Pensamento Computacional do Pré-Escolar ao Secundário: uma revisão do estado da área

Shuchi Grover[1] e Roy Pea[1,2]

Traduzido por Leonel Morgado
30-06-2019

O artigo influente de Jeannette Wing sobre pensamento computacional, há 6 anos, defendeu a necessidade de incluir esta nova competência na capacidade analítica de cada criança, como ingrediente vital à aprendizagem de ciências, tecnologia, engenharia e matemática (STEM: *science, technology, engineering, & mathematics*). O que é o pensamento computacional? Porque é que este artigo ecoou junto de tantas pessoas e serviu de grito de combate para educadores, investigadores de educação e decisores políticos? Como interpretaram a definição de Wing e que avanços houve desde que foi publicado o seu artigo? Neste artigo, enquadra-se o estado atual do discurso sobre pensamento computacional no ensino desde o nível pré-escolar ao nível secundário, examinando as obras académicas publicadas mais recentemente que se baseiam no artigo de Wing's. Identificam-se as lacunas de investigação e articulam-se as prioridades para esforços futuros.

Palavras-chave: pensamento computacional; ensino da informática; literacia computacional; computadores e aprendizagem; currículos do ensino pré-escola, básico e secundário; ambientes de aprendizagem; resolução de problemas; aprendizagem STEM; cognição dos alunos; tecnologia

Introdução

Há seis anos, o artigo sucinto e influente de Jeannette Wing, "Computational Thinking", surgiu na rubrica Viewpoint do fascículo de março de 2006 da revista *Communications of the ACM*, pronunciando: "Representa uma atitude e uma competência universalmente aplicável, que todos, não apenas informáticos ansiaria aprender e utilizar" (p. 33).

Os argumentos de Wing chamaram a atenção da uma ampla comunidade académica. Lançado pelo seu artigo e por uma crescente comunidade de investigadores, educadores e decisores políticos, o pensamento computacional, enquanto conceito com um programa de investigação associado, testemunhou cada vez mais atenção e investigação. Os ventos de popa do ambiente geral enfunaram este interesse crescente. A questão do ensino da informática desde o nível pré-escolar ao secundário assumiu a primazia quando um relatório veemente, intitulado *Andar sem combustível: o fracasso do ensino da informática do pré-escolar ao secundário na era digital* (Wilson, Sudol, Stephenson, & Stehlik, 2010), revelou os números abismalmente baixos de mulheres na informática e que mais de dois terços do país tenha escassas recomendações sobre informática ao nível do ensino secundário. A preocupação com estas estatísticas aprofundava as previsões da direção-geral norte-americana de estatísticas do trabalho (Bureau of Labor Statistics, http://www.bls.gov/ooh/) de que a informática seria um dos mercados laborais de mais rápido crescimento até 2018. Este imperativo da educação em informática encaixou diretamente com as políticas de ciência que se vinham focando desde o início do séc. XXI na aprendizagem das áreas das ciências, tecnologia, engenharia e matemática (STEM: *science, technology, engineering, & mathematics*). Encarando o pensamento computacional como algo no cerne de todas as disciplinas STEM (Henderson, Cortina, Hazzan, & Wing, 2007), parece que a informática do pré-escolar ao secundário é uma ideia cujo tempo propício chegou.

Evidentemente, a ideia do pensamento computacional não é nova. Já nos anos 1960 Alan Perlis tinha sustentado a necessidade de os alunos universitários de todas as áreas aprenderem a programar e aprenderem a "teoria da computação" (Guzdial, 2008). Contudo, no contexto do ensino não superior, a informática só começou a ter acolhimento generalizado a partir do trabalho de Seymour Papert no MIT, nos anos 1980. Papert foi o pioneiro da ideia de as crianças desenvolverem o pensamento procedimental através da programação em LOGO (Papert, 1980, 1991). Este recente ressurgimento traz ao tema uma perspetiva fresca, de "séc. XXI", constituindo o artigo de Wing em 2006 um ponto de partida lógico para a nossa análise crítica do estado atual da área da programação computacional no ensino não superior. As secções seguintes examinam as obras académicas mais salientes, publicadas recentemente, que usaram o artigo de Wing como base. Este artigo também relata os esforços principais sobre ensino da informática em nível não superiores.

Dada a ambiguidade das definições de pensamento computacional, que tem vindo a afetar este conceito, e dado que é imperativo para o ensino escolar, a próxima secção olha

[1]Universidade de Stanford, Escola de Educação, Stanford, CA, EUA
[2]Instituto H-STAR, Stanford, CA, EUA

Educational Researcher, vol. 42, n.º 1, pp. 38–43
DOI: 10.3102/0013189X12463051
© 2013 AERA. http://er.aera.net

profundamente para as várias perspetivas e para a evolução das definições de pensamento computacional, para os argumentos de desenvolver o pensamento computacional das crianças em idade escolar e as críticas mais frequentes contra o pensamento computacional nas escolas. Este artigo faz então um levantamento das investigações recentes sobre pensamento computacional (incluindo algumas que não usam a expressão *pensamento computacional* propriamente dita, mas contudo debruçam-se sobre as competências computacionais em crianças), dos vários ambientes e ferramentas que se crê apoiarem o desenvolvimento do pensamento computacional, e avaliam-se estudos que tentam medir o pensamento computacional. Finalmente, este artigo expõe as prioridades para ampliar o discurso sobre o pensamento computacional do ensino pré-escolar ao secundários, com base nas lacunas existentes na investigação atual.

O quê e o porquê do pensamento computacional

Segundo Wing (2006), "o pensamento computacional lida com a resolução de problemas, a conceção de sistemas e a compreensão do comportamento humano, apoiando-se nos conceitos fundamentais da informática" (p. 33). A essência do pensamento computacional é pensar como um informático, um cientista computacional, quando confrontado com um problema.

O apelo à ação de Wing em prol do pensamento computacional na educação escolar serviu como ponto de partida para duas oficinas da Academia Nacional de Ciências norte-americana, reunindo investigadores de craveira das áreas da educação, ciências da aprendizagem e ciências da computação/informática, e líderes empresariais do setor da informática, para explorar "a natureza do pensamento computacional e as suas implicações cognitivas e educativas" (National Research Council [NRC], 2010, p. viii) e os aspetos pedagógicos do pensamento computacional (NRC, 2011). Na primeira oficina, as noções iniciais sobre pensamento computacional, que se centravam no pensamento procedimental e na programação (Papert, 1980, 1991), embora ainda consideradas válidas, foram revisitadas e ampliadas, para abrangerem vários conceitos nucleares da informática, que a levem além da "mera programação". A oficina, contudo, deixou bem vincada a falta de consenso que parece ter infernizado esta área. Algumas das questões centrais que ficaram por responder na oficina foram: como se pode reconhecer o pensamento computacional? qual é a melhor pedagogia para promover o pensamento computacional nas crianças? é possível separar, legitimamente, a programação, os computadores e o pensamento computacional? (NRC, 2010). Algumas destas questões foram reexaminadas na oficina seguinte, que se focou na melhoria da definição da área, reunindo e sintetizando inspirações de educadores que abordaram o pensamento computacional no seu trabalho com professores e alunos do ensino não superior. O objetivo da oficina foi a partilha de exemplos e boas práticas de pedagogias e ambientes para o ensino do pensamento computacional, tendo revelado uma multiplicidade de perspetivas, refletindo as várias ferramentas e pedagogias que são candidatas legítimas a serem usadas no desenvolvimento destas competências.

Wing (2011) revisitou o tema e clarificou-o, da seguinte forma: "o pensamento computacional são os processos de pensamento envolvidos na formulação de problemas e na sua resolução, para que as soluções possam ser representadas sob uma forma que possa ser realizada eficazmente por um agente processador de informação." Aho (2012) simplificou esta formulação, definindo o pensamento computacional como os processos de pensamento envolvidos na formulação de problemas para que "as suas soluções possam ser representadas como passos computacionais e algoritmos" (p. 832).

Recentemente, a Sociedade Real britânica (Royal Society, 2012) avançou com uma definição sucinta e manejável, que captura a essência do pensamento computacional: "o pensamento computacional é o processo de reconhecimento de aspetos de computação no mundo que nos rodeia, e a aplicação de ferramentas e técnicas das ciências da computação para compreender e raciocinar sobre sistemas e processos naturais e artificiais" (p. 29).

Uma perspetiva valiosa que decompõe o significado do pensamento computacional, especialmente para os currículos do terceiro ciclo do ensino básico ou do ensino secundário, surge na disciplina Princípios das Ciências da Computação, em fase-piloto por parte do College Board e da National Science Foundation (NSF) (http://www.csprinciples.org/). Esta disciplina foca-se nas *práticas* do pensamento computacional e baseia-se em nas sete "grades ideias" da informática/computação:

1. A computação é uma atividade humana criativa
2. A abstração reduz a informação e o detalhe, para se focar em conceitos relevantes à compreensão e resolução de problemas
3. Os dados e a informação facultam a criação de conhecimento
4. Os algoritmos são ferramentas para desenvolver e expressar soluções para problemas computacionais
5. A programação é um processo criativo que produz artefactos computacionais
6. Os dispositivos e sistemas digitais, e as redes que os interligam, permitem e alimentam abordagens computacionais à resolução de problemas
7. A computação permite a inovação noutras áreas, como as ciências naturais e exatas, as ciências sociais, as humanidades, as artes, a medicina, a engenharia e a gestão.

Na sequência de oficinas organizadas pela Computer Science Teachers Association (CSTA) e pela International Society for Technology in Education (ISTE), Barr e Stephenson (2011) avançaram de forma semelhante com uma "definição operacional do pensamento computacional" focada nos professores do ensino não superior, que incluía uma lista de verificação, explicativa, dos significados do pensamento computacional ao longo de uma enumeração de conceitos e capacidades centrais do pensamento computacional, acompanhada de exemplos de como poderiam ser integrados em atividades de várias disciplinas.

Vale a pena fazer notar que a potente ideia de "literacia computacional" (diSessa, 2000) é anterior à proposta de Wing de pensamento computacional para todos. Embora a essência, de ambos os conceitos, tenha por alvo esta nova competência da era digital, diSessa separa as ferramentas "materiais" (como os ambientes de programação) dos aspetos "cognitivos" e "sociais" da literacia computacional. Além disso, diSessa sublinha o uso da "computação como um meio" para explorar outros domínios como a matemática e a ciência, à semelhança de Kay e Goldberg (1977) na sua exploração da matemática, da ciência e da arte através de

programação em Smalltalk. Esta noção é frequentemente negligenciada nas definições populares do pensamento computacional. O termo *literacia computacional* pode, talvez, ser suscetível de ser confundido com anteriores, como *literacia informática, literacia informativa* ou *literacia digital*, que foram assumindo vários significados ao longo dos anos e ficam bem aquém do que diSessa exige da literacia computacional. Embora a expressão e a noção *pensamento computacional* agora pareça ser a preferida, relativamente a *literacia computacional*, nas práticas e na investigação atuais os dois termos são usados, com frequência, como sinónimos.

A literacia procedimental é outro avatar do pensamento computacional que foi proposto originalmente em 1980 por B. A. Sheil nos laboratórios Xerox PARC. Nas nossas leituras, pouco há que distinga a literacia procedimental do pensamento computacional, quando se aplica este principalmente à criação de videojogos e outros artefactos multimédia ou, de forma mais ampla, da prática de pensamento computacional no contexto das artes dos novos média e da sua conceção.

Os investigadores e educadores de informática, na sua maioria, trabalham agora genericamente com as supra mencionadas descrições recentes do pensamento computacional. O valor da *abstração* como pedra de toque do pensamento computacional (diferenciando-o de outros tipos de pensamento) não é contestado. Abstrair é "definir padrões, generalizar a partir de casos específicos" e uma chave para lidar com a complexidade (Wing, 2011). Os elementos que se seguem são atualmente amplamente aceites como constituindo o pensamento computacional, pelo que forma a base dos currículos que visam apoiar a sua aprendizagem e avaliar o seu desenvolvimento:

- Abstrações e generalização de padrões (incluindo modelos e simulações)
- Processamento sistemático de informações
- Sistemas simbólicos e representações
- Noções algorítmicas de fluxo de controlo
- Decomposição estruturada de problemas (modularizar)
- Pensamento iterativo, recursivo e paralelo
- Lógica condicional
- Restrições ao desempenho e eficiência
- Depuração e deteção sistemática de erros

A programação não é apenas uma competência fundamental da informática e uma ferramenta central para apoiar as tarefas cognitivas envolvidas no pensamento computacional, mas também uma demonstração de competências computacionais. Esforços notáveis como o CS Unplugged (http://csunplugged .org/), que introduzem conceitos computacionais sem usar computadores, embora estejam a proporcionar atividades introdutórias valiosas que expõem as crianças à natureza da informática, podem estar a manter os alunos arredados das experiências computacionais cruciais ligadas à prática habitual do pensamento computacional.

Finalmente, embora haja um reconhecimento geral que a computação perpassa todos os aspetos da economia global, o seu lugar como parte obrigatória do currículo escolar está longe de assegurado. Muitas críticas têm surgido em redor destas várias interpretações do pensamento computacional e falta de clareza entre os educadores sobre a informática enquanto disciplina. Outra preocupação válida é se há argumentos suficientemente fortes para que todas as crianças, incluindo as que alegarem não ter qualquer interesse em seguir carreiras de informática ou STEM, desenvolvam competências computacionais na escola. No mapa do currículo escolar, se soma zero, como devem os decisores políticos encontrar espaço num currículo escolar já sobrecarregado? Também há falta de concordância sobre se o pensamento computacional deve acabar por ser integrado na educação como um tema geral, um tema dentro de uma disciplina ou um tema multidisciplinar (NRC, 2011). Por fim, questiona-se se o pensamento computacional é suficientemente distinto de outras formas de pensamento que as crianças estão a desenvolver. Os proponentes do pensamento computacional reconhecem que embora partilhe elementos com as práticas de pensamento matemáticas, de engenharia e a até do *design* e conceção, e acolha um legado rico de enquadramentos similares, também expande cada uma dessas competências cognitivas de forma única (Lee et al., 2011). Denning e Freeman (2009) notam que embora o paradigma da computação "contenha ecos da engenharia, das ciências e da matemática, é claramente diferente por causa do seu foco central no processos da informação" (p. 30) e que a interpretação de Wing sobre o pensamento computacional se enquadra bem neste sistema de práticas.

Alegamos que a abordagem à resolução de problemas, que geralmente se descreve como pensamento computacional, é uma omissão reconhecível e crucial das capacidades que se espera que as crianças desenvolvam durante a sua formação habitual não superior em ciências e matemática (embora o pensamento computacional tenha sido mencionaldo por fim, embora brevemente, no enquadramento da educação em ciências para os níveis pré-escolar ao secundário 2012 NRC). Se a literacia básica em matemática e em ciências pode ser considerada essencial para que todas as crianças compreendam como funciona o nosso mundo, porque deveria a educação escolar evitar levantar o capô dos dispositivos computacionais que são agora omnipresentes? Acreditamos que quem possuir competência computacionais estará melhor posicionado para tirar partido de um mundo com computação ubíqua. As primeiras experiências com este modo de resolução de problemas, além de aliviar os problemas que são relatados no ensino superior nas disciplinas introdutórias de informática, irão gerar interesse e preparar os alunos para serem bem-sucedidos neste campo em expansão, pleno de oportunidades.

As notícias recentes nos meios de comunicação e no setor empresarial sugerem que o movimento para tornar a programação uma competência mais comum, para todos, e introduzir os 'ritmos (abreviatura de algoritmos) como quarto "r" da literacia do século XXI está a adquirir um ímpeto global. Israel há muito que faz alarde de ter um currículo escolar exemplar com informática obrigatória. Países como a Rússia, a África do Sul, a Nova Zelândia e a Austrália já encontraram espaço para a informática no currículo dos níveis de ensino não superior. Mais recentemente, o Reino Unido iniciou programas-piloto para ensinar a computação a todas as crianças em idade escolar, na sequência de uma determinação política de 2012 da Sociedade Real britânica.

Resumo da investigação relevante sobre pensamento computacional no ensino não superior

Havendo definições genericamente aceites sobre o pensamento computacional no ensino não superior, o foco deslocou-se

recentemente para como abordar as questões mais práticas da promoção e acesso ao desenvolvimento do pensamento computacional. Há muitas obras das últimas três décadas que lidam com as questões do ensino e aprendizagem da programação e da informática. O cerne da investigação em ensino da informática, contudo, situa-se no contexto das salas de aula do ensino superior. Embora haja muito a aprender sobre o pensamento computacional no ensino não superior, quer dos estudos sobre programação com crianças nos anos 1980 (usando linguagens como LOGO e BASIC), quer das primeiras experiências de programação e informática com alunos universitários, as restrições de espaço impostas por este ensaio bem como um foco no ressurgimento recente do pensamento computacional fazem com que esta revisão se limite à *investigação recente envolvendo ferramentas do século XXI e crianças em idade escolar.*

Ambientes e ferramentas que apoiam o pensamento computacional

A ideia de "chão baixo, teto alto" como um dos princípios diretores da criação de ambientes de programação para crianças tem estado presente desde o tempo do LOGO. Essencialmente, significa que embora deva ser fácil para um principiante atravessar o limitar da criação de programas que funcionam (chão baixo), a ferramenta também deve ser suficientemente poderosa e expansível para satisfazer as necessidades de programadores avançados (teto alto). Os ambientes computacionalmente ricos e as ferramentas eficazes de pensamento computacional para crianças em idade escolar têm de ter um limitar baixo e um teto alto, andaimes, permitir a transferência de conceitos, apoiar a equidade e serem sistémicas e sustentáveis (Repenning, Webb, & Ioannidou, 2010). Várias ferramentas de programação cumprem estes critérios, até certo ponto. São populares entre elas os ambientes de programação gráfica, como o Scratch, o Alice, o Game Maker, o Kodu, e o Greenfoot; as ferramentas de criação de simulações baseadas na Web como o Agentsheets e o Agentcubes; e os conjuntos robóticos e os meios tangíveis, como as placas Arduino e Gogo. Os ambientes de programação gráfica são relativamente fácies de usar e permitem que as primeiras experiências se centrem na conceção e na criação, evitando os problemas da sintaxe da programação. Ao permitir a iniciantes criar programas por encaixe de blocos gráficos que controlam as ações de vários atores dinâmicas num ecrã, os ambientes como o Scratch, a popular ferramenta do MIT, permitem programar facilmente – literalmente, de "estalo" (de peças).

Várias destas experiências computacionais introdutórias usam a progressão trietápica "usar-modificar-criar" para ajudar o aluno a passar de utilizador a modificador a criador de artefactos computacionais (Lee et al., 2011), uma progressão que foi pela primeira vez usada de forma ampla na aplicação Hypercard da Apple, entre meados dos anos 1980 e o início dos anos 1990. Atividades curriculares como a conceção de jogos e a robótica costumam servir bem como meio de exploração iterativa do pensamento computacional, tornando-as ideais não apenas para motivar e envolver as crianças em idade escolar, mas também para as introduzir à informática. As experiências visuais e tangíveis de programação costumam dar lugar à exposição a linguagens de programação de alto nível como o Python, o Java ou o Scheme.

As recomendações para envolver as raparigas através da computação em contexto (Margolis & Fisher, 2002; vd. também Cooper & Cunningham, 2010) proporcionam uma lógica cativante para ferramentas que almejam tapar a falha de género no campo da informática. Os ambientes computacionais emergentes estão prestes a proporcionar mais oportunidades para envolvimento do pensamento computacional em situações formais e informais, envolvendo simultaneamente as raparigas. Os têxteis eletrónicos e outros conjuntos de "artesanato computacional", que usam equipamentos pequenos e potentes, como o Lilypad Arduino, permitem que as crianças combinem as artes e ofícios tradicionais, ocmo a costura e o esboço a lápis com a computação e a eletrónica. O MIT App Inventor, um ambiente de programação visual que usa blocos gráficos de código como o Scratch e permite criar aplicações móveis para o sistema Android, é mais neutro em termos de género e mais completo do que a maior parte das ferramentas. Define um chão baixo para a produção criativa de aplicações móveis (algo que todos os adolescentes, incluindo as raparigas, anseiam fazer) mas ao mesmo tempo lidam com conceitos complexos de pensamento computacional incluindo abstração de procedimentos e de dados, pensamento iterativo e recursivo, decomposição estruturada de tarefas, pensamento condicional e lógico, e depuração.

Apesar da sua crescente popularidade para promover muitas competências do séx. XXI no ensino não superior (NRC, 2012), os videojogos como plataformas para examinar o pensamento computacional nas crianças têm sido subutilizados na investigação recente. Holbert e Wilensky (2011) desenvolveram e testaram com êxito um protótipo de videojogo, FormulaT, que visou servir como plataforma para aprender princípios da cinemática, bem como "estratégias computacionais sistemáticas". O FormulaT usou o NetLogo, um ambiente computacional para modelação baseada em agentes. As atividades de abstração de comportamentos pertinentes em agentes, de aplicação de regras e de avaliação dos resultados através de modelação e de simulação, são formas-chave de envolvimento no pensamento computacional. Blikstein (2010) demonstra o aproveitamento dos modelos computacionais do NetLogo para a aprendizagem de ciências em turmas do ensino secundário. Contudo, a modelação baseada em agentes continua a ser relativamente subutilizada na investigação do pensamento computacional.

Sem surpresa, as ferramentas computacionais atuais variam na sua eficácia em permitir o envolvimento com os vários componentes do pensamento computacional. Maloney, Peppler, Kafai, Resnick e Rusk (2008) relataram a demonstração de vários elementos do pensamento computacional, como a lógica condicional, o pensamento iterativo e paralelo, e a abstração de dados, em programas de Scratch criados por jovens citadinos em contextos de atividade pós-letivas. Contudo, o Scratch não tem forma de abstrair as funcionalidades sob a forma de funções e procedimentos, o que originou uma versão chamada Snap!, de Berkeley, que visa lidar com este problema. Talvez uma situação imperiosa da informática no ensino não superior alimente o desenvolvimento de novas ferramentas, criadas expressamente para apoiar o pensamento computacional nas crianças em idade escolar. Estas ferramentas não deverão apenas incorporar todas as características de ferramentas eficazes de pensamento computacional e promover o desenvolvimento de todas as competências já identificadas como integrando o pensamento computacional, mas também ser guiadas pela investigação recente sobre o entendimento do senso-comum humano sobre a computação e sobre como as crianças explicam as sua abordagens

à resolução de problemas (Pane, Ratanamahatana, & Myers, 2001; Simon, Chen, Lewandowski, McCartney, & Sanders, 2007).
Por fim, apesar da variedade de ambientes nos quais decorre a atual investigação sobre pensamento computacional, muitas áreas promissores permanecem inexploradas; os movimentos Fab Labs, Makerspaces, e DIY/Faça-Você-Mesmo, como as Maker Faire e os Instructables, que promovem a construção de artefactos computacionais tangíveis, eventos informais de "hacking" para jovens, bem como os ubíquos e potentes *smartphones*, tudo são possibilidades entusiasmantes.

Avaliação do pensamento computacional

Sem dar atenção à avaliação, o pensamento computacional não terá grande esperança de entrar com êxito em qualquer currículo do ensino não superior. Além disso, para ajuizar sobre a eficácia de qualquer currículo que incorpore o pensamento computacional, é necessário validar métricas que permitam aos educadores avaliar aquilo que a criança aprendeu.

As investigações mais recentes a abordar as questões da avaliação do pensamento computacional, como a de Werner, Denner, Campe, and Kawamoto's (2012), *Fairy Assessment* no Alice, empregaram artefactos de programação criados por alunos, ou artefactos de programação pré-concebidos, para avaliar a compreensão e emprego, por parte dos alunos, de abstração, lógica condicional, pensamento algorítmico e outros conceitos do pensamento computacional para resolução de problemas. Há muito que são consideradas apelativas, no meio educativo, as ideias de desconstrução, retroengenharia e depuração, para avaliar a compreensão das crianças em contextos informáticos. Fields, Searle, Kafai e Min (2012) avaliaram as competências de engenharia e de programação dos alunos, à medida que estes depuravam projetos defeituosos, previamente construídos, de têxteis eletrónicos. Han Koh, Basawapatna, Bennett e Repenning (2010) tentaram, com algum êxito, avaliar a questão espinhosa da transferência de conceitos, para responder a perguntas como "Agora que o aluno consegue programas os Invasores Espaciais, será que consegue programar uma simulação científica?"

Nas últimas duas décadas, a "conversa de académicos" foi alavancada para promover e avaliar a literacia científica e matemática. O desenvolvimento do uso pelos alunos do vocabulário e da linguagem da informática ao longo do processo de envolvimento com atividades computacionalmente ricas proporciona um instrumento adicional para medir o desenvolvimento do pensamento computacional (Grover, 2011).

Ensino da informática do pré-escolar ao secundário

Wilson e Guzdial (2010) sustentam que embora a urgência nacional norte-americana para fortalecer os domínios STEM no ensino não superior se tenha traduzido em milhares de milhões de dólares de financiamento, a investigação específica sobre o ensino da informática continua a estar subfinanciada. As iniciativas da NSF, como a CPATH, a BPC e mais recentemente a CE21 já fizeram muitos para dar força a projetos que visam trazer conceitos de pensamento computacional e informática para o nível secundário. Um impulso adicional para conduzir os alunos interessados, no segundo e terceiro ciclo do ensino básico, para as carreiras informáticas, encontra-se em iniciativas da DARPA, como a CS-STEM ou a FIRE da Universidade Carnegie Mellon University (*Fostering Innovation through Robotics Exploration*, apoiar a inovação através da exploração robótica).

Embora a investigação em curso no desenvolvimento do pensamento computacional apoie e influencie os currículos de informática no ensino não superior, preparar os professores para a educação em informática e assegurar a equidade de géneros continuam a ser desafios enormes. A iniciativa CS10K da NSF pretende acrescentar 10.000 novos professores de informática nas escolas do terceiro ciclo e secundárias norte-americanas até 2015. A aliança Georgia Computes! está na vanguarda de esforços de âmbito nacional norte-americano para a formação de professores, desenvolvimento de currículos de pensamento computacional e informática do pré-escolar ao secundário e também na motivação das alunas para a informática. O investigador Guzdial, do Instituto de Tecnologia da Geórgia (EUA), defende no seu blogue (http://computinged.wordpress.com/) que entre os desafios para cumprir as metas CS10K se incluem as respostas a questões como as seguintes: de que precisam os professores para se tornarem professores de informática bem-sucedidos? que tipo de pedagogia se adequa às vidas de professores em exercício no terceiro ciclo do ensino básico e secundário? o que é conhecimento sobre o conteúdo pedagógico em informática?

Em termos curriculares, além da disciplina Princípios das Ciências da Computação para a formação profissional em informática, o currículo Exploring CS (http://www.exploringcs.org) visa ser um programa de 1 ano propedêutico do ensino superior, destinado a alunos do secundário. Outras iniciativas que visam introduzir a informática nas escolas incluem a CS4HS (http://www.cs4hs.com/) e a Computing in the Core (http://www.computinginthecore .org/), ambas sendo colaborações entre a academia, organismos nacionais e organizações como a Microsoft e a Google. O currículo-modelo da CSTA para informática do pré-escolar ao secundário proporciona sugestões curriculares para ajudar a criar interesse, envolver e motivar os alunos para a informática. Além disso, o sítio Web Exploring Computational Thinking da Google (www.google.com/edu/computational-thinking) tem uma grande riqueza de ligações para recursos de pensamento computacional na web. A ACM também criou recentemente uma nova coluna, a EduBits, no seu periódico quinzenal ACM Inroads, que realça as principais atividades educativas no seio da ACM e suas organizações afiliadas.

Alargar o âmbito do discurso e as prioridades para investigação empírica

É assim bastante evidente que muito do trabalho recente sobre pensamento computacional se centrou principalmente na questão das definições, e nas ferramentas que apoiam o desenvolvimento do pensamento computacional. Alguns passos foram dados no âmbito da definição de currículos para alimentar as competências computacionais e a avaliação do seu desenvolvimento. Contudo, ainda permanecem grande lacunas que apelam a investigação empírica.

Sob um ponto de vista que foi ecoado por Alfred Aho, Wing disse que "é necessária uma aplicação da investigação em ciências da aprendizagem à conceção de currículos adequados às várias idades e ciclos, para que o pensamento computacional maximize o

seu impacte significado para os alunos do pré-escolar ao secundário" (NRC, 2011, p. 4). Salvo alguns estudos recentes como os de Fadjo, Lu e Black (2009) e de Berland e Lee (2011), poucos tiveram em conta a investigação contemporâneas das ciências da aprendizagem sobre aprendizagem sócio-cultural e situada, cognição distribuída e corporizada, bem como as análises de atividades, interação e discurso. Os aspetos cognitivos das crianças e dos iniciantes na aprendizagem do conceitos computacionais já foram extensamente estudados nos anos 1980, em questões como o desenvolvimento de competências cognitivas (Kurland, Pea, Clement, & Mawby, 1986); depuração (Pea, Soloway, & Spohrer, 1987); problemas de transferência de conceitos (Clements & Gullo, 1984; Pea & Kurland, 1984); uso de andaimes adequados para uma transferência bem-sucedida (Klahr & Carver, 1988), entre muitos outros. Esse corpo de saber deve ser recuperado para influenciar a investigação do séc. XXI sobre pensamento computacional.

Também subinvestigada é a ideia da computação como meio para ensinar outros temas (encaixando a introdução ao pensamento computacional no ensino não superior com a transferência de competências de resolução de problemas para outros domínios). Trabalhos anteriores nesta área incluem a demonstração do êxito das crianças na conceção de programa de LOGO para ensinar frações (Harel & Papert, 1990) e ciências (Kafai, Ching, & Marshall, 1997), bem como no uso de software de modelação em ciências (Metcalf, Krajcik, & Soloway, 2000).

Os estudos empíricos sobre pensamento computacional nas crianças em idade escolar poderiam apoiar-se na extensa investigação sobre os tipos de problema que os alunos iniciantes da programação no ensino superior enfrentam nas suas primeiras experiências, que vão além de problema de sintaxe. Existirão barreiras bem definidas ou alvos de dificuldade que ocorram no percurso para o desenvolvimento de alguns elementos do pensamento computacional nas crianças (por ex., recursão)? Se sim, quais são e como se pode lidar com eles?

Também está largamente por explorar o território das disposições (ou atitudes) e estereótipos sobre pensamento computacional e informática, e a relação destes com o desenvolvimento da identidade do aluno (Mercier, Barron, & O'Connor, 2006). Quão cruciais são na nossa luta para proporcionar tanto a raparigas como a rapazes experiências de aprendizagem que visam alimentar competências de pensamento computacional? Trabalhos recentes, incipientes, no levantamento de atitudes dos alunos face à informática são apenas um início no processo de obter maior compreensão sobre isto.

Claramente, muito falta fazer para ajudar a desenvolver uma compreensão mais lúcida, teórica e prática, das competências computacionais das crianças. Por exemplo, o que podemos esperar que as crianças saibam ou façam melhor, depois de participarem num currículo concebido para desenvolver o pensamento computacional? E como pode isso ser avaliado? Estas são talvez algumas das perguntas mais importantes que aguardam resposta, antes de se poder fazer qualquer tentativa séria para introduzir em grande escala nas escolas currículos para o desenvolvimento do pensamento computacional. É hora de abordar as lacunas e ampliar o discurso académico do séc. XXI sobre o pensamento computacional.

Agradecimento

Agradecemos graciosamente o financiamento deste trabalho pelo LIFE Center da National Science Foundation (NSF0835854).

REFERÊNCIAS

Aho, A. V. (2012). Computation and computational thinking. *Computer Journal*, *55*, 832–835.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads*, *2*, 48–54.

Berland, M., & Lee, V. (2011). Collaborative strategic board games as a site for distributed computational thinking. *International Journal of Game-Based Learning*, *1*(2), 65–81.

Blikstein, P. (2010). Connecting the science classroom and tangible interfaces: the bifocal modeling framework. In *Proceedings of the 9th International Conference of the Learning Sciences*, Chicago, IL, EUA, 128–130.

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognitions. *Journal of Educational Psychology*, *76*, 1051–1058.

Cooper, S., & Cunningham, S. (2010). Teaching computer science in context. *ACM Inroads*, *1*, 5–8.

Denning, P., & Freeman, P. (2009). Computing's paradigm. *Communications of the ACM*, *52*(12), 28–30.

diSessa, A. A. (2000). *Changing minds: Computers, learning, and literacy*. Cambridge, MA, EUA: MIT Press.

Fadjo, C. L., Lu, M., & Black, J. B. (2009, June). *Instructional embodiment and video game programming in an after school program*. Artigo apresentado na World Conference on Educational Multimedia, Hypermedia & Telecommunications, Chesapeake, VA, EUA.

Fields, D. A., Searle, K. A., Kafai, Y. B., & Min, H. S. (2012). Debuggems to assess student learning in e-textiles. In *Proceedings of the 43rd SIGCSE Technical Symposium on Computer Science Education*. New York, NY: ACM Press.

Grover, S. (2011, April). *Robotics and engineering for middle and high school students to develop computational thinking*. Artigo apresentado no encontro anual da American Educational Research Association, Nova Orleães, LA, EUA.

Guzdial, M. (2008). Paving the way for computational thinking. *Communications of the ACM*, *51*(8), 25–27.

Han Koh, K., Basawapatna, A., Bennett V., & Repenning, A. (2010). Towards the automatic recognition of computational thinking for adaptive visual language learning. In *Proceedings of the 2010 Conference on Visual Languages and Human Centric Computing (VL/HCC 2010)* (pp. 59–66). Madrid, Espanha: IEEE Computer.

Harel, I., & Papert, S. (1990). Software design as a learning environment. *Interactive Learning Environments*, *1*, 1–32.

Henderson, P. B., Cortina, T. J., Hazzan, O., and Wing, J. M. (2007) Computational thinking. In *Proceedings of the 38th ACM SIGCSE Technical Symposium on Computer Science Education (SIGCSE '07)*, 195–196. Nova Iorque, NY, EUA: ACM Press.

Holbert, N. R., & Wilensky, U. (2011, April). *Racing games for exploring kinematics: a computational thinking approach*. Artigo apresentado no encontro anual da American Educational Research Association, Nova Orleães, LA, EUA.

Kafai, Y. B., Ching, C. C., & Marshall, S. (1997). Children as designers of educational multimedia software. *Computers & Education*, *29*, 117–126.

Kay, A., & Goldberg, A. (1977). Personal dynamic media. *IEEE Computer*, *10*, 31–41.

Klahr, D., & Carver, S. M. (1988). Cognitive objectives in a LOGO debugging curriculum: Instruction, learning, and transfer. *Cognitive Psychology*, *20*, 362–404.

Kurland, D. M., Pea, R. D., Clement, C., & Mawby, R. (1986). A study of the development of programming ability and thinking skills in high school students. *Journal of Educational Computing Research*, *2*, 429– 458.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J.,Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads*, *2*, 32–37.

Maloney, J., Peppler, K., Kafai, Y. B., Resnick, M., & Rusk, N. (2008). Programming by choice: Urban youth learning programming with Scratch. In *Proceedings of SIGCSE '08*. New York, NY: ACM Press.

Margolis, J., & Fisher, A. (2002). *Unlocking the clubhouse: Women in computing*. Cambridge: MIT Press.

Mercier, E. M., Barron, B., & O'Connor, K. M. (2006). Images of self and others as computer users: The role of gender and experience. *Journal of Computer Assisted Learning*, *22*, 335–348.

Metcalf, J. S., Krajcik, J., & Soloway, E. (2000). Model-It: A design retrospective. In M. J. Jacobson & R. B. Kozma (Eds.), *Innovations in science and mathematics education* (pp. 77–115). Mahwah, NJ, EUA: Lawrence Erlbaum.

National Research Council. (2010). *Committee for the Workshops on Computational Thinking: Report of a workshop on the scope and nature of computational thinking*. Washington, DC, EUA: National Academies Press.

National Research Council. (2011). *Committee for the Workshops on Computational Thinking: Report of a workshop of pedagogical aspects of computational thinking*. Washington, DC, EUA: National Academies Press.

National Research Council. (2012). *A framework for K–12 science education: Practices, crosscutting concepts, and core ideas*. Washington, DC, EUA: National Academies Press.

Pane, J. F., Ratanamahatana, C. A., & Myers, B. A. (2001). Studying the language and structure in non-programmers' solutions to programming problems. *International Journal of Human-Computer Studies*, *54*, 237–264.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Nova Iorque, NY, EUA: Basic Books.

Papert, S. (1991). Situating constructionism. In I. Harel & S. Papert (Eds.), *Constructionism*. (pp. 1–11). Norwood, NJ, EUA: Ablex.

Pea, R. D., & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology*, *2*, 137–168.

Pea, R. D., Soloway, E., & Spohrer, J. C. (1987). The buggy path to the development of programming expertise. *Focus on Learning Problems in Mathematics*, *9*, 5–30.

Repenning, A., Webb, D., & Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education (SIGCSE '10)*, 265–269. Nova Iorque, NY, EUA: ACM Press.

Royal Society. (2012). Shut down or restart: The way forward for computing in UK schools. Obtido em http://royalsociety.org/education/policy/computing-in-schools/report/

Simon, B., Chen, T., Lewandowski, G., McCartney, R., & Sanders, K. (2007, March). *Commonsense computing: What students know before we teach (Episode 1: Sorting)*. Artigo apresentado na Second International Workshop on Computing Education Research, Canterbury, Reino Unido.

Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012). The Fairy performance assessment: Measuring computational thinking in middle school. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education (SIGCSE '12)*, 215-220. Nova Iorque, NY, EUA: ACM.

Wilson, C., & Guzdial, M. (2010). How to make progress in computing education. *Communications of the ACM*, *53*(5), 35–37.

Wilson, C., Sudol, L. A., Stephenson, C., & Stehlik, M. (2010). *Running on empty: The failure to teach K-12 computer science in the digital age*. Nova Iorque, NY, EUA: The Association for Computing Machinery and the Computer Science Teachers Association.

Wing, J. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–36.

Wing, J. (2011). Research notebook: Computational thinking—What and why? *The Link Magazine*, Spring. Universidade Carnegie Mellon, Pittsburgh, EUA. Obtido em http://link.cs.cmu.edu/article.php?a=600

AUTORES

SHUCHI GROVER é doutorando na Escola de Educação da Universidade de Stanford, 485 Lasuen Mall, Stanford, CA 94305-3096, EUA; *shuchig@stanford.edu*. A sua investigação centra-se no apoio às crianças para obterem literacia computacional (estudar os processos sociais, culturais e cognitivos que ajudam a desenvolver competências computacionais) e na ferramentas e ambientes que contribuem para o seu desenvolvimento.

ROY PEA é o Professor "David Jacks" de ciências da educação e da aprendizagem, na Universidade de Stanford, Escola de Educação, e de Escola de Ciências da Computação (convidado), e Diretor do Instituto H-STAR, Wallenberg Hall, 450 Serra Mall, Bldg. 160, Stanford, CA 94305, EUA; *roypea@stanford.edu*. O seu trabalho nas ciências da aprendizagem centra-se no avanço das tecnologias, descobertas, ferramentas e práticas para a aprendizagem apoiada por tecnologia nos domínios complexos.

Manuscrito recebido a 14 de abril de 2012
Revisões recebidas a 13 de junho de 2012 e a 19 de julho de 2012
Aceite a 6 de setembro de 2012