ADAPTIVE cells have one run each. The split rate ranges from 5% to 25.5% across readers, so it is a property of the reader as much as of the corpus. Appendix C.4 compares the two answering prompts within FIXED.

Sample size and variance. FC-MH has 100 questions, and a 6-point difference on 100 items is not significant on the least favourable pair of runs (Table 2). At temperature 0 the decomposer still rewrites part of its plans between identical runs, and scores spread by up to 4 SubEM on FC-MH and 0.4 F1 on MuSiQue across runs (Appendix C.5). The MuSiQue increments are measured on 2,417 items.

Evidence for the hidden hop. The contrast between the fact corpus and the paragraph corpus is observational: the two differ in domain and construction as well as in record type. The hidden-hop annotation (§6.3) was made by one author, on 100 questions per dataset, and 35 of the 100 FC-MH targets do not follow the corpus under its own recency rule (Appendix B). The atomised MuSiQue follow-up keeps the questions fixed across its procedures, but changes the store, retriever and prompts relative to the original paragraph experiment. Its higher split rate and negligible gain do not establish whether reasoning-path structure is the cause.

Datasets. FC-MH is artificial: its facts are counterfactual and the corpus is built to contradict itself, so the rate of hidden hops we find there (17 of 100 questions) is not an estimate for any real store. Its form, one relation per record, is that of an agent memory of extracted facts (Chhikara et al., 2025; Rasmussen et al., 2025), which public benchmarks rarely test. On natural text, hidden hops were rare in MuSiQue (1 of 100) and the split fired on 3.2% of questions; on EX-FEVER it fired on 6.55% of claims and changed little. We therefore have no natural-text dataset on which the repair pays.

Retrieval side and scope. Indexing one fact per record on FC-MH is a choice we made and held fixed. It raises the single-query baseline above the chunked-index rows of previous work, so our increments are measured over a stronger retrieval side than the one under which the benchmark was first reported, and the design measures increments against that fixed side rather than systems against systems. FC-MH is counterfactual by construction and every prompt forbids answering from world knowledge; as a check, the reader scores 4.0 SubEM with no evidence shown. All datasets are English, and the adaptive plan costs $1.7\times$ the reader calls of the fixed plan on FC-MH and $1.5\times$ on MuSiQue.

Acknowledgments

The Commonwealth of Australia (represented by the Advanced Strategic Capabilities Accelerator) supports this research through a Defence Science Partnerships agreement.

References

Paul J. L. Ammann, Jonas Golde, and Alan Akbik. 2025. Question decomposition for retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Shulin Cao, Jiajie Zhang, Jiaxin Shi, Xin Lv, Zijun Yao, Qi Tian, Juanzi Li, and Lei Hou. 2023. Probabilistic tree-of-thought reasoning for answering knowledge-intensive complex questions. *arXiv preprint arXiv:2311.13982*. Accepted at EMNLP 2023.

Tong Chen, Hongwei Wang, Sihao Chen, Wenhao Yu, Kaixin Ma, Xinran Zhao, Hongming Zhang, and Dong Yu. 2024. Dense X retrieval: What retrieval granularity should we use? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building production-ready AI agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*.

Justus-Jonas Erker, Nils Reimers, and Iryna Gurevych. 2026. GRITHopper: Decomposition-free multi-hop dense retrieval. In *Proceedings of the 2026 Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. HippoRAG: Neurobiologically inspired long-term memory for large

language models. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*.

Qisheng Hu, Quanyu Long, and Wenya Wang. 2025. Decomposition dilemmas: Does claim decomposition boost or burden fact-checking performance? In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics (NAACL)*.

Yuanzhe Hu, Yu Wang, and Julian McAuley. 2026. Evaluating memory in LLM agents via incremental multi-turn interactions. In *The Fourteenth International Conference on Learning Representations (ICLR)*.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. 2024. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Yuelyu Ji, Zhuochun Li, Rui Meng, and Daqing He. 2026. Retrieval–reasoning processes for multi-hop question answering: A four-axis design framework and empirical trends. *arXiv preprint arXiv:2601.00536*.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Shuguang Jiao, Xinyu Xiao, Yunfan Wei, Shuhan Qi, Chengkai Huang, Quan Z. Michael Sheng, and Lina Yao. 2026. PruneRAG: Confidence-guided query decomposition trees for efficient retrieval-augmented generation. *arXiv preprint arXiv:2601.11024*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Jueun Kim, Sungho Park, and Wook-Shin Han. 2026. Hi-Q: Hierarchical evidence-guided query refinement for multi-hop question answering. *arXiv preprint arXiv:2608.30468*.

Minghan Li, Junjie Zou, Xinxuan Lv, Chao Zhang, and Guodong Zhou. 2026. S2G-RAG: Structured sufficiency and gap judging for iterative retrieval-augmented QA. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yanhong Li, David Yunis, David McAllester, and Jiawei Zhou. 2025. Context-efficient retrieval with factual decomposition. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics (NAACL)*.

Yanming Liu, Xinyue Peng, Xuhong Zhang, Weihao Liu, Jianwei Yin, Jiannan Cao, and Tianyu Du. 2024. RA-ISF: Learning to answer and understand from retrieval augmentation via iterative self-feedback. *arXiv preprint arXiv:2403.06840*.

Huanhuan Ma, Weizhi Xu, Yifan Wei, Liuji Chen, Liang Wang, Qiang Liu, Shu Wu, and Liang Wang. 2024. EX-FEVER: A dataset for multi-hop explainable fact verification. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Anuj Maharjan, Devinder Kaur, and Richard Molyet. 2026. Agent-orchestrated adaptive RAG: A comparative study on structured and multi-hop retrieval. *arXiv preprint arXiv:2606.05658*.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2023. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*.

Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. 2024. ADaPT: As-needed decomposition and planning with language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. 2025. Zep: A temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*.

Vikas Reddy and Sumanth Reddy Challaram. 2026. Reliable post-retrieval assembly for agent memory: Separating evidence extraction from policy execution. In *Lifelong Agent Workshop at COLM 2026*.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. RAPTOR: Recursive abstractive processing

for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Sheroz Shaikh. 2026. Dissecting agentic RAG: A component ablation for multi-hop QA with a local 7B model. *arXiv preprint arXiv:2606.21553*.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024. DRAGIN: Dynamic retrieval augmented generation based on the information needs of large language models. *arXiv preprint arXiv:2403.10081*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jingjin Wang and Jiawei Han. 2025. PropRAG: Guiding retrieval with beam search over proposition paths. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oğuz. 2020. Answering complex open-domain questions with multi-hop dense retrieval. *arXiv preprint arXiv:2009.12756*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Jiahao Zhang, Haiyang Zhang, Dongmei Zhang, Yong Liu, and Shen Huang. 2024. End-to-end beam retrieval for multi-hop question answering. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Jiajie Zhang, Shulin Cao, Tingjia Zhang, Xin Lv, Jiaxin Shi, Qi Tian, Juanzi Li, and Lei Hou. 2023. Reasoning over hierarchical question decomposition tree for explainable question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*.

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

A Prompts

The prompts below are copied from the notebooks that produced the reported numbers, per corpus.

A.1 FC-MH

Decompose (all decomposing procedures).

```
You decompose a multi-hop question into an ordered list of single-hop subquestions.
Rules:
- One subquestion per line, numbered '1.', '2.', ...
- A later subquestion refers to the ANSWER of an earlier one with #k (e.g. #1 = answer to
subquestion 1).
- Each subquestion must be answerable on its own once the #k references are filled in.
- If the question is already single-hop, output exactly one line.
Output ONLY the numbered list, nothing else.
```

User message: `DECOMP_FEWSHOT + '\nQuestion: ' + question`

Decompose few-shot.

```
Question: What is the country of origin of the sport played by Christian Abbiati?
1. What is the sport played by Christian Abbiati?
2. What is the country of origin of #1?

Question: Who is the spouse of the director of Inception?
1. Who is the director of Inception?
2. Who is the spouse of #1?

Question: What is the capital of the country where the Eiffel Tower is located?
1. In which country is the Eiffel Tower located?
2. What is the capital of #1?
```

Commit (SINGLE and FIXED).

```
You are a knowledge management system. Answer the Question using ONLY the Knowledge Pool. All
entities are synthetic or counterfactual, so your own world knowledge is WRONG here and must NEVER
be used, even as a fallback. You may combine facts if needed. Your answer MUST be a word or phrase
that literally appears inside the Knowledge Pool text. Each fact starts with a serial number; a
HIGHER serial number means a MORE RECENT fact. Before you answer, check whether two facts give
different values for the same relation of the same entity. If they do, answer with the one that has
the HIGHER serial number - even when the lower-serial value is the one that matches the real world,
which is the usual case. Give a very concise answer - one word or short phrase only.
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nAnswer:`

Answer (ADAPTIVE).

```
You are a knowledge management system. Answer the Question using ONLY the Knowledge Pool. All
entities are synthetic or counterfactual, so your own world knowledge is WRONG here and must NEVER
be used, even as a fallback. You may combine facts if needed. Your answer MUST be a word or phrase
that literally appears inside the Knowledge Pool text. If the pool truly contains nothing related,
output exactly: unknown. Each fact starts with a serial number; a HIGHER serial number means a MORE
RECENT fact. Before you answer, check whether two facts give different values for the same relation
of the same entity. If they do, answer with the one that has the HIGHER serial number - even when
the lower-serial value is the one that matches the real world, which is the usual case. Give a very
concise answer - one word or short phrase only.
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nAnswer:`

Commit (ADAPTIVE).

```
You are a knowledge management system. Answer the Question using ONLY the Knowledge Pool. All
entities are synthetic or counterfactual, so your own world knowledge is WRONG here and must NEVER
be used, even as a fallback. You may combine facts if needed. Your answer MUST be a word or phrase
that literally appears inside the Knowledge Pool text. Even when the pool is a poor match, give the
closest word or phrase in it; never refuse and never answer 'unknown'. Each fact starts with a
serial number; a HIGHER serial number means a MORE RECENT fact. Before you answer, check whether
two facts give different values for the same relation of the same entity. If they do, answer with
the one that has the HIGHER serial number - even when the lower-serial value is the one that matches
```

```
the real world, which is the usual case. Give a very concise answer - one word or short phrase only.
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nAnswer:`

FindBridge (ADAPTIVE).

```
You are a knowledge management system working ONLY over the given Knowledge Pool. All entities are
synthetic or counterfactual, so your own world knowledge is WRONG here; never use it.
A reader has already tried and failed to answer the Question from this pool. Do NOT answer the
question. Your only job is to find the stepping stone.
The pool often does not record the asked relation for the asked entity, but does record some OTHER
relation of that entity, which names a NEW entity - and the asked relation is recorded for the new
entity instead. The pool may not say what sport someone plays but may say what position they play;
it may not give a spouse's citizenship but may name the spouse; it may not say who runs a product
but may name the company that made it.
Find the ONE fact in the pool that names that new entity, and copy it verbatim. The fact will
normally mention an entity the question already names - that is how it connects. What makes it a
stepping stone is the NEW name it introduces.
A fact is NOT a stepping stone if it introduces no new name, or if it is about an entity the
question is not asking about.
Output EXACTLY ONE line:
BRIDGE: <one fact from the pool, copied verbatim>
NONE
If no fact in the pool names such an entity, output exactly: NONE
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nOutput:`

Split (ADAPTIVE).

```
A question cannot be answered directly from the Knowledge Pool, but a bridge fact was found. Split
the question into EXACTLY TWO simpler subquestions using that bridge fact.
All entities are synthetic or counterfactual; never use your own world knowledge to decide what to
ask.
Rules:
- Subquestion 1 must be answerable by the bridge fact.
- Subquestion 2 must contain the placeholder #1 (the answer of subquestion 1) and, once #1 is
filled in, must answer the original question.
- Both subquestions must be about entities named in the question or the bridge fact you were given.
Never mention an entity from the examples.
- Output EXACTLY two lines, nothing else:
1. <subquestion 1>
2. <subquestion 2 containing #1>
```

User message: `SPLIT_FEWSHOT + '\nQuestion: {question}\nBridge fact: {bridge}'`

Split few-shot.

```
### EXAMPLES (format only - do NOT answer these, do NOT reuse their entities)
Question: In which city was the author of The Ashen Ledger born?
Bridge fact: 4102. The author of The Ashen Ledger is Ilvane Sabreth.
1. Who is the author of The Ashen Ledger?
2. In which city was #1 born?

Question: Which country is the spouse of Marren Volkov a citizen of?
Bridge fact: 7731. Marren Volkov is married to Orrin Fennwick.
1. Who is Marren Volkov married to?
2. Which country is #1 a citizen of?

Question: Which time zone does Harrowmere use?
Bridge fact: 5120. Harrowmere is a village in the province of Calvessa.
1. Which province is Harrowmere a village in?
2. Which time zone does #1 use?

Question: In which branch of the armed forces did Corvin Ashdale serve?
Bridge fact: 1210. Corvin Ashdale held the rank of Vane Marshal.
1. Which rank did Corvin Ashdale hold?
2. Which branch of the armed forces uses the rank of #1?

### END EXAMPLES
```

```
Now the REAL question. Both subquestions must be about the entities in THIS question and its bridge
fact, and nothing from the examples above.
```

A.2 MuSiQue

Decompose.

```
You decompose a multi-hop question into an ordered list of single-hop subquestions.
Rules:
- One subquestion per line, numbered '1.', '2.', ...
- Every subquestion must name an entity that appears in the question you were given, or refer to an
earlier answer with #k (e.g. #1 = answer to subquestion 1).
- Each subquestion must be answerable on its own once the #k references are filled in.
- Some questions need two INDEPENDENT lookups rather than a chain. When neither fact depends on the
other, ask for them in SEPARATE subquestions with no #k placeholder, then combine them in a later
subquestion that references both (for example '... #1 ... #2 ...'). Do not force a parallel
question into a single chain.
- If the question is already single-hop, output exactly one line.
Output ONLY the numbered list, nothing else.
```

User message: `DECOMP_FEWSHOT + '\nQuestion: ' + question`

Decompose few-shot.

```
### EXAMPLES (format only - do NOT answer these)
Question: In which city was the author of The Ashen Ledger born?
1. Who is the author of The Ashen Ledger?
2. In which city was #1 born?

Question: Which country is the spouse of Marren Volkov a citizen of?
1. Who is Marren Volkov married to?
2. Which country is #1 a citizen of?

Question: When was the first establishment that McDonaldization is named after, open in the country
Horndean is located?
1. What is McDonaldization named after?
2. Which country is Horndean located in?
3. When did the first #1 open in #2?

Question: What language does Orrin Fennwick speak?
1. What language does Orrin Fennwick speak?
### END EXAMPLES

Now decompose the REAL question below. Decompose only that question - do not answer it and do not
reproduce any example above.
```

Answer (ADAPTIVE).

```
You answer questions using ONLY the passages you are given. Answer the Question from the Passages.
You may combine passages if the answer needs more than one of them. Your answer MUST be a word or
phrase that appears in the passage text. If the passages truly do not contain the answer, output
exactly: unknown. Never answer from your own knowledge of the world, even when you are confident
and even as a fallback: an answer that is not supported by these passages is wrong here, whether or
not it is true. Give a very concise answer - one word or short phrase only, with no explanation.
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nAnswer:`

Commit (SINGLE, every hop of FIXED, ADAPTIVE fallback).

```
You answer questions using ONLY the passages you are given. Answer the Question from the Passages.
You may combine passages if the answer needs more than one of them. Your answer MUST be a word or
phrase that appears in the passage text. Even when the passages are a poor match, give the closest
word or phrase in them; never refuse and never answer 'unknown'. Never answer from your own
knowledge of the world, even when you are confident and even as a fallback: an answer that is not
supported by these passages is wrong here, whether or not it is true. Give a very concise answer -
one word or short phrase only, with no explanation.
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nAnswer:`

FindBridge (ADAPTIVE).

```
You work ONLY over the passages you are given, never from your own knowledge of the world.
A reader has already tried and failed to answer the Question from these passages. Do NOT answer the
question. Your only job is to find the stepping stone.
The passages often do not state the asked relation for the asked entity, but do state some OTHER
relation of that entity, which names a NEW entity – and the asked relation is recorded for the new
entity instead. They may not say which country a film was made in but may name its director; they
may not give a person's birthplace but may name the band they founded; they may not say when a
building opened but may name the company that built it.
Find the ONE sentence in the passages that names that new entity, and copy it verbatim. It will
normally mention an entity the question already names – that is how it connects. What makes it a
stepping stone is the NEW name it introduces.
A sentence is NOT a stepping stone if it introduces no new name, or if it is about an entity the
question is not asking about.
Output EXACTLY ONE line:
BRIDGE: <one sentence from the passages, copied verbatim>
NONE
If no sentence in the passages names such an entity, output exactly: NONE
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nOutput:`

Split (ADAPTIVE).

```
A question cannot be answered directly from the passages, but a bridge sentence was found. Split
the question into EXACTLY TWO simpler subquestions using that bridge sentence.
Rules:
- Subquestion 1 must be answerable by the bridge sentence.
- Subquestion 2 must contain the placeholder #1 (the answer of subquestion 1) and, once #1 is
filled in, must answer the original question.
- Both subquestions must be about entities named in the question or the bridge sentence you were
given. Never mention an entity from the examples.
- Output EXACTLY two lines, nothing else:
1. <subquestion 1>
2. <subquestion 2 containing #1>
```

User message: `SPLIT_FEWSHOT + '\nQuestion: {question}\nBridge fact: {bridge}'`

Split few-shot.

```
### EXAMPLES (format only – do NOT answer these, do NOT reuse their entities)
Question: In which city was the author of The Ashen Ledger born?
Bridge sentence: The Ashen Ledger was written by Ilvane Sabreth.
1. Who is the author of The Ashen Ledger?
2. In which city was #1 born?

Question: Which country is the spouse of Marren Volkov a citizen of?
Bridge sentence: Marren Volkov married Orrin Fennwick in 1998.
1. Who is Marren Volkov married to?
2. Which country is #1 a citizen of?

Question: Which time zone does Harrowmere use?
Bridge sentence: Harrowmere is a village in the province of Calvessa.
1. Which province is Harrowmere a village in?
2. Which time zone does #1 use?

Question: In which branch of the armed forces did Corvin Ashdale serve?
Bridge sentence: Corvin Ashdale held the rank of Vane Marshal.
1. Which rank did Corvin Ashdale hold?
2. Which branch of the armed forces uses the rank of #1?
### END EXAMPLES

Now the REAL question. Both subquestions must be about the entities in THIS question and its bridge
sentence, and nothing from the examples above.
```

A.3 EX-FEVER

Decompose.

```
You turn a multi-hop CLAIM into the ordered list of questions someone would have to answer in order
to check it.
A multi-hop claim usually describes an entity instead of naming it—'the comedian who founded Apatow
Productions', 'the director of that film'—and its evidence is spread over several Wikipedia
```

```
articles. The early questions resolve those descriptions into names; the later ones ask about the
properties the claim asserts of them.
Rules:
- One question per line, numbered ‘1.’, ‘2.’, ...
- A later question can refer to an earlier answer with #k (e.g. #1 = the answer to question 1).
- Resolve every thing the claim describes but does not name, before asking about it.
- Ask about EVERY property the claim asserts—dates, nationalities, roles, genres, places. A
property nobody asks about is never checked, and a false claim is often false precisely there.
- Ask, never assert. Nothing in the claim may be assumed true: that is what is being tested.
- Each question must be answerable on its own.
Output ONLY the numbered list, nothing else.
```

User message: `DECOMP_FEWSHOT + ‘\nClaim: {claim}\n’`

Decompose few-shot.

```
### EXAMPLES (format only - do NOT answer these, do NOT reuse their entities)
Claim: The Ashen Ledger, a 1998 mystery novel, was written by the Kerelian journalist who founded
Northwind Cartography.
1. Who founded Northwind Cartography?
2. Who wrote The Ashen Ledger?
3. What is the nationality of #1?
4. What is the occupation of #1?
5. In what year was The Ashen Ledger published?
6. What genre is The Ashen Ledger?
Claim: The lead actress of Glass Aviary was born in Dunmarrow.
1. Who is the lead actress of Glass Aviary?
2. Where was #1 born?

Claim: Orrin Fennwick is known for his work in a type of stage play performed without spoken
dialogue.
1. What type of stage play is performed without spoken dialogue?
2. What is Orrin Fennwick known for?
### END EXAMPLES

Now the REAL claim. Every question must be about the entities in THIS claim and nothing from the
examples above.
```

Answer (ADAPTIVE).

```
You answer questions using ONLY the passages you are given. Answer the Question from the Passages.
You may combine passages if the answer needs more than one of them. Your answer MUST be a word or
phrase that appears in the passage text. If the passages truly do not contain the answer, output
exactly: unknown. Never answer from your own knowledge of the world, even when you are confident
and even as a fallback: an answer that is not supported by these passages is wrong here, whether or
not it is true. Give a very concise answer - one word or short phrase only, with no explanation.
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nAnswer:`

Commit (FIXED and ADAPTIVE fallback).

```
You answer questions using ONLY the passages you are given. Answer the Question from the Passages.
You may combine passages if the answer needs more than one of them. Your answer MUST be a word or
phrase that appears in the passage text. Even when the passages are a poor match, give the closest
word or phrase in them; never refuse and never answer ‘unknown’. Never answer from your own
knowledge of the world, even when you are confident and even as a fallback: an answer that is not
supported by these passages is wrong here, whether or not it is true. Give a very concise answer -
one word or short phrase only, with no explanation.
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nAnswer:`

FindBridge (ADAPTIVE).

```
You work ONLY over the passages you are given, never from your own knowledge of the world.
A reader has already tried and failed to answer the Question from these passages. Do NOT answer the
question. Your only job is to find the stepping stone.
The passages often do not state the asked relation for the asked entity, but do state some OTHER
relation of that entity, which names a NEW entity—and the asked relation is recorded for the new
```

```
entity instead. They may not say which country a film was made in but may name its director; they
may not give a person's birthplace but may name the band they founded; they may not say when a
building opened but may name the company that built it.
Find the ONE sentence in the passages that names that new entity, and copy it verbatim. It will
normally mention an entity the question already names—that is how it connects. What makes it a
stepping stone is the NEW name it introduces.
A sentence is NOT a stepping stone if it introduces no new name, or if it is about an entity the
question is not asking about.
Output EXACTLY ONE line:
BRIDGE: <one sentence from the passages, copied verbatim>
NONE
If no sentence in the passages names such an entity, output exactly: NONE
```

User message: `[Knowledge Pool]\n{pool}\n\nQuestion: {question}\nOutput:`

Split (ADAPTIVE).

```
A question cannot be answered directly from the passages, but a bridge sentence was found. Split
the question into EXACTLY TWO simpler subquestions using that bridge sentence.
Rules:
- Subquestion 1 must be answerable by the bridge sentence.
- Subquestion 2 must contain the placeholder #1 (the answer of subquestion 1) and, once #1 is
filled in, must answer the original question.
- Both subquestions must be about entities named in the question or the bridge sentence you were
given. Never mention an entity from the examples.
- Output EXACTLY two lines, nothing else:
1. <subquestion 1>
2. <subquestion 2 containing #1>
```

User message: `SPLIT_FEWSHOT + '\nQuestion: {question}\nBridge fact: {bridge}\n'`

Split few-shot.

```
### EXAMPLES (format only - do NOT answer these, do NOT reuse their entities)
Question: In which city was the author of The Ashen Ledger born?
Bridge sentence: The Ashen Ledger was written by Ilvane Sabreth.
1. Who is the author of The Ashen Ledger?
2. In which city was #1 born?

Question: Which country is the spouse of Marren Volkov a citizen of?
Bridge sentence: Marren Volkov married Orrin Fennwick in 1998.
1. Who is Marren Volkov married to?
2. Which country is #1 a citizen of?
### END EXAMPLES

Now the REAL question. Both subquestions must be about the entities in THIS question and its bridge
sentence, and nothing from the examples above.
```

Verdict.

```
You are a fact-checking verdict model. Decide how the evidence bears on the CLAIM.
Reply with exactly one label and nothing else:
- SUPPORT: every checkable part of the claim is stated by, or directly follows from, the evidence.
- REFUTE: the evidence states something INCOMPATIBLE with the claim—a different value for something
the claim asserts. Evidence that is merely silent about part of the claim is not a contradiction.
- NOT ENOUGH INFO: the evidence neither states nor contradicts a fact the claim depends on.
Three traps, in the order they cost the most:
1. RANKING AND ORDER. Being the first, the second, the only, the youngest or the largest of
something can only be settled by evidence that establishes the whole ordering. An article that
names some of the items does not settle it. Where the evidence does not settle a ranking, the
answer is NOT ENOUGH INFO, never REFUTE.
2. A SPECIFIC QUANTITY. If the claim gives a count, an age, a year or a total and the evidence
never states that number, the answer is NOT ENOUGH INFO, not SUPPORT. Evidence about the same
subject is not evidence for the number.
3. DESCRIPTION INSTEAD OF A NAME. A claim often refers to something by description rather than by
name—'the second-longest river in North America', 'the frontman of that band'. If the description
fits something the evidence does name, treat them as the same thing. Not finding the description
word-for-word is not a contradiction.
```

```
Judge the substance of the claim. Paraphrase, different wording, and extra detail in the evidence
do not matter, and an incidental modifier the evidence happens not to repeat is not by itself a
reason to answer NOT ENOUGH INFO.
Use ONLY the evidence provided. Your own world knowledge must not be used.
Output the label only.
```

Verdict few-shot.

```
### EXAMPLES (format only - invented entities, do NOT reuse them)
[Search record]
Q: Who directed Glass Aviary?
A: Ilvane Sabreth
Q: What nationality is Ilvane Sabreth?
A: Kerelian

CLAIM: Glass Aviary was directed by a Kerelian director.
Label: SUPPORT

[Search record]
Q: What kind of institution is Dunmarrow Museum?
A: art museum
Q: Is Dunmarrow Museum the largest museum in Kerelia?
A: unknown

CLAIM: Dunmarrow Museum is the largest museum in Kerelia.
Label: NOT ENOUGH INFO

[Search record]
Q: How many regional titles has Quillon United won?
A: unknown

CLAIM: Quillon United has won 12 regional titles.
Label: NOT ENOUGH INFO

[Search record]
Q: Where was Marren Volkov born?
A: Ostravia

CLAIM: Marren Volkov was born in Brenmark.
Label: REFUTE
### END EXAMPLES

Now the REAL one. Judge only the supplied passages and/or search record below.
```

User message: `VERDICT_FEWSHOT + [Evidence]\n{passages} + [Working notes]\n{resolved question-answer pairs} + \n\nCLAIM: {claim}\nLabel:`

B Hidden-Hop Annotation Guideline

We release the annotations together with our code at https://github.com/anvu2211/hidden-hop.

B.1 FC-MH

We manually traced all 100 questions through the fact corpus and marked a hop as hidden when the relation the question asks for is not recorded for the current entity but can be reached through an intermediate relation, as *sport* is reached through *position* in Table 1.

Each annotation contains two parts separated by a colon. The left side records the ordered relation chain expressed by the question. The right side records the serial numbers of the facts used to instantiate that chain.

Relation order. The symbol $\rightarrow$ gives the order in which relations must be resolved. For example,

$$\textit{sport} \rightarrow \textit{country_of_origin}$$

first identifies a sport and then its country of origin.

Hidden hops. A parenthesised relation sequence on the right of a hop marks a hidden hop. For example,

$$\textit{sport}\ (\textit{position} \rightarrow \textit{sport}) \rightarrow \textit{country_of_origin}$$

means that the question asks directly for a sport, but the corpus reaches it by first retrieving a position and then mapping that position to a sport. An item is positive if at least one requested hop requires such an intermediate relation.

Fact traces. Serial numbers on the right give the selected facts in traversal order. For example,

$$10906 \rightarrow 7164 \rightarrow 7774$$

records the facts selected for the position, sport, and country-of-origin steps. Because FC-MH contains superseded facts, the annotation follows the highest-serial fact for a relation whenever conflicting values are available.

Alternative and missing facts. A serial number in parentheses on the right records a different fact used by the released target chain. For example, "4 (1)" means that serial 4 is the value selected under the most-recent-fact rule, whereas the dataset target follows serial 1. None means that no fact completing the intended chain could be located. A slash between relations records alternative interpretations of the question rather than an additional ordered hop.

Split	Hidden hop	n	FIXED	ADAPTIVE	Δ_2
Fired	yes	31	6.5	32.3	+25.8
Fired	no	71	14.1	38.0	+23.9
Not fired	yes	37	16.2	16.2	+0.0
Not fired	no	261	68.2	68.6	+0.4

Table 5: FC-MH SubEM by split decision and hidden-hop annotation, over the four ADAPTIVE runs at k=3. n counts question–run decisions.

Wrong labels. WRONG marks a suspected error in the released output label. It does not indicate that a model prediction is wrong. We apply this label when the released answer follows a superseded fact, selects an inconsistent branch, or cannot be reached through the relation chain expressed by the question. Ambiguous cases are retained with an explanatory note rather than silently adjudicated.

B.2 MuSiQue

We annotated a sample of 100 questions from the development split. For each question we first wrote a decomposition from the question alone, then revealed the gold decomposition and the candidate paragraphs. We marked a hop as hidden only when the gold route added a relation that the question-derived decomposition lacks and the paragraphs required that relation to reach the answer.

B.3 Dataset-label audit

The FC-MH annotation also distinguishes dataset errors from model errors. We mark a target as *wrong label* when the released answer does not follow the corpus chain under the most-recent-fact rule; this concerns the benchmark label and not the prediction of any procedure. The audit flags 35 of the 100 targets, one of them borderline, and ten of the 17 hidden-hop questions have a flagged target. We keep the released labels in every reported score.

C Additional Results

C.1 Published FC-MH Baselines

System	Reader	Evidence	SubEM
Hu et al. (2026)			
BM25	`gpt-4o-mini`	512-tok.	3.0
text-emb-3-small	`gpt-4o-mini`	512-tok.	3.0
text-emb-3-large	`gpt-4o-mini`	512-tok.	4.0
Contriever	`gpt-4o-mini`	512-tok.	7.0
HippoRAG 2	`gpt-4o-mini`	512-tok.	5.0
MemoRAG	`gpt-4o-mini`	512-tok.	7.0
MemGPT	`gpt-4o-mini`	512-tok.	3.0
Mem0	`gpt-4o-mini`	4096-tok.	2.0
Zep	`gpt-4o-mini`	4096-tok.	3.0
Full context	`gpt-5-mini`	full	28.0
Reddy and Challaram (2026)			
CAR	`gpt-4o-mini`	facts	27.0
CAR	`gpt-4o`	facts	41.0
HIDDEN HOP	`gpt-4o-mini`	facts	55.5 $\pm$1.7

Table 6: FC-MH SubEM of previous work and HIDDEN HOP, over all 100 questions. Rows from previous work keep their own reader and evidence condition and are shown for context. Evidence is what the reader is given: chunks of 512 or 4,096 tokens, the full context, or single facts.

C.2 Retrieval Depth Sweep

Table 7 gives every procedure at each depth we ran. The best depth is chosen on the questions we report. The increments, however, do not depend on the choice. At matched depth, Δ_1 is between $+30.5$ and $+41.0$ SubEM on FC-MH and between $+4.1$ and $+8.8$ F1 on MuSiQue, and Δ_2 is $+6.5$, $+4.0$ and $+4.2$ SubEM on FC-MH at k=3, 5 and 10, against -0.5, $+0.8$ and -0.6 F1 on MuSiQue.

C.3 Atomised MuSiQue

We use the 100 previously annotated MuSiQue questions and 500 additional development questions sampled by annotated hop count. `gpt-5.6-luna` rewrites 7,931 unique candidate paragraphs into mostly single-relation facts, then checks the facts against each source paragraph twice. After deduplication, the shared store contains 99,112 facts, including 18,332 FC-MH distractor facts with serial numbers removed.

Retrieval uses `text-embedding-3-large` at 1,024 dimensions with cosine similarity. The MuSiQue decomposer produces cached plans shared across the procedures. The `gpt-4o-mini` reader uses answer, bridge and split prompts adapted from FC-MH: the rules about counterfactual entities, recency and conflicting updates are removed. The original MuSiQue experiment uses a different retrieval setting and the full 2,417 questions, so its F1 scores are not directly comparable with this table.

Procedure	k	Runs	Score
FC-MH, all 100 questions, SubEM			
SINGLE	3	2	8.0
SINGLE	5	2	11.0
SINGLE	10†	2	**11.5**
FIXED	3†	2	**49.0**
FIXED	5	2	44.0
FIXED	10	2	42.0
ADAPTIVE	3†	4	**55.5**
ADAPTIVE	5	1	48.0
ADAPTIVE	10	4	46.2
MuSiQue, full development split (n=2,417), F1			
SINGLE	3	1	45.5
SINGLE	5	1	52.2
SINGLE	10†	2	**53.0**
FIXED	3	1	54.3
FIXED	5	1	56.3
FIXED	10†	2	**57.4**
ADAPTIVE	3	1	53.9
ADAPTIVE	5†	2	**57.1**
ADAPTIVE	10	1	56.8

Table 7: Every procedure at each depth, with `gpt-4o-mini`. † marks the depth used in Tables 2 and 3. Multi-run cells are means.

	k=5	k=10	k=35
SINGLE	30.16	36.87	39.79
FIXED	43.08	41.36	45.78
ADAPTIVE	43.24	41.66	45.32
Δ_1	$+12.92$	$+4.48$	$+5.99$
p	$< 10^{-3}$	0.03	0.003
Δ_2	$+0.16$	$+0.31$	-0.46
p	0.85	0.71	0.62
Split %	17.0	19.2	23.8

Table 8: F1 on 600 atomised MuSiQue development questions. One run per depth. p - paired sign-flip test.

C.4 Answering Prompt Control

ADAPTIVE falls back on a COMMIT prompt that forbids refusing, whereas SINGLE and FIXED answer with their own COMMIT prompt (Appendix A), so part of Δ_2 could come from the prompt rather than from the split. Table 9 reruns FIXED on FC-MH with the fallback prompt on every hop, for each reader of Table 2. The two arms share the plan, decomposer, pool, depth, temperature and seed. The two prompts differ by at most 3.5 SubEM in either direction, and none of the paired tests detect a significant difference.

Reader	Answering prompt	k=3	k=5	k=10	Δ	95% CI	p	Refusals
`gpt-4o-mini`	own	49.0†	44.0	42.0	—	—	—	4/5
	fallback	45.5†	44.0	39.5	−3.5	[−7.5, 0.0]	0.124	3/3
`gpt-5.6-luna`	own	—	54.0	54.5†	—	—	—	1/1
	fallback	—	54.5	54.5†	+0.0	[−4.5, 4.0]	1.000	0/0
Qwen3.5-35B-A3B	own	49.5†	44.0	—	—	—	—	5/5
	fallback	52.5†	47.5	—	+3.0	[−0.5, 7.0]	0.185	2/3
Gemma-4-26B-A4B-it	own	54.0	54.5†	54.0	—	—	—	22/22
	fallback	50.5	54.5†	52.5	+0.0	[−1.5, 1.5]	1.000	19/18

Table 9: FIXED on FC-MH with its own answering prompt, as in every other table, against the ADAPTIVE fallback prompt on every hop. SubEM over all 100 questions, mean of two runs per cell. † marks the depth at which Δ (fallback − own), its paired bootstrap CI and a sign-flip p are computed. *Refusals* gives the number of questions left unanswered in each of the two runs at that depth.

C.5 Plan Variance Between Runs

Table 10 gives every run behind Tables 2 and 3 and the spread between the highest and lowest run of each procedure. Table 11 shows how a spread arises: the decomposer, at temperature 0, writes a different plan for the same question in two runs of identical code, and one changed sub-question is enough to change the answer.

Run	Hop 1	Hop 2	Hop 3	Answer
A. FC-MH q22, FIXED at k=3. *"What is the country of origin of the sport associated with John Beilein's occupation?"* gold: *Japan*				
1	*What is the occupation of John Beilein?*	*What is the country of origin of the sport associated with #1?*	—	*association football*
2	*What is the occupation of John Beilein?*	*What is the sport associated with #1?*	*What is the country of origin of #2?*	*Japan* = gold
B. FC-MH q95, ADAPTIVE at k=3. *"Which company did the employer of the creator of Link work for?"* gold: *University of Oxford*				
1	*Who is the creator of Link?*	*Who was the employer of #1?*	—	*University of Oxford* = gold
2	*Who is the creator of Link?*	*Who is the employer of #1?*	*Which company does #2 work for?*	*Frankfurt*
C. MuSiQue `2hop__68590_27619`, FIXED at k=5. *"Where is the mission for the agency that controls the monetary policy of the EU found?"* gold: *Article 2 of the Statute of the ECB*				
1	*What is the agency that controls the monetary policy of the EU?*	*Where is the mission **of** European Central Bank found?*	—	*Article 2 of the Statute of the ECB*, F1 1.00
2	the same sub-question	*Where is the mission **for** European Central Bank found?*	—	*Frankfurt, Germany*, F1 0.00

Table 11: The same question in two runs of identical code, with a different plan and a different answer. In A only the three-hop plan can be answered from the corpus; in B the plans differ in length, while in C one word of one sub-question changes what the retriever returns.

Dataset	Procedure	Runs	Score in each run	Spread
FC-MH	SINGLE (k=10)	2	11.0, 12.0	1.0
FC-MH	FIXED (k=3)	2	48.0, 50.0	2.0
FC-MH	ADAPTIVE (k=3)	4	57.0, 56.0, 56.0, 53.0	4.0
MuSiQue	SINGLE (k=10)	2	52.91, 53.06	0.15
MuSiQue	FIXED (k=10)	2	57.24, 57.56	0.32
MuSiQue	ADAPTIVE (k=5)	2	57.31, 56.94	0.36

Table 10: The score of every run behind Tables 2 and 3, and the spread between the highest and lowest run.

D Additional Traces

Table 13 gives three ADAPTIVE traces on FC-MH from the logs, with the pool shown at every step. In A and B the split repairs a hidden hop; in C it separates two relations the corpus stores as separate facts. Table 12 gives a MuSiQue question on which the refusal costs the answer: the reader refuses on a pool that contains it, as §4.1 anticipates.

Step	Sub-question	Retrieved	Reader output
A refusal on a pool that contained the answer. MuSiQue `2hop__487706_40501` at k=5. *"How many refugees emigrated to the participant of Battle of Kentish Knock?"* gold: *75,000 to 100,000*			
hop 1	*Who was the participant of the Battle of Kentish Knock?*	p13* *a naval battle between the fleets of the Dutch Republic and England*; p3, p6, p15, p18	*Dutch Republic and England*, under both
hop 2, FIXED	*How many refugees emigrated to Dutch Republic and England?*	p16* *the Dutch Republic received the largest group of Huguenot refugees, an estimated total of 75,000 to 100,000 people*; p1, p11, p14, p19	COMMIT → *75,000 to 100,000* = gold, F1 1.0, 3 calls
hop 2, ADAPTIVE	the same sub-question	the same pool	ANSWER → UNKNOWN
FINDBRIDGE	—	the same pool	p16 itself, the sentence that states the number
SPLIT 2a	*How many Huguenot refugees did the Dutch Republic receive?*	p16*; p1, p4, p11, p19	*75,000 to 100,000*, the gold string
SPLIT 2b	*How many refugees emigrated to Dutch Republic and England, including 75,000 to 100,000?*	p16*; p1, p11, p14, p19	ANSWER → UNKNOWN; FINDBRIDGE returns p16 again and SPLIT repeats the same pair
depth cap	the same sub-question at $d_{\max}$	the same pool	COMMIT → *Four thousand* (run 1), *Hundreds of thousands* (run 2), F1 0.0, 10 calls

Table 12: A MuSiQue question on which a refusal made the answer worse than that of FIXED, at k=5. pn is candidate paragraph n, and * marks a supporting paragraph. The pool at hop 2 is the same under both procedures and contains the gold paragraph; both ADAPTIVE runs refuse on it and end at F1 0.0.

Step	Sub-question	Retrieved	Reader output
A. FC-MH q0. *"What is the country of origin of the sport played by Christian Abbiati?"* gold: *Philippines*; FIXED: *Not specified*			
hop 1	*What is the sport played by Christian Abbiati?*	6954 *Abbiati: goalkeeper*; 10906 *Abbiati: cornerback*; 18026 (Buffon)	ANSWER → UNKNOWN
FINDBRIDGE	—	the same pool	6954, the superseded position
SPLIT 1a	*What position does Christian Abbiati play?*	6954, 10906, 17425	*cornerback*, the higher serial
SPLIT 1b	*What is the sport played by cornerback?*	3070 *cornerback: American football*; 7164 *cornerback: field hockey*; 17536	*field hockey*
hop 2	*What is the country of origin of field hockey?*	1923 *field hockey: England*; 7774 *field hockey: Philippines*; 16113	*Philippines* = gold, 7 calls
B. FC-MH q69. *"Which city is the capital of the country where the genre, which Benjamin Sisko performed, originated?"* gold: *Rajanpur*; FIXED: *Harare*			
hop 1	*What is the genre that Benjamin Sisko performed?*	9952 *Sisko: performed by Avery Brooks*; 11662 *Sisko: performed by Extreme*; 14935 *Avery Brooks: jazz*	ANSWER → UNKNOWN
FINDBRIDGE	—	the same pool	9952, the superseded performer
SPLIT 1a	*Who performed Benjamin Sisko?*	9952, 11662, 17489	*Extreme*, the higher serial
SPLIT 1b	*What is the genre that Extreme performed?*	711 *Extreme: heavy metal*; 5558 *Extreme: jazz*; 18024	*jazz*
hop 2	*In which country did jazz originate?*	4602 *jazz: United States*; 11957; 16135 *jazz: Romania*	*Romania*
hop 3	*What is the capital of Romania?*	3557 *Bucharest*; 4332 *Rajanpur*; 15941	*Rajanpur* = gold, 8 calls
C. FC-MH q5. *"What is the language officially spoken in the country where the spouse of the Chief Executive Officer of CBS Corporation holds citizenship?"* gold: *Walloon*; FIXED: *Hebrew*			
hop 1	*Who is the Chief Executive Officer of CBS Corporation?*	352 *CEO of CBS Corporation: Les Moonves*; 12008; 16008	*Les Moonves*
hop 2	*What is the citizenship of the spouse of Les Moonves?*	8506 *Moonves: married to Julie Chen*; 12849 *Moonves: married to Louis Aragon*; 15555	ANSWER → UNKNOWN
FINDBRIDGE	—	the same pool	8506, the superseded spouse
SPLIT 2a	*Who is Les Moonves married to?*	8506, 12849, 15555	*Louis Aragon.*, the higher serial
SPLIT 2b	*What is the citizenship of Louis Aragon.?*	3072 *Aragon: France*; 12744 *Aragon: Italy*; 13690	*Italy*
hop 3	*What is the language officially spoken in Italy?*	4286 *Italy: Italian*; 8583 *Italy: Walloon*; 15544	*Walloon* = gold, 8 calls

Table 13: Three FC-MH questions that ADAPTIVE (k=3) split and answered correctly and FIXED answered wrongly. Numbers are fact serials, and each pool is the one the reader was shown. A and B are hidden hops; in C the split separates two relations the corpus stores as separate facts.