Sistema multiagente para a geração automática de questões de programação

Júlio César da S. Dantas[1], **Eduardo Henrique da Silva Aranha**[1]

[1]Departamento de Informática e Matemática Aplicada (DIMAp)
Universidade Federal do Rio Grande do Norte, Natal - RN.

dantas.j.c.s@gmail.com, eduardo@dimap.ufrn.br

Resumo. *A elaboração de questões abertas de programação é uma tarefa complexa e demandante de tempo, exigindo conhecimentos pedagógicos e técnicos. Este trabalho propõe um sistema multiagentes para a geração automática dessas questões, utilizando a LangGraph como ferramenta de orquestração. A arquitetura desenvolvida permite que agentes especializados criem enunciados, verifiquem coerência, elaborem casos de teste e padronizem código de forma colaborativa, reduzindo a necessidade de intervenção humana e aumentando a consistência e qualidade pedagógica do material. Resultados parciais indicam que o uso de múltiplos agentes permite contornar limitações comuns de modelos de IA generativa, como inconsistências nos enunciados e erros de código, e possibilita adaptação a diferentes níveis de dificuldade e contextos educacionais. O estudo contribui para a área de sistemas multiagentes e oferece uma estratégia replicável para integração de múltiplos LLMs ou agentes inteligentes em tarefas educacionais automatizadas.*

Abstract. *Creating open-ended programming questions is a complex and time-consuming task that requires both pedagogical and technical expertise. This paper proposes a multi-agent system for the automatic generation of such questions, using LangGraph as an orchestration tool. The developed architecture allows specialized agents to collaboratively create problem statements, verify coherence, generate test cases, and standardize code, reducing the need for human intervention while improving the consistency and pedagogical quality of the generated material. Preliminary results indicate that the multi-agent approach overcomes common limitations of generative AI models, such as statement inconsistencies and coding errors, and allows adaptation to different difficulty levels and educational contexts. This research contributes to the field of multi-agent systems and presents a replicable strategy for integrating multiple LLMs or intelligent agents in automated educational tasks.*

1. Introdução

A resolução de problemas tem um papel importante no aprendizado, em especial de disciplinas fundamentalmente práticas, como a programação. Por meio dela, o estudante não apenas consolida conhecimentos teóricos, mas também desenvolve habilidades cognitivas como raciocínio lógico, pensamento crítico e capacidade de abstração. Além disso, a prática constante na solução de problemas contribui para a fixação de conceitos, a aplicação de técnicas de forma contextualizada e a construção de estratégias próprias para enfrentar desafios complexos [Thalheimer 2003].

DOI: 10.5753/cbie_estendido.2025.16241

No entanto, a elaboração de questões de programação demanda investimentos significativos em termos de tempo e dedicação de pessoal qualificado. Esse processo não se limita apenas à formulação do enunciado, mas envolve a criação de contextos adequados, a definição de requisitos claros, a preparação de casos de teste abrangentes e a verificação da correção e pertinência das soluções esperadas. Além disso, há o desafio adicional de produzir questões que sejam, ao mesmo tempo, desafiadoras, contextualizadas e pedagogicamente relevantes, de forma a manter o engajamento dos estudantes e atender aos objetivos de aprendizagem propostos.

Sendo assim, um campo recente de estudos que tem despertado crescente interesse é o da geração automática de questões, embora a maior parte das pesquisas ainda se concentre em disciplinas e formatos distintos da programação. Por exemplo, [Huang and Mostow 2015] exploram a criação de questões do tipo "complete a sentença" com base em critérios gramaticais; [Odilinye et al. 2015] propõem um método que combina técnicas semânticas e uso de templates para formular enunciados; e [Kumar et al. 2015] investigam a geração automática de questões objetivas utilizando medidas de similaridade semântica obtidas com word2vec para selecionar as melhores alternativas de distratores.

Com os avanços recentes dos modelos de inteligência artificial generativa, como ChatGPT, Mixtral, LLAMA, Gemini e outros, esse campo de estudo tem experimentado uma guinada, ampliando as possibilidades de criação automática de questões. Diferentemente das abordagens tradicionais, que dependiam de regras fixas, templates prédefinidos ou recursos limitados de processamento de linguagem natural, os modelos generativos atuais conseguem compreender e produzir texto com maior fluidez, coerência e contextualização. Isso possibilita não apenas a formulação de questões mais variadas e adaptadas a diferentes níveis de dificuldade, mas também a geração de enunciados contextualizados, exemplos práticos, casos de teste e até explicações de gabarito. Por exemplo, [Chan et al. 2023] utilizaram o ChatGPT para a geração de questões abertas de programação, destacando o potencial dos modelos generativos para agilizar essa tarefa, mas enfatizando a necessidade de intervenção humana ao longo do processo. De forma semelhante, [Meißner et al. 2024] chegaram a conclusões próximas, reconhecendo essas ferramentas como potencializadoras do trabalho docente, mas não como substitutas no processo de elaboração de questões. Em um trabalho mais recente, [Santana et al. 2025a] avaliaram diferentes técnicas de prompt para a criação de questões abertas e corroboraram o consenso existente na literatura sobre a rapidez proporcionada pelo uso dessas ferramentas.

Entretanto, assim como os demais estudos mencionados, também identificaram limitações significativas, como a falta de clareza em alguns enunciados, a presença de inconsistências, erros de código e descumprimento de padrões de programação, evidenciando que a qualidade do material gerado ainda depende de revisão e curadoria humanas. Diante desse cenário, torna-se relevante investigar como modelos de IA generativa podem ser aplicados especificamente à geração automatizada de questões de programação, explorando estratégias para mitigar suas limitações e potencializar seus benefícios no contexto educacional.

Dito isso, o objetivo geral do trabalho investigar a capacidade do uso de sistemas multiagentes para a geração automática de questões abertas de programação, utilizando

a LangGraph como ferramenta de orquestração, visando aumentar a eficiência, a consistência e, de maneira geral, a qualidade pedagógica do material produzido, ao mesmo tempo em que minimiza a necessidade de intervenção humana. Para isso, deve ser construída uma arquitetura multiagente capaz de executar tarefas especializadas na criação de questões, avaliar o impacto da utilização da LangGraph na organização e integração dos agentes; comparar a abordagem multiagente com métodos tradicionais de prompt em termos de robustez, redução de erros, consumo de tokens, e adaptação de níveis de dificuldade, contextos e estilos das questões; analisar a qualidade pedagógica das questões geradas, verificando sua relevância, clareza e aderência aos objetivos de aprendizagem; e explorar estratégias para mitigar limitações conhecidas em modelos de IA generativa, como inconsistências nos enunciados, erros de código e descumprimento de padrões de programação.

2. Motivação

Como discutido anteriormente, o estado atual das pesquisas sobre geração automática de questões abertas de programação ainda se baseia predominantemente em estratégias relativamente simples de prompt, como chain-of-thought, one-shot e few-shot, ou depende de intervenção humana durante o processo de criação, como observado em [Chan et al. 2023]. Embora esses estudos tenham alcançado resultados promissores, eles também evidenciaram limitações importantes que ainda precisam ser superadas.

Nesse contexto, o presente trabalho amplia as abordagens anteriores ao adotar uma metodologia de multiagentes para a geração automática de questões, buscando reduzir os erros observados em métodos anteriores e minimizar a necessidade de intervenção humana direta. A participação humana permanece restrita à seleção e curadoria das questões geradas, permitindo que todo o processo de criação seja realizado de forma automatizada, mais escalável e consistente, ao mesmo tempo em que mantém a qualidade pedagógica do material produzido.

Diferentemente das abordagens tradicionais de prompt, que seguem esquemas rígidos, os sistemas multiagentes oferecem maior adaptabilidade na geração de questões. Cada agente pode desempenhar funções especializadas, como criação de enunciados, elaboração de casos de teste, verificação de coerência e padronização de código, permitindo múltiplas validações de forma paralela e colaborativa. Essa arquitetura não-linear pode aumentar a robustez do processo, reduzir erros comuns em métodos centralizados e possibilitar a adaptação automática de níveis de dificuldade, contextos e estilos das questões, aproximando a produção de material ao comportamento de um corpo docente humano sem necessidade de intervenção constante. Conjectura-se que o problema se beneficie de uma abordagem não-linear vista que o estado atual de pesquisas revela deficiências associadas a geração linear, apontando, por exemplo, erros de semântica e sintaxe em trechos de código e enunciados confusos, como em [Santana et al. 2025a] e [Meißner et al. 2024]; enquanto algumas desses trabalhos propõem *humans-in-the-loop* como solução, a proposta aqui se mune de agentes extras e ciclos de comunicação para mitigar esses problemas.

Para orquestrar os agentes, propõe-se o uso da LangGraph, uma biblioeca baseada na plataforma LangChain, que integra de forma eficiente os agentes especializados e facilita a execução paralela e coordenada das tarefas [Oshin and Campos 2025], potencial-

izando a qualidade das questões geradas. Muitos *frameworks* existem para a orquestração de agentes, como o ReAct - proposto por [Yao et al. 2023]. Nele, o agente combina etapas explícitas de raciocínio (Reason), que permitem organizar o pensamento e planejar soluções, com etapas de ação (Act), que envolvem interações com o ambiente ou ferramentas externas para obter informações adicionais ou executar verificações. Esse ciclo iterativo de raciocínio e ação torna os agentes mais robustos, capazes de lidar com problemas que exigem tanto reflexão quanto operações práticas [Yao et al. 2023]. O sistema proposto neste trabalho, embora compartilhe a inspiração na ideia de ciclos iterativos de raciocínio, diferencia-se do ReAct por adotar uma arquitetura multiagente baseada na divisão de tarefas especializadas e não contar com a chamada a ferramentas externas (Act). Em vez de um único agente alternando entre raciocinar e agir, a solução utiliza múltiplos agentes com papéis distintos — como gerar enunciados, validar coerência, elaborar casos de teste e padronizar código — que colaboram de forma orquestrada por meio da LangGraph. Essa configuração possibilita maior paralelismo, especialização e consistência no processo de geração de questões de programação.

As questões de pesquisa incluem: Qual é a eficácia de sistemas multiagentes na geração de questões abertas de programação, considerando consistência, clareza, relevância pedagógica e redução da necessidade de intervenção humana? E como as questões geradas por essa abordagem se comparam a métodos tradicionais de prompt, como few-shot, one-shot e chain-of-thought, outras abordagens que prevêem humanos no ciclo de desenvolvimento e até questões geradas completamente por humanos?

3. Metodologia

A presente pesquisa adota a Design Science Research (DSR) como metodologia, por buscar simultaneamente resolver um problema prático em um contexto específico — a geração automática de questões abertas de programação — por meio de um artefato, e produzir novo conhecimento científico aplicável em contextos semelhantes [Pimentel et al. 2020].

O ciclo da DSR será seguido conforme as fases clássicas. Primeiramente, a identificação do problema e a definição dos objetivos se baseiam na demanda educacional por ferramentas de apoio à elaboração de questões abertas e no desafio de garantir qualidade e relevância em perguntas geradas automaticamente. Em seguida, na fase de design e desenvolvimento, haverá a construção iterativa do artefato, integrando as diretrizes propostas por [Shah 2024] para mitigar vieses e aumentar a transparência no uso de LLMs. Essas diretrizes estabelecem que as respostas desejadas devem ser verificadas em múltiplas dimensões e para amostras suficientemente grandes, por pelo menos dois avaliadores humanos, e que o prompt utilizado deve gerar respostas consistentes, sendo justificável, explicável e verificável.

Além da proposta metodológica apresentada por [Shah 2024], o percurso metodológico para o desenvolvimento do artefato inclui a realização de entrevistas semiestruturadas com professores, bem como a condução de experimentos envolvendo protótipos e questionários de usabilidade - como o avaliado em [Bangor et al. 2008]. Essa abordagem permite não apenas coletar percepções detalhadas e contextuais dos docentes sobre as funcionalidades e usabilidade do artefato, mas também testar iterativamente soluções propostas, garantindo que o desenvolvimento seja orientado por *feedback*

contínuo e alinhado às necessidades reais dos usuários.

Na fase de avaliação, será verificado se o artefato resolve o problema, por meio de estudos de caso envolvendo professores e alunos, analisando a qualidade, a clareza e a aplicabilidade pedagógica das questões geradas. Por fim, a fase de comunicação abrangerá a documentação e a divulgação dos resultados, incluindo as lições aprendidas, o codebook desenvolvido e recomendações metodológicas para pesquisas que empreguem LLMs na educação. A avaliação das questões, junto a alunos e professores, deve ser realizada a partir de entrevistas semiestruturadas e de questionários com perguntas de múltipla escolha sobre as questões geradas, baseada, em especial, no trabalho de [Santana et al. 2025b].

4. Resultados parciais e esperados

Após um ano de pesquisa sobre o uso de inteligência artificial generativa para a criação de questões abertas em programação, foi possível identificar erros recorrentes cometidos pelos modelos e mapear estratégias com diferentes níveis de eficácia para contorná-los. Esse trabalho resultou na publicação de dois artigos que apresentam resultados preliminares, contribuindo para o entendimento das limitações e potencialidades dos LLMs nesse contexto. Um dos resultados mais relevante foi o encadeamento de chamadas entre assistentes para alcançar diminuir a dependência de um avaliador humano no ciclo de geração, indicando possibilidades de automação parcial do processo sem comprometer a qualidade das questões geradas - a metodologia multiagente ergue-se a partir dessa experiência, deixando-a menos linear e mais ciclíca.

No entanto, alguns desafios permanecem. Um deles é construir uma proposta que seja rígida o suficiente para garantir a geração consistente de questões apropriadas, mas flexível o bastante para se adaptar a diferentes níveis de complexidade e tipos de questões. Outro desafio é estabelecer critérios de avaliação que permitam medir com precisão a qualidade das questões geradas, seja em comparação com questões produzidas por humanos outros esquemas de geração automática ou por métricas próprias, garantindo que a eficácia do artefato seja avaliada de forma relevante para contextos educacionais distintos. Outro desafio é conhecer que tipos de interação professor-sistema são melhores avaliadas pelos usuários e os intervalos de tempo aceitáveis para a geração das questões.

Os próximos passos envolvem a conclusão da primeira versão do sistema multiagentes, em contato com professores da área, e então começar a realizar estudos de caso com outros professores e alunos, de maneira a avaliar o uso da aplicação e a efetividade das questões geradas.

Espera-se, com isso, oferecer uma proposta de solução para a geração automática de questões abertas de programação que minimize ou até elimine a necessidade de intervenção humana no processo de criação e seleção das questões. A intenção é que o sistema seja capaz de produzir questões consistentes, relevantes e pedagogicamente adequadas de forma autônoma, garantindo qualidade e diversidade nos tipos de perguntas geradas. Além disso, busca-se que a solução permita adaptação a diferentes níveis de dificuldade, contextos educacionais e objetivos de aprendizagem, oferecendo um mecanismo confiável para apoiar professores e instituições de ensino na avaliação de competências de programação de forma mais eficiente, escalável e sistemática.

A pesquisa visa também avançar na área de sistemas multiagentes, apresentando

uma proposta estruturada que organiza a interação entre agentes e pode ser replicado e adaptado a diferentes contextos e domínios de aplicação; facilitando a implementação de soluções complexas envolvendo múltiplos LLMs ou outros agentes inteligentes, reduzindo a necessidade de intervenções humanas no processo.

References

Bangor, A., Kortum, P. T., and Miller, J. T. (2008). An empirical evaluation of the system usability scale. *International Journal of Human–Computer Interaction*, 24(6):574–594.

Chan, W., An, A., and Davoudi, H. (2023). A case study on chatgpt question generation. In *2023 IEEE International Conference on Big Data (BigData)*, pages 1647–1656.

Huang, Y. T. and Mostow, J. (2015). Evaluating human and automated generation of distractors for diagnostic multiple-choice cloze questions to assess children's reading comprehension. In Conati, C., Heffernan, N., Mitrovic, A., and Verdejo, M., editors, *Artificial Intelligence in Education*, pages 155–164. Springer International Publishing, Cham.

Kumar, G., Banchs, R., and D'Haro, L. F. (2015). Automatic fill-the-blank question generator for student self-assessment. In *IEEE Frontiers in Education Conference (FIE)*, pages 1–3.

Meißner, N., Speth, S., and Becker, S. (2024). Automated programming exercise generation in the era of large language models. In *36th International Conference on Software Engineering Education and Training (CSEET)*, pages 1–5, Würzburg, Germany.

Odilinye, L., Popowich, F., Zhang, E., Nesbit, J., and Winne, P. H. (2015). Aligning automatically generated questions to instructor goals and learner behaviour. In *IEEE 9th International Conference on Semantic Computing (ICS)*, pages 216–223.

Oshin, M. and Campos, N. (2025). *Learning LangChain: Building AI and LLM Applications with LangChain and LangGraph*. O'Reilly Media, Inc., Sebastopol, CA, USA. Copyright © 2025 Olumayowa "Mayo" Olufemi Oshin. All rights reserved. Printed in the United States of America.

Pimentel, M., Filippo, D., and Santoro, F. M. (2020). Design science research: fazendo pesquisas científicas rigorosas atreladas ao desenvolvimento de artefatos computacionais projetados para a educação. In Jaques, P. A., Pimentel, M., Siqueira, S., and Bittencourt, I., editors, *Metodologia de Pesquisa Científica em Informática na Educação: Concepção de Pesquisa*, volume 1 of *Série Metodologia de Pesquisa em Informática na Educação*. SBC, Porto Alegre.

Santana, A., Silva, F., Dantas, J., Souza, J., and Aranha, E. (2025a). Geração automática de questões de programação usando llm: Um relato de experiência. In *Anais do XXXIII Workshop sobre Educação em Computação*, pages 1415–1425, Porto Alegre, RS, Brasil. SBC.

Santana, A., Silva, F., Dantas, J., Souza, J., and Aranha, E. (2025b). Geração de questões de programação baseada em templates e ia generativa. In *Anais do Workshop de Informática na Escola, CBIE*, Porto Alegre, RS, Brasil. SBC. No prelo.

Shah, C. (2024). From prompt engineering to prompt science with human in the loop. In *Proceedings of ACM Conference on Human in the Loop*, Seattle, WA, USA. arXiv preprint arXiv:2401.04122, May 10, 2024.

Thalheimer, W. (2003). The learning benefits of questions. Technical report, Work Learning Research.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. (2023). React: Synergizing reasoning and acting in language models.