Programação sem Sobrecarga: Otimizando o Ensino através da Teoria da Carga Cognitiva

João Henrique Berssanette

Instituto Federal do Paraná (IFPR) – Campus Curitiba /
Centro de Referência Ponta Grossa – PR – Brasil

joao.berssanette@ifpr.edu.br

Abstract. *This short course presents an approach based on Cognitive Load Theory (CLT) to optimize the teaching and learning process of computer programming, an area often associated with high dropout rates. Structured in four moments that integrate theory and practice, it is intended for teachers who work in programming education at different educational levels. The proposal integrates theoretical aspects of CLT and its practical application in programming teaching, offering evidence-based guidelines and practical strategies for effective cognitive load management during the development of programming content.*

Resumo. *Este minicurso apresenta uma abordagem fundamentada na Teoria da Carga Cognitiva (TCC) para otimização do processo de ensino e aprendizagem de programação de computadores, área frequentemente associada a altas taxas de evasão. Estruturado em quatro momentos que integram teoria e prática, destina-se a docentes que atuam no ensino de programação em diferentes níveis educacionais. A proposta integra aspectos teóricos da TCC e sua aplicação prática no ensino de programação, oferecendo diretrizes baseadas em evidências e estratégias práticas para o gerenciamento efetivo da carga cognitiva durante o desenvolvimento dos conteúdos de programação.*

1. Tipo de Minicurso

Minicurso teórico-prático.

2. Justificativa

O processo de ensino e aprendizagem de programação de computadores constitui um desafio significativo no cenário da educação em computação, com taxas de insucesso que giram em torno de 30% em disciplinas introdutórias (Simon *et al.* 2019; Watson e Li 2014). Essa alta taxa de insucesso reflete o fato de que aprender a programar é um desafio notável, dada a complexidade cognitiva envolvida.

A programação demanda múltiplos domínios cognitivos, como resolução de problemas, abstração e raciocínio lógico. Aprendizes iniciantes frequentemente enfrentam dificuldades com estruturas condicionais (*if*, *else*), pois precisam lidar com sintaxe, lógica booleana e fluxo de execução ao mesmo tempo, gerando alta carga cognitiva.

Para mitigar essa sobrecarga cognitiva, é fundamental que os docentes utilizem abordagens que se mostrem efetivas na redução da carga cognitiva, como exemplos resolvidos e atividades guiadas. Contudo, muitos docentes dessa área têm como origem cursos de bacharelado e tecnologia (GIRAFFA e MORA, 2016), que não contemplam os conhecimentos pedagógicos relacionados a essas abordagens. Isso reforça a importância de capacitação em teorias de aprendizagem como a TCC, para que possam planejar e conduzir o ensino de forma mais eficaz.

Nesse sentido, a Teoria da Carga Cognitiva (TCC), de Sweller (1988), surge como uma solução pedagógica promissora. Ela contribui com a organização didática e metodológica do ensino, fundamentando-se na Psicologia Cognitiva para otimizar o processo de aprendizagem.

Para auxiliar na superação desses desafios mencionados, a Teoria da Carga Cognitiva (TCC), proposta por Sweller (1988), fundamenta-se na Psicologia Cognitiva para otimizar o processo de ensino e aprendizagem. A TCC defende que a aprendizagem é mais eficaz quando alinhada ao processo cognitivo humano (Sweller, Merriënboer, e Paas 2019). Este princípio é crucial em áreas com alta demanda cognitiva, como a programação, onde a complexidade das tarefas pode sobrecarregar a memória de trabalho.

No ensino de programação, a TCC é especialmente relevante devido à alta carga cognitiva intrínseca da área. Essa carga está associada à complexidade inerente às tarefas, como a escrita e depuração de código. A depuração, por exemplo, exige que o aprendiz compreenda simultaneamente o código, o comportamento esperado e as mensagens de erro, sobrecarregando a memória de trabalho.

Essa sobrecarga da memória de trabalho, causada pela alta carga intrínseca, precisa ser gerenciada. Diferentemente da carga extrínseca, que pode ser reduzida com um design instrucional adequado, a carga intrínseca exige estratégias que facilitem o processamento da informação.

Nesse contexto, intervenções pedagógicas bem estruturadas são fundamentais, podendo aumentar significativamente as taxas de aprovação (Vihavainen, Airaksinen, e Watson 2014). A TCC se apresenta como uma ferramenta valiosa para o desenvolvimento dessas intervenções, com foco no gerenciamento da carga cognitiva.Intervenções

Diante desse cenário, este minicurso propõe uma abordagem baseada na TCC, integrando teoria e prática para o gerenciamento da carga cognitiva no ensino de programação, visando oferecer aos participantes tanto a fundamentação teórica quanto estratégias práticas para aplicação imediata em sala de aula.

As contribuições esperadas são: (i) qualificar docentes em práticas pedagógicas mais efetivas para o ensino de programação; (ii) reduzir a retenção e evasão; e (iii) disseminar essas práticas na comunidade acadêmica, impactando positivamente a área.

3. Sumário Estendido

O minicurso está organizado em quatro momentos, alternando teoria e prática, com foco na integração efetiva dos princípios da Teoria da Carga Cognitiva (TCC) no ensino de programação. O primeiro momento apresenta uma visão geral do ensino de programação e seus desafios. Serão discutidos os objetivos das disciplinas introdutórias, com ênfase no desenvolvimento de soluções computacionais. Será abordado o perfil dos docentes da

área, destacando os desafios enfrentados por aqueles com formação técnica sem preparo pedagógico.

No segundo momento, serão explorados os fundamentos cognitivos da aprendizagem, com foco na arquitetura cognitiva humana. Serão realizadas demonstrações práticas das limitações da memória de trabalho, evidenciando suas restrições. Será apresentado o papel da memória de longo prazo na construção de esquemas mentais e na automatização, ressaltando sua importância para a expertise em programação.

O terceiro momento aprofunda a TCC e suas implicações para o ensino. A partir dos fundamentos da teoria e sua concepção de aprendizagem, serão analisadas as três formas de carga cognitiva (intrínseca, extrínseca e relevante). Serão explorados os princípios e efeitos da TCC, como o efeito de elemento interativo e exemplo resolvido, entre outros, demonstrando sua aplicação no ensino de programação.

A parte final é dedicada à aplicação prática da TCC no ensino de programação. Os participantes serão previamente preparados para o uso das plataformas BeeCrowd Academic e GitHub, com tutoriais e suporte técnico. A plataforma BeeCrowd Academic será utilizada para demonstrar a implementação de estratégias baseadas nos efeitos de exemplo resolvido e conclusão, com exemplos concretos de problemas de programação e suas soluções otimizadas. Os participantes também aprenderão a usar o GitHub Space-Codes para desenvolvimento e colaboração.

Os participantes desenvolverão materiais didáticos, considerando os princípios cognitivos estudados. A integração com o GitHub Space-Codes será demonstrada para organização e compartilhamento dos materiais. As atividades combinarão exposição dialogada nos momentos teóricos, com demonstrações práticas e exercícios práticos nos momentos de aplicação, garantindo o equilíbrio entre teoria e prática.

4. Público-Alvo

O minicurso é direcionado a docentes de programação de computadores em diversos níveis (técnico, superior e pós-graduação), especialmente aqueles que lecionam disciplinas introdutórias, estruturas de dados, algoritmos e áreas afins.

Muitos desses profissionais possuem formação superior, como bacharelado ou tecnólogo em áreas da computação, e podem não ter formação pedagógica específica. O minicurso, portanto, oferece uma oportunidade de desenvolvimento profissional, com foco em práticas pedagógicas baseadas em evidências científicas, visando auxiliar docentes que enfrentam altas taxas de retenção e evasão.

É necessário que os participantes tenham experiência prévia em ensino de programação e conhecimentos básicos de Python (linguagem das demonstrações práticas). Não é necessário conhecimento prévio sobre a Teoria da Carga Cognitiva. Para garantir a qualidade das interações e o aproveitamento das atividades, o número de participantes será limitado a 30.

O minicurso foi estruturado para atender a esse público, combinando fundamentação teórica com estratégias práticas e ferramentas aplicáveis em sala de aula, para otimizar o ensino e aprendizagem de programação.

5. Biografia do Autor

João Henrique Berssanette é Professor de Ensino Básico, Técnico e Tecnológico no Instituto Federal do Paraná (IFPR), atuando nos campi de Telêmaco Borba e Curitiba, e no Centro de Referência de Ponta Grossa. Sua atuação docente abrange diferentes níveis de ensino, da formação técnica à pós-graduação.

Leciona disciplinas de Programação de Computadores, bem como disciplinas relacionadas aos processos de Ensino e Aprendizagem. É Doutor em Ensino de Ciência e Tecnologia (UTFPR, 2021) e Mestre pela mesma instituição (2016). Possui especializações em Docência da Educação Profissional Técnica e Tecnológica e em Gestão de Pessoas, além de graduação em Tecnologia em Processamento de Dados.

Currículo Lattes: http://lattes.cnpq.br/4957636385989608

ORCID: 0000-0002-7622-3003

6. Organização do Minicurso

Para garantir o máximo aproveitamento deste minicurso, os participantes terão acesso prévio a um ambiente virtual de aprendizagem (AVA) exclusivo. Disponível em profeberssa.com (senha: EduComp2025), o AVA contém materiais preparatórios, incluindo tutoriais passo a passo para criação de contas e configuração do ambiente nas plataformas BeeCrowd Academic e GitHub, que serão utilizadas na parte prática. Além disso, o AVA conterá materiais de apoio, leituras complementares e um espaço para dúvidas e discussão.

O minicurso, em si, terá duração de três horas, divididas em quatro momentos que integram teoria e prática. O primeiro momento (30 minutos) contextualiza o cenário atual do ensino de programação. São explorados os desafios das disciplinas introdutórias, o perfil dos docentes e as implicações pedagógicas desse contexto.

O segundo momento (30 minutos) aprofunda os fundamentos cognitivos essenciais à aprendizagem. Inclui demonstrações práticas sobre as limitações da memória de trabalho, a construção de esquemas mentais e a automatização cognitiva.

O terceiro momento (60 minutos) é dedicado à Teoria da Carga Cognitiva. São abordados seus princípios, os tipos de carga cognitiva e seus efeitos no ensino de programação (ex: efeito de elemento interativo, exemplo resolvido).

Os 60 minutos finais são dedicados à aplicação prática. Os participantes utilizarão BeeCrowd Academic e GitHub Space-Codes para desenvolver e compartilhar materiais didáticos otimizados, com base em exemplos concretos de problemas de programação e suas soluções.

As atividades combinarão exposição dialogada nos momentos teóricos, com demonstrações práticas e exercícios práticos nos momentos de aplicação, garantindo o equilíbrio entre teoria e prática.

7. Idioma

O minicurso será ministrado em língua portuguesa.

8. Infraestrutura e Materiais Necessários

A participação no minicurso requer computador com conexão estável à internet e navegador atualizado.

Conforme detalhado na seção "Organização do Minicurso", os participantes deverão criar contas nas plataformas GitHub (utilizando o GitHub Space Codes) e BeeCrowd Academic antes do início das atividades. Tutoriais para a criação dessas contas estão disponíveis no ambiente virtual de aprendizagem (AVA).

Referências

GIRAFFA, Lucia Maria Martins, e Michael da Costa MORA. (2016). "Evasão na disciplina de algoritmo e programação: um estudo a partir dos fatores intervenientes na perspectiva do aluno". em *Congresos CLABES*.

Simon, Andrew Luxton-Reilly, Vangel V. Ajanovski, Eric Fouh, Christabel Gonsalvez, Juho Leinonen, Jack Parkinson, Matthew Poole, e Neena Thota. (2019). "Pass Rates in Introductory Programming and in other STEM Disciplines". P. 53–71 em *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education. ACM*.

Sweller, John. (1988). "Cognitive load during problem solving: Effects on learning". *Cognitive Science* 12(2):257–85. doi: 10.1016/0364-0213(88)90023-7.

Sweller, John, Jeroen J. G. van Merriënboer, e Fred Paas. (2019). "Cognitive Architecture and Instructional Design: 20 Years Later". *Educational Psychology Review* 31(2):261–92. doi: 10.1007/s10648-019-09465-5.

Vihavainen, Arto, Jonne Airaksinen, e Christopher Watson. (2014). "A systematic review of approaches for teaching introductory programming and their influence on success". P. 19–26 em *Proceedings of the tenth annual conference on International computing education research - ICER '14*. ACM Press.

Watson, Christopher, e Frederick W. B. Li. (2014). "Failure rates in introductory programming revisited". P. 39–44 em *Proceedings of the 2014 conference on Innovation & technology in computer science education - ITiCSE '14*. ACM Press.