Online-Journals.org

iJET International Journal of Emerging Technologies in Learning

iJET | eISSN: 1863-0383 | Vol. 19 No. 6 (2024) | OPEN ACCESS

https://doi.org/10.3991/ijet.v19i06.49829

PAPER

Integration of AI and Metaheuristics in Educational Software: A Hybrid Approach to Exercise Generation

Blanka Láng(✉), Balázs Dömsödi

Corvinus University of Budapest, Budapest, Hungary

blanka.lang@uni-corvinus.hu

ABSTRACT

This study explores the integration of generative artificial intelligence (AI) with Exercise Generation Algorithm+ (EGAL+), a multi-objective harmony search (HS) metaheuristic-based algorithm capable of composing high-quality exercises. These exercises are characterized by their diversity, consistent difficulty, and comprehensive coverage of the source material, tailored to user preferences. One of the main challenges of using metaheuristics to compile exercises efficiently is the initial creation of a large question bank, which often demands significant time and effort from instructors. To overcome this challenge, the integration of a readily available existing generative AI module is proposed. This module is accessed through its application programming interface, autonomously populating the question bank. This sets the stage for EGAL+ to fine-tune the selection and assembly of specific exams. The resulting program enables educators to create an extensive question bank from any educational material, independent of the subject, and subsequently compose exercises with minimal effort. This approach leverages the synergistic benefits of both generative AI and metaheuristic-based optimization, offering a robust and efficient solution for exercise generation.

KEYWORDS

automatic question generation (AQG), artificial intelligence (AI), multi-objective optimization, exercise generation, metaheuristics, harmony search (HS)

1 INTRODUCTION

The manual generation of high-quality test sets, whether single or multiple-choice question (MCQ) sets, is a tedious and time-consuming task. Consequently, a significant amount of research has been dedicated to automatic question generation (AQG) to save teachers considerable time and effort. Compiling question sets is currently a routine task for many educators, who then utilize them for assessments. However, creating high-quality question sets is a laborious process involving numerous repetitive tasks. As teachers' time and energy could be better utilized in tasks that leverage their expertise to create more value, many algorithmic solutions for question

Láng, B., Dömsödi, B. (2024). Integration of AI and Metaheuristics in Educational Software: A Hybrid Approach to Exercise Generation. *International Journal of Emerging Technologies in Learning (iJET)*, 19(6), pp. 38–51. https://doi.org/10.3991/ijet.v19i06.49829

Article submitted 2024-03-24. Revision uploaded 2024-05-10. Final acceptance 2024-05-10.

set generation have been developed to streamline this process [1], [2], and [3]. The issue became more pronounced during the COVID-19 pandemic, with a surge in online exams conducted without in-person supervision [4], [5].

Some studies present general question-generating solutions that can be applied across all fields, while others design question-generating solutions tailored to specific disciplines [5], [6], and [7]. There may be significant variations in effectiveness when these solutions are applied to different fields. Some researchers concentrate solely on question generation, while others also examine the answers derived from the questions. The following recent publications review the literature and explore future research directions in the field of question generation.

A methodology is presented that combines generative software engineering principles with the process of feature-oriented product line engineering, specifically for question generation, as detailed in [4]. This generator allows for the creation of families of single-choice questions based on written templates, defined features, and parameters. According to the authors, examiners and instructors can easily utilize this generator to generate various question variants.

A model for generating programming questions that utilizes the local knowledge graph and abstract syntax tree is proposed in [7]. The generated questions were well-received, earning positive feedback from a group of experienced instructors.

In [8], a practical toolkit for question-and-answer generation (QAG) is provided. Generating questions with answers from a text can be useful in many areas, but it is not easy to achieve. This paper introduces an online service for multilingual QAG for end-users and a comprehensive Python package for fine-tuning and generating models for developers.

Several research papers have addressed the challenges and opportunities of utilizing large language models (LLMs) [2], [5], [9], [10], and [11]. In [9], it is argued that this fundamental technology is a key driver of innovation and that its application in education can bring many benefits. The research explores how these models can help create educational content and engage learners. It is pointed out that the use of LLMs in education requires participants to understand both the technology and its limitations.

In [10], the issue of MCQs generated from textbooks is explored in the context of using LLMs. The study investigates whether LLMs, due to recent advancements, can produce questions that are comparable to those generated by humans. Two LLMs are analyzed, highlighting their distinct issues: Macaw tends to duplicate answer choices, while Bing Chat often omits the correct answer from the choices provided. The findings indicate that the performance of the two LLMs does not significantly differ from human performance.

In [5], the study explores the generation of MCQs using LLMs specifically for computer science courses. It highlights that crafting high-quality MCQs is a time-consuming task and that LLMs are efficient in assisting instructors in creating questions that align with courses and learning goals. The findings indicate that GPT-4 outperformed GPT-3 in generating answers solely based on the question text.

In [12], the study explores the utilization of AQG Web services and AQG models to produce a Quran question-answer dataset. Four freely available tools—Explore Artificial Intelligence (AI), Cathoven, Questgen, and Lumos Learning—were assessed for this purpose. They were employed to generate a dataset consisting of 40,585 questions with answers derived from the English version of the Quran. The tools exhibited varying levels of performance, with some displaying superior capabilities compared to others. It is noteworthy that although the Cathoven Question Generator had its strengths, the outcomes were not consistent across all tools, highlighting the diverse range of capabilities and results.

In [13], a systematic review of student question generation (SQG) systems is provided. It has been found that since 2000, there has been a growing interest in the development of SQG systems. The review aims to present a comprehensive overview of existing SQG systems. By utilizing a two-dimensional classification scheme, the study identifies the most commonly integrated additional features and design characteristics among 54 SQG learning systems. The research reveals significant variations in design features among SQG systems: the majority are domain-specific, support only one type of question generation, and do not allow the inclusion of multimedia files in student question generation.

In [14], the discipline of AQG, methodologies, datasets, evaluation metrics, and various applications of natural language question generation are discussed. These systems can aid chatbots in generating questions from text, enabling the automatic generation of questions. One potential classification of question generation systems includes visual question generation, stand-alone question generation, and conversation-oriented question generation. The literature presented in the review contributes to understanding the diverse question generation systems and delving into the existing data collections.

The emergence of ChatGPT has sparked intense debate and attention on AI worldwide and has further facilitated the widespread adoption of AI [15], [16], [17], [18], [19], and [20].

In [6], a literature review discusses the impact of ChatGPT in the field of education. ChatGPT is an AI-powered chatbot that was launched in November 2022. It has the capability to generate coherent and informative responses that simulate human conversation. The paper outlines ChatGPT's functionalities across various subjects, its application in education, and the initial challenges faced within the first three months. Through the analysis of 50 articles, it was found that ChatGPT's performance varied, with issues arising from generating inaccurate information and bypassing plagiarism detection systems. The paper recommends that educational institutions promptly revise their evaluation methods and policies and emphasize the importance of providing adequate training to teachers and students for responsible use of ChatGPT in educational environments.

Despite a large body of research and many useful results, there are still areas for further research and unresolved issues in the area of task generation that remain to be addressed. In addition to the indications in the above articles, for example, in [9], the importance of critical thinking and fact-checking is highlighted, and recommendations are made on how to use these models in a responsible and ethical manner in education. The challenges facing the field, such as occasional meaninglessness and the lack of naturalness in the context of information extraction from generated questions, are emphasized in [14]. The biggest problem is that the quality of the questions cannot currently be trusted enough to be used without constant human supervision and regular human review [9].

As can be seen from the above studies, researchers have recently been searching for new and applicable results and answers regarding the utilization of AI [5], [6], [8], [10], [12], and [14]. AI can effectively process lengthy materials and textbooks and generate questions from them. Unfortunately, the quality of AI-generated questions cannot currently be trusted enough to be used without human review. Several studies have also compared different question generation applications, highlighting the strengths and weaknesses of each option. In conclusion, despite significant recent progress in automated question generation, none of the solutions can be considered fully reliable yet. Therefore, it is worthwhile to continue research in this area and leverage the potential of AI tools.

Based on the literature review, it can be assumed that the capabilities of the previously developed Exercise Generation Algorithm+ (EGAL+) could be greatly enhanced by integrating an AI tool for generating questions, given that its scalability has also increased. The goal of EGAL+ was to automate the generation of exams used in educational institutions for assessment, ensuring that they are sufficiently varied, contain tasks that can be included together based on a user-specified set of criteria, and that the exams have the same aggregate difficulty values, ensuring fairness. In a previous literature review, it was found that an application precisely addressing the stated issue does not yet exist, so the significant contribution of EGAL+ is well-founded [21].

Although EGAL+ has proven to be a useful tool, it has several limitations, including (i) restricted input parameters and user parameterization options, (ii) the need for manual question creation by the instructor, and (iii) difficulty in populating the voluminous preference matrix.

This led to the formulation of the following three research objectives for this paper:

i) Enhance the program's logic to improve scalability, enabling it to generate high-quality task sequences from a question bank of at least a thousand questions. This enhancement should maintain similar execution times as with shorter inputs and eliminate previous parameter restrictions.
ii) Integrate an AI tool capable of generating at least a thousand textbook questions, formatted and ready for assembly in EGAL+, thereby eliminating the need for manual question creation.
iii) Enhance support for populating the preference matrix.

This paper presents a novel solution that synergizes modern AI capabilities with a metaheuristic tool. It leverages automated question generation and minimizes the need for human review by processing a whole body of knowledge. This approach makes the questions easily reusable for further high-quality exams with the inclusion of Exercise Generation Algorithm+.

2 THE PURPOSE AND PROCEDURES OF EGAL+

The purpose of EGAL+ is to automate the generation of exams used during assessments in educational institutions. It ensures that the exams are diverse, contain tasks that can be grouped together based on user-specified criteria, and that the task sequences have equal aggregated difficulty values to create fair exercises [21].

The EGAL+ algorithm utilizes a harmony search (HS) metaheuristic approach because of the effectiveness of metaheuristics in solving intricate optimization problems where conventional methods may be suboptimal. In particular, the HS algorithm has shown its capability to efficiently explore the solution space and pinpoint near-optimal solutions for complex problems [22], [23], and [24].

Harmony Search is a metaheuristic optimization algorithm that was initially proposed in 2001. The algorithm is inspired by the way musicians improvise to find pleasing harmony. The idea behind HS is to mimic the behavior that naturally occurs when musicians play their instruments or create music together [25].

Harmony Search, a relatively recent addition to optimization algorithms, has gained widespread usage in various fields for global optimization tasks, owing to its operational simplicity and impressive performance. The HS method is characterized by its efficiency in finding solutions. It has been effectively applied in various fields,

such as mechanical structure design, pipe network optimization, data classification system optimization, and function optimization [26].

The effectiveness of such procedures largely depends on their specific implementation in terms of code. Since they are optimization procedures, numerous algorithms can be developed to yield satisfactory results, with the key distinction among them lying in the requirements for computational power, storage space, and processing time.

Recently, EGAL+ has undergone major improvements through the implementation of mutual dependence conditions on the inputs. This allows anyone without expertise in the field of mathematical optimization to parameterize the program with inputs that lead to a successful run. A run is deemed successful if it produces an output within an acceptable time using the limited resources defined in the given case. In this scenario, all task sequences are equally difficult, and there are no pairs of tasks in any sequence that are prohibited from being included together based on user input. A task sequence is considered of higher quality the more it differs from others and the more it includes tasks that the user has identified as more desirable for inclusion together.

To implement this, the program requires the tasks themselves (*task contents*) as input, from which the task sequences will be assembled. It also needs the difficulty values of the individual tasks on a scale of 1–5 (*task difficulties*) and the preference for the joint inclusion of each pair of tasks on a scale of 0–10 (*coexistence preferences*). Here, 0 represents a prohibitive value, while a higher number indicates a stronger desire for joint inclusion. Additionally, the program needs to know the number of tasks in a task sequence (*exercise length*) and the quantity of task sequences to be generated (*population size*). The combination of task contents and task difficulties will be referred to as the *question bank*.

As the next step, the target difficulty of the task sequences is determined, which is calculated as the sum of the task difficulties included in the individual task sequences. The user is presented with three options: low, medium, and high total difficulty values. Identifying these three options is not a simple task, as it is essential to ensure that, with the selected sum of task difficulties, a population size of task sequences with the exercise length can be created from the provided question bank, taking into account any potential user restrictions on task pairs.

After the user selects the desired target difficulty level for the task sequences, the initial *population* is generated. The initial population comprises the specified number of task sequences, with each sequence containing the designated number of tasks. The total difficulty of each task sequence matches the target difficulty level, which is calculated based on the sum of the difficulties of all tasks within the sequence. Additionally, no task sequence includes any prohibited task pairs. At this stage, it is ensured that a variety of tasks of equal difficulty, meeting all criteria, have been extracted from the question bank. Subsequently, the program proceeds to the quality enhancement module.

As mentioned above, the quality of the task sequences is determined by the differences compared to each other and the coexistence preferences expressed by the user. The higher the coexistence value, the better the quality. To improve the quality of the population, the program utilizes an HS metaheuristic algorithm. Within this framework, the algorithm aims to enhance the quality of the task sequences of the population for a specified number of generations (the default value is 50). During this quality improvement process, the program first generates a random number between 0 and 1. If this number is greater than *HMCR* (default value 0.5), a completely new task sequence is created. If the number is less than or equal to HMCR, one of the existing task sequences is selected and copied, then modified with a *PAR*

probability (default value of 0.2) for a task. It ensures that the task sequence remains valid, whether it is a completely new one or a modified copy of an existing one.

If the program decides to modify one of the existing task sequences, it may not be certain that an element can be found in the question bank within an acceptable short time to be inserted in place of another element in the task sequence or that the replacement is possible at all. In such cases, after a parameterizable time, the program discards the idea of modifying the task sequence and leaves it unchanged.

At this point, the quality improvement attempt can reach one of three states randomly: it can either create a completely new task sequence that fits into the population, copy one without making any changes, or copy one and randomly change one of the tasks included to another appropriate element from the question bank.

Whichever one of the three possibilities arises, the next step of the program is to compare the newly created task sequence with the worst quality of the existing task sequences. If the new task sequence is of better quality than the current worst one, it replaces it with the new one and discards the previous one. If it is not of better quality than the worst one, then it discards the new one.

Regardless of whether a task sequence has been replaced or not, the program compares whether the average quality of the population has increased by at least epsilon (default value: 0.0000000001). If the average quality of the population has not increased by at least epsilon for the failed *epsilon check limit* times (default value: 10) consecutively, the program aborts the execution of new quality improvement iterations and outputs the finished task sequences. Additionally, if the generation limit is reached, the program concludes the quality improvement iterations and outputs the finished task sequences.

3 IMPROVEMENTS REGARDING EFFICIENCY

A major shortcoming of EGAL+ has been its inability to handle the necessity of creating long task sequences for many students from a question bank with a large number of elements. This is a critical situation where teachers could benefit the most from automation due to the complexity of the task. Prior to the advancements introduced in this study, the program was severely limited, with a maximum population size of 100, a question bank of 50, and a highly restricted number of zero values in coexistence preferences. These limitations constrained the input elements and user parameterization significantly.

One of the objectives of these most recent developments was to eliminate all input size restrictions. This allows users to enter an extensive question bank, potentially in the thousands, created from a large preexisting knowledge base, and generate high-quality task sequences, as previously defined, for a substantial number of students, even numbering in the hundreds.

The ability to question the fundamentals of a field of science is particularly beneficial in an educational setting. This is especially true when these fundamentals either remain constant or change only very slowly, requiring in-depth examination by many students on a regular basis. This scenario is commonly observed in undergraduate university education across various disciplines like mathematics, micro- and macroeconomics, anatomy, law, and history. Moreover, with the rise of online education, the capacity to engage multiple students in questioning simultaneously is further amplified.

To eliminate the input restrictions, one of the most important modifications implemented in the program was to change the operating logic during the initial

population generation. As described in Section 2, EGAL+ first creates an initial population based on the specified parameters, which is valid in terms of the requirements set by the coexistence preferences and the aggregated difficulty goal. So far, the generation of the initial population has been characterized by the fact that the algorithm selected the exercise length and number of elements in a single operation from all the possible tasks in the question bank for a task sequence and then checked whether it was valid to include these tasks together. It is easy to see that the more prohibitive values are in the coexistence preferences, and the more different the individual difficulty values given for each task are, the less likely it is that the correct result will follow from selecting the tasks to be inserted into the task sequence at the same time, because these factors all reduce the number of acceptable combinations.

For this reason, the program was modified so that, during the creation of the initial population, the exercise length and number of tasks are not selected all at once. Instead, the selection process is organized into iterations to choose only one new task at a time that meets the requirements set by the coexistence preferences. This process continues until enough tasks are successfully chosen for a task sequence.

Even in this solution, additional control operations are required. For example, preparations must be made for the possibility that occasionally a task is chosen that, although it corresponds to the ones chosen up to that point, does not allow the completion of the whole task sequence. In this case, it must be ensured that the program does not enter an infinite loop. Overall, a better result can be achieved with this new approach than with the previously used method.

Although at first it may seem like a more computationally demanding task to check the task sequence each time after a component is added to see if it is built correctly, it still results in a much more efficient run than checking it once after it has been created. This is because it greatly reduces the probability of creating unnecessary task sequences, thereby reducing the overall time required for generating the output.

The method of storing population elements has also been changed. Previously, individual elements were represented as 0s and 1s in a vector, where a value of 1 indicated an included task and a value of 0 indicated a non-included task. This resulted in a vector with a length equal to the number of questions in the question bank. Now, instead of using 0s and 1s, the indexes of the included tasks are stored. This change reduces the size of population elements and streamlines processing, as the length of a task sequence is now determined by the exercise length rather than the potentially larger size of the question bank.

The third significant modification concerning the efficiency of the program is that it now generates the starting population while searching for target difficulty options. Instead of initially finding the potential population target difficulty options in a separate operation and then creating the initial population after the user's selection, the difficulty options are now determined by analyzing randomly generated task sequences and classifying them. This process results in creating an initial population for each possible target difficulty option. After the selection, this initial population can be treated as such, allowing the program to commence enhancing its quality. Consequently, this approach significantly reduces the overall operational requirements.

With the help of the developments outlined in this chapter, the need for input size limitations has been successfully eliminated. It is now possible to generate exams for hundreds of students from a question bank containing thousands of questions in just seconds of execution time, without any solvable coexistence prohibitions. In the previous state of the program, such an attempt would have led to practically infinite running.

For the exact implementation, the source code of EGAL+ can be found in the following GitHub repository: https://github.com/balazs-domsodi/EGALPP.

4 INTEGRATING GENERATIVE AI FOR STREAMLINING QUESTION BANK CREATION

Although recent developments have significantly enhanced the program's scalability, as discussed in Section 3, these advancements have also introduced new challenges that must be addressed before EGAL+ can be deployed in a real-world setting. One of these challenges is that the creation and categorization of numerous questions to fully leverage the program's benefits may exceed instructors' resources in most scenarios.

Given the substantial challenge in this case of creating a large volume of questions in textual form based on an extensive text source, it was identified that this issue can be addressed through the integration of an AI module to generate questions from a knowledge base, such as a textbook. Subsequently, the recently described metaheuristic algorithm can compile the specific exercises, providing a comprehensive solution for generating exams, which is the primary purpose of Exercise Generation Algorithm+.

In the field of natural language processing, AQG applications are prominent. These applications autonomously generate questions from text sources and images, guided by a particular subject or concept. They have been extensively tested in educational environments and are regularly used in machine reading comprehension tools and conversational systems. This has led to a significant increase in their popularity due to their versatility and effectiveness. Closed- and open-domain AQG applications can be distinguished. For closed-domain question generation, queries are formulated specific to a field such as medicine or educational literature, drawing upon knowledge that is unique to the domain and bound by an ontology. In contrast, open-domain question generation is not tied to any specific domain and permits the creation of questions regardless of the domain, necessitating only global ontologies [14]. Because the goal of this research is to create an application for general use, open-domain options could be considered.

The utilization of LLMs for AQG has undergone significant development in recent years, evolving from early approaches that treated MCQ generation as a pipeline of subtasks to more recent deep learning approaches. These models are now capable of generating questions, answers, and distractor answer options for MCQs by providing them with sentences from a textbook. Despite some challenges, studies have shown that LLMs are remarkably capable of creating MCQs matching human performance on most metrics [10]. Considering these recent findings and the necessity for high-quality questions in the EGAL+ question bank that can replace human-generated content, along with the potential for future automatic evaluation, the objective was to incorporate an AQG solution using LLM for generating multiple-choice questions.

For selecting the appropriate tool in a specific context of generating MCQs, research studies are actively comparing various alternatives to LLMs such as GPT-3 and GPT-4, as well as different AQG applications. These studies measure their effectiveness in generating relevant and accurate questions and answers for MCQs. The findings suggest that while some tools outperform others, there is considerable potential in this field [5] [12].

Given the availability of numerous suitable tools, it could be beneficial to incorporate multiple alternatives into EGAL+ in the future. This would allow users to select the application programming interface that best fits their specific use case. The primary objective of this research is to demonstrate that an AI-based question generation tool can be effectively combined with a metaheuristic algorithm for mass-producing high-quality exams. To illustrate this concept, PrepAI is selected and integrated, a popular tool available at https://prepai.io. Notably, PrepAI meets all the previously established criteria for a tool suitable for implementation in EGAL+, known for its versatility and ability to process PDF inputs.

5 PRACTICAL DEMONSTRATION OF RESULTS

In this section, the capabilities of the recently improved EGAL+ are demonstrated, particularly its efficiency in integrating an AI module for processing numerous questions and describing each of the program's three main modes. The current UI of the program listing its modes is shown in Figure 1.

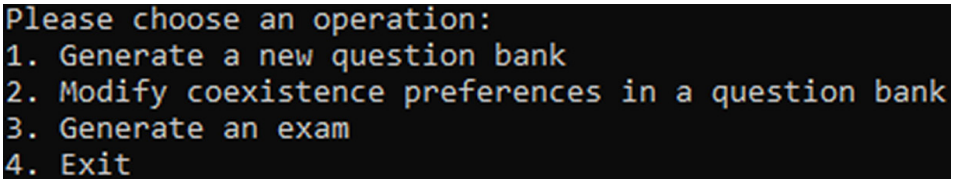

Fig. 1. Screenshot showing the current UI of the program, listing its modes in a menu structure

The first step of the program is to generate new question banks. This process requires providing a name for the question bank to be created, a topic name, the input file (typically a textbook), the number of questions to be generated, and the page count and/or specific page ranges to be included from the input file for question generation.

Thanks to the advanced capabilities of PrepAI, it is now possible to import hundreds of pages from a PDF document and generate up to a thousand questions. Additionally, due to its seamless integration with EGAL+, it can be operated directly from within this program. This integration guarantees that the resulting text file containing the question bank is already formatted for further processing in EGAL+. Each row in this file includes the generated question, the suggested difficulty value for the question (ranging from 1–2 as per PrepAI), and coexistence preferences ranging from 0 to 10, separated by semicolons. Each coexistence preference value indicates the desired joint inclusion of that question with the preceding ones in order, with the default value being 10 at the time of question generation. The answer options are also separated by semicolons, with the correct answer denoted by an asterisk. All four elements in each row are separated by tab characters. The parameterization of the first mode can be seen in Figure 2.

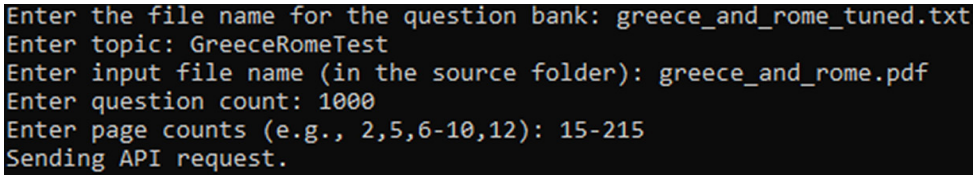

Fig. 2. Screenshot showing an example parameterization of the first mode

The second mode is a new improvement aimed at enhancing the support for populating the preference matrix by allowing the optional modification of coexistence preferences. This mode requires the question bank itself as an input. Users can define as many question groups as they desire by specifying the question numbers and/or ranges, along with the intended coexistence preferences for each group. Once the user finishes defining the question groups, they are prompted to provide coexistence preferences for the disjoint ones from the created groups. Users are asked to specify the coexistence preference values for each pair, and the corresponding values in the question bank are then adjusted based on the user input. The parameterization of the second mode can be seen in Figure 3.

```
Enter the file name of the question bank: greece_and_rome_tuned.txt
Enter the group of questions (e.g., 2,5,6-10,12): 1-38
Enter the joint inclusion preference (0-10): 6
Do you want to define another group of questions? (yes/no): yes
Enter the group of questions (e.g., 2,5,6-10,12): 39-140
```

Fig. 3. Screenshot showing an example parameterization of the second mode

The generation of the exams is the third and final mode, which involves HS metaheuristic optimization. The program takes the generated question bank with optionally modified coexistence preferences as input. It also prompts the user for the number of desired questions in an exam and the number of exams to be generated. The program then identifies three achievable total difficulty goals for the exams with the largest possible difference between them to provide the user with diverse options. Once the desired difficulty goal is selected, the program generates the exams. It considers all coexistence preferences, potential prohibitions on including specific questions, strictly maintains the difficulty goal, and aims to create the most varied exams possible while adhering to all criteria. The parameterization of the third mode can be seen in Figure 4.

```
Enter the name of the question bank: greece_and_rome_tuned.txt
Please specify the desired exercise length:
30
Please specify the desired population size:
100
Please choose from the difficulty options below:
67
83
99
```

Fig. 4. Screenshot showing an example parameterization of the third mode

For this demonstration, a question bank consisting of 1000 questions was generated from the first 200 pages of the book “Myths and Legends of Ancient Greece and Rome” by E. M. Berens [27] using the program’s first mode. This book was accessed through a volunteer-led initiative dedicated to the preservation and digitization of culturally significant works known as Project Gutenberg, accessible at https://gutenberg.org. The topic selection was guided by its inclusion in general education to ensure the broad interpretability of the demonstration.

For the generated 1000 questions, the following question groups were created in the second mode: G1: 1–38, G2: 39–140, G3: 141–279, G4: 280–540, G5: 541–635, G6: 636–1000, and G7: 637–649. The coexistence preferences in each group were set to 6, except for G6, where it was set to 3, and G7, where it was set to 0. The coexistence preferences between disjoint groups were set to 10 in each case, except for G4–G5, where they were set to 0, and for G6, which for each disjoint other group was set to 7.

The coexistence preference values used for the questions derived from the book are randomly selected rather than expertly determined. However, this randomness does not impact the successful operation of the program.

To demonstrate the ability of EGAL+ to handle difficulty values for single questions in the range of 1–5, the initial value of 1 for PrepAI in the question bank was randomly adjusted within the range of 1–2, and the initial two values were randomly adjusted within the range of 3–5.

In the third mode, 100 exams of 30 questions were generated by selecting the medium difficulty option out of the three total difficulty options. The program’s quality improvement module can enhance the results by approximately 1.3% (as detailed in Chapter 4), surpassing the initial population that already meets all strict criteria.

It is notable that after generating 1000 questions from the 200 pages using the integrated PrepAI module, the generation of 100 exams of 30 questions was completed in less than a second by the HS module of the program, meeting all the specified user criteria. This showcases the power and efficiency of EGAL+ when integrated with an AI module to produce high-quality exams rapidly, requiring minimal manual effort from the users.

All the results in this demonstration were obtained from a laptop equipped with an Intel Core i7-9750H 2.6 GHz CPU and 16.0 GB of RAM. The resulting question bank is available at https://github.com/balazs-domsodi/EGALPP/blob/main/databank/greece_and_rome.txt, and the resulting task sequences are available at https://github.com/balazs-domsodi/EGALPP/blob/main/output/tasks.txt. The task indices of the initial population are available at https://github.com/balazs-domsodi/EGALPP/blob/main/output/initial.txt, and the task indices of the enhanced population are available at https://github.com/balazs-domsodi/EGALPP/blob/main/output/enhanced.txt, with the chosen aggregated difficulty value in the beginning of both files.

It's important to note that while the modes for adjusting coexistence preferences and generating exams only require suitable inputs, creating new question banks necessitates a PrepAI account. Users must possess such an account and input the required credentials into a client_id.txt and client_secret.txt file located in the program's root directory.

This study presents a method for educators to automatically process textbooks and generate question banks, enabling the creation of high-quality exams in seconds. However, it's important to note that the AI used for question generation may introduce errors. The authors propose an initial, one-time investment of time for an expert to review and correct every AI-generated question bank. This step is crucial until generative AI achieves full professional reliability. The primary advantage of EGAL+ is that once the generated question bank is proofread and the coexistence preference values are fine-tuned, educators can repeatedly generate high-quality exams from specific course materials for an extended period of time in subjects that don't change rapidly. This process ensures long-term, consistent quality in assessments, freeing up educators' time spent assembling exams from textbooks.

6 CONCLUSION AND FUTURE IMPROVEMENTS

This study addresses a practical problem in educational institutions: the manual compilation of exercises often leaves educators with less time for tasks that require their personal attention, such as providing individualized student support. Alternatively, if insufficient time is allocated to exercise preparation, the quality of these materials may be compromised, potentially hindering optimal student assessment.

The research problem identified is the computational automation of exercise assembly from educational resources. This complex problem presents numerous challenges, including ensuring the quality and appropriateness of the assembled exercises and handling diverse educational resources.

A review of the literature revealed that while many sophisticated exercise-generating systems exist, none are capable of addressing the specific problem identified in this study: generating multiple, distinct subsets of predetermined tasks based on quality criteria such as difficulty and categorization. This represents a research gap, as none of the solutions or ongoing research discovered by the authors aim to address exercise generation and compilation in such a generic and standardizable manner.

The objectives of this research were formulated as follows:

i) Enhance the program's logic to improve scalability, enabling it to generate high-quality task sequences from a question bank of at least 1000 questions. This enhancement should maintain similar execution times as with shorter inputs and eliminate previous parameter restrictions.
ii) Integrate an AI tool capable of generating at least 1000 textbook questions, formatted and ready for assembly in EGAL+, thereby eliminating the need for manual question creation.
iii) Enhance support for populating the preference matrix.

The research objectives were successfully met by enhancing the program's logic and integrating a generative AI tool into the program's operations, as documented and demonstrated in the paper. To achieve the research objectives, the authors further developed an algorithm they had previously created for generating task sequences, Exercise Generation Algorithm+.

The improvements presented in this study focus on enhancing the program's scalability to process more data and integrating an AI-based text-processing module capable of generating a large number of MCQs from PDF-based sources. The study demonstrates and documents the effectiveness and efficiency of the resulting program, indicating that all the research objectives were successfully achieved. All information necessary to reproduce the results is made available.

The enhanced EGAL+ has proven capable of automatically processing textbooks and generating questions, a feat made possible by the integrated AI module. Following a single proofreading session and fine-tuning the coexistence preference values, educators can leverage the implemented metaheuristic algorithm to consistently generate high-quality exams from the provided course materials in a remarkably short time. This process ensures the long-term, consistent quality of assessments and reduces the time teachers spend compiling exams from textbooks.

However, there are still many opportunities for further development of the program. One of the most significant opportunities is updating difficulty values based on the results achieved by participants on the prepared assessments. This would enable the question bank to have increasingly accurate difficulty values after each use, facilitating the self-improvement of the question bank through usage. Another potential enhancement for the future involves integrating the AI tool's generated answer options into the program and implementing an automatic assessment feature to provide additional support to instructors. Other potential improvements include exploring integration possibilities with learning management systems and designing a user interface that caters to the needs of educators.

7 REFERENCES

[1] V. Nentwich, N. Fischer, A. C. Sonnenbichler, and A. Geyer-Schulz, "Computer aided exercise generation – A framework for human interaction in the automated exercise generation process," in *Proceedings of the 13th International Joint Conference on e-Business and Telecommunications (ICETE 2016)*, 2016, pp. 57–63. https://doi.org/10.5220/0005947700570063

[2] Y.-A. Bachiriand and H. Mouncif, "Artificial intelligence system in aid of pedagogical engineering for knowledge assessment on MOOC platforms: Open EdX and Moodle," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 18, no. 5, pp. 144–160, 2023. https://doi.org/10.3991/ijet.v18i05.36589

[3] S. Pandraju and S. G. Mahalingam, "Answer-aware question generation from tabular and textual data using T5," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 16, no. 18, pp. 256–267, 2021. https://doi.org/10.3991/ijet.v16i18.25121

[4] N. Willert and J. Thiemann, "Template-based generator for single-choice questions," *Technology, Knowledge and Learning*, vol. 29, pp. 355–370, 2024. https://doi.org/10.1007/s10758-023-09659-5

[5] A. Tran, K. Angelikas, E. Rama, C. Okechukwu, D. H. Smith, and S. MacNeil, "Generating multiple choice questions for computing courses using large language models," in *2023 IEEE Frontiers in Education Conference (FIE)*, College Station, TX, USA, 2023, pp. 1–8. https://doi.org/10.1109/FIE58773.2023.10342898

[6] C. K. Lo, "What is the impact of ChatGPT on education? A rapid review of the literature," *Education Sciences*, vol. 13, no. 4, p. 410, 2023. https://doi.org/10.3390/educsci13040410

[7] C.-Y. Chung, I.-H. Hsiao, and Y.-L. Lin, "AI-assisted programming question generation: Constructing semantic networks of programming knowledge by local knowledge graph and abstract syntax tree," *Journal of Research on Technology in Education*, vol. 55, no. 1, pp. 94–110, 2022. https://doi.org/10.1080/15391523.2022.2123872

[8] A. Ushio, F. Alva-Manchego, and J. Camacho-Collados, "A practical toolkit for multilingual question and answer generation," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, *System Demonstrations*, Toronto, Canada. Association for Computational Linguistics, vol. 3, 2023, pp. 86–94. https://doi.org/10.18653/v1/2023.acl-demo.8

[9] E. Kasneci *et al.*, "ChatGPT for good? On opportunities and challenges of large language models for education," *Learning and individual differences*, vol. 103, p. 102274, 2023. https://doi.org/10.1016/j.lindif.2023.102274

[10] A. M. Olney, "Generating multiple choice questions from a textbook: Llms match human performance on most metrics," in *AIED Workshops*, 2023.

[11] J. Heilala, A. Shibani, and A. Gomes de Freitas, "The requirements for heutagogical attunement within STEAM education," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 18, no. 16, pp. 19–35, 2023. https://doi.org/10.3991/ijet.v18i16.42313

[12] S. S. M. Alnefaie, E. Atwell, and M. A. Alsalka, "Using automatic question generation web services tools to build a Quran question-and-answer dataset," *International Journal on Islamic Applications in Computer Science and Technology*, vol. 11, no. 2, pp. 1–12, 2023.

[13] F.-Y. Yu and C.-W. Kuo, "A systematic review of published student question-generation systems: Supporting functionalities and design features," *Journal of Research on Technology in Education*, vol. 56, no. 2, pp. 172–195, 2022. https://doi.org/10.1080/15391523.2022.2119448

[14] N. Mulla and P. Gharpure, "Automatic question generation: A review of methodologies, datasets, evaluation metrics, and applications," *Progress in Artificial Intelligence*, vol. 12, pp. 1–32, 2023. https://doi.org/10.1007/s13748-023-00295-9

[15] H. H. Thorp, "ChatGPT is fun, but not an author," *Science*, vol. 379, no. 6630, pp. 313–313, 2023. https://doi.org/10.1126/science.adg7879

[16] E. A. Van Dis, J. Bollen, W. Zuidema, R. van Rooij, and C. L. Bockting, "ChatGPT: Five priorities for research," *Nature*, vol. 614, no. 7947, pp. 224–226, 2023. https://doi.org/10.1038/d41586-023-00288-7

[17] A. Tatnall and A. Fluck, "Twenty-five years of the education and the information technologies journal: Past and future," *Education and Information Technologies*, vol. 27, pp. 1359–1378, 2022. https://doi.org/10.1007/s10639-022-10917-9

[18] T. T. A. Ngo, "The perception by university students of the use of ChatGPT in education," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 18, no. 17, pp. 4–19, 2023. https://doi.org/10.3991/ijet.v18i17.39019

[19] O. Abu Khurma, N. Ali, and R. Hashem, "critical reflections on ChatGPT in UAE education: Navigating equity and governance for safe and effective use," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 18, no. 14, pp. 188–199, 2023. https://doi.org/10.3991/ijet.v18i14.40935

[20] L. H. Al-Obaydi, M. Pikhart, and B. Klimova, "ChatGPT and the general concepts of education: Can artificial intelligence-driven chatbots support the process of language learning?" *International Journal of Emerging Technologies in Learning (iJET)*, vol. 18, no. 21, pp. 39–50, 2023. https://doi.org/10.3991/ijet.v18i21.42593

[21] B. Láng and B. Dömsödi, "Development of the improved exercise generation metaheuristic algorithm EGAL+ for end users," *International Journal of Emerging Technologies in Learning*, vol. 17, no. 11, pp. 210–224, 2022. https://doi.org/10.3991/ijet.v17i11.28099

[22] M. Dubey, V. Kumar, M. Kaur, and T. P. Dao, "A systematic review on harmony search algorithm: Theory, literature, and applications," *Mathematical Problems in Engineering*, vol. 2021, no. 1, p. 5594267, 2021. https://doi.org/10.1155/2021/5594267

[23] F. Qin, A. M. Zain, and K. Q. Zhou, "Harmony search algorithm and related variants: A systematic review," *Swarm and Evolutionary Computation*, vol. 74, p. 101126, 2022. https://doi.org/10.1016/j.swevo.2022.101126

[24] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Applied mathematics and computation*, vol. 188, no. 2, pp. 1567–1579, 2007. https://doi.org/10.1016/j.amc.2006.11.033

[25] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001. https://doi.org/10.1177/003754970107600201

[26] X. Z. Gao, V. Govindasamy, H. Xu, X. Wang, and K. Zenger, "Harmony search method: Theory and applications," *Computational intelligence and neuroscience*, vol. 2015, no. 1, p. 258491, 2015. https://doi.org/10.1155/2015/258491

[27] E. M. Berens, "Myths and Legends of Ancient Greece and Rome," Urbana, Illinois: Project Gutenberg. Retrieved from https://www.gutenberg.org/ebooks/22381

8 AUTHORS

Blanka Láng is an Associate Professor at the Corvinus University of Budapest. Her research areas include harmony and genetic metaheuristic algorithm development, linear regression model selection, and exercise generation problems (E-mail: blanka.lang@uni-corvinus.hu).

Balázs Dömsödi is a Ph.D. student in Business Informatics Engineering at Corvinus University of Budapest. His research interests include harmony metaheuristic algorithm development and exercise generation problems (E-mail: balazs.domsodi@stud.uni-corvinus.hu).