LANGUAGE MODELS USE BOTH PROACTIVE AND RETROACTIVE MECHANISMS TO MAKE COMPOSITIONAL INFERENCES

Anonymous authors
Paper under double-blind review

ABSTRACT

One of the most powerful features of modern language models (LMs) is their ability to generalize about novel in-context information. A fundamental component of this ability is *compositional inference*: the capacity to combine multiple new relations such as 'Tom's father is Steve' and 'Jim's father is Tom' in order to arrive at an unstated inference such as 'Jim's grandfather is Steve'. In this work, we study how LMs make these kinds of inferences. Using a variety of causal interventions on Llama 3.1 405B Instruct, we first demonstrate that 405B Instruct uses two functionally independent mechanisms to do so: a *proactive mechanism* that makes generalizations as soon as information is available in-context, and a *retroactive mechanism* that reasons about what is in-context only when it is necessary for predicting the immediate next token. We then consider *why* 405B Instruct uses two different mechanisms to make compositional inferences. We find that 405B Instruct uses contextual information to choose when to use the proactive mechanism, and that the proactive mechanism enables it to make deeper inferences than what the retroactive mechanism supports on its own. Taken together, our work finds that LMs use two distinct and interpretable mechanisms to infer novel relations from in-context information, and that LMs use contextual cues to decide when to precompute relevant information for later inferences.

1 INTRODUCTION

Much of the success of language models (LMs) can be attributed to their ability to generalize about new information provided in-context (Brown et al., 2020; Garg et al., 2022). For example, hundreds of millions of LM users now populate LM context windows every day with personal data and esoteric queries that could not have possibly appeared in the model's training data. Yet, LMs respond flexibly and compellingly to these novel contexts, motivating their meteoric rise as general purpose systems.

What mechanisms enable this flexibility, and what are their underlying primitives? To answer these questions, one popular line of work has studied *entity binding*, the ability of an LM to understand direct relations between objects described in-context (Kim & Schuster, 2023). In this work, we study another important ability that builds on entity binding: *compositional inference*, the ability to combine multiple new relations described in-context to make a new inference (Sinha et al., 2019).

Compositional inference is a fundamental primitive of generalization and complex reasoning (Fodor & Pylyshyn, 1988; Lake et al., 2016). Consider, for example, the following inputs:

> Tom's father is Steve. Jim's father is Tom. Who is Jim's grandfather?
> The bus is in the warehouse. The pen is in the bus. Which building is the pen in?
> All F love P. All C are a type of F. What do all C love?

Here, entity binding enables LMs to relate pairs of tokens such as 'Tom' and 'Steve' or 'pen' and 'bus', but it is compositional inference that allows LMs to combine these bound relations to extract higher level inferences such as 'Jim's grandfather is Steve' or 'All C love P' in order to respond correctly to the questions. Although seemingly mundane, these examples represent a most basic computation that pervades both human and LM thought. How, then, do LMs do it?

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

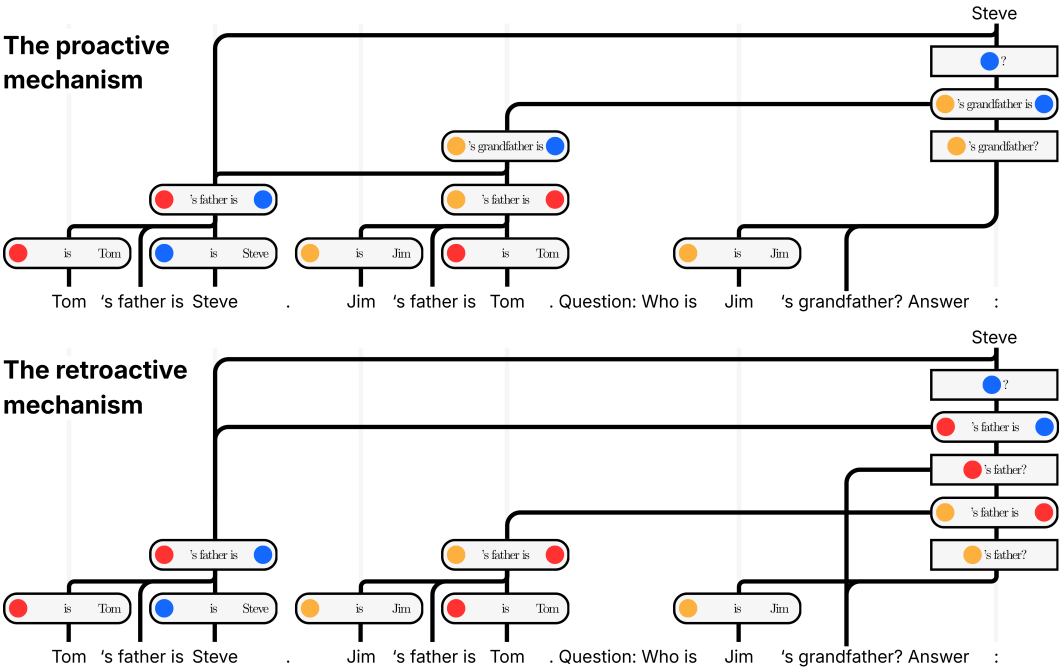

Figure 1: **How do LMs make compositional inferences?** We identify two mechanisms for doing so. The proactive mechanism forms compositional representations of new information as it is presented in-context, and uses these representations later to make next token predictions. In contrast, the retroactive mechanism reasons latently to make compositional inferences, and uses this immediately to make next token predictions. We describe these mechanisms in detail in Section 2.

The core contribution of this work is to demonstrate that LMs use two functionally independent mechanisms to make these kinds of compositional inferences. The first mechanism, which we refer to as the *proactive mechanism*, is reminiscent of the 'world models' perspective of LM in-context learning (Li et al., 2021; 2022). Here, the LM proactively composes relations as they are encountered and before this is necessary for predicting the immediate next token (e.g. by representing a full family tree as it processes individual relations). The second mechanism, which we refer to as the *retroactive mechanism*, is reminiscent of 'latent reasoning' perspectives of LM inference (Biran et al., 2024; Yang et al., 2024). Here, the LM retroactively composes relations by reasoning across what is in-context only when it is necessary for predicting the immediate next token, in a manner akin to a vertical chain of thought (i.e. by chaining together a sequence of smaller inferences).

We propose causal models of both mechanisms (Figure 1) and use causal interventions to verify their existence within the Llama 3.1 family of models (Grattafiori et al., 2024), with particular focus on Llama 3.1 405B Instruct ('405B-I'). We find that both mechanisms are independently sufficient for enabling 405B-I to make compositional inferences (Figure 2), and that we can intervene upon the intermediate variables utilized by each mechanism such that 405B-I's next token predictions change in ways that closely align with the predictions of our causal models (Figure 3). These experiments demonstrate that 405B-I uses both mechanisms to make compositional inferences, and provide evidence that the proposed mechanisms are a causal abstraction of 405B-I's computations.

We also consider *why* 405B-I uses two functionally independent mechanisms for the same problem. In particular, the proactive mechanism seems redundant when compared to the more flexible retroactive mechanism. We first demonstrate that the proactive mechanism is utilized on the basis of the model's expectations about what its future task will be (Figure 4A). We then demonstrate that the proactive mechanism can enable the model to make deeper compositional inferences than the retroactive mechanism can make alone, due to the capacity constraints of a single token position (Figure 4B). Taken together, our results therefore not only highlight that LMs such as 405B-I use two distinct and interpretable mechanisms to make compositional inferences, but also demonstrate that contextual cues can determine when LMs precompute information for downstream inference.

2 THE PROACTIVE AND RETROACTIVE MECHANISMS

We begin by describing the proactive and retroactive mechanisms for compositional inference. To make our description easier to follow, we will focus on the 'grandfather' inference that we used in the introduction both here and throughout the rest of this paper. Of course, this example represents only one of countless compositional inferences that LMs can make. We demonstrate that our mechanisms and experiments generalize to other tasks in Appendix A.3.2.

Figure 1 illustrates our causal models of the proactive and retroactive mechanisms. Each causal model describes a hierarchy of operations distributed across token positions and model layers. In the rounded boxes, both causal models posit the existence of abstract representations that either associate positional IDs with token values (e.g. '● `is Tom`' and '● `is Steve`'), or mark relationships between pairs of positional IDs (e.g. '●`'s father is` ●'). In the rectangular boxes, both causal models also posit the existence of abstract queries for these representations, such as a query for the token value of ●, '●`?`', or a query for the positional ID that represents ●'s father, '●`'s father?`'. Finally, with the black lines, both causal models posit that these abstract representations and queries interact with one another. Lower level representations such as '● `is Tom`' and '● `is Steve`' can inform the construction of higher level representations such as '●`'s father is` ●', while a query at one token position for '●`'s father?`' can be addressed by moving the representation '●`'s father is` ●' from an earlier token position to the querying token position.

These basic mechanisms, which are inspired by past studies on entity binding (e.g. Feng & Steinhardt, 2024; Dai et al., 2024; Prakash et al., 2026; Gur-Arieh et al., 2026), can be extended to explain how LMs solve compositional inference tasks in two ways:

The proactive mechanism. First, the proactive mechanism posits that LMs can combine multiple representations of relationships between positional IDs, such as '●`'s father is` ●' and '●`'s father is` ●', into a new representation that specifies an unstated relationship, such as '●`'s grandfather is` ●'. As the first row of Figure 1 depicts, this occurs as soon as the information necessary for forming this representation is available in-context and before the representation is necessary for generating a next-token prediction. It is in this sense that the mechanism is proactive. When the LM must later generate a token prediction to follow the token sequence "Who is Jim's grandfather? Answer:", it can then form a query at the final ':' token position for '●`'s grandfather?`' and directly retrieve '●`'s grandfather is` ●' from the earlier position. The LM then queries for the token value of '●' and outputs 'Steve' as its next token prediction.

The retroactive mechanism. Second, the retroactive mechanism posits that LMs can successively chain together past evidence at a single token position in order to make a compositional inference. As the second row of Figure 1 illustrates, this occurs on-demand at the token position at which the compositional inference is immediately required to make a next token prediction. In our example, this corresponds to the ':' token. Here, the model forms two successive queries to determine who "Jim's grandfather" is: first it queries for "Jim's father" ('●`'s father?`'), and then it uses the answer to that query to query for Tom's father ('●`'s father?`'). The model then finally queries for the token value associated with ● and outputs 'Steve' as its next token prediction.

3 EXPERIMENTAL EVIDENCE

Given these descriptions of the proactive and retroactive mechanisms, we now turn to verifying their existence within LMs using a range of causal interventions. We first outline our general approach before discussing two sets of results: evidence for the existence and functional independence of each mechanism, which we explore through causal mediation, and evidence for their reliance on the intermediate variables that we claim they rely on, which we explore through causal abstraction.

3.1 METHODS

Models. Our main experiments are run on Llama 3.1 405B Instruct (405B-I; Grattafiori et al., 2024) via NDIF[1](Fiotto-Kaufman et al., 2025). We chose 405B-I because it is the largest and most capable model that we could access, achieving 100% next token prediction accuracy on all of our tasks.

[1]`https://ndif.us/`

In Figure 6 and Appendix A.3.1 we also report results for two smaller models, Llama 3.1 70B Instruct and Llama 3.1 8B Instruct. There, we show that the retroactive mechanism can clearly be identified within these smaller models, but that the proactive mechanism is less pronounced. The relationship between the proactive mechanism and model size is an interesting area of exploration which we consider in Appendix A.3.1.

Dataset. We continue to focus on the 'grandfather' inference that we introduced in Section 1. Specifically we consider the following slightly more complicated prompt template:

> Use the following information to answer a question. Answer with a single word.
>
> Information: {male_3}'s mother is {female_1}. {male_3}'s father is {male_4}. {male_1}'s mother is {female_0}. {male_1}'s father is {male_2}. {male_0}'s father is {male_1}.
>
> Question: Who is {male_0}'s grandfather?
> Answer:

(1)

This prompt template includes two premises for the compositional inference alongside three distractor premises that are not necessary for answering the question. Its correct next token prediction is always '{male_2}'. Tokens such as '{male_2}' and '{female_0}' represent slots for real single-token names; we instantiate prompts from this template by sampling from a list of 135 male names and 44 female names. The full name lists are available in Appendix A.2. For all experiments, we filter out instances of this prompt where the LM under question does not predict '{male_2}' as its next token prediction for the source prompt, as well as instances where it predicts '{male_2}' as its next token prediction for the target prompt. For 405B-I this represents less than 1% of prompts; a small minority. We sample 100 instantiations of this prompt for each of our experiments.

We use this template because it allows us to design counterfactual prompts that clearly delineate between the proactive and retroactive mechanisms. We shall explore these counterfactuals in the coming sections. The inclusion of multiple distractors also makes it less likely that our LMs can respond to the prompt by relying on other simple heuristics. Although we focus on this single prompt template in our main text, we also verify our analyses on a wide range of other tasks in Appendix A.3.2. We find that the results for all tasks are consistent with our core analysis here.

Experiments. Our experiments rely on the framework of causal abstraction (Geiger et al., 2025), which we describe in greater detail in Appendix A.1.2. Broadly, we use activation patching at the residual stream. Doing so using carefully designed pairs of 'source' and 'target' prompts allows us to verify the causal models that we describe in Section 2. This is because our causal models posit the existence of specific representations within the residual stream, and therefore predict specific behavioral outcomes when these representations are removed, added, or altered via patches to it.

3.2 The existence and functional independence of the proactive mechanism

We first use the activation patching experiment illustrated in red and pink in Figure 2 to verify the existence and functional independence of the proactive mechanism. This experiment uses our base prompt template 1 as the source prompt and constructs from it a target prompt that swaps the second occurrence of '{male_1}' for '{male_5}' (the underlined token in the left panel of Figure 2).

Modifying this token gives us a source prompt where the premise information is sufficient for inferring that the answer should be '{male_2}' and a target prompt where the premise information is insufficient for inferring any answer at all. The specific prediction of our causal model for the proactive mechanism is that the third '{male_1}' token in prompt template 1 (the token highlighted in red in Figure 2), should hold a representation that attributes the positional ID associated with '{male_2}' as the grandfather of the positional ID associated with'{male_0}'. Patching from source to target at this token position should therefore drive 405B-I to predict '{male_2}' as the next token prediction for the target prompt. This is because the model can answer the question using the patched-in representation even though the target prompt does not itself contain any information to support a prediction.

Result. As Figure 2 demonstrates, we find two token positions where patching from source to target drives 405B-I to predict '{male_2}' while processing the target prompt: one at the third '{male_1}' token, highlighted in red, and another at the 'grandfather' token, highlighted in pink.

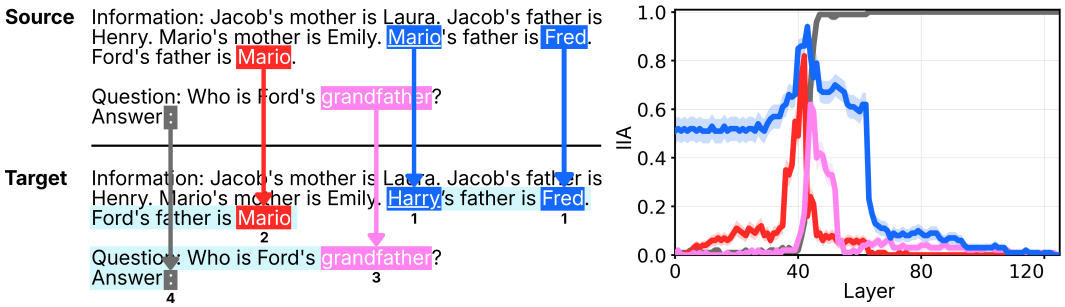

Figure 2: **Evidence for the existence and functional independence of the proactive and retroactive mechanisms.** Left: a specific instantiation of the prompt template. Right: IIA as a function of intervention layer across 100 instantiations of this template. Ribbons display SEM. Source and target prompts differ at the underlined token. Three activation patching experiments are shown. Light blue tokens are frozen for the blue patch only. There, we find that 405B-I can make a compositional inference retroactively and without the ability to reason proactively. In the red and pink patches we find that it can make the same inference proactively and without the ability to reason retroactively. The gray patch demonstrates that this occurs immediately before final outputs are determined.

Interchange intervention accuracy (IIA) peaks at layer 42 for the former and at layer 45 for the latter. In Appendix A.3.3, we explain why the representation at the latter is likely a copy of the former.

This result demonstrates that there are proactive representations that can drive 405B-I to predict '{male_2}' as the answer to a query that is not supported by the raw input and hence is not possible under the retroactive mechanism. It therefore also demonstrates the functional independence of the proactive mechanism, as the patch from source to target solely affects our causal model of the proactive mechanism and not the retroactive mechanism. As we note in Appendix A.3.4, it cannot be explained by the third '{male_1}' token storing both retroactive relations.

3.3 The existence and functional independence of the retroactive mechanism

We next use the activation patching experiment illustrated in blue in Figure 2 to verify the existence and functional independence of the retroactive mechanism. This experiment uses source and target prompts that are identical to the prompts that we use for the proactive experiment in Section 3.2. Here, however, we simultaneously patch at two token positions, the second occurrence of '{male_1}' and the first occurrence of '{male_2}', and freeze all other positions aside from the final position. We freeze tokens via a self-patch where all layers of 405B-I's residual stream are patched back into itself while it is processing the target prompt. This allows us to remove any effect that the patch from source to target has on token positions other than the final ':' token.

Under the retroactive hypothesis, the final ':' token does not rely on intermediate representations that are formed at the frozen token positions and that therefore do not exist in the patched target run. The patched target run is thus answerable only via the retroactive rather than proactive mechanism.

Result. As Figure 2 demonstrates, this patching setup drives 405B-I to predict '{male_2}' as its next token prediction if the patch occurs at early layers of the model. The effect peaks at layer 43 with an IIA of approximately 0.94. This indicates that the LM can make a compositional inference at the final ':' token position even if none of the tokens before it had access to the information needed to make that inference. This result is therefore evidence for the existence of the retroactive mechanism. It also supports the functional independence of the retroactive mechanism, as it is not possible for proactive representations to be driving this effect.

3.4 The intermediate variables of the proactive and retroactive mechanisms

Our experiments in Sections 3.2 and 3.3 support the conclusion that the proactive and retroactive mechanisms each exist and are functionally independent. However, they do not provide strong evidence for *how* either mechanism works. As Figure 1 illustrates, our causal model of the proac-

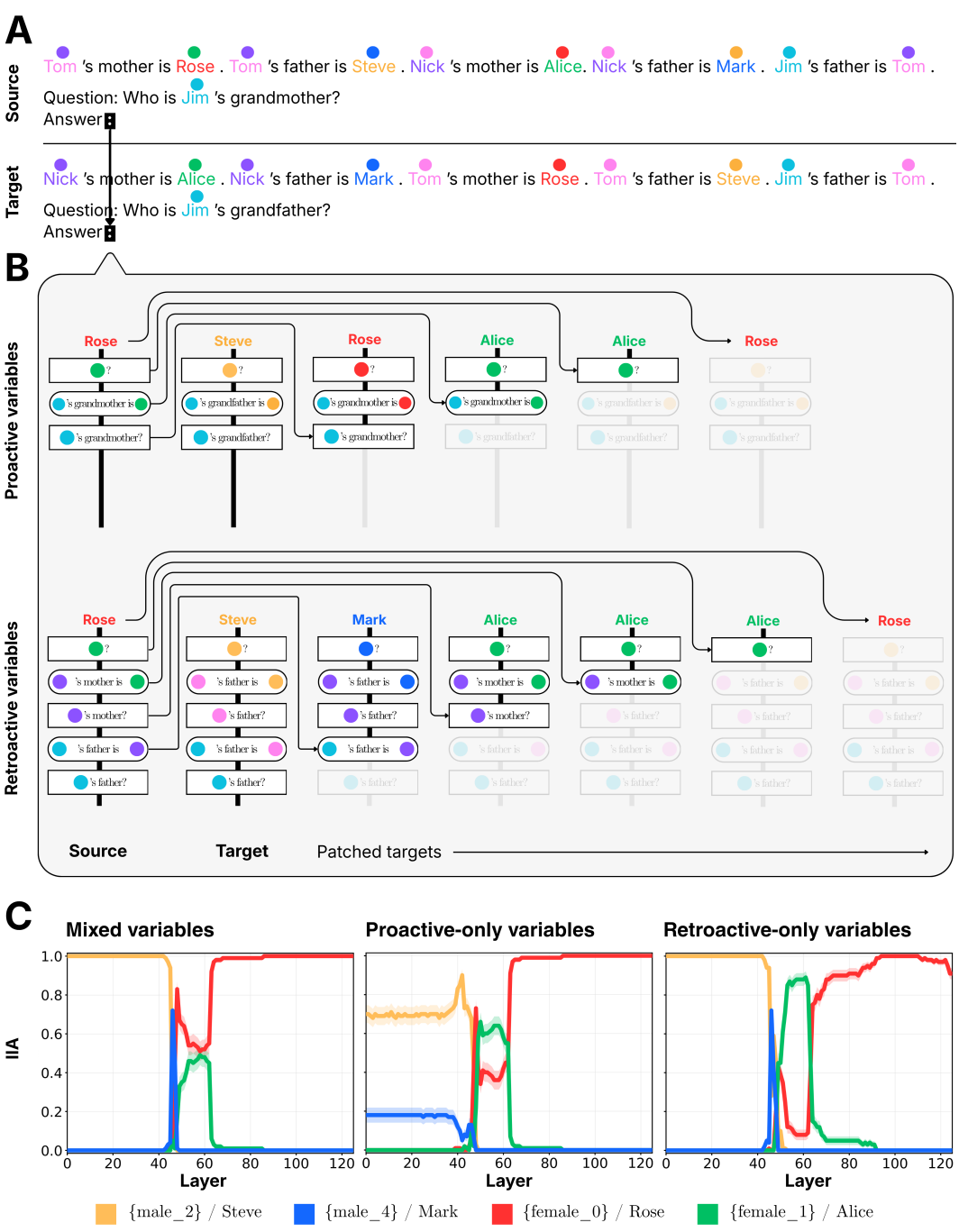

Figure 3: **Evidence for the intermediate variables of the proactive and retroactive mechanisms. A)** One instantiation of our source and target prompts for this experiment. Colored circles mark positional IDs and colored words mark the token values associated with these IDs in the target prompt. Patching occurs at the final ':' token. **B)** Our causal models for the proactive (top row) and retroactive (bottom row) mechanisms posit that the LM will hold different intermediate variables while processing the ':' token of the source (column 1) and target (column 2) prompts. They therefore predict that patching from source to target at increasing layers of the LM should drive it to produce a specific pattern of changing next token predictions (the colors of the names from the second column onward). **C)** Running this experiment produces behavior that is highly consistent with these predictions. Further details on the proactive-only and retroactive-only variables experiments are available in Appendix A.2.2.

tive mechanism makes the explicit claim that at the final ':' token an LM will query for '● 's grandfather?', retrieve a representation that '● 's grandfather is ●', query for '●?', and then retrieve 'Steve' as its next token prediction. Likewise, our model of the retroactive mechanism makes the explicit claim that the LM will query for '● 's father?', retrieve a representation that '● 's father is ●', query for '● 's father?', retrieve a representation that '● 's father is ●', query for '●?', and then finally retrieve 'Steve' as its next token prediction. In this section, we identify and intervene upon these intermediate variables.

This experiment is illustrated in Figure 3A. Here, we use prompt template 1 as our target prompt. For our source prompt, we construct a variation of prompt template 1 where we swap the positions of the first and second premises with the positions of the third and fourth premises. We also change the query from 'grandfather' to 'grandmother'. As the first two columns of Figure 3B demonstrate, this produces a setup where our causal model of the proactive mechanism predicts different intermediate variables at every equivalent layer of the LM's internal stack at the final ':' token and between the source and target prompts. Our model of the retroactive mechanism also predicts different intermediate variables at all but the early layers of the LM's internal stack. Patching layer by layer from source to target at the ':' token position should therefore allow us to individually modify these variables. Because we are modifying specific variables and not *all* variables, doing so should then produce follow-on effects that change LM predictions in observable and predictable ways. We illustrate these effects from the third column onward in Figure 3B. As we patch from source to target at increasing layers of the ':' token position, the proactive mechanism should drive the LM to predict '{male_2}', then '{female_0}', then '{female_1}', and finally '{female_0}' again. In contrast, the retroactive mechanism should drive the LM to predict '{male_2}', then '{male_4}', then '{female_1}', and finally '{female_0}'.

Result. Figure 3C demonstrates that these effects are precisely what we see when we run this patching experiment. When we naively patch from source to target at the ':' token (labeled 'mixed variables' in Figure 3C), we observe effects that are consistent with the predictions of *both* the proactive and retroactive mechanisms: namely, an initial period where '{male_2}' occupies the highest IIA followed by a brief spike in IIA for '{male_4}', another spike in IIA for '{female_0}', a spike for '{female_1}', and a final period where '{female_0}' maintains ceiling IIA. This suggests that 405B-I relies on both mechanisms simultaneously. To isolate the intermediate variables of each mechanism, we use a combination of the intervention described here and the 'freezing' intervention described in Section 3.3 so that the necessary information is only available to one of the mechanisms (we provide more details on these 'double source' interventions in Appendix A.2.2). The results of these experiments, shown as 'proactive-only variables' and 'retroactive-only variables' in Figure 3C, are even more clearly aligned with the predictions of each mechanism. Our results are therefore not only consistent with the existence and functional independence of the two mechanisms, but also with their specific intermediate variables. This also pushes back against alternate accounts of the proactive patch-in experiment, which we discuss in Appendix A.3.4.

4 Why think proactively?

We have thus far demonstrated that 405B-I uses both a proactive and a retroactive mechanism to solve our compositional inference task. This is surprising given that the retroactive mechanism seems so much more flexible than the proactive mechanism: to make inferences proactively, 405B-I must anticipate an unbounded space of potential future inferences, but to make inferences retroactively, it must only attend to what is already at hand. If both mechanisms achieve the same outcome, then why should an LM ever make inferences proactively?

We adopt a resource rational lens to answer this question. First, we demonstrate that proactive representations are utilized by 405B-I on the basis of expected utility: given small alterations to the prompt for our grandfather inference task, we find that we can manipulate the extent to which our proactive patch-in experiment from Section 3.2 has an effect (Figure 4A). Second, we demonstrate that proactive representations can allow 405B-I to accomplish tasks that require greater computational depth: given limited processing capacity at any single token position of the underlying transformer, proactive representations allow sub-computations to be offloaded to earlier token positions (Figure 4B). Together, these experiments demonstrate that 405B-I uses context to determine whether to proactively precompute information that enables it to perform more complex tasks.

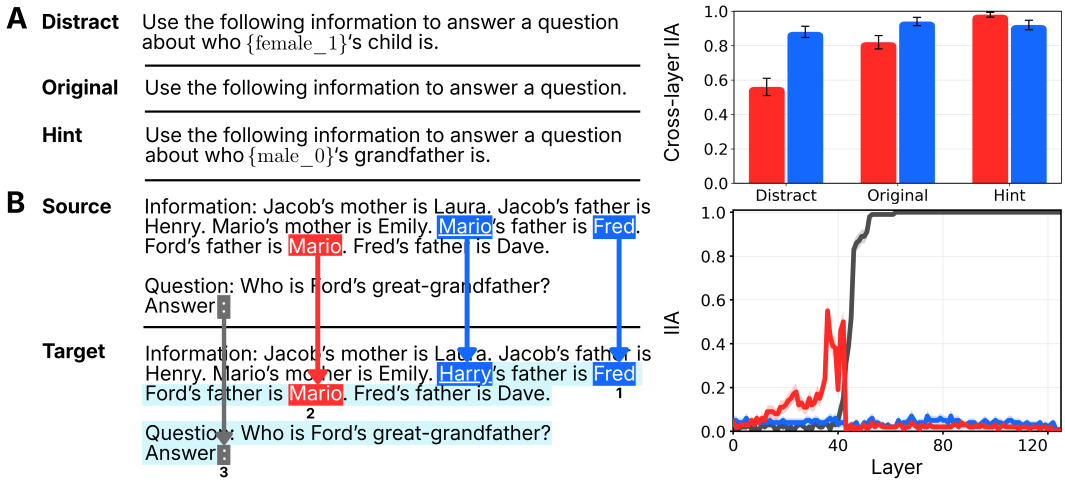

Figure 4: **Proactive representations are used rationally and support deeper compositions. A)** Left: three different preambles that we use for our proactive patch-in experiment from Section 3.2 (red) and our retroactive patch-in experiment from Section 3.3 (blue). All experiments ultimately ask the same question about who {male_0}'s grandfather is. Right: giving 405B-I a distraction or a hint before giving it the information leads to large changes in the effectiveness of the proactive patch-in (red) but not the retroactive patch-in experiment (blue). **B)** Left: an expansion of our patch-in experiment that requires an inference about {male_0}'s great-grandfather. Right: patching in the proactive representation for {male_0}'s grandfather can enable this inference when the information is insufficient (red), but enabling the retroactive mechanism does not (blue).

4.1 Proactive representations are used rationally

We first demonstrate that 405B-I's tendency to use the proactive mechanism is influenced by its expectations about what task it will be asked to accomplish and therefore what is useful to proactively compute. We manipulate these expectations by modifying the instructions that we provide the model. Specifically, we create variants of prompt template 1's preamble 'Use the following information to answer a question'. As we illustrate in the left panel of Figure 4A, we create one variation that distracts from what the question will eventually be ('answer a question about who {female_1}'s child is.'). We create another variation that hints at what the question will be ('answer a question about who {male_0}'s grandfather is.'). If 405B-I's expectations about the task influence its likelihood of using the proactive mechanism, then these modifications should affect the proactive patch-in experiment from Section 3.2 but not the retroactive patch-in experiment from Section 3.3.

Result. The right panel of Figure 4A shows that this is true: for the proactive patch-in experiment, cross-layer IIA (the proportion of prompts where '{male_2}' was the top prediction for at least one intervened layer) falls from 0.82 to 0.56 when we use the distraction preamble but rises to 0.98 when we use the hint preamble. In contrast, cross-layer IIA remains relatively stable at 0.88, 0.94, and 0.92 for the retroactive experiment. The utility of the proactive mechanism is thus clearly affected by the preambles that we provide 405B-I in a way that the retroactive mechanism is not.

4.2 Proactive representations make deeper tasks possible

Finally, we consider a patching experiment that extends prompt template 1 by adding a sixth premise, '{male_2}'s father is {male_6}.' and changing the query to be for '{male_0}'s great-grandfather' rather than '{male_0}'s grandfather'. An example of this extended prompt is shown in the left panel of Figure 4. To generate a source and target prompt, we maintain the same manipulation as before where we change the second occurrence of the '{male_1}' token such that the source prompt is sufficient while the target prompt is insufficient for answering the question.

This extension allows us to examine whether 405B-I relies upon the proactive mechanism, the retroactive mechanism, or both mechanisms to solve this harder '3-hop' inference task. If 405B-

I relies upon a proactive representation that '{male_0}'s grandfather is {male_2}' in order to infer who '{male_0}'s great-grandfather' is, then the same patch as in Section 3.2 (and shown in red in Figure 4) should enable 405B-I to predict '{male_6}' as its output under the target prompt. If 405B-I relies upon the retroactive mechanism to infer at the final ':' token position who '{male_0}'s great-grandfather' is, then the same patch as in Section 3.3 (and shown in blue in Figure 4) should enable 405B Instruct to predict '{male_6}' as its output under the target prompt.

Result. As the right panel of Figure 4 demonstrates, the proactive patch-in experiment can drive 405B-I to predict '{male_6}' under the target prompt while the retroactive patch-in experiment has largely no effect on its likelihood to predict '{male_6}' under the same prompt. This suggests that 405B-I can build upon a proactive representation of the 2-hop grandfather inference in order to make a 3-hop great-grandfather inference. In contrast, it does not seem capable of making the 3-hop inference by using only the retroactive mechanism at the final token position. Thus, one function of the proactive mechanism may be that it allows LMs to make deeper inferences with limited resources than they might otherwise be able to make with the retroactive mechanism alone.

5 Related work

Entity binding. We view this work first and foremost as an extension of the literature on LM mechanisms for entity binding. Compositional inference is a distinct task that sits downstream of entity binding, but the mechanisms we present here build on the ideas of positional IDs (Feng & Steinhardt, 2024; Dai et al., 2024) and lookbacks Prakash et al. (2026); Gur-Arieh et al. (2026) from past studies on entity binding. In particular, we extend the idea that positional IDs can be used to bind pairs of tokens by demonstrating that they can also be used to mark semantic relationships, inform the construction of higher-order representations, and enable multi-step reasoning.

World models and multi-hop reasoning. Our proactive mechanism connects with past work that examines whether LMs (and transformers more broadly) form 'world models' that proactively track the implicit state of what is specified in-context (Li et al., 2021; 2022; Tang et al., 2026). Our retroactive mechanism also connects with past work that studies how LMs perform latent multi-hop reasoning (Biran et al., 2024; Yang et al., 2024; Yao et al., 2025; Yu et al., 2025; Lindsey et al., 2025). Guo et al. (2025) in particular study a task setting similar to ours, albeit with much smaller finetuned LMs and toy transformers that lead them to different conclusions. In broader terms, our work also connects with both classic (Plotkin, 1975) and recent (Sen Sharma et al., 2026) ideas about eager and lazy computation. Our work unifies these perspectives by demonstrating that both can be true for the same task. Further, it extends the world models perspective by proposing a concrete specification for how, when, and why proactive representations are formed, and it differs from existing multi-hop reasoning work by examining multi-hop inferences about novel in-context information rather than factual recall that relies on in-weights knowledge.

Causal mediation and abstraction. Our work builds on a tradition of causal approaches to interpretability (Mueller et al., 2025; 2026). In particular, our approach to verifying the existence of the proactive and retroactive mechanisms is based on a causal mediation analysis (Vig et al., 2020; Wang et al., 2022; Meng et al., 2022), while our approach to verifying the intermediate variables of both mechanisms is based on causal abstraction (Geiger et al., 2021; 2025; Prakash et al., 2026; Gur-Arieh et al., 2026). Our work demonstrates how these approaches can continue to bear fruit in our collective efforts to understanding the internal computations of LMs.

6 Conclusion

In this work we present two mechanisms that allow LMs such as 405B-I to make compositional inferences. We demonstrate that each mechanism is functionally independent, and that each mechanism utilizes intermediate variables that we can causally intervene upon. We further demonstrate that 405B Instruct utilizes the proactive mechanism rationally – it is more likely to use proactive representations when its context indicates that they will be useful, and its use of proactive representations enables it to make deeper inferences than what the retroactive mechanism can support on its own. In summary, our work here sheds light on the emergent symbolic machinery that powers a fundamental primitive of LM reasoning abilities, and characterizes the duality of this machinery as a rational adaptation to task demands and capacity constraints.

AI USE STATEMENT

Experiment code was written with the assistance of GPT-5.6 Sol and GPT-6 Astra and then human verified. We also occasionally used ChatGPT to help with literature review. The token lists that we sampled from to generate instances of our prompt templates were also created in ChatGPT. Aside from these three use cases, all ideas, writing, and figures were human generated.

ACKNOWLEDGMENTS

We direct enormous gratitude towards the team at NDIF for building and maintaining the compute infrastructure that made this work possible.

REFERENCES

Eden Biran, Daniela Gottesman, Sohee Yang, Mor Geva, and Amir Globerson. Hopping too late: Exploring the limitations of large language models on multi-hop queries. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 14113–14130, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Qin Dai, Benjamin Heinzerling, and Kentaro Inui. Representational analysis of binding in language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17468–17493, 2024.

Jiahai Feng and Jacob Steinhardt. How do language models bind entities in context? In *International Conference on Learning Representations*, volume 2024, pp. 36391–36413, 2024.

Jaden Fiotto-Kaufman, Alexander Loftus, Eric Todd, Jannik Brinkmann, Koyena Pal, Dmitrii Troitskii, Michael Ripa, Adam Belfki, Can Rager, Caden Juang, et al. Nnsight and ndif: Democratizing access to open-weight foundation model internals. In *International Conference on Learning Representations*, volume 2025, pp. 92337–92370, 2025.

Jerry A Fodor and Zenon W Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71, 1988.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in neural information processing systems*, 35:30583–30598, 2022.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. *Advances in neural information processing systems*, 34:9574–9586, 2021.

Atticus Geiger, Duligur Ibeling, Amir Zur, Maheep Chaudhary, Sonakshi Chauhan, Jing Huang, Aryaman Arora, Zhengxuan Wu, Noah Goodman, Christopher Potts, and Thomas Icard. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 26(83):1–64, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Tianyu Guo, Hanlin Zhu, Ruiqi Zhang, Jiantao Jiao, Song Mei, Michael I Jordan, and Stuart Russell. How do llms perform two-hop reasoning in context? *arXiv preprint arXiv:2502.13913*, 2025.

Yoav Gur-Arieh, Mor Geva, and Atticus Geiger. Mixing mechanisms: How language models retrieve bound entities in-context. In *International Conference on Learning Representations*, volume 2026, pp. 27757–27786, 2026.

Najoung Kim and Sebastian Schuster. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3835–3855, 2023.

Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *arXiv preprint arXiv:1604.00289*, 2016.

Belinda Z Li, Maxwell Nye, and Jacob Andreas. Implicit representations of meaning in neural language models. In *Proceedings of the 59th annual meeting of the Association for Computational Linguistics and the 11th international joint conference on natural language processing (Volume 1: Long papers)*, pp. 1813–1827, 2021.

Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. *arXiv preprint arXiv:2210.13382*, 2022.

Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Aaron Mueller, Atticus Geiger, Sarah Wiegreffe, Dana Arad, Iván Arcuschin, Adam Belfki, Yik Siu Chan, Jaden Fiotto-Kaufman, Tal Haklay, Michael Hanna, et al. Mib: A mechanistic interpretability benchmark. *arXiv preprint arXiv:2504.13151*, 2025.

Aaron Mueller, Jannik Brinkmann, Millicent Li, Samuel Marks, Koyena Pal, Nikhil Prakash, Can Rager, Aruna Sankaranarayanan, Arnab Sen Sharma, Jiuding Sun, Eric Todd, David Bau, and Yonatan Belinkov. The quest for the right mediator: Surveying mechanistic interpretability for NLP through the lens of causal mediation analysis. *Computational Linguistics*, 52(1):331–378, March 2026. doi: 10.1162/coli.a.572. URL https://aclanthology.org/2026.cl-1.10/.

Gordon D. Plotkin. Call-by-name, call-by-value and the λ-calculus. *Theoretical computer science*, 1(2):125–159, 1975.

Nikhil Prakash, Natalie Shapira, Arnab Sen Sharma, Christoph Riedl, Yonatan Belinkov, Tamar Shaham, David Bau, and Atticus Geiger. Language models use lookbacks to track beliefs. In *International Conference on Learning Representations*, volume 2026, pp. 7657–7694, 2026.

Arnab Sen Sharma, Giordano Rogers, Natalie Shapira, and David Bau. Llms process lists with general filter heads. In *International Conference on Learning Representations*, volume 2026, pp. 63227–63263, 2026.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L Hamilton. Clutrr: A diagnostic benchmark for inductive reasoning from text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4506–4515, 2019.

Zilu Tang, Qiao Zhao, Gabriel Franco, Derry Wijaya, Aaron Mueller, Sebastian Schuster, and Najoung Kim. Do language models track entities across state changes? *arXiv preprint arXiv:2605.30233*, 2026.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Simas Sakenis, Jason Huang, Yaron Singer, and Stuart Shieber. Causal mediation analysis for interpreting neural nlp: The case of gender bias. *arXiv preprint arXiv:2004.12265*, 2020.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. Do large language models latently perform multi-hop reasoning? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10210–10229, 2024.

Yuekun Yao, Yupei Du, Dawei Zhu, Michael Hahn, and Alexander Koller. Language models can learn implicit multi-hop reasoning, but only if they have lots of training data. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 9695–9713, 2025.

Zeping Yu, Yonatan Belinkov, and Sophia Ananiadou. Back attention: Understanding and enhancing multi-hop reasoning in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 11268–11283, 2025.

Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. In *International Conference on Learning Representations*, volume 2024, pp. 1651–1678, 2024.

A Appendix

A.1 Definitions

A.1.1 Compositional inference

We define compositional inferences as inferences that depend on combining two or more relations provided in-context. Let $\mathcal{E}$ denote a set of entities, $r_1, \ldots, r_k$ denote possible relations over elements of $\mathcal{E}$, q denote the relation expressed by a query, $e_0 \in \mathcal{E}$ denote the entity to which the query applies, and C denote a composition rule such that $C(r_1, \ldots, r_k) = r_k \circ \cdots \circ r_1 = q$. This rule may be supplied by prior knowledge, specified in-context, or inferred from the task itself. Given in-context facts of the form $r_1(e_0) = e_1, \ldots, r_k(e_{k-1}) = e_k$, a compositional inference is one that finds an answer entity e_k that satisfies $q(e_0) = e_k$. For example, given $r_{\text{father}}(\text{Jim}) = \text{Steve}$, $r_{\text{father}}(\text{Joe}) = \text{Jim}$, and $C(r_{\text{father}}, r_{\text{father}}) = r_{\text{father}} \circ r_{\text{father}} = r_{\text{grandfather}}$, one such compositional inference is the answer to the query $r_{\text{grandfather}}(\text{Joe})$, which in this case is Steve.

A.1.2 Causal abstraction

Our investigation of proactive and retroactive mechanisms relies on the framework of causal abstraction (Geiger et al., 2021; Zhang & Nanda, 2024). At the heart of this framework lies an *abstract causal model* of each mechanism: a hypothesized set of intermediate variables and causal dependencies that describes how information for a given input sequence X is transformed by an LM into an output token Y. The causal model is useful because it allows us to posit the existence of mediating variables M where $X \rightarrow M \rightarrow Y$ (Mueller et al., 2026), and more generally it allows us to represent the LM's underlying computations as a directed graph $G = (V, E)$ whose variables $V = X, M_1, \ldots, M_n, Y$ represent computational states at specific layers and token positions of the LM, and whose edges specify their causal dependencies (Geiger et al., 2025). Importantly, these variables do not need to correspond to individual neurons or components of the LM, and can instead be used to describe higher-level states that may be instantiated by specific groups of neurons.

We use *interchange interventions* to verify our causal models. Consider a *source* input x_s and a *target* input x_t with different expected LM outputs $Y(x_s)$ and $Y(x_t)$. Interchange interventions involve the construction of inputs x_s and x_t such that a mediator in the causal model takes different values $M(x_s)$ and $M(x_t)$. Because our causal models make explicit claims about how their variables depend on one another, we can derive from them a concrete prediction about how the LM's output $Y_{M \leftarrow M(x_s)}(x_t)$ should change when we replace $M(x_t)$ with $M(x_s)$ while processing x_t. Because our causal models also make explicit claims about the absolute token position and the relative layer position of their variables, we can also derive from them a concrete prediction about *where* this intervention on the LM should produce the predicted change in output. We can then test

these predictions by patching internal activations from the source run into the target run across token positions and layers; here, an intervention that produces the predicted behavioral change at the predicted location and not at other locations provides evidence for both the existence and location of a hypothesized mediator. In aggregate, these experiments can then provide evidence for a given causal model as a whole.

A.1.3 POINTERS

The causal models that we refer to in this work build on prior work from the entity binding literature that introduces the concept of a *pointer* (Prakash et al., 2026; Gur-Arieh et al., 2026). The pointer mechanism posits that a given in-context entity token at position i is marked with an abstract positional ID z_i which can be passed to a later token when information about the first entity is encountered at the later token. This serves two purposes. First, it allows the LM to represent that two tokens at different positions within an input sequence are referring to the same underlying entity. Second, it supports retrieval: z_i can be transformed into a key $k(z_i)$ at the original token position and a query $q(z_i)$ at a later token position that allows attention heads at the later position to attend back to the original position and copy information stored there through its value projection. In this way, the mechanism allows the LM to pass forward a lightweight reference, ie. a 'pointer', to an earlier entity and retrieve information associated with that entity when needed.

In our causal models of the proactive and retroactive mechanisms, we use the pointer mechanism as a primitive for transmitting intermediate relational information between token positions. We extend the pointer mechanism such that positional IDs z_i can be transformed into keys and queries that mark relations of IDs $r_k(z_i)$ in addition to just the IDs themselves.

A.2 EXTENDED METHODS

A.2.1 ENTITY NAMES

Here we list the token lists that we sample from to instantiate prompts from our prompt templates (i.e. prompt template 1, and the additional templates shown in Figure A.3.2). In total we use lists of 16 tokens for building types, 44 tokens for female names, 135 tokens for male names, 10 tokens for tools, and 13 tokens for vehicle types. Lists of candidate words were initially generated with the assistance of ChatGPT before they were filtered down to single token words according to Llama 3.1's tokenizer.

female names

```
Mary, Elizabeth, Jennifer, Susan, Sarah, Jessica, Karen, Lisa, Anna,
Emily, Carol, Michelle, Laura, Emma, Amy, Virginia, Alice, Rachel,
Maria, Grace, Victoria, Rose, Ann, Charlotte, Jane, Ruby, Crystal,
Anne, April, Chelsea, Beth, Georgia, Ana, Ada, Hope, Kate, Summer,
May, Hillary, Una, Meg, Liv, Arial, Vy
```

male names

```
James, Michael, Robert, John, David, Christopher, Richard, Mark,
Jason, Thomas, Charles, Gary, Daniel, Brian, Andrew, Jacob, Justin,
Larry, Joseph, Scott, Kevin, Brandon, George, Paul, Anthony, Adam,
Eric, Stephen, Jeremy, Edward, Alexander, Jerry, Frank, Christian,
Todd, Aaron, Patrick, Bruce, Roger, Jose, Jack, Jackson, Keith, Mike,
Sean, Peter, Henry, Steve, Craig, Arthur, Luke, Albert, Connor, Harry,
Jeff, Alan, Carl, Tony, Juan, Joe, Howard, Roy, Leo, Johnny, Louis,
Tim, Carlos, Ian, Fred, Jesus, Philip, Martin, Cole, Marcus, Jim,
Rick, Greg, Jay, Allen, Tom, Dean, Jon, Roman, Bill, Jake, Don, Grant,
Gene, Mario, Bob, Lewis, King, Leon, Brad, Kent, Andre, Ty, Guy,
Clinton, Theo, Marco, Neil, Ace, Ron, Ross, Doug, Ken, Ted, Simon,
Hugh, Junior, Wilson, Chuck, Ben, Neal, Prince, Dick, Nick, Pat, Matt,
Will, Ed, Phil, Ford, Case, Hal, Tre, Trace, Josh, Rod, Al, Stan,
Mitch, Mac, Russ
```

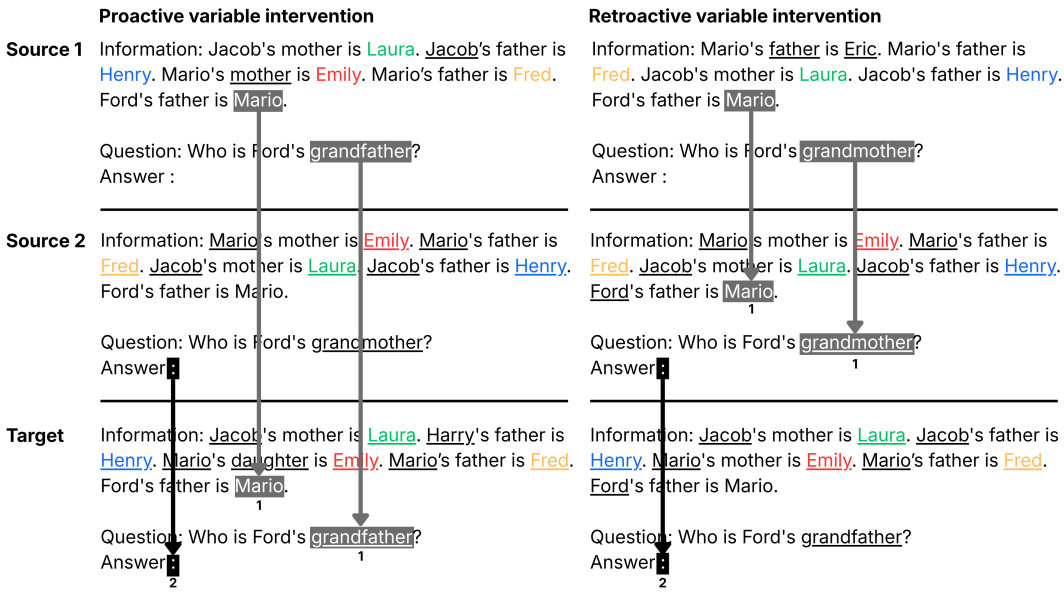

Figure 5: **Double-source activation patching prompt examples.** Gray patches are executed at all layers, while black patches are executed layer by layer.

tools

```
hammer, axe, knife, hoe, file, level, brush, clamp, pick, torch
```

vehicles

```
car, truck, van, bus, train, boat, ship, jet, bike, cab, tractor,
tram, rocket
```

buildings

```
warehouse, shed, house, store, shop, factory, station, tower, castle,
school, church, hospital, hotel, office, library, arena
```

A.2.2 'DOUBLE-SOURCE PATCHING' TO DISENTANGLE INTERMEDIATE VARIABLES

In Figure 3C we include results for two patching experiments that rely on a 'double source' patching design that we did not describe in main text: the 'proactive-only variables' experiment, and the 'retroactive-only variables' experiment. These experiments were motivated by our realization that patching from source to target in the naive mixed variables setting (depicted in Figure 3A) was muddying our results because it involved patching both proactive and retroactive representations which sometimes pulled in different directions. Although the final result in the left panel of Figure 3C did resemble a mix of both mechanisms, we were unsatisfied with this sole result and wanted to design experiments that could intervene on only the proactive variables, or only the retroactive variables.

These experiments are depicted in Figure 5. The broad idea here is that we can patch from two source prompts, either sequentially or in parallel, in order to negate the effect of swapping representations for one mechanism while maintaining the effect of swapping representations of the other, under the experimental design of Figure 3A. In the proactive-only variable intervention (left column of Figure 5), we use a parallel approach where we patch from the first source prompt into the target prompt at all layers while patching from the second source prompt into the target prompt incrementally. Here, the two source prompts and the target prompt is designed to prevent the model from producing outputs via the retroactive mechanism, while enabling it to use the proactive mechanism. In the retroactive-only variable intervention (right column of Figure 5), we use a sequential approach

where we first patch from the first source prompt into the second source prompt at all layers, and then we patch from the modified second source prompt into the target prompt. Here, the prompts are designed to prevent the model from using the proactive mechanism, while enabling it to use the retroactive mechanism.

A.3 Extended results

A.3.1 Other models

We ran the experiments detailed in Sections 3.2, 3.3, and 3.4 across the Llama 3.1 Instruct family of models and display their results in Figure 6. Results for 405B Instruct, which are identical to what is shown and discussed in main text, are included for ease of comparison. Our results demonstrate that both Llama 3.1 8B Instruct and Llama 3.1 70B Instruct do not have a clear tendency to use the type of proactive representation that we detail in Section 2. Although 8B Instruct achieves an IIA of just under 0.5 when we perform the proactive patch-in experiment at token 46 (red line, top left panel of Figure 6) and an IIA of just under 0.7 when we perform the retroactive patch-in experiments at tokens 36 and 40 (with all other tokens frozen, blue line of top left panel), its proactive-only and retroactive-only patches produce results that diverge from the predictions of our causal models of either mechanism. 70B Instruct, on the other hand, produces results that are clearly consistent with a retroactive-only approach to solving the grandfather inference: there is little effect for the proactive-patchin experiment (red and pink lines, top-middle panel), while the mixed and retroactive-only variables experiments are perfectly aligned with the predictions of the retroactive mechanism, and the proactive-only variables experiment is neatly aligned with the predictions of a missing proactive mechanism.

These results correspond with differences in task accuracy: across 1000 instantiations of prompt template 1, 8B Instruct predicts {`male_2`} as its next token prediction 81.9% of the time while 70B Instruct predicts {`male_2`} 92.9% of the time and 405B Instruct predicts {`male_2`} 100% of the time. They therefore support (but do not confirm) the hypothesis that the proactive and retroactive mechanisms are necessary for making robust compositional inferences. Our broader hypothesis is that 8B Instruct utilizes a mixture of heuristics to make the inference, ie. it does not necessarily rely on a reasoning process that is as amenable to the clean and symbolic characterization of the proactive and retroactive mechanisms, and that the proactive and retroactive mechanisms may emerge with scale, with the retroactive mechanism emerging first and the proactive mechanism emerging later.

A.3.2 Other tasks

We also ran the experiments detailed in Sections 3.2 and 3.3 on a number of other tasks using 405B Instruct. These tasks include a granddaughter inference task, a syllogisms task, a containment task, and a locations task, detailed in Figure 7A. 405B Instruct achieves 100% accuracy for each of these tasks. As Figure 7B demonstrates, our proactive and retroactive patch-in experiments tend to work just as well on these tasks as they do on the grandfather inference task that we focus on in main text. Our results therefore indicate that the proactive and retroactive mechanisms apply to each of these additional tasks.

One caveat to our task selection is that all of our tasks share the same higher level structure, where: 1. the final two premises are the ones required for the compositional inference, 2. the final premise corresponds to the first premise in the inference (ie. 'B $\rightarrow$ C. A $\rightarrow$ B'), and 3. the target prompt modifies the source prompt by changing the first entity in the second last premise. In exploratory work, we found that these details do sometimes matter with respect to the utility of the proactive and retroactive mechanisms. For example, separating the two premises required for the compositional inference by presenting a number of irrelevant premises in between them can lead to decreased utility of the proactive mechanism. We suspect that this relates to our point in Section 4; it seems plausible that by separating the two premises, we also reduce the LM's expectation that it will be asked a question that requires it to connect them. Another example is that our task setup is sensitive to which token is perturbed between the source and target prompts. If we modify the first entity of the final premise rather than the first entity of the second last premise, we find that 405B Instruct under the patched target overwhelmingly prefers to predict ' Not' rather than any of the entity tokens. This preference is strong enough to wash away the effects of the proactive and retroactive patch-ins. Here we hypothesize that there are other mechanisms at play, such as

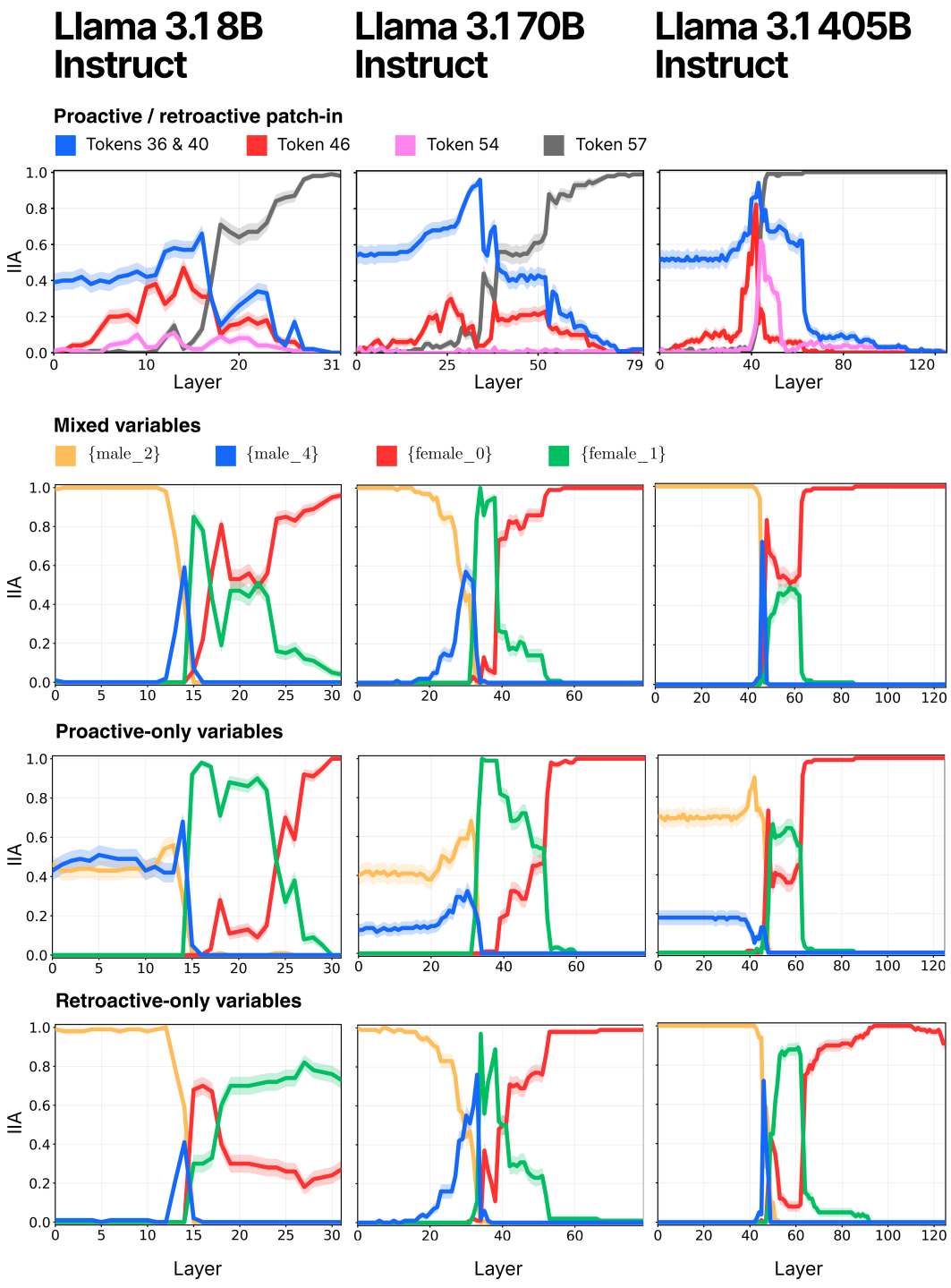

Figure 6: **Experimental results for other Llama 3.1 Instruct models on the grandfather task.** We run the same experiments as in Sections 3.2, 3.3, and 3.4 on Llama 3.1 8B Instruct (column 1) and Llama 3.1 70B Instruct (column 2). The same results for Llama 3.1 405B Instruct that we displayed in the main text are also included here in column 3 for ease of comparison. 70B Instruct clearly uses the retroactive mechanism to solve the grandfather task while 8B Instruct does not appear to fit either mechanism's predictions well.

a mechanism that independently determines whether the question is answerable given the premise statements regardless of what retroactive and proactive representations are available to the LM, and that this mechanism for some reason activates more strongly in one condition than the other.

Broadly, then, we conclude that the mechanisms we discuss in this work do indeed generalize across different task settings. However, understanding how these mechanisms interact with other mechanisms and expanding our understanding of the resource-rational perspective on proactive computation remain as interesting topics of future investigation that we are excited to pursue.

A.3.3 The proactive representation at token position 54

We were initially surprised to find an effect at token position 54, i.e. the 'grandfather' token in the question, which is not specifically predicted by our causal model of the proactive mechanism in Figure 1. Is the proactive representation at position 54 independently formed, or is it copied from position 46 (the third '`male_{1}`' token)? To answer this question we ran one additional experiment where we masked 405B-I's ability to attend from token position 54 to token position 36 (the token that differs between source and target) while processing the source prompt, and another where we masked its ability to attend from token position 54 to token position 46 while processing the source prompt. We found that performing the same activation patching experiment did not affect the former, but collapsed the effect for the latter. This suggests that the proactive representation at position 54 is a later copy of the representation at 46, rather than an independently formed representation.

A.3.4 The 'Attribute enrichment' hypothesis

One alternate hypothesis that could also explain our proactive patch-in results from Section 3.2 is that 405B-I aggregates relations at the third '`{male_1}`' position but does not actually compose the information proactively (i.e. in Figure 1, it may store 'Jim's father is Tom' and 'Tom's father is Steve' over 'Tom', but not 'Jim's grandfather is Tom'). We will refer to this hypothesis as the 'attribute enrichment' hypothesis. Although Figure 2 demonstrates that the third '`{male_1}`' token position mediates the answer to the query 'Who is `{male_0}`'s grandfather', it could also be true that it does so by storing both '`{male_1}`'s father is `{male_2}`' and '`{male_0}`'s father is `{male_1}`'. Under the attribute enrichment hypothesis, our proactive mechanism would just be a different instance of the retroactive mechanism, where the mediating variables compress from two token positions into one.

The strongest piece of evidence that we have against this hypothesis is that the proactive patch-in experiment leads to only a 5% cross-layer IIA for '`{male_2}`' if we change the question to be 'Who is `{male_1}`'s father' rather than 'Who is `{male_0}`'s grandfather'. In other words, patching at the third '`{male_1}`' position, i.e. the same position that we claim carries a proactive representation, does not lead to an effect that is predicted by attribute enrichment. In addition, Figure 3 also illustrates what is stored within the final ':' token: here, the early red peak in IIA that is visible in both the mixed and proactive-only columns of Figure 3C suggests that our patching experiments are replacing the query 'Who is `{male_0}`'s grandfather' with 'Who is `{male_0}`'s grandmother'. This is the only explanation that we have for the early red peak, which is not otherwise predicted by either the attribute enrichment hypothesis or the retroactive mechanism. The attribute enrichment hypothesis also does not explain or predict our great-grandfather intervention experiment from Section 4.2 – under this hypothesis, there is no clear reason why the proactive patch-in experiment should produce an effect when the retroactive patch-in experiment does not. We therefore treat the attribute enrichment hypothesis as an interesting hypothesis for a potential third mechanism that operates in parallel to our proactive and retroactive mechanisms, but not as one that displaces the validity of the proactive mechanism.

A

	Source	Target
Grand-father	Use the following information to answer a question. Answer with a single word. Information: {male_3}'s mother is {female_1}. {male_3}'s father is {male_4}. {male_1}'s mother is {female_0}. {male_1}'s father is {male_2}. {male_0}'s father is {male_1}. Question: Who is {male_0}'s grandfather? Answer:	Use the following information to answer a question. Answer with a single word. Information: {male_3}'s mother is {female_1}. {male_3}'s father is {male_4}. {male_1}'s mother is {female_0}. {male_5}'s father is {male_2}. {male_0}'s father is {male_1}. Question: Who is {male_0}'s grandfather? Answer:
Grand-daughter	Use the following information to answer a question. Answer with a single word. Information: {female_3}'s son is {male_2}. {female_3}'s daughter is {female_4}. {female_1}'s son is {male_1}. {female_1}'s daughter is {female_2}. {male_2}'s daughter is {female_1}. Question: Who is {male_2}'s granddaughter? Answer:	Use the following information to answer a question. Answer with a single word. Information: {female_3}'s son is {male_2}. {female_3}'s daughter is {female_4}. {female_1}'s son is {male_1}. {female_5}'s daughter is {female_2}. {male_2}'s daughter is {female_1}. Question: Who is {male_2}'s granddaughter? Answer:
Syllogisms	Use the following information to answer a question. Answer with a single word. Information: All {letter_4} are taller than {letter_5}. All {letter_4} are shorter than {letter_6}. All {letter_1} are taller than {letter_2}. All {letter_1} are shorter than {letter_3}. All {letter_0} are a type of {letter_1}. Question: What are all {letter_0} shorter than? Answer:	Use the following information to answer a question. Answer with a single word. Information: All {letter_4} are taller than {letter_5}. All {letter_4} are shorter than {letter_6}. All {letter_1} are taller than {letter_2}. All {letter_7} are shorter than {letter_3}. All {letter_0} are a type of {letter_1}. Question: What are all {letter_0} shorter than? Answer:
Containers	Use the following information to answer a question. Answer with a single word. Information: The {vehicle_1} is owned by {person_1}. The {vehicle_1} is in the {building_1}. The {vehicle_0} is owned by {person_0}. The {vehicle_0} is in the {building_0}. The {tool_0} is in the {vehicle_0}. Question: Which building contains the {tool_0}? Answer:	Use the following information to answer a question. Answer with a single word. Information: The {vehicle_1} is owned by {person_1}. The {vehicle_1} is in the {building_1}. The {vehicle_0} is owned by {person_0}. The {vehicle_2} is in the {building_0}. The {tool_0} is in the {vehicle_0}. Question: Which building contains the {tool_0}? Answer:
Locations	Use the following information to answer a question. Answer with a single word. Information: {person_3} is one floor above {person_4}. {person_3} is one floor below {person_5}. {person_1} is one floor above {person_6}. {person_1} is one floor below {person_2}. {person_0} is one floor below {person_1}. Question: Who is {person_0} two floors below? Answer:	Use the following information to answer a question. Answer with a single word. Information: {person_3} is one floor above {person_4}. {person_3} is one floor below {person_5}. {person_1} is one floor above {person_6}. {person_7} is one floor below {person_2}. {person_0} is one floor below {person_1}. Question: Who is {person_0} two floors below? Answer:

B

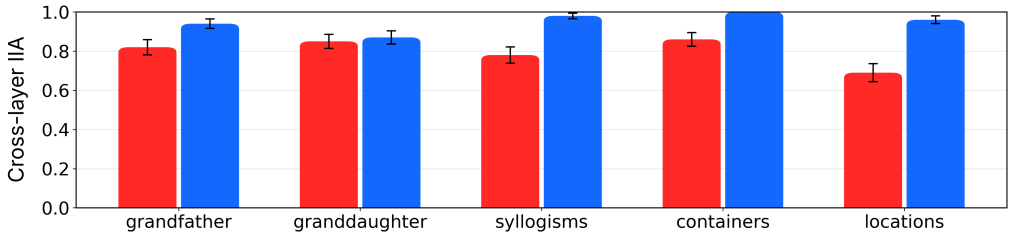

Figure 7: **Other tasks.** **A)** Prompt templates for four other compositional inference task settings, alongside our original grandfather inference task in the top row. **B)** Cross-layer IIA for our proactive (red) and retroactive (blue) patch-in experiments for each prompt template.