$$S_{ij}^{\text{SHARDED}} = \begin{cases} 100, & \text{if } 1 \leq i \leq 3 \\ 100, & \text{if } 4 \leq i \leq 8, 1 \leq j \leq 6 \\ 0, & \text{otherwise} \end{cases} .$$

In situation 3, $\overline{P} = 60\%$, with an aptitude of $A = 80\%$, and a unreliability of $U = 60\%$. Note that situation 3 leads to a larger increase in unreliability (from 0% to 60%) than a decrease in aptitude (from 90% to 80%) when compared to fully-specific simulations. This corresponds in practice to our observation: drops in performance are explained by small drops in aptitude and large drops in reliability.

Finally, we note that though this concrete example we provide uses binary scores (0 and 100) for simulated conversation outcomes, aptitude (A) and unreliability (U) can equally be applied to continuous metrics (such as BLEU).

Appendix F Qualitative Analyses of Simulation Logs

In the following subsections, we report qualitative analyses on the corpus of simulations from the main experiment (Section 6.1). The purpose of the analyses is to discern root causes in model behavior that lead to performance degradation. We identify four behaviors below and provide the analysis for each item in the rest of the section:

1. LLMs attempt to answer the entire problem prematurely.
2. LLMs overly rely on previous (incorrect) answer attempts, leading to lengthier "bloated" answers.
3. LLMs overly adjust their answers based on the last conversation turn, materialized by a pronounced forgetting of middle-turns.
4. LLMs produce answers that are overly verbose, which likely introduce problem assumptions that detract attention from user-utterances.

F.1 Premature Answer Attempts

	Conversation Progress At First Answer Attempt				
Model	0-20%	20-40%	40-60%	60-80%	80-100%
First answer attempt is ...	earliest	early	midway	late	latest
3.1-8B	16.1	24.0	35.3	39.6	39.7
OLMo2	17.6	32.7	37.7	47.3	26.4
3-Haiku	27.1	35.6	47.4	58.9	70.3
4o-mini	30.2	39.2	48.4	58.2	59.9
3.3-70B	33.3	40.1	51.2	60.0	69.3
Phi-4	25.7	33.1	47.0	53.0	57.9
CMD-A	38.0	42.9	56.5	65.5	73.5
4-Scout	39.8	36.8	51.0	57.9	64.8
o3	21.0	37.9	51.9	58.4	68.0
3.7-Sonnet	29.2	35.6	55.3	68.0	71.6
R1	39.5	43.1	53.5	66.4	50.2
4o	36.0	41.4	56.2	65.6	90.4
2.5-Flash	39.0	48.6	60.2	70.8	74.6
4.1	33.9	52.7	60.6	69.0	78.6
2.5-Pro	41.1	45.7	53.5	64.6	63.8
Average	30.9	40.5	51.7	60.4	64.4

Table 6: Averaged performance ($\overline{P}$) breakdown, based on how early in the conversation the LLM makes its first answer attempt. Analysis conducted on simulations of two tasks: Code and Math.

During SHARDED simulation, responses are classified according to a seven-class conversation strategy categorization. In particular, each assistant response is tagged as being a formal *answer attempt* or not (as answer attempts require further processing: extraction and evaluation by the task-specific evaluator).

On the onset of conversation, LLMs have the least amount of information (highest level of underspecification), and are least likely to formulate correct answer attempts. Proposing a solution early might therefore plant certain incorrect elements in it, which wrongly influences the interaction later in the conversation.

To evaluate this hypothesis, we bin all simulated conversations from our experiments based on how early in the conversation the first answer attempt is generated by the LLM. Specifically, we create five bins: `0-20%` if the first answer attempt occurs within the first 20% turns of the conversation, and `20-40%`, `40-60%`, `60-80%`, and `80-100%` if it occurs in later turns of the conversation. Of the six tasks included in our experiments, only two (Math and Code) observed a significant range in LLM behavior for answer attempt timing. For the other four tasks, models attempt an answer from the first turn in most of the time, rendering analysis on this parameter impossible.

Analysis results for the two remaining tasks are presented in Table 6. We observe that for every single model, conversations with a later first answer attempt lead to higher averaged performance. Across all models, conversations with the first attempt being made in the first 20% of conversations achieve a score of 30.9, less than half of the 64.4 when the LLM waits for the last 20% of the conversation to make an answer attempt.

In other words, we find that premature answer attempts detract LLM performance. Conversations where the model clarifies user instructions or discusses the problem at a high-level before moving to generating complete answer attempts lead to higher performance. We hypothesize that this is due to the model making incorrect assumptions in premature solutions, which conflict with subsequent user instructions in later turns.

F.2 Answer Bloat in Multi-Turn Conversation

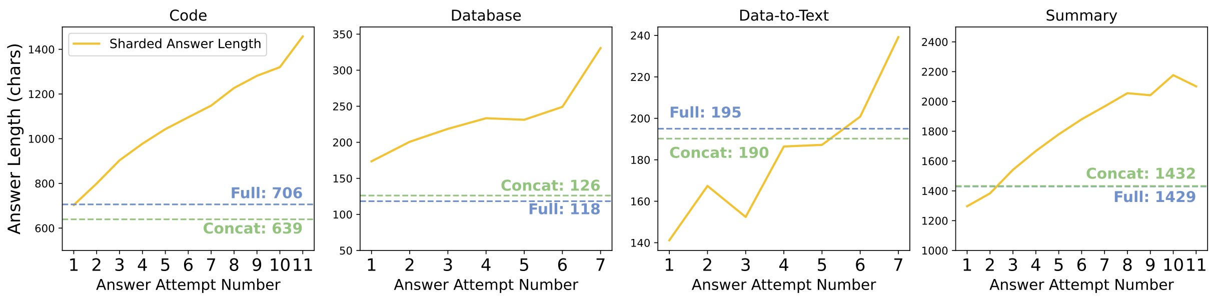

Figure 9: Average length (in number of characters) of answer attempts across four tasks (Code, Database, Data-to-text, and Summary) in SHARDED conversations. Answer attempts in the FULL and CONCAT settings tend to be shorter on average than those from SHARDED setting. SHARDED answer attempts increase in length as the LLMs make more answer attempts.

In multi-turn conversation simulations, the LLM might make multiple answer attempts, with each subsequent attempt being potentially based on previous attempts. In contrast, single-turn conversations constrain conversation dynamics, with the LLM making a single, first-and-final answer attempt.

To understand multi-turn conversation dynamics, we calculate the average length of answer attempts in each simulation type. For the SHARDED setting, we calculate average length for each attempt within a simulation (*i.e.*, average length of the first attempt, second attempt, third attempt, etc.). We note for readers here that the analysis is conducted on extracted answer attempts (output of the Answer Extractor module in Figure 3) rather than the entire assistant responses. The extracted answer more accurately measures dynamics in answer attempts (i.e., generated SQL query, or Python function) rather than the entire responses, which might contain varying amounts of unrelated content.

Results of the analysis are plotted in Figure 9. Across the four tasks, we find that answer lengths in the FULL and CONCAT settings tend to be similar, typically within 2-10% of each other. On three of the analyzed tasks (Code, Database, Summary), the first answer attempt in the SHARDED setting has a similar length to FULL and CONCAT counterparts, yet for each subsequent answer attempt, we observe an increase in average answer length. The effect is such that the final answer attempts in SHARDED conversations (right portion of the four plots) tend to be 20-300% longer than the solutions generated in the FULL and CONCAT settings. We name this observation the *answer bloat effect*: as a multi-turn conversation progresses, the LLM generates incorrect answer attempts, making assumptions about portions of the instruction that remain unspecified. As the user reveals additional information in succeeding turns, the LLM does not successfully invalidate its prior assumptions and overly relies on its previous attempts. Answer bloat in multi-turn, underspecified conversation leads to longer solutions compared to single-turn equivalents.

We perform an additional analysis, focusing only on the Code and Database tasks and filtering to simulations where the LLM reaches an entirely correct solution (score of 100.0). For Code task, correct programs obtained from SHARDED setting are on average 850 characters long, which is 27% more characters than the correct solutions generated in the FULL setting (668 characters on average). For Database, correct SQL queries in the SHARDED setting are on average 129 characters, 14% more characters than those from the FULL setting (113 characters). In summary, LLMs are less likely to reach a correct solution in multi-turn settings (lower $\overline{P}$), and when they do, the final solutions they reach are longer (bloated), hinting that the solutions are qualitatively worse.

F.3 Over-adjust based on Last Turn of Conversation

Because the summary task requires the assistant to attribute its summary back to documents through citation, the task offers a unique opportunity to analyze what turns of information LLMs pay attention to as the multi-turn conversation progresses. As a reminder, the summary task involves a user introducing new documents at each turn. The focus of our analysis is therefore to understand whether document introduction order (across turns) affects the likelihood of the LLM citing a document.

In Figure 10, we plot the the results of our analysis. Each row corresponds to the analysis of summaries generated at a given turn in the sharded simulation. At turn 1 (top row), 96% of the cited documents were introduced in the first turn. The missing 4% correspond to hallucinated citation to documents that were not introduced, and explains why none of the rows' distribution sum to 100%. At turn two (second row from the top), summaries include citation in roughly equal proportion for turn-1 and turn-2 documents (i.e., 48% and 49%).

We interpret this to mean that in 2-turn conversations, LLMs pay roughly equal attention to documents in either turn. Analysis of summaries generated in turns 3-8 of sharded simulations reveal an imbalance in the documents the LLM cites to. In eighth-turn summaries, 20% of citations are to documents introduced in turn 8, compared to 8% from turn 2 and 3 (150% difference). At a high-level, as the conversation progresses, LLMs are most likely to cite either documents in the first or last turns, and less likely to cite documents introduced in intermediary (middle) turns. This finding mirrors findings of a *loss-in-the-middle* phenomena of LLMs paying more attention to documents at the start or end of their provided context, at the cost of middle-context content [29, 50, 40]. In short, we observe that the lost-in-the-middle phenomena occurs not only in single-turn long-context settings, but also in multi-turn conversation. We name this phenomenon *loss-in-middle-turns*.

We note that the analysis presented in Figure 10 averages numbers across the 15 LLMs included in our main experiment. Even though we observe some loss-in-middle-turns in all models, the magnitude of the effect varies across models, typically with more performant models having a more muted effect, showing they have better capabilities of handling attribution across multiple turns of context. We do not include model-specific analyses in this work and leave it for future work.

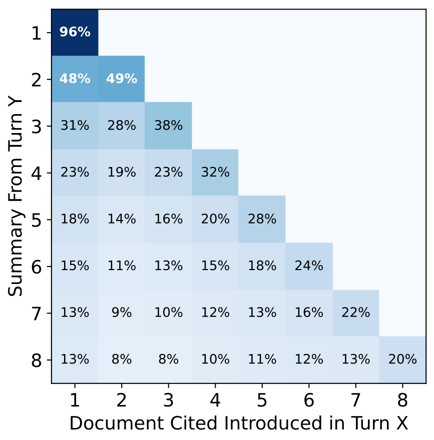

Figure 10: Analysis of citation patterns in summaries generated by LLMs with the SHARDED simulation. At each turn, the LLM generates an updated summary (y-axis), which includes citations from the documents that have been revealed up to this turn. Percentages in a row do not add up to 100% due to citation hallucinations that occur for some models.

F.4 Overly-verbose Assistant Responses

Task	Relative Assistant Verbosity				
	0-20%	20-40%	40-60%	60-80%	80-100%
Assistants responses are ...	shortest	short	median	long	longest
Code	55.3	52.3	48.9	46.9	42.5
Math	62.9	64.0	62.1	60.9	56.1
Database	43.8	40.0	37.3	34.3	31.3
Actions	41.5	49.6	54.2	53.6	50.8
Data-to-Text	25.0	24.3	24.0	23.1	21.8
Summary	15.4	14.7	13.5	12.0	10.3
Average	40.7	40.8	40.1	38.6	35.6

Table 7: Averaged performance ($\overline{P}$) of LLMs on the six experimental tasks, arranged based on model relative verbosity (length of response). Performance degrades when models generate longer responses on five of the six tasks.

When simulating multiple conversations based on a common instruction, we observe variation in responses, particularly in the length of the response generated by the LLM. To understand how verbosity (length of a response) affects model performance, we perform a verbosity analysis.

One difficulty with assessing verbosity is that different tasks and instructions might require different levels of verbosity. For example, generating a Python function likely requires a longer than generating an SQL query. In order to regularize for task-specific variations, we assign a *verbosity tag* calculated for each (LLM, instruction) tuple. For each simulated sharded conversation involving an LLM on an instruction, we calculate the average length of the per-turn response (number of total characters in assistant responses divided by number of turns). We then bin conversations into quintiles according to this metric. More specifically, since we simulated $N = 10$ conversations for each (model, instruction) pair, we assign 2 simulations per quintile, which we name: shortest, short, median, long, and longest. We then calculate averaged performance ($\overline{P}$) on the six experimental tasks, arranged based on this verbosity tag. Results are summarized in Table 7.

On five of the six tasks, performance is 10-50% higher in simulated conversations with shortest response length, compared to conversations with longest response length. As assistant responses get longer (left to right in the Table), performances gradually drop. The Actions task is the only task where such an effect is not observed, and where shortest response length from the assistant is detrimental.

Predominantly however, models achieve higher performance when they generate shorter responses. We hypothesize that deterioration due to over-verbosity is due to longer responses typically containing more assumptions or hypotheses from the assistant, which can lead to confusion in following turns. On the other hand, short turns tend to be focused (e.g, a single clarification question), and keep the conversation on track.

Deterioration due to over-verbosity is note-worthy, as besides deteriorating underlying model performance, longer responses also take longer for users to read, which is undesirable. The finding therefore indicates that longer LLM responses are bad both for models and end-users.