Integrating LLM-based Explanations into Open-Ended Graph Practice Exercises for Increased Learning and Engagement

Ali Mahmoud Shokry[1][a], Anan Schütt[2][b] and Elisabeth André[2][c]
[1]*German University of Cairo, Egypt*
[2]*University of Augsburg, Germany*

Keywords: Open-Ended Learning Environment, Large Language Model, Maximum Independent Set.

Abstract: Open-ended tasks are effective for learners to explore different actions and practice their skills. However, learners might not be able to use them effectively without proper guidance throughout the learning episode. In this paper, we explore using an LLM to generate explanations for the solution of the maximum independent set, a graph problem that learners attempt to solve. We combine LLM with a solver program to make the explanation accurate, while keeping the natural language usage and setting an educational tone of writing. We evaluate the method in a user study, which shows improved learning and engagement.

1 INTRODUCTION

Open-ended learning environments allow learners to explore different paths to solve a problem (Hannafin, 1995). Although there are many ways a student can interact with the problems, different works have found that how a student explores these different paths influences their learning outcomes (Fratamico et al., 2017; Nasir et al., 2021; Schütt et al., 2024). Generally, students who don't actively engage with the learning task doesn't learn effectively. Their disengagement might be caused by a lack of motivation, or possibly even simply because they lack the guidance to get started on the task. Likewise, research has established that feedback that details the problem-solving process helps guide the learner to a better course of action and to improve the learning process (Merrill et al., 1995; van der Kleij et al., 2015; Ajjawi et al., 2022).

Several previous works have applied Large Language Models (LLM) as tutoring agents to give feedback to learners (Munshi et al., 2023; Koutcheme et al., 2025). However, in domains where the task requires more complex reasoning, and especially in domains that are not part of the LLM's training, hallucinations especially become an issue that could diminish the tutoring potential (Li et al., 2024), causing inaccurate explanations and feedback, as observed in (Steiss et al., 2024; Smith et al., 2025).

To address the accuracy issue, we propose using an external tool to provide a rigidly formulated solution path as the first step to the explanation, then use the LLM to deliver the solution in natural language. An example domain where such an external tool exists is the maximum independent set problem, which is a classic and well-studied problem in computer science. Following this, we explore using these solvers to provide information for the LLM to generate factually correct, natural language feedback. Conceptually, we focus on studying the LLM's ability to formulate pedagogically effective explanations rather than its reasoning ability.

Aside from accuracy, using LLMs as part of the learning process potentially has detrimental effects on the learners, as having an LLM available could lead them to offload the cognitive work to the LLM and be unengaged in learning (Poitras et al., 2024; Kosmyna et al., 2025). The effects on the learner, especially in the context of a problem-solving task, are not yet well understood (Kumar et al., 2025). Thus, our work also studies the effects of applying LLM as the explanation provider in the context of an open-ended practice task.

In this paper, we present a method to combine an external solver program with an LLM to generate pedagogically helpful explanations for the Maximum Independent Set (MIS) problem, a novel approach in the tutoring context. We evaluate the effects of the generated feedback in a user study, where participants

[a] https://orcid.org/0009-0002-2610-5284
[b] https://orcid.org/0009-0006-2459-719X
[c] https://orcid.org/0000-0002-2367-162X

Shokry, A. M., Schütt, A. and André, E.
Integrating LLM-based Explanations into Open-Ended Graph Practice Exercises for Increased Learning and Engagement.
DOI: 10.5220/0014814700004021
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 18th International Conference on Computer Supported Education (CSEDU 2026) - Volume 1*, pages 84-93
ISBN: 978-989-758-833-4; ISSN: 2184-5026
Proceedings Copyright © 2026 by SCITEPRESS – Science and Technology Publications, Lda.

practice solving the MIS tasks, measuring the affective, behavioural, and learning aspects. We address the following questions:

- **RQ1:** How does the LLM-generated feedback affect the learning gain and the practice behavior of the participants?
- **RQ2:** How are the LLM-generated feedback messages perceived by the participants?

2 RELATED WORKS

In open-ended learning tasks, learners are open to try out different actions in the environment and see the effects for themselves (Hannafin, 1995). Because of the openness, Alfieri et al. (2011) identified that such a learning environment needs to be accompanied by a feedback mechanism or explanation to guide the learner for the learning to be effective. Different works have studied the effects of different types of feedback, and generally found that both cognitive feedback (e.g., correcting a misconception, suggesting actions) and affective feedback (e.g., telling the student to keep on trying) contribute to effective learning (Anders Ericsson, 2008; D'mello and Graesser, 2013; Munshi et al., 2023). We follow these findings, adding explanations to our open-ended learning task, and implementing both the cognitive and affective guidelines.

The active research into LLMs has led to many applications in tutoring, one main field of which includes giving hints and feedback to the learner. LLMs have been shown to be capable of giving feedback to the learner, as previously explored in domains like programming (Phung et al., 2023) and algebra (McNichols et al., 2023). Effectively, these works rely on the LLM both for the cognitive task of generating a solution to the exercise that the learner attempts, and for the pedagogical task of delivering the feedback and explanation to the learner. For this reason, niche domains, where LLMs were not extensively trained on, might cause them to give inaccurate feedback (Estévez-Ayres et al., 2025). Apart from that, LLMs can give erroneous explanations of getting to the solution, even if it can answer the question correctly (Gupta et al., 2025), which could pose a problem if LLM's training data only evaluates the answer correctness.

The domain of interest in this paper is the graph theory problem named Maximum Independent Set (MIS), which is potentially a domain that LLMs cannot solve reliably, as was also the case in other similar rule-based puzzles (Giadikiaroglou et al., 2024).

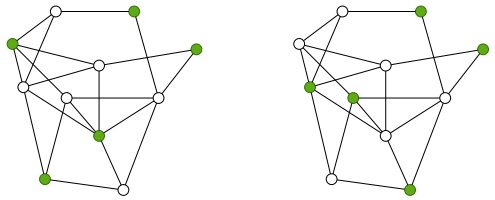

Figure 1: A graph with the maximum independent set drawn in green. The two different selections each form a valid solution on the same graph.

Li et al. (2024) identified that LLMs struggle particularly with puzzles that need longer multi-step reasoning, and points out hallucination and the fact that LLMs seem to copy examples rather than deduce the underlying rules of the puzzle as the root of the issue. To address the LLM's low reliability on rule-based puzzles, Ding et al. (2023) proposed the Everything-of-Thought framework, which combines the LLM with an AlphaGo Zero (Silver et al., 2017) style of policy/value network and Monte-Carlo tree search. While this approach improves LLM's performance in solving rule-based puzzles and would likely also be applicable in MIS, hallucination is still an unresolved issue, and thus, the correctness is not guaranteed like a solver program specifically written for the particular problem domain. Since our goal lies in generating an explanation in natural language, it is more direct and reliable to use the LLM's lingual abilities to translate an externally-generated, rigidly-formatted solution into a natural explanation text, which can also be seen as an instance of tool usage (Qin et al., 2024).

3 MAXIMUM INDEPENDENT SET

We take the Maximum Independent Set problem as the focus point of our work. The goal of the task is to select the set of vertices in a graph, such that no pair of the chosen vertices is connected by an edge, and the set contains as many vertices as possible. Figure 1 shows an example of the task, where the green vertices form a maximum independent set in this graph. The rules of the task are very simple to explain, while solving the task itself can be difficult, and has no fixed, efficient procedure that always leads to the correct solution, making it an interesting problem-solving task to experiment with practice methods.

To give more context to the task, we present some terms and some basic intuition for solving. In a graph G with some selection S, we call the vertices that are not adjacent to any vertex in S the free vertices. Correspondingly, we call the number of free vertices adjacent to a vertex v the free degree of v. Roughly speak-

ing, the free degree of a vertex v tells how many vertices become "unusable", if v were added to the selection. This leads to a guideline, that selecting a vertex with low free degree is often a good action. Although this guideline works in general, simply repeatedly doing this does not always lead to a correct solution. In many cases, some trial-and-error is involved, for example, by forcing a selection other than the one with the lowest free degree.

4 METHODS

Our proposed framework for generating adaptive explanations is divided into two parts. First, the graph exercise is given to the solver, which finds a maximum independent set and outputs a solver trace, describing how to reach the solution. The input to the solver differs, depending on the learner's attempt at the problem. Second, the solver trace is inserted in a prompt for the LLM, which outputs an explanation to give to the learner. The text in this section explains the details of the two parts, and describes the user study to evaluate the generated explanations.

4.1 Solver

The solver algorithm was built upon the Iterated Local Search (ILS) algorithm as proposed by Andrade et al. (2012). The ILS works by combining a deterministic local search with stochastic solution perturbation, going through the following sub-procedures iteratively to refine the candidate solution:

1. Conflict Resolution (Pre-computation): The algorithm checks if the current candidate solution includes conflicts, i.e., adjacent nodes in the selection. If conflicts exist, then the vertex with the highest conflict count is iteratively removed until there is no conflict left.
2. Local Search (2-Improvement): The algorithm greedily inserts free vertices until the candidate solution is maximal. To further optimize, it utilizes a "2-improvement" strategy, which looks for selected vertices that can be removed to allow two new vertices to be selected.
3. Perturbation: To escape local optima, the algorithm applies a "force routine", inserting k randomly chosen non-free vertices and removing their neighbors from the candidate solution. The number of forced insertions k is stochastic, and typically is $k = 1$.
4. Acceptance: The new candidate solution is accepted directly if it is larger than the current candidate. However, to encourage exploration, the algorithm still accepts the new candidate solution at a probability, even if it isn't larger than the current candidate.

The steps of the algorithm, particularly the 2-improvement and perturbation, build up a strategy that human solvers can also intuitively use when solving by hand; one tries to make small local changes to get a better solution, and if that doesn't work, then one can switch to force some changes to the current solution to try other configurations that might reach a better solution. ILS also takes a candidate solution as a starting point, so it can take the learner's attempt and show how to improve upon it to get a valid solution, generating an explanation that extends from the learners current process, following the suggestion from the principles of deliberate practice (Anders Ericsson, 2008).

The solver generates a solution trace to serve as the basis for the LLM's natural language explanation. If the learner's attempt was incorrect, we pass the learner's attempt as the starting point for generating the solution trace. Otherwise, we generate a solution trace with the empty set as the starting point. We run the ILS solver with at most 10 iterations to generate a trace to limit the length of the trace, and consequently, the explanation. We restart the solver until a valid solution is generated.

The solution trace includes:

- the response whether the learner's initial solution is independent and maximum
- the list of actions that the solver takes to come to a valid solution, along with side information, such as the sub-procedure that made that action, or the vertices that are free neighbours of the selected vertex.
- the list of free vertices after each action

An example of the solver trace, illustrating the specific logging of the "2-improvement" sub-procedure passed to the LLM, is provided in Appendix A.

4.2 Large Language Model (LLM)

We use the LLM Deepseek-R1 with 8B parameters to generate natural-language explanations. We chose this model because of its open availability, its accessibility through lower hardware requirements, and its performance in comparison to similar models in our pilot test. We use a temperature setting of $t = 0.6$ with the nucleus sampling probability $top_p = 0.95$ to allow for low variability, while showing enough fluency in the language use.

```
Your task is to give a natural language explanation of the
    provided step-by-step trace for solving the maximum
    independent set problem on a specific graph.

Tell the student that the solution provided was incorrect
    and then the following explanation will build from
    the student's solution.

The original graph is (represented as an adjacency list)
<Adjacency list>

Trace of the solver's steps where the initial set is the
    student's solution:
<Solver trace log>

When explaining the solution trace:
1. For each step, explain the reasoning behind each vertex
    selection or rejection, and describe how the graph
    and the independent set change.
2. Highlight the strategic thinking and connect decisions
    to broader principles of graph theory.
3. Maintain a clear, positive, and motivating tone, and
    write as if you are the solver explaining to the
    student.
4. Format the response as a markdown file (no boxed items)
5. Mention the importance of the two-improvement strategy
    when it is used.
6. Do not include the adjacency list in the solution.
7. As you explain, consider: how vertex degrees influence
    decisions, the impact on neighbors, relationships
    between vertices, and the two-improvement strategy.
     - If a high-degree node is chosen, explain it may
    be a trial and could be removed later.
     - Typical reasons for node choice: low-degree
    (eliminates few), neighbors a high-degree node, or is
    the only free vertex.
8. Limit to a maximum of 2 consecutive newlines in the
    response.
After the trace, provide a concise summary of key
    strategic insights and the reasoning behind each
    vertex in the final solution, helping students
    understand the implicit reasoning pattern.


Below is an example response given another trace. This is
    only a guide for you to understand how the response
    should look.
Example:
<Example explanation>
```

Figure 2: The prompt template used to generate explanations when the learner submits an incorrect solution.

The goal of the explanation is to show the learner actions that lead to a valid solution, give reasons for the suggested actions, and avoid excessive information, such as outputting the entire adjacency list to the student, following the guidelines from Hattie and Timperley (2007). The explanation should be formulated in a friendly and positive manner to keep an atmosphere conducive to learning (McLaren et al., 2011). To align the model's output with these pedagogical goals, the LLM's prompt includes some instructions and an example of the explanation in the desired form, following the one-shot learning method (Brown et al., 2020). See Figure 2 for the prompt in case the learner's submitted solution is incorrect. The prompt for a learner's correct solution is similar, but the explanation starts from an empty selection instead of the learner's submission, showing a potentially different path to the solution from the learner's.

4.3 User Study

To evaluate the pedagogical effectiveness and the learner's perception of the LLM-generated explanations, we conducted a between-subject user study.

4.3.1 Procedure

The experimental session was conducted using a custom-built web interface, shown in Figure 3. The participant works on the task by clicking on vertices to select/deselect them. The participant has a time limit of 90 seconds for each graph, which is shown in a timer. The participant can submit their selection before the time runs out, or their final state will be taken as their selection once the time runs out.

Figure 4 shows the phases of the user study. The first phase informs the participant about the study and collects their consent. Then we collect the demographic information, which includes the age, gender, level of education, and their self-reported knowledge of graph theory. After that, the application teaches the participant about the MIS problem, giving the definition of the problem and the requirements of the solution. To pass this phase, the participant must give the correct MIS on a small sample graph, making sure that they understand the task.

The participant is then led to the pre-test, where the participant attempts to find the MIS on three graphs, one after the other. These three graphs were chosen to contain one easy, one medium, and one difficult graph, and are the same for every participant. This phase collects the participant's ability before practice, and therefore, there is neither feedback nor explanation in this phase.

Next is the practice phase, where the participant gets eight graphs to practice on. These eight graphs are set to be in an increasing order of estimated difficulty, and are the same sequence of graphs for every participant. In this phase, the web application gives feedback on the submission, i.e., whether it is maximum and independent, and also gives an explanation of the correct solution.

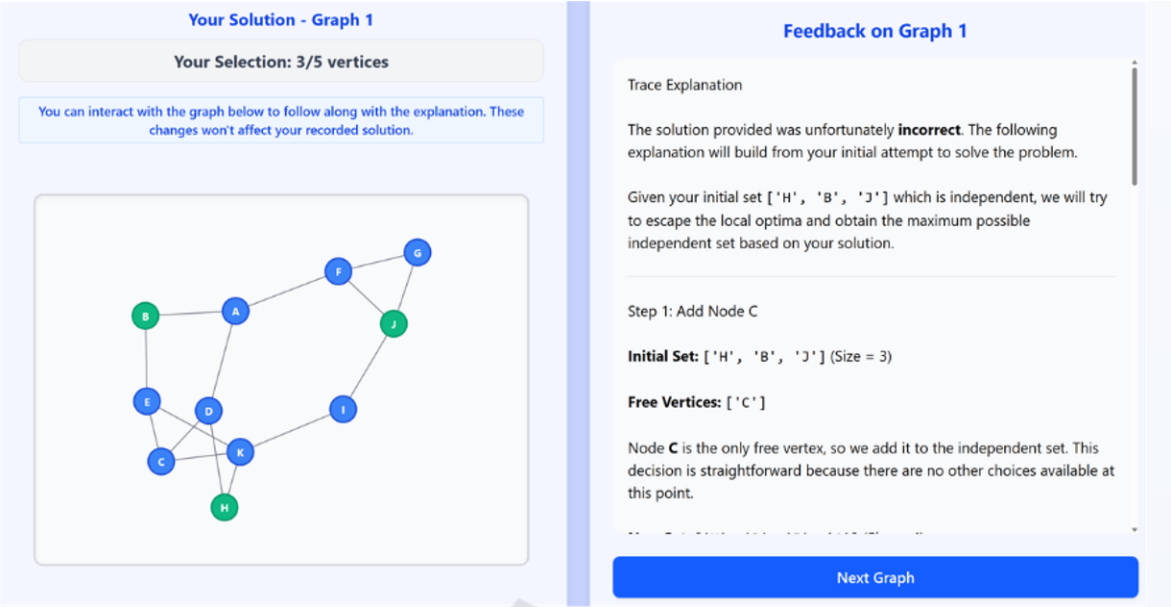

Figure 3: Screenshot of a page of the user study web application. This page appears after the learner has submitted his solution, and is getting an explanation. The left panel of the screen shows the learner's submission, which can still be interacted with. The right panel shows the explanation text.

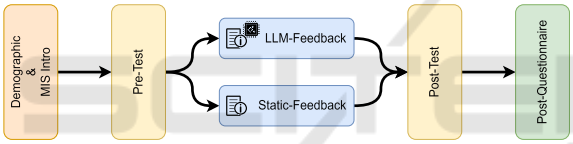

Figure 4: Diagram showing the phases of the user study.

The fourth phase is the post-test, which is set up just like the pre-test phase, but with a different set of graphs, to collect the participants' ability after practicing.

Finally, participants fill out a post-study questionnaire in the sixth phase to evaluate their subjective perception of explanations.

4.3.2 Conditions

Participants were randomly assigned to one of two feedback conditions during the intervention phase: the LLM-feedback or static-feedback condition. The LLM-feedback condition receives the LLM-generated explanation during the practice phase, which lists the steps to solving the problem, adapting the steps to the student's submission. The static-feedback condition only receives the correctness feedback on their submission, along with the fixed, expert-written guidelines for finding the MIS in general.

4.3.3 Measures

We measure the pedagogical effectiveness by the pre- and post-test scores. For each test problem, a participant can earn a maximum of two points, one for submitting an independent set, and one for submitting a set that is also maximum. This system gives a maximum of six points in each test phase.

Additionally, we record the number of actions in the graphs (select, deselect, and reset) as another dependent variable. This reflects how engaged the participant is with the exercise, and is linked to improved learning in similar learning tasks.

We also use a 5-point Likert scale questionnaire with 16 items to collect the subjective perception of the feedback, following the items proposed by Chang et al. (2024), which is divided into the following categories:

- Accuracy: The factual correctness of the explanations regarding graph theory concepts and the validity of the specific solution steps described.
- Relevance: The degree to which the feedback directly aided the participant's problem-solving process and addressed their specific learning needs.
- Fluency and Transparency: The clarity, readability, and linguistic quality of the generated text, measuring how easy the explanations were to follow.
- Human Alignment: A measure of educational effectiveness, assessing how well the explanations matched participants' expectations of a supportive human tutor.

- Engagement: The extent to which the explanations motivated the student and sustained their attention throughout the task.

We also have an open-text field for the participant to freely give further feedback regarding the study.

4.3.4 Participants

We recruited 30 students at the University of Augsburg as participants in the study.

The sample was predominantly composed of undergraduate Computer Science students (83.3%), with a smaller proportion of graduate students (16.7%). The mean age of the participants was 22.2 years ($SD = 2.50$, range: 19–31), and the gender distribution was 76.7% male and 23.3% female. Recruitment was conducted on a voluntary basis through university student channels, with no monetary compensation provided. The participants are only required to be fluent in English, without prerequisites regarding graph theory.

5 RESULTS

5.1 Learning Outcomes

Table 1: Comparison of test scores, normalized learning gain, and in-practice action counts between conditions.

	Condition	
Construct	LLM	Static
Pre-test	3.50 ± 1.83	3.44 ± 1.55
Post-test	4.71 ± 1.38	3.50 ± 1.79
Actions	101.07 ± 27.71	79.00 ± 20.67

We present the test results and the number of in-practice actions in Table 1. The pre-test scores are comparable between the two groups, with no significant differences from the two-tailed t-test ($t = 0.097$, $p = 0.923$). The post-test score is significantly higher in the LLM-feedback group, compared to the static-feedback group with a medium effect size, as shown by the one-tailed t-test ($t = 2.050$, $p = 0.025$, Cohen's $d = 0.75$).

We also analyzed the actions of the participants during the practice phase, counting the number of actions (select and deselect) that they made on the graph. The mean and standard deviations of the counts are also shown in Table 1. The one-tailed t-test shows that the LLM-feedback group makes significantly more actions on the graph than the expert-feedback group, with a large effect size ($t = 2.493$, $p = 0.009$, Cohen's $d = 0.912$).

5.2 Subjective Evaluation

We report the Likert-scale measures in Table 2. We test for significant differences in two measures: engagement and fluency, in accordance with the hypotheses. The one-tailed t-tests show no significant differences in these two measures (Relevance: $t = 0.563$, $p = 0.289$, engagement: $t = -0.28$, $p = 0.609$).

Table 2: Response from the 5-point Likert scale questionnaire, aggregated in each category

	Condition	
Construct	LLM	Static
Accuracy	4.24 ± 1.15	4.08 ± 0.70
Relevance	4.00 ± 1.18	3.78 ± 0.91
Fluency	4.18 ± 0.93	4.12 ± 0.85
Human Align.	4.11 ± 0.98	3.81 ± 0.83
Engagement	3.69 ± 1.09	3.79 ± 0.86

5.3 Analysis of Open-Ended Feedback

To contextualize the quantitative results, we read through the open-text comments to find the main opinions on the explanation. First, many participants found the latency of the LLM's explanation to be an issue "waiting time for getting the explanation is borderline too long", which further led them to skip some of the explanations. Some participants also remarked that the explanations are too long to read. Furthermore, one participant remarked that the explanation was incorrect "It made me a little mad that I had to read through this hallucinated generated "explanation".". The log data from this interaction shows that the LLM largely copied the example included in the prompt and presented that as the explanation, instead of taking information from the solver trace. Following this, we examined 50 additional log instances and found no other instances where the LLM copies from the example explanation.

In the static-feedback condition, participants also pointed out that the guidelines remained fixed "solution advice stayed the same... regardless of my solution", which again discouraged them from reading over and over again.

6 DISCUSSION

We start with the first research question, which addresses the learning gain and practice behavior. Results from Table 1 show a trend that the LLM-feedback group has higher learning gains than the static-feedback group. This hints that the proposed

framework of combining the LLM with the solver as an input source can be an effective method to give feedback. Furthermore, since the solver already handles the graph problem's solution, a small LLM like DeepseekR1 with 8B parameters is sufficient to generate feedback, aligning with the research interest in using smaller LLMs for tutoring (Wei et al., 2025). As for practice behavior, Table 1 shows that learners in the LLM-feedback group perform more actions on the graph problems during the practice phase, revealing the higher engagement. This engagement was also found to be an effective means for learning in MIS (Schütt et al., 2024), showing promise for embedding LLMs in a similar application for practicing skills.

The second research question addresses the subjective perception of the learners towards the feedback. The high accuracy, relevance, and human alignment in Table 2 for both conditions, as given by the participants, show that the LLM-feedbacks seem as pedagogically useful as the expert-written static feedback. Also, the high fluency shows that the explanation is easy to understand, and high engagement shows that students don't find the feedback boring or disturbing. The open-text comments suggest that the main value that the LLM feedback provided is that they adapt to the graph being worked on, pointing out key steps in the exercise, while the static feedback only gives general guidelines that are the same for every graph, fitting with Munshi et al. (2023). Still, the LLM sometimes gives overly lengthy explanations, which should be summarized so that the reason behind each choice is recognizable, rather than sequentially listing the actions done on the graph. Despite our attempt to make the explanations factually correct by providing a solver trace as the ground truth, the LLM's explanation could still sometimes be unreliable, as shown by the case where the LLM took the example explanation as the explanation to give to the student. One of the participants also reports hallucinated explanations, despite the framework that should guarantee the correctness by following an algorithm's output, which shows the possibility for the LLM's failure even in an almost purely linguistic use case. Finally, our setup required a few seconds to generate the explanation, which was enough to be viewed negatively by the participants, and might even counteract the benefits of having adaptive feedback.

Our user study was mainly limited by the number of participants, with a sample size too small to deliver statistically significant results. The participant pool is also homogeneous, consisting mainly of undergraduate students from the same country, which might particularly affect the subjective perception of the feedback.

7 CONCLUSION

In this paper, we studied using LLMs in combination with a solver program to give learners an explanation on the maximum independent set problem, adapting the explanation to the learner's current attempt. Our results from the user study suggest that using LLMs to give a more detailed and adaptive explanation not only improves the learning outcome, but also leads the learners to interact with the task more, which is also helpful for learning. Despite being generally well-received, the quality of the explanation, specifically the verbosity and, surprisingly, the accuracy, is still a point of potential improvement.

Future work could explore similar tasks that can benefit from our proposed framework, with the LLM working in tandem with an external solver in a more flexible manner, extending from our fixed and rigid domain of MIS, which makes integrating an external solver simple. These tasks should be problem-solving tasks with multi-step solutions, where there is a reasoning behind each step and a solving algorithm exists. Similar graph theory tasks or logical tasks, such as logic grid puzzles or chess puzzles, and even K-12 level mathematics questions, could be interesting and feasible domains to study.

REFERENCES

Ajjawi, R., Kent, F., Broadbent, J., Tai, J. H.-M., Bearman, M., and Boud, D. (2022). Feedback that works: a realist review of feedback interventions for written tasks. *Studies in Higher Education*, 47(7):1343–1356.

Alfieri, L., Brooks, P. J., Aldrich, N. J., and Tenenbaum, H. R. (2011). Does discovery-based instruction enhance learning? *Journal of educational psychology*, 103(1):1.

Anders Ericsson, K. (2008). Deliberate practice and acquisition of expert performance: a general overview. *Academic emergency medicine*, 15(11):988–994.

Andrade, D. V., Resende, M. G., and Werneck, R. F. (2012). Fast local search for the maximum independent set problem. *Journal of Heuristics*, 18(4):525–547.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, Red Hook, NY, USA. Curran Associates, Inc.

Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y., Ye, W.,

Zhang, Y., Chang, Y., Yu, P. S., Yang, Q., and Xie, X. (2024). A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Ding, R., Zhang, C., Wang, L., Xu, Y., Ma, M.-J., Zhang, W., Qin, S., Rajmohan, S., Lin, Q., and Zhang, D. (2023). Everything of thoughts: Defying the law of penrose triangle for thought generation. *ArXiv*, abs/2311.04254.

D'mello, S. and Graesser, A. (2013). Autotutor and affective autotutor: Learning by talking with cognitively and emotionally intelligent computers that talk back. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 2(4):1–39.

Estévez-Ayres, I., Callejo, P., Hombrados-Herrera, M. Á., Alario-Hoyos, C., and Kloos, C. D. (2025). Evaluation of LLM tools for feedback generation in a course on concurrent programming. *Int. J. Artif. Intell. Educ.*, 35(2):774–790.

Fratamico, L., Conati, C., Kardan, S., and Roll, I. (2017). Applying a framework for student modeling in exploratory learning environments: Comparing data representation granularity to handle environment complexity. *International Journal of Artificial Intelligence in Education*, 27:320–352.

Giadikiaroglou, P., Lymperaiou, M., Filandrianos, G., and Stamou, G. (2024). Puzzle solving using reasoning of large language models: A survey. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N., editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11574–11591, Miami, Florida, USA. Association for Computational Linguistics.

Gupta, A., Reddig, J. M., Calò, T., III, D. W., and MacLellan, C. J. (2025). Beyond final answers: Evaluating large language models for math tutoring. In Cristea, A. I., Walker, E., Lu, Y., Santos, O. C., and Isotani, S., editors, *Artificial Intelligence in Education - 26th International Conference, AIED 2025, Palermo, Italy, July 22-26, 2025, Proceedings, Part I*, volume 15877 of *Lecture Notes in Computer Science*, pages 323–337. Springer.

Hannafin, M. J. (1995). Open-ended learning environments: Foundations, assumptions, and implications for automated design. In *Automating instructional design: Computer-based development and delivery tools*, pages 101–129. Springer.

Hattie, J. and Timperley, H. (2007). The power of feedback. *Review of educational research*, 77(1):81–112.

Kosmyna, N., Hauptmann, E., Yuan, Y. T., Situ, J., Liao, X., Beresnitzky, A. V., Braunstein, I., and Maes, P. (2025). Your brain on chatgpt: Accumulation of cognitive debt when using an AI assistant for essay writing task. *CoRR*, abs/2506.08872.

Koutcheme, C., Dainese, N., Sarsa, S., Hellas, A., Leinonen, J., Ashraf, S., and Denny, P. (2025). Evaluating language models for generating and judging programming feedback. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*, SIGCSETS 2025, page 624–630, New York, NY, USA. Association for Computing Machinery.

Kumar, H., Rothschild, D. M., Goldstein, D. G., and Hofman, J. M. (2025). Math education with large language models: Peril or promise? In Cristea, A. I., Walker, E., Lu, Y., Santos, O. C., and Isotani, S., editors, *Artificial Intelligence in Education - 26th International Conference, AIED 2025, Palermo, Italy, July 22-26, 2025, Proceedings, Part IV*, volume 15880 of *Lecture Notes in Computer Science*, pages 60–75. Springer.

Li, Y., Wang, H., and Zhang, C. (2024). Assessing logical puzzle solving in large language models: Insights from a minesweeper case study. In Duh, K., Gómez-Adorno, H., and Bethard, S., editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 59–81. Association for Computational Linguistics.

McLaren, B. M., DeLeeuw, K. E., and Mayer, R. E. (2011). A politeness effect in learning with web-based intelligent tutors. *Int. J. Hum. Comput. Stud.*, 69(1-2):70–79.

McNichols, H., Zhang, M., and Lan, A. S. (2023). Algebra error classification with large language models. In Wang, N., Rebolledo-Mendez, G., Matsuda, N., Santos, O. C., and Dimitrova, V., editors, *Artificial Intelligence in Education - 24th International Conference, AIED 2023, Tokyo, Japan, July 3-7, 2023, Proceedings*, volume 13916 of *Lecture Notes in Computer Science*, pages 365–376. Springer.

Merrill, D. C., Reiser, B. J., Merrill, S. K., and Landes, S. (1995). Tutoring: Guided learning by doing. *Cognition and Instruction*, 13(3):315–372.

Munshi, A., Biswas, G., Baker, R., Ocumpaugh, J., Hutt, S., and Paquette, L. (2023). Analysing adaptive scaffolds that help students develop self-regulated learning behaviours. *Journal of Computer Assisted Learning*, 39(2):351–368.

Nasir, J., Kothiyal, A., Bruno, B., and Dillenbourg, P. (2021). Many are the ways to learn identifying multi-modal behavioral profiles of collaborative learning in constructivist activities. *International Journal of Computer-Supported Collaborative Learning*, 16(4):485–523.

Phung, T., Cambronero, J., Gulwani, S., Kohn, T., Majumdar, R., Singla, A., and Soares, G. (2023). Generating high-precision feedback for programming syntax errors using large language models. In Feng, M., Käser, T., Talukdar, P. P., Agrawal, R., Narahari, Y., and Pechenizkiy, M., editors, *Proceedings of the 16th International Conference on Educational Data Mining, EDM 2023, Bengaluru, India, July 11-14, 2023*. International Educational Data Mining Society.

Poitras, E., Crane, B., and Siegel, A. (2024). Generative ai in introductory programming instruction: Examining the assistance dilemma with llm-based code generators. In *Proceedings of the 2024 on ACM Virtual*

Global Computing Education Conference V. 1, pages 186–192.

Qin, Y., Liang, S., Ye, Y., Zhu, K., Yan, L., Lu, Y., Lin, Y., Cong, X., Tang, X., Qian, B., Zhao, S., Hong, L., Tian, R., Xie, R., Zhou, J., Gerstein, M., Li, D., Liu, Z., and Sun, M. (2024). Toolllm: Facilitating large language models to master 16000+ real-world apis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Schütt, A., Huber, T., Nasir, J., Conati, C., and André, E. (2024). Does difficulty even matter? investigating difficulty adjustment and practice behavior in an open-ended learning task. In *Proceedings of the 14th Learning Analytics and Knowledge Conference*, LAK '24, page 253–262, New York, NY, USA. Association for Computing Machinery.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T. P., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., and Hassabis, D. (2017). Mastering the game of go without human knowledge. *Nat.*, 550(7676):354–359.

Smith, S. B., Wei, H., O'Neill, A., Durai, A., DeNero, J., Zamfirescu-Pereira, J. D., and Norouzi, N. (2025). Spotting AI missteps: Students take on LLM errors in CS1. In Stone, J. A., Yuen, T. T., Shoop, L., Rebelsky, S. A., and Prather, J., editors, *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 2, SIGCSE TS 2025, Pittsburgh, PA, USA, 26 February 2025 - 1 March 2025*, pages 1627–1628. ACM.

Steiss, J., Tate, T., Graham, S., Cruz, J., Hebert, M., Wang, J., Moon, Y., Tseng, W., Warschauer, M., and Olson, C. B. (2024). Comparing the quality of human and chatgpt feedback of students' writing. *Learning and Instruction*, 91:101894.

van der Kleij, F. M., Feskens, R. C. W., and Eggen, T. J. H. M. (2015). Effects of feedback in a computer-based learning environment on students' learning outcomes: A meta-analysis. *Review of Educational Research*, 85(4):475–511.

Wei, Y., Carvalho, P., and Stamper, J. C. (2025). Small but significant: On the promise of small language models for accessible AIED. *CoRR*, abs/2505.08588.

A Solver Trace

The following demonstrates the raw log output generated by the ILS solver. This structured text serves as the input context for the Large Language Model.

Initial set: `[]`
Initial free vertices: `['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P']`

add node F to the independent set with neighbors excluded: `['B', 'E', 'J', 'L', 'O']`
current set: `['F']` length = 1
free vertices: `['A', 'C', 'D', 'G', 'H', 'I', 'K', 'M', 'N', 'P']`

add node A to the independent set with neighbors excluded: `['H', 'P']`
current set: `['F', 'A']` length = 2
free vertices: `['C', 'D', 'G', 'I', 'K', 'M', 'N']`

add node N to the independent set with neighbors excluded: `['D']`
current set: `['F', 'A', 'N']` length = 3
free vertices: `['C', 'G', 'I', 'K', 'M']`

add node K to the independent set with neighbors excluded: `['C', 'M']`
current set: `['F', 'A', 'N', 'K']` length = 4
free vertices: `['G', 'I']`

add node G to the independent set
current set: `['F', 'A', 'N', 'K', 'G']` length = 5
free vertices: `['I']`

add node I to the independent set
current set: `['F', 'A', 'N', 'K', 'G', 'I']` length = 6
free vertices: `[]`

remove node K from the independent set reason: two improvements
by removing this node, the two vertices `['C', 'K', 'M']` would be free and can be added to the independent set without any conflicts between them
current set: `['F', 'A', 'N', 'G', 'I']` length = 5
free vertices: `['C', 'K', 'M']`

add node C to the independent set reason: two improvements with neighbors excluded: `['K']`
current set: `['F', 'A', 'N', 'G', 'I', 'C']` length = 6
free vertices: `['M']`

add node M to the independent set reason: two improvements
current set: `['F', 'A', 'N', 'G', 'I', 'C', 'M']` length = 7
free vertices: `[]`

Max Independent set: `['F', 'A', 'N', 'G', 'I', 'C', 'M']`
Size = 7

B Example of LLM-Generated Explanation

Trace Explanation

Given the original graph and your initial solution, let's break down how we can improve it to find the maximum independent set.

Initial Set: ['H', 'O', 'F', 'P', 'M', 'D', 'L']

The initial free vertices are empty, meaning all selected nodes must form an independent set with no conflicts. However, upon closer inspection, there are conflicts within this set (specifically involving L and D), so we need to resolve these to maintain independence.

—

Step 1: Conflict Resolution

We remove nodes **L** and **D** from the independent set because they conflict with other selected nodes. This step ensures that all remaining nodes in the set are non-adjacent.

- Current Set: ['F', 'H', 'M', 'O', 'P'] (Size = 5)
- Free Vertices: []

—

Step 2: Two-Improvement Strategy

Now, we look for opportunities to improve the independent set by removing one node and adding two non-conflicting nodes. Removing **O** allows us to add **I** and **Q**, which are not adjacent to each other or any nodes in the current set.

- Current Set: ['H', 'F', 'P', 'M', 'I', 'Q'] (Size = 6)
- Free Vertices: []

—

Step 3: Two-Improvement Strategy Again

Next, we remove **F** from the set to add two new nodes, **B** and **E**, which are non-conflicting. This step increases the size of the independent set by one node (from 6 to 7).

- Current Set: ['H', 'P', 'M', 'I', 'Q', 'B', 'E'] (Size = 7)
- Free Vertices: []

—

Final Independent Set

The final maximum independent set is **['H', 'P', 'M', 'I', 'Q', 'B', 'E']**, with a size of 7. This is the largest possible independent set for the given graph.

—

Summary of Strategic Insights

1. **Conflict Resolution**: Removing conflicting nodes (like L and D) ensures that the remaining nodes form a valid independent set.
2. **Two-Improvement Strategy**: Replacing one node with two non-conflicting nodes significantly improves the size of the independent set. This strategy is key to escaping local optima and finding larger solutions.
3. **Node Selection**: Prioritizing nodes like B, E, and F (which have fewer neighbors or allow adding multiple new nodes) helps maximize the independent set's size.

This approach demonstrates how combining conflict resolution with strategic replacements can lead to an optimal solution in graph theory problems like the maximum independent set.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS