Revista Brasileira de Informática na Educação, Volume 24, Número 1, 2016

Problemas e Dificuldades no Ensino e na Aprendizagem de Programação: Um Mapeamento Sistemático

Problems and Weaknesses in the Teaching and Learning of Programming: A Mapping Review

Draylson Micael Souza
Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de Sâo Paulo (USP)
draylson@icmc.usp.br

Marisa Helena da Silva Batista
Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de Sâo Paulo (USP)
mahh.isa@gmail.com

Ellen Francine Barbosa
Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de Sâo Paulo (USP)
francine@icmc.usp.br

Resumo *O ensino e aprendizagem de programação é uma tarefa complexa. Diversas pesquisas apontam problemas que vão desde a dificuldade dos alunos em compreender os conceitos de programação até a falta de motivação dos mesmos em realizar adequadamente a atividade de programação. O objetivo deste artigo é apresentar os resultados de uma revisão da literatura conduzida para coletar e avaliar evidências em problemas e dificuldades no ensino e aprendizagem de programação. A ideia é fornecer uma síntese compreensível desses problemas e das soluções propostas para minimizá-los. Para atingir esse objetivo, foi realizado um mapeamento sistemático (MS), uma vez que ele é um método de pesquisa para identificar estudos primários, relacionados a uma área de pesquisa. Foram revisados 519 artigos, sendo que 70 trabalhos foram identificados como relevantes em relação aos objetivos do MS conduzido. Os resultados apontam para tendências e necessidades de pesquisa em ensino e aprendizagem de programação, bem como fornecem um catálogo de trabalhos que propõem soluções para os problemas enfrentados em cenários reais de ensino e aprendizagem. De modo geral, é possível perceber uma tendência de pesquisas em ferramentas de visualização para ajudar os alunos a entenderem os conceitos de programação.*

Palavras-Chave: *Ensino e Aprendizagem de Programação, Mapeamento Sistemático*

Abstract *The programming education is a complex task. Several studies point to problems ranging from the difficulty of the students in understanding the programming concepts to the lack of motivation of them in performing the programming activity. The objective of this paper is to present the results of a literature review conducted to collect and evaluate evidence-problems and difficulties in teaching and learning of programming. The idea is to provide a comprehensive summary of these problems and the solutions proposed to minimize them. To achieve this goal, a systematic mapping (MS) was performed, since he is a research method that establishes procedures for identifying studies related to a research question. We reviewed 519 articles and identified 70 identified as relevant in relation to the objectives of MS. The results point to trends and research needs in teaching and learning of programming and provides educators with a catalog of works that propose solutions to the problems faced in real scenarios of teaching and learning. In general, mapping results suggest a trend of research in methods and visualization tools to help students understand the concepts of programming.*

Keywords: *Mapping Review, Teaching and Learning, Programming*

DOI: 10.5753/RBIE.2016.24.01.39

1 Introdução

O ensino e a aprendizagem de programação é considerado uma tarefa complexa e, como consequência, os cursos de programação frequentemente têm altas taxas de reprovação e desistência [56]. Em geral, os alunos têm dificuldades em entender determinados conceitos de programação, tais como ponteiros, recursão, declaração de variáveis, dentre outros [27, 48, 55]. No entanto, alguns autores ressaltam que muitos alunos entendem os conceitos de programação, mas têm dificuldades em aplicá-los durante a construção de programas [30, 34, 52]. Ainda, a atividade de programação não está limitada apenas à construção de programas, sendo necessário o desenvolvimento de habilidades de compreensão e análise de programas, as quais são pouco exploradas nas disciplinas de programação [39, 49, 68]. Por fim, outro aspecto bastante comentado é a motivação dos alunos, uma vez que muitos deles veem a programação como uma atividade cansativa e tediosa [18, 33, 62].

Embora alguns problemas sejam comumente conhecidos, é difícil ter uma visão geral sobre todos eles e identificar quais as soluções possíveis e mais adequadas para saná-los. Alguns trabalhos, como o de Soloway e Spohrer [63] e o de Robins et al. [58], procuram reunir as observações sobre vários autores com relação aos problemas e dificuldades no ensino e na aprendizagem de programação. No entanto, tais trabalhos não classificam e organizam as informações de forma a facilitar a identificação de tendências e necessidades na área, bem como não fornecem uma visão atual sobre os problemas e dificuldades no ensino e na aprendizagem de programação, uma vez que não abrangem as pesquisas realizadas nesta década.

Visando atender as questões mencionadas, um mapeamento sistemático (MS) foi conduzido a fim de identificar e classificar os estudos atuais existentes sobre as dificuldades e problemas no ensino e na aprendizagem de programação, bem como as soluções propostas para minimizá-los. Basicamente, um mapeamento sistemático provê uma visão geral de uma área de pesquisa para avaliar a quantidade de evidência existente com relação a um tópico de interesse [37]. Assim, este estudo pode ser útil para os pesquisadores identificarem as tendências e necessidades de pesquisa na área de ensino e de aprendizagem de programação. Ainda, por meio desse estudo, educadores em programação poderão beneficiar-se de um catálogo de trabalhos que propõe soluções para os problemas e dificuldades enfrentadas em cenários reais de ensino e de aprendizagem.

Este artigo apresenta a descrição do MS conduzido e a discussão dos seus resultados. A principal contribuição do trabalho é apresentar um panorama sobre o atual estado da arte em problemas e dificuldades no ensino e na aprendizagem de programação. A partir deste panorama, é possível elicitar diferentes linhas de pesquisa e estabelecer uma agenda de pesquisa para atender as atuais limitações da área.

O restante desse artigo está organizado da seguinte forma. Na Seção 2 são apresentadas as questões de pesquisa e os processos que foram realizados para selecionar e extrair os dados dos estudos. Na Seção 3 são apresentados e discutidos os resultados da análise dos dados extraídos. Por fim, na Seção 4 são apresentadas as conclusões desse trabalho e algumas perspectivas de trabalhos futuros.

2 Trabalhos Relacionados

Existem vários trabalhos que revisam a literatura a fim de identificar aspectos relacionados ao ensino e à aprendizagem de programação. Aureliano et al. [74], por exemplo, realizaram uma revisão sistemática, analisando o crescimento do interesse no processo de ensino e aprendizagem, as regiões do Brasil que mais realizam pesquisas nesse tema, em que nível de educação as pesquisas são realizadas e os tipos de pesquisa realizada.

Por outro lado, Robins et al. [58] realizam uma revisão e discussão da literatura focando nos aspectos psicológicos e educacionais, tais como as diferenças entre os programadores novatos e experientes, estratégias e conhecimentos em programação, dentre outros. Destaca-se também a revisão sistemática de Vihavainen et al. [75], que identifica intervenções no ensino de programação descritas na literatura e verificam se elas melhoram as taxas de aprovação em disciplinas de programação.

Em uma perspectiva diferente, este trabalho apresenta um mapeamento sistemático que visa identificar os principais problemas e dificuldades no ensino e na aprendizagem de programação, bem como as soluções que vêm sendo propostas a fim de amenizá-los.

3 Mapeamento Sistemático

O mapeamento sistemático (MS) conduzido está em conformidade com as diretrizes propostas por Kitchenham e Charters [37]. Um MS começa com a fase de planejamento, que inclui a formulação das questões de pesquisa e a definição dos critérios de inclusão e exclusão, seguido pela busca e revisão dos estudos.

3.1 Questões de Pesquisa

Uma abordagem comumente utilizada para formular as questões de pesquisa é a utilização dos critérios de

PICO. Usando o PICO as questões de pesquisa são estruturadas em quatro aspectos: população (***P****opulation*), intervenção (***I****ntervention*), comparação (***C****omparison*) e resultados (***O****utcomes*) [37]. A Tabela 1 mostra os atributos do PICO definidos para o mapeamento realizado. Para identificar o atual estado da arte em problemas e dificuldades no ensino e na aprendizagem de programação foram consideradas duas questões de pesquisa (QPs). As QPs e suas motivações são apresentadas na Tabela 2.

Tabela 1: Resumo do PICO

População	Estudos atuais sobre o ensino e a aprendizagem de programação
Intervenção	Aspectos do ensino e da aprendizagem de programação
Comparação	(não se aplica ao propósito deste estudo)
Resultados	Problemas e dificuldade atuais no ensino e na aprendizagem de programação

3.2 Processo de Busca

Os termos e sinônimos considerados na construção da *string* de busca foram obtidos por meio da identificação dos principais termos nas questões de pesquisa (PICO) e palavras-chave em um grupo de artigos conhecidos sobre o ensino e a aprendizagem de programação (Tabela 3).

Tabela 2: Artigos sobre o ensino e a aprendizagem de programação

Autor(es)	**Ano**	**Palavras-chave**
Panwong e Kemavuthanon [52]	2014	student, learning, programming, problem
Bain e Barnes [11]	2014	student, learning, programming, problem
Hidalgo-Céspedes [28]	2014	student, learning, programming, difficult
Piteira e Costa [55]	2013	student, novice, programming, difficult
Costa e Aparicio [19]	2014	student, learning, programming, difficult
Piteira e Costa [54]	2012	student, novice, learning, programming, difficult, problem

Ressalta-se que testes foram realizados considerando o grupo de controle e observou-se que somente o termo "*learning*" era suficiente para retornar os estudos sobre ambos o ensino e a aprendizagem de programação.

A Tabela 4 mostra a string de busca construída com base nos termos e sinônimos identificados.

Tabela 3: *String* de busca

***String* final**
(student OR novice) AND learning AND programming AND (difficulty OR problem)

Depois da definição dos termos de pesquisa, o processo de identificação da literatura relevante foi iniciado. A busca foi realizada considerando a base eletrônica SCOPUS, pois ela indexa as principais bases eletrônicas em Ciência da Computação. No entanto, o estudo é preliminar, e deverá ser atualizado considerando buscas em outras bases.

Além disso, os seguintes filtros foram considerados: (1) publicações entre 2010 e 2014, uma vez que os objetivos e QPs visam caracterizar os problemas mais atuais; (2) publicações da área de Ciência da Computação, eliminando os estudos provenientes de outras áreas.

Inicialmente, os potenciais estudos primários foram pesquisados na base. Como resultado, 519 estudos foram identificados. Excluindo os resultados indisponíveis e duplicados, foram obtidos 482 artigos.

3.3 Critérios de Inclusão e Exclusão

Para realizar a seleção dos estudos, foram definidos critérios de inclusão e exclusão. Assim, estudos foram incluídos no MS se eles satisfazem os seguintes critérios de inclusão (CI):

CI1: O artigo discute um problema ou dificuldade no ensino ou na aprendizagem de programação.

CI2: O artigo apresenta uma possível solução para amenizar um problema ou dificuldade no ensino ou na aprendizagem de programação.

Com relação ao critério de exclusão (CE), estudos foram excluídos se:

CE1: O artigo não menciona algum problema ou alguma solução relacionada ao ensino ou à aprendizagem de programação.

Tabela 4: Questões de Pesquisa

Questões de Pesquisa	**Principal motivação**
QP1: Quais problemas no ensino e na aprendizagem de programação vêm sendo observados atualmente?	Esta questão de pesquisa visa fornecer uma visão geral dos problemas e dificuldades mais atuais no ensino e na aprendizagem de programação.
QP2: Quais as soluções são atualmente propostas para amenizar os problemas e dificuldades no ensino e na aprendizagem de programação?	Esta questão de pesquisa complementa a anterior, fornecendo uma visão geral das soluções mais atuais propostas para amenizar os problemas e dificuldades no ensino e na aprendizagem de programação.

3.4 Seleção e Classificação dos Estudos

A atividade de seleção dos estudos foi conduzida em dois estágios. Durante o estágio 1, dois revisores realizaram a revisão dos meta-dados dos artigos. Foi realizado uma revisão minuciosa do título, resumo e

palavras-chave dos estudos, bem como outros meta-dados que viessem a ser úteis para a classificação dos estudos. Estudos identificados como relevantes, foram selecionados para leitura completa. Em reuniões, os revisores discutiam sobre possíveis discordâncias sobre a seleção ou não de um estudo. Procurou-se eliminar o máximo possível de estudos irrelevantes, tomando cuidado para que nenhum estudo relevante fosse descartado. Como resultado, um total de 112 estudos foram selecionados.

No estágio 2, foram obtidos os textos completos dos estudos primários selecionados no estágio 1. O texto completo de cada estudo primário selecionado foi lido pelos revisores, decidindo se este deveria ser incluído ou excluído. Os estudos primários incluídos na seleção final correspondem aos artigos relevantes que atendiam às QPs abordadas neste MS. Assim, foram identificados 70 estudos relevantes. Os artigos identificados como relevantes são apresentados nas tabelas 5 e 6.

Finalmente, a atividade de classificação foi conduzida em dois estágios. No primeiro estágio os artigos foram classificados considerando os problemas e dificuldades que eles abordam. Um conjunto de seis categorias foi utilizado para classificar os estudos:

- **Aprendizagem de Conceitos (P1):** Dificuldades relacionadas à aprendizagem de conceitos de programação, tais como recursão, ponteiros, estruturas de repetição, classes e objetos.
- **Aplicação de Conceitos de Programação (P2):** Dificuldades em utilizar os conceitos de programação aprendidos durante a construção de programas.
- **Compreensão de Programas (P3):** Dificuldades em ler e entender programas.
- **Fatoração e Refatoração de Programas (P4):** Dificuldades em dividir o programa em módulos, funções, classes, etc.
- **Motivação (P5):** Dificuldades relativas à falta de interesse e/ou desânimo dos alunos em realizar a atividade de programação.
- **Dificuldades Relacionadas ao Professores (P6):** Dificuldades dos professores em ensinar os conceitos de programação, desenvolver materiais e exercícios de apoio, bem como avaliar os trabalhos de programação.

No segundo estágio, os artigos foram classificados considerando as soluções propostas. Um conjunto de treze soluções foi utilizado para classificar os estudos:

- **Visualização (S1):** Utilização de animações para demonstrar as sequências de ações dos programas e algoritmos.
- ***Serious Games* (S2):** Utilização e desenvolvimento de jogos para o ensino e a aprendizagem de programação.
- **Ambientes de Desenvolvimento Pedagógico (S3):** Utilização e desenvolvimento de ambientes ambientes que provêem funcionalidades para construção e execução de programas, mas visando o ensino e a aprendizagem de programação..
- **Colaboração (S4):** Estratégias de ensino e de aprendizagem em que os alunos possam aprender uns com os outros.
- ***Scaffolding* (S5):** Estratégias de ensino em que o professor vai adaptando a tarefa de acordo com o nível de habilidade dos alunos, realimentando-a por contínuo *feedback* durante a progressão da tarefa.
- **Notações (S6):** Consiste em ensinar os conceitos de programação por meio de notações e linguagens mais familiares aos aprendizes, tais como as linguagens naturais e as notações musicais.
- **Reflexão (S7):** Estratégias de ensino que envolvem a aprendizagem por meio da reflexão de experiências anteriores.
- ***Feedback* (S8):** Envolve a utilização de *feedback* mais significativo aos alunos com respeito à qualidade dos seus programas.
- **Instrução Ancorada (S9):** Estratégia de ensino em que os alunos devem resolver problemas considerando um material instrucional previamente fornecido.
- **Interatividade (S10):** Estratégias que promovem uma maior interação entre os alunos e dos alunos com o professor.
- **Problemas Reais (S11):** Consiste em prover atividades em que os alunos irão construir programas que podem ser utilizados para alguma finalidade ao contrário dos *toy programs* que após as atividades constumam ser descartados.

Tabela 5: Artigos Incluídos

Id	Autor	Título	Ano
1	Abid et al. [1]	Using computer aided language software for teaching and self-learning	2011
2	Affandya et al. [2]	A study of tracing and writing performance of novice students in introductory programming	2011
3	Ajayi et al. [3]	Development and testing of a graphical FORTRAN learning tool for novice programmers	2010
4	Alberola e Garcia-Fornes [5]	Achieving individual feedback through the on-line educational platform	2012
5	Al-Fedaghi [4]	Conceptual framework for recursion in computer programming	2012
6	Ambrósio e Costa [7]	Evaluating the impact of PBL and tablet PCs in an algorithms and computer programming course	2010
7	Ambrósio et al. [6]	Identifying cognitive abilities to improve CS1 outcome	2011
8	Anderson et al. [8]	Introductory programming meets the real world: Using real problems and data in CS1	2014
9	Aris [9]	Object-oriented programming semantics representation utilizing agents	2011
10	Aris e Nazeer [10]	Object-oriented programming semantics education based on intelligent agents	2011
11	Bain e Barnes [11]	Why Is programming so hard to learn?	2014
12	Begosso et al. [12]	An approach for teaching algorithms and computer programming using Greenfoot and Python	2012
13	De Oliveira Brandao et al. [21]	A system to help teaching and learning algorithms	2012
14	Brito e De Sá-Soares [13]	Assessment frequency in introductory computer programming disciplines	2014
15	Cambranes [14]	Supporting novice programmers with natural language in the early stage of programming	2013
16	Camp e Woodward [15]	ICT enhanced learning experience for an introductory Object Oriented Programming course: A case study	2011
17	Chang et al. [16]	How to teach software programming? Using affective teaching method and social network to enhance the learning motivation in programming courses - An example on facebook	2012
18	Cisar et al. [17]	Teaching computer science in a web-based environment	2013
19	Corney et al. [18]	Engaging students in programming	2010
20	Costa e Aparicio [19]	Evaluating success of a programming learning tool	2014
21	Coull e Duncan [20]	Emergent requirements for supporting introductory programming	2011
22	Enström [22]	Dynamic programming - Structure, difficulties and teaching	2013
23	Gomes e Mendes [23]	A study on student performance in first year CS courses	2010
24	Gomes e Mendes [24]	Studies and proposals about initial programming learning	2010
25	Haatainen et al. [25]	A practice for providing additional support in CS1	2013
26	Hartanto e Reye [26]	Incorporating anchored learning in a C# Intelligent Tutoring System	2013
27	Helminen e Malmi [27]	Jype - A program visualization and programming exercise tool for python	2010
28	Hidalgo-Céspedes [28]	Allegories for learning abstract programming concepts	2014
29	Horváth [29]	Teaching one language in more depth is better than many languages superficially	2011
30	Hu et al. [30]	The design and implementation of learner models in online peer assessment to support learning	2012
31	Hu et al. [31]	Towards scaffolding problem-solving implementation process in undergraduate programming course	2013
32	Huang et al. [32]	Programming plush toys as an introduction to computer science: The (Fraug!htp) question of motivation	2011
33	Ibrahim e Jaafar [33]	Using educational games in learning introductory Programming: A pilot study on students' perceptions	2010
34	Jantan e Aljunid [34]	An experimental evaluation of scaffolded educational games design for programming	2012
35	Karapinar et al. [35]	Binary apple tree: A game approach to tree traversal algorithms	2012

Tabela 6: Artigos Incluídos (Continuação)

Id	Autor	Titulo	Ano
36	Kaucic e Asic [36]	Improving introductory programming with Scratch?	2011
37	Ko e Lee [38]	Using music notation for teaching computer programming	2013
38	Kollmansberger [39]	Helping students build a mental model of computation	2010
39	Liu et al. [40]	Anchor-based promgramming teaching embedded with Ch platform	2010
40	Liu et al. [41]	Research on teaching of college programming courses in network environment	2011
41	Lo et al. [42]	Learning beginning programming with cloud-based cloze programming practices	2013
42	Maleko et al. [43]	Understanding and analysing programmer interactions in a facebook programming group	2014
43	Malliarakis et al. [44]	Towards a new massive multiplayer online role playing game for introductory programming	2013
44	Martins et al. [45]	Student reflections as an influence in the dynamics of an introductory programming course	2011
45	Matthews et al. [46]	Merits and pitfalls of programming learning objects: A pilot study	2012
46	Matthíasdóttir e Geirsson [47]	The novice problem in computer science	2011
47	Meerbaum-Salant et al. [48]	Learning computer science concepts with Scratch	2010
48	Mei [49]	The application of trail-and-error learning in C language curriculum for non-majors in computer	2013
49	Moura e Van Hattum-Janssen [50]	Teaching a CS introductory course: An active approach	2011
50	Oliveira et al. [51]	Can natural language be utilized in the learning of programming fundamentals?	2013
51	Panwong e Kemavuthanon [52]	Problem-based learning framework for junior software developer: Empirical study for computer programming students	2014
52	Pears e Rogalli [53]	mJeliot: ICT support for interactive teaching of programming	2011
53	Piteira e Costa [55]	Computer Programming and Novice Programmers	2012
54	Piteira e Costa [54]	Learning computer programming: Study of difficulties in learning programming	2013
55	Piteira e Haddad [56]	Innovate in your program computer class: An approach based on a serious game	2011
56	Pyshkin [57]	Teaching programming: What we miss in academia - Or what would Mr. Platt say?	2011
57	Salgado e Castro [59]	An approach to support algorithms learning using virtual worlds	2012
58	Santos et al. [60]	A class record and reviewing system designed to promote programming learning	2011
59	Settle et al. [61]	Three Views on Motivation and Programming	2014
60	Shi e White [62]	Work in Progress: Learning to Program in a Connected World	2013
61	Stone e Clark [64]	The impact of Problem-Oriented Animated Learning Modules in a CS1-style course	2011
62	Sudol-DeLyser et al. [65]	Learning looping: From natural language to worked examples	2012
63	Tamada et al. [66]	A framework for programming process measurement and compiling error interpretation for novice programmers	2011
64	Tsukamoto et al. [67]	Change of students' motivation in an introductory programming course for non-computing majors	2012
65	Vrachnos e Jimoyiannis [68]	Design and evaluation of a web-based dynamic algorithm visualization environment for novices	2013
66	Wirth e McCuaig [69]	Making programs with the Raspberry Pi	2014
67	Xinogalos [71]	Programming techniques and environments in a technology management department	2012
68	Yadin [72]	Reducing the dropout rate in an introductory programming course	2011
69	Yang et al. [73]	Teaching method and practice about the course of program language design in college	2010
70	Wu [70]	Practice and experience in the application of problem-based learning in computer programming course	2010

- **Representações Semânticas (S12):** Estratégias que visam fornecer aos alunos representações dos códigos dos programas em uma lnguagem mais natural.

- **Outros (S13):** Soluções menos citadas. Incluem: apredizagem baseada em problemas, estimular os alunos a aprender desenvolvendo programas ou modificando programas existentes, aprendizagem por meio de um ambientes de rede, aprendizagem por tentativa e erro, avaliação em pares ou semanais, e-learning, sessões adicionais de ajuda e fornecimento de uma melhor visibilidade do progresso de avaliação.

Ressalta-se que algumas categorias, como Colaboração e Visualização, foram inicialmente definidas durante a elaboração dos formulários de extração. Outras, como Instrução Ancorada e Representações Semânticas, puderam ser incluídas durante a extração dos dados.

3.5 Extração dos Dados

Com o conjunto final de estudos primários definido, a atividade de extração dos dados foi realizada sobre os 70 artigos incluídos. A atividade de extração foi conduzida em dois estágios. Durante o primeiro estágio, os dados foram extraídos utilizando formulários de extração previamente elaborados. O conteúdo de tais formulários é sumarizado na Tabela 7. Depois disso, no segundo estágio, foram elaborados resumos e representações visuais dos dados, a fim de facilitar a análise dos dados.

Tabela 7: Sumário dos formulários de extração

Campo	Raciocínio
Metadados do artigo	Utilizado para gerenciar e localizar os artigos.
Problemas abordados	Relevante para enteder quais são os problemas e dificuldades no ensino de programação.
Categorias dos problemas	Utilizado para classificar os estudos de acordo com os problemas abordados.
Soluções propostas	Relevante para entender quais soluções estão sendo propostas para amenizar os problemas e dificuldades no ensino de programação.
Categorias das soluções	Utilizado para classificar os estudos de acordo com as soluções propostas.

4 Resultados e Discussões

Nesta seção, os resultados obtidos no MS são apresentados e discutidos.

4.1 Análise Geral

A Figura 1 mostra a porcentagem de artigos classificados em cada categoria de problemas. É possível notar uma grande predominância de problemas relativos à aprendizagem de conceitos de programação (P1), seguido de problemas relacionados à aplicação desses conceitos na construção de programas (P2) e de motivação (P5). Os artigos referentes a essas três categorias correspondem a mais de 80% do total de artigos selecionados. Desse modo, percebe-se que a maioria dos problemas está relacionada a essas três categorias, sendo importante aprofundar o estudo de técnicas para saná-las ou minimizá-las.

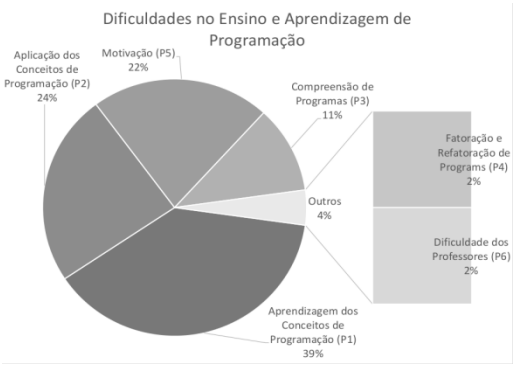

Figura 1: Porcentagem de Artigos Classificados em cada Categoria de Problemas

Também foi possível observar que os problemas motivacionais estão relacionados com as dificuldades na aprendizagem e aplicação dos conceitos de programação. Como apresentado na Figura 2, dos 28 trabalhos envolvendo os problemas motivacionais, 22 também envolvem os problemas na aprendizagem e aplicação dos conceitos de programação.

Também foi possível observar que os problemas motivacionais estão relacionados com as dificuldades na aprendizagem e aplicação dos conceitos de programação. Como apresentado na Figura 2, dos 28 trabalhos envolvendo os problemas motivacionais, 22 também envolvem os problemas na aprendizagem e aplicação dos conceitos de programação.

Ainda, nota-se uma relação entre essas duas categorias de problema, em que: (1) dos 49 trabalhos que envolvem as dificuldades na aprendizagem dos conceitos de programação, apenas 25 não relacionam este problema com as dificuldades em aplicar os conceitos aprendidos; e (2) dos 31 trabalhos que envolvem as dificuldades em aplicar os conceitos de programação, apenas 7 não relacionam este problema com as dificuldades na aprendizagem de conceitos de programação.

Já a Figura 3 mostra a porcentagem de artigos classificados em cada categoria de soluções. As categorias de soluções mais sugeridas pelos artigos em ordem decrescente foram: (1) Visualização (S1), com 21% das soluções propostas; (2) *Serious Games* (S2), com 15%; (3) Ambientes de Desenvolvimento Pedagógicos (S3), com 13%.

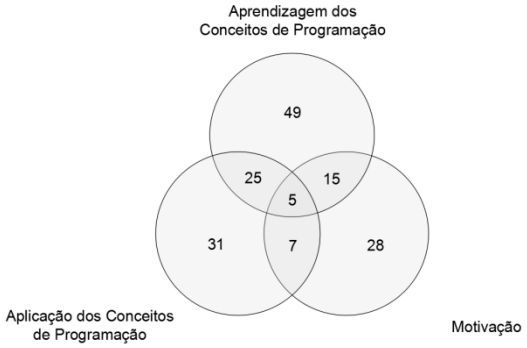

Figura 2: Intersecção entre as Categorias de Problemas

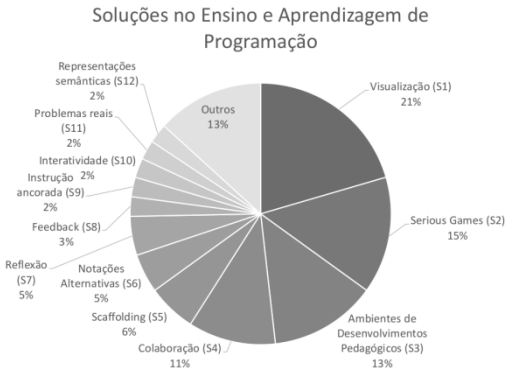

Figura 3: Porcentagem de Artigos Classificados em cada Categoria de Soluções

A utilização de técnicas de visualização foi a mais sugerida pelos autores, dentre eles Meerbaum-Salant et al. [48] enfatizam que ambientes visuais de programação facilitam o desenvolvimento de software em um contexto "divertido" e "não ameaçador" aos alunos. De fato, a utilização de ferramentas de apoio que usam interação e animação auxiliam os alunos a entender melhor os conteúdos ensinados, possibilitando cada um com diferentes níveis de potencial de aprendizagem a trabalhar em seu próprio ritmo, de forma individualizada [12].

Apesar da maior tendência de pesquisas visando à utilização de visualização, ambientes de programação pedagógicos e *serious games*, foi possível identificar várias outras propostas que poderiam ser melhor investigadas pelos pesquisadores. Por exemplo, os trabalhos de Cambranes [14] e Oliveira et al. [51] propõem a utilização de técnicas de representação como linguagem natural, que segundo Cambranes [14] demonstrou uma melhora no "ganho de aprendizagem" dos alunos. Para Oliveira et al. [51], sua utilização é considerada uma alternativa atraente e promissora, pois o aluno não precisa aprender uma gramática formal para aprender os fundamentos da programação, diminuindo assim o seu nível de esforço em escrever um programa.

Ao contrário do que ocorreu com as categorias de problemas, poucas foram as intersecções entre as categorias de soluções. Como pode ser observado na Figura 4, mesmo entre as três categorias mais citadas, poucos trabalhos propuseram soluções que envolvessem mais de uma categoria.

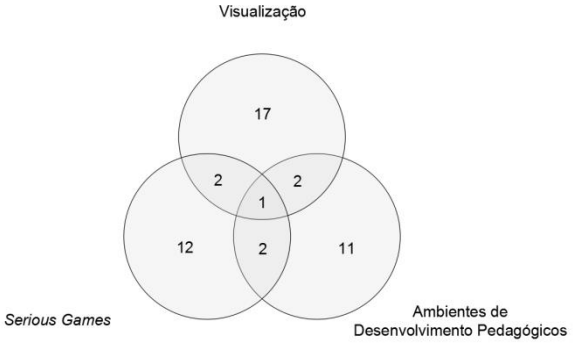

Figura 4: Intersecção entre as Categorias de Soluções

4.2 Análise das Relações entre as Categorias

Como discutido anteriormente, alguns artigos puderam ser classificados em mais de uma categoria de problema, uma vez que os autores relacionavam a existência de um problema como a causa de outro. A Figura 5 visa relacionar as categorias de problemas, contabilizando quantos artigos puderam ser classificados em ambos elementos de cada par de categorias de problemas.

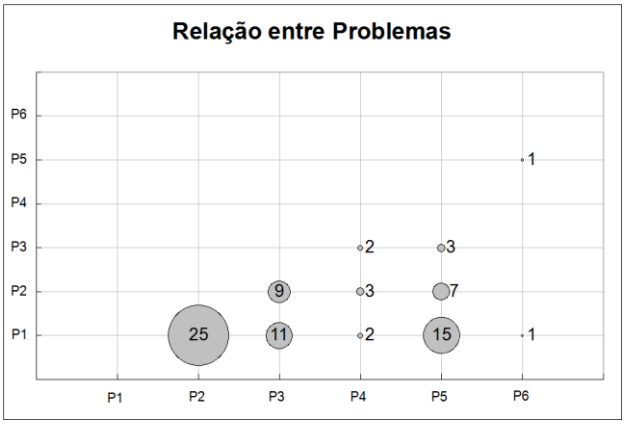

Figura 5: Relação entre as Categorias de Problemas

As categorias de problemas que mais tiveram artigos em comum foram a Aprendizagem dos Conceitos de Programação (P1) e a Aplicação de Conceitos de Programação (P2), com 25 artigos classificados simultaneamente em ambas categorias. A Aprendizagem de Conceitos de Programação (P1) também possui um número considerável de artigos em comum com a categoria Motivação (P5), com 15 artigos classificados simultaneamente em ambas categorias. Já a terceira combinação com mais artigos em comum é entre a

Aplicação de Conceitos de Programação (P2) e a Compreensão de Programas (P3), com 8 artigos classificados simultaneamente. A presença de um número grande de artigos em comum entre duas categorias pode fornecer indícios de que os problemas associados a elas podem estar relacionados.

Por sua vez, a Figura 6 contabiliza quantos artigos puderam ser classificados em ambos elementos de cada par de categorias de soluções. Diferentemente dos problemas, as categorias de soluções não apresentam muitos artigos em comum.

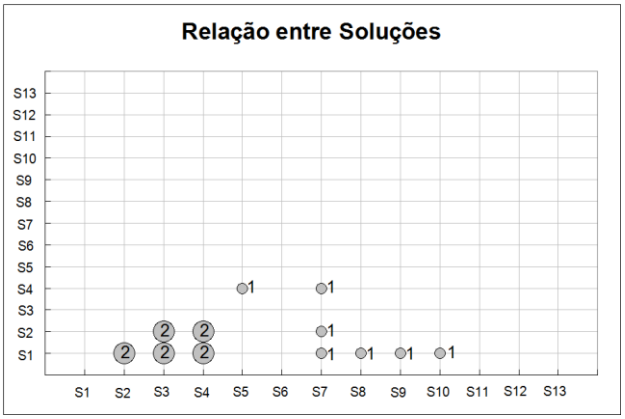

Figura 6: Relação entre as Categorias de Soluções

Por fim, a Figura 7 mostra a quantidade de artigos classificados em cada par de categorias de problemas e de soluções. Observa-se que a maior relação foi entre problemas associados à Aprendizagem de Conceitos de Programação (P1) com soluções relacionadas à Visualização (S1). A Visualização (S1) também é bastante sugerida para minimizar as dificuldades com relação à Aplicação de Conceitos de Programação (P2) e à Compreensão de Programas (P3).

Outras soluções bastante sugeridas foram: (i) a utilização de *Serious Games* (S2) para amenizar os problemas de Aprendizagem dos Conceitos de Programação (P1) e Motivação (P5); e (ii) a utilização de Ambientes de Desenvolvimento Pedagógicos (S3) para amenizar os problemas de Aprendizagem dos Conceitos de Programação (P1) e de Aplicação de Conceitos de Programação (P2).

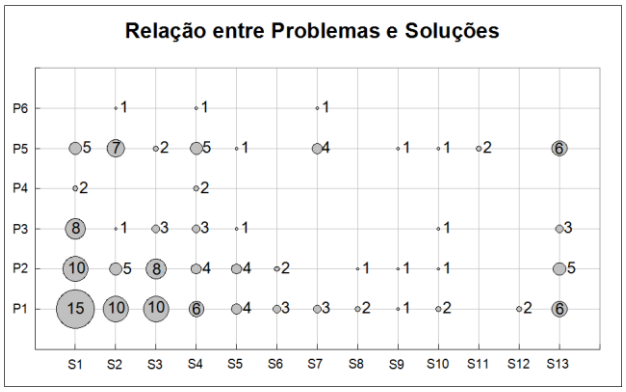

Figura 7: Relação entre as Categorias de Problemas e Soluções

4.3 Análise por Ano

Os resultados também foram analisados em relação ao ano de publicação (Figura 8). A categoria de problema mais referenciada em todos os anos foi a Aprendizagem em Conceitos de Programação, apenas igualando-se com Motivação em 2014. No caso das categorias de soluções não há uma predominância de nenhuma delas em todos os anos.

No ano de 2011 está o maior número de ocorrências de trabalhos sobre as dificuldades em aprender os conceitos de programação (P1), tema considerado em 15 artigos publicados nesse ano. Em 2011 também está o maior número de ocorrências de trabalhos relacionados ao problema da motivação (P5), considerado por 9 artigos publicados nesse ano. Já as dificuldades em aplicar os conceitos de programação (P2) teve um número maior de artigos relacionados em 2010.

Ainda no ano de 2011 está o maior número de ocorrências de trabalhos propondo a utilização de visualização (S1) para minimizar as dificuldades dos alunos no ensino e na aprendizagem de programação, sendo propostos em 6 artigos publicados nesse ano. Já a utilização de *serious games* (S2) teve o maior número de ocorrências em 2010 e 2012, sendo proposta por 4 artigos publicados em cada um desses anos. Por fim, os ambientes de desenvolvimento pedagógicos (S3) também teve o maior número de ocorrências em 2011, sendo propostos por 4 artigos publicados nesse ano.

Ressalta-se que o mapeamento foi realizado no segundo semestre de 2014, portanto existe a possibilidade de artigos publicados em 2014 não estarem disponíveis durante a condução do mapeamento.

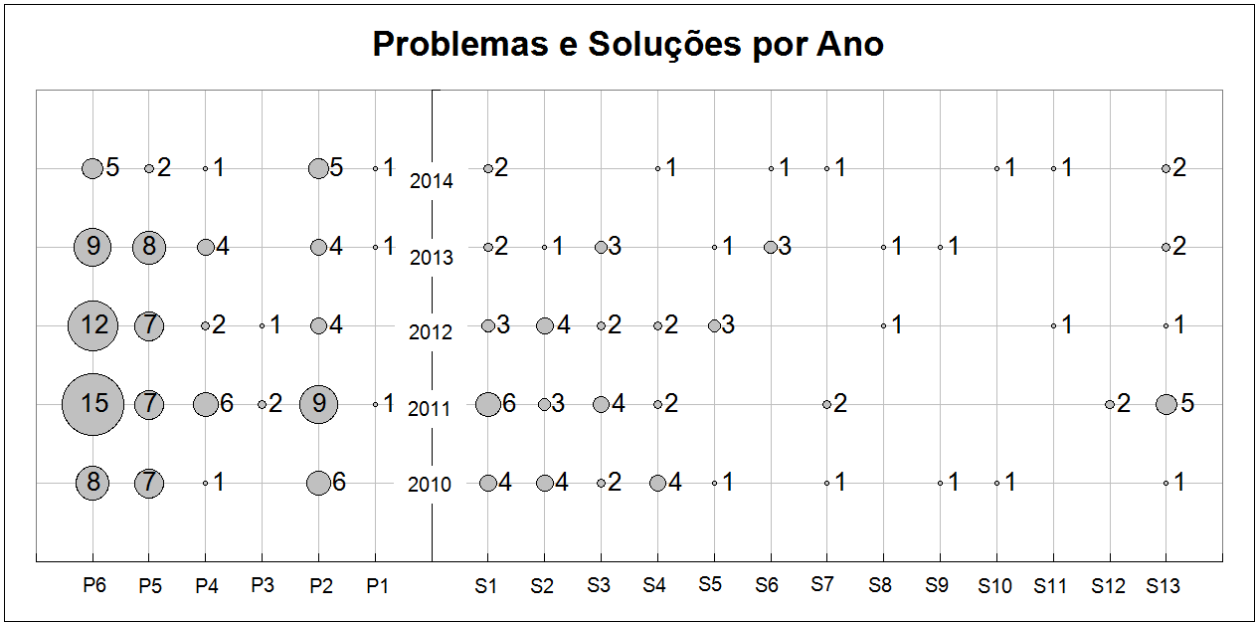

Figura 8: Problemas e Soluções por Ano

5 Conclusões e Trabalhos Futuros

Neste artigo foram apresentados e discutidos os resultados de um mapeamento sistemático conduzido com o objetivo de identificar os principais problemas e dificuldades no ensino e na aprendizagem de programação, bem como as soluções que vêm sendo propostas a fim de amenizá-los.

Um mapeamento sistemático é passível de falhas, especialmente quando a identificação dos estudos primários é feita unicamente por buscas automatizadas. Para amenizar essas limitações, procurou-se observar as referências dos artigos selecionados a fim de verificar se algum artigo importante não foi incluído.

Em geral, o mapeamento contribuiu para um melhor entendimento do atual estado da arte em ensino e aprendizagem de programação e para a identificação de limitações de pesquisa e futuras direções. Os resultados mostraram um grande número de artigos que discutem problemas na aprendizagem e propõe abordagens de ensino para minimizar as dificuldades na aprendizagem. Foi possível concluir que os principais problemas no ensino e na aprendizagem de programação investigados atualmente são: (1) as dificuldades dos alunos em aprender os conceitos de programação, (2) a dificuldade dos alunos na aplicação desses conceitos durante a construção de programas e (3) a falta de motivação entre os alunos na realização da atividade de programação. Para amenizar tais problemas e dificuldades, foram identificadas tendências de pesquisa principalmente com relação à: (1) utilização de visualização de programas e algoritmos, (2) utilização de *serious games* (3) o desenvolvimento de ambientes pedagógicos para o ensino e aprendizagem de programação.

A principal contribuição desse trabalho está na obtenção de uma visão geral dos problemas e dificuldades no ensino e na aprendizagem de programação, bem como das possíveis soluções propostas. Os resultados podem contribuir para o aperfeiçoamento da docência na área e incentivar novas pesquisas que testem empiricamente as soluções mapeadas.

Como trabalhos futuros, pretende-se complementar os resultados obtidos nesse MS por meio da busca de novos artigos em outras bases eletrônicas e por meio da busca manual em anais de congressos na área. Além disso, o projeto de um curso introdutório de programação deverá ser elaborado visando integrar as soluções propostas. A integração de ferramentas que apoiem as soluções identificadas em ambientes de aprendizagem também deverá ser investigada.

Agradecimentos

Os autores agradecem o apoio financeiro das agências de fomento: FAPESP, CAPES e CNPq.

Referências

[1] S. H. Abid, S. Zehra, and H. Iftikhar. Using computer aided language software for teaching and self-learning. In 2011 14th International Conference on Interactive Collaborative Learning, ICL 2011 - 11th International Conference Virtual University, VU'11, pages 102–106, 2011.

[2] Affandya, N. S Herman, S. B Salam, and E Noersasongko. A study of tracing and writing performance of novice students in introductory programming. Communications in Computer and Information Science, 181 CCIS(PART 3):557–570, 2011.

[3] A. O Ajayi, E. A Olajubu, D. F Ninan, S. A Akinboro, and H. A Soriyan. Development and testing of a graphical fortran learning tool for novice programmers. Interdisciplinary Journal of Information, Knowledge, and Management, 5:277–291, 2010.

[4] S. Al-Fedaghi. Conceptual framework for recursion in computer programming. Journal of Theoretical and Applied Information Technology, 46(2):983–990, 2012.

[5] J. M. Alberola and A. Garcia-Fornes. Achieving individual feedback through the on-line educational platform. In 2012 International Symposium on Computers in Education, SIIE 2012, 2012.

[6] A. P Ambrósio, F. M Costa, L Almeida, A Franco, and J Macedo. Identifying cognitive abilities to improve cs1 outcome. In Proceedings - Frontiers in Education Conference, FIE, 2011.

[7] A. P. L. Ambrósio and F. M. Costa. Evaluating the impact of pbl and tablet pcs in an algorithms and computer programming course. In SIGCSE'10 - Proceedings of the 41st ACM Technical Symposium on Computer Science Education, pages 495–499, 2010.

[8] R. Anderson, M. D. Ernst, R. Ordóéz, P. Pham, and S. A. Wolfman. Introductory programming meets the real world: Using real problems and data in cs1. In SIGCSE 2014 - Proceedings of the 45th ACM Technical Symposium on Computer Science Education, pages 465–466, 2014.

[9] T. N. M. Aris. Object-oriented programming semantics representation utilizing agents. Journal of Theoretical and Applied Information Technology, 31(1):10–20, 2011.

[10] T. N. M Aris and S. A Nazeer. Object-oriented programming semantics education based on intelligent agents. In 2011 5th Malaysian Conference in Software Engineering, MySEC 2011, pages 404–407, 2011.

[11] G. Bain and I. Barnes. Why is programming so hard to learn? In ITICSE 2014 - Proceedings of the 2014 Innovation and Technology in Computer Science Education Conference, page 356, 2014.

[12] L. C. Begosso, L. R. Begosso, E. M. Goncalves, and J. R. Goncalves. An approach for teaching algorithms and computer programming using greenfoot and python. In Proceedings - Frontiers in Education Conference, FIE, 2012.

[13] M. A. Brito and F. De Sá-Soares. Assessment frequency in introductory computer programming disciplines. Computers in Human Behavior, 30:623–628, 2014.

[14] E. Cambranes. Supporting novice programmers with natural language in the early stage of programming. In Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC, pages 173–174, 2013.

[15] O. Camp and R. Woodward. Ict enhanced learning experience for an introductory object oriented programming course: A case study. In CSEDU 2011 - Proceedings of the 3rd International Conference on Computer Supported Education, volume 2, pages 16–25, 2011.

[16] C. C Chang, I. C Chen, and H. C. K Lin. How to teach software programming? using affective teaching method and social network to enhance the learning motivation in programming courses - an example on facebook. In Workshop Proceedings of the 20th International Conference on Computers in Education, ICCE 2012, pages 345–351, 2012.

[17] S. M Cisar, R Pinter, P Cisar, and D Radosav. Teaching computer science in a web-based environment. In SISY 2013 - IEEE 11th International Symposium on Intelligent Systems and Informatics, Proceedings, pages 415–418, 2013.

[18] M. Corney, D. Teague, and R. N. Thomas. Engaging students in programming. In Conferences in Research and Practice in Information Technology Series, volume 103, pages 63–72, 2010.

[19] C. J Costa and M b Aparicio. Evaluating success of a programming learning tool. In ACM International Conference Proceeding Series, pages

73–78, 2014.

[20] N. J Coull and I. M. M Duncan. Emergent requirements for supporting introductory programming. ITALICS Innovations in Teaching and Learning in Information and Computer Sciences, 10(1):78–85, 2011.

[21] L. De Oliveira Brandao, R. Da Silva Ribeiro, and A. A. F. Brandao. A system to help teaching and learning algorithms. In Proceedings - Frontiers in Education Conference, FIE, 2012.

[22] E. Enström. Dynamic programming - structure, difficulties and teaching. In Proceedings - Frontiers in Education Conference, FIE, pages 1857–1863, 2013.

[23] A Gomes and A. J Mendes. Studies and proposals about initial programming learning. In Proceedings - Frontiers in Education Conference, FIE, pages S3F1–S3F6, 2010.

[24] A Gomes and A. J Mendes. A study on student performance in first year CS courses. In ITiCSE'10 - Proceedings of the 2010 ACM SIGCSE Annual Conference on Innovation and Technology in Computer Science Education, pages 113–117, 2010.

[25] S. Haatainen, A. J. Lakanen, V. Isomottonen, and V. Lappalainen. A practice for providing additional support in cs1. In Proceedings - 2013 Learning and Teaching in Computing and Engineering, LaTiCE 2013, pages 178–183, 2013.

[26] B. Hartanto and J. Reye. Incorporating anchored learning in a C# intelligent tutoring system. In Doctoral Student Consortia - Proceedings of the 21st International Conference on Computers in Education, ICCE 2013, pages 5–8, 2013.

[27] J. Helminen and L. Malmi. Jype - a program visualization and programming exercise tool for python. In Proceedings of the ACM Conference on Computer and Communications Security, pages 153–162, 2010.

[28] J. Hidalgo-Céspedes. Allegories for learning abstract programming concepts. In ICER 2014 - Proceedings of the 10th Annual International Conference on International Computing Education Research, pages 149–150, 2014.

[29] R. Horváth. Teaching one language in more depth is better than many languages superficially. In ICETA 2011 - 9th IEEE International Conference on Emerging eLearning Technologies and Applications, Proceedings, pages 71–74, 2011.

[30] L. L Hu, S. S Tseng, and T. J Lee. Towards scaffolding problem-solving implementation process in undergraduate programming course. In Proceedings - 2013 IEEE 13th International Conference on Advanced Learning Technologies, ICALT 2013, pages 417–418, 2013.

[31] Q Hu, Y Huang, and C Liu. The design and implementation of learner models in online peer assessment to support learning. In 2012 2nd International Conference on Consumer Electronics, Communications and Networks, CECNet 2012 - Proceedings, pages 2961–2963, 2012.

[32] Y Huang, J Meyers, W DuBow, Z Wu, and M Eisenberg. Programming plush toys as an introduction to computer science: The (fraught) question of motivation. In IADIS International Conference on Cognition and Exploratory Learning in Digital Age, CELDA 2011, pages 195–202, 2011.

[33] R Ibrahim and A Jaafar. Using educational games in learning introductory programming: A pilot study on students' perceptions. In Proceedings 2010 International Symposium on Information Technology - Visual Informatics, ITSim'10, volume 1, 2010.

[34] S. R b Jantan and S. A Aljunid. An experimental evaluation of scaffolded educational games design for programming. In 2012 IEEE Conference on Open Systems, ICOS 2012, 2012.

[35] Z. Karapinar, A. Senturk, S. Zavrak, R. Kara, and P. Erdogmus. Binary apple tree: A game approach to tree traversal algorithms. In 2012 International Conference on Information Technology Based Higher Education and Training, ITHET 2012, 2012.

[36] B Kaucic and T Asic. Improving introductory programming with scratch? In MIPRO 2011 - 34th International Convention on Information and Communication Technology, Electronics and Microelectronics - Proceedings, pages 1095–1100, 2011.

[37] Barbara A. Kitchenham and S. Charters. Guidelines for performing systematic literature reviews in software engineering. Technical report, School of Computer Science and Mathematics, Keele University, Keele, UK, 2007.

[38] E. Ko and K. Lee. Using music notation for teaching computer programming. In Proceedings of the 21st International Conference on Computers in Education, ICCE 2013, pages 475–478, 2013.

[39] S. Kollmansberger. Helping students build a mental model of computation. In ITiCSE'10 - Proceedings of the 2010 ACM SIGCSE Annual Conference on Innovation and Technology in Computer Science Education, pages 128–131, 2010.

[40] J. Liu, H. Li, and L. Chen. Research on teaching of college programming courses in network environment. Communications in Computer and Information Science, 201 CCIS(PART 1):210–215, 2011.

[41] L. Liu, Z. Wang, and X. Jiang. Anchor-based promgramming teaching embedded with ch platform. In Proceedings of 2010 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications, MESA 2010, pages 49–52, 2010.

[42] Y. J Lo, C. C Lin, L Hou, J. D Wu, Y. C Feng, and G. C Lee. Learning beginning programming with cloud-based cloze programming practices. In Proceedings - 2013 Learning and Teaching in Computing and Engineering, LaTiCE 2013, pages 235–236, 2013.

[43] M. Maleko, M. Hamilton, D. D'Souza, and F. Scholer. Understanding and analysing novice programmer interactions in a facebook programming group. In Proceedings - 2014 International Conference on Teaching and Learning in Computing and Engineering, LATICE 2014, pages 112–119, 2014.

[44] C. Malliarakis, M. Satratzemi, and S. Xinogalos. Towards a new massive multiplayer online role playing game for introductory programming. In ACM International Conference Proceeding Series, pages 156–163, 2013.

[45] S. W. Martins, A. J. Mendes, and A. D. Figueiredo. Student reflections as an influence in the dynamics of an introductory programming course. In Proceedings - Frontiers in Education Conference, FIE, 2011.

[46] R Matthews, H. S Hin, and K. A Choo. Merits and pitfalls of programming learning objects: A pilot study. In ACM International Conference Proceeding Series, pages 293–296, 2012.

[47] Á. Matth´ıasdóttir and H. J Geirsson. The novice problem in computer science. In ACM International Conference Proceeding Series, volume 578, pages 570–576, 2011.

[48] O. Meerbaum-Salant, M. Armoni, and M. Ben-Ari. Learning computer science concepts with scratch. In ICER'10 - Proceedings of the International Computing Education Research Workshop, pages 69–76, 2010.

[49] F. Mei. The application of trail-and-error learning in c language curriculum for non-majors in computer. In Proceedings of the 8th International Conference on Computer Science and Education, ICCSE 2013, pages 1239–1242, 2013.

[50] I. C Moura and N Van Hattum-Janssen. Teaching a cs introductory course: An active approach. Computers and Education, 56(2):475–483, 2011.

[51] O. L Oliveira, A. M Monteiro, and N. T Roman. Can natural language be utilized in the learning of programming fundamentals? In Proceedings - Frontiers in Education Conference, FIE, pages 1851–1856, 2013.

[52] P. Panwong and K. Kemavuthanon. Problem-based learning framework for junior software developer: Empirical study for computer programming students. Wireless Personal Communications, 76(3):603–613, 2014.

[53] A. Pears and M. Rogalli. mjeliot: Ict support for interactive teaching of programming. In Proceedings - Frontiers in Education Conference, FIE, 2011.

[54] M Piteira and C Costa. Computer programming and novice programmers. In ACM International Conference Proceeding Series, pages 51–53, 2012.

[55] M Piteira and C Costa. Learning computer programming: Study of difficulties in learning programming. In ACM International Conference Proceeding Series, pages 75–80, 2013.

[56] M Piteira and S. R Haddad. Innovate in your program computer class: An approach based on a serious game. In ACM International Conference Proceeding Series, pages 49–54, 2011.

[57] E. Pyshkin. Teaching programming: What we miss in academia - or what would mr. platt say? In 2011 7th Central and Eastern European Software Engineering Conference, CEE-SECR 2011, 2011.

[58] Anthony Robins, Janet Rountree, and Nathan Rountree. Learning and teaching programming: A review and discussion. Computer Science Education, 13(2):137–172, 2003.

[59] N. Salgado and T. Castro. An approach to support algorithms learning using virtual worlds. In Proceedings - 9th Brazilian Symposium on Collaborative Systems, SBSC 2012, pages 16–19, 2012.

[60] Á. Santos, A. Gomes, and A. Mendes. A class

record and reviewing system designed to promote programming learning. In Proceedings - Frontiers in Education Conference, FIE, 2011.

[61] A Settle, A Vihavainen, and J Sorva. Three views on motivation and programming. In ITICSE 2014 - Proceedings of the 2014 Innovation and Technology in Computer Science Education Conference, pages 321–322, 2014.

[62] J. Shi and S. White. Work-in-progress: Learning to program in a connected world. In Proceedings - 2013 Learning and Teaching in Computing and Engineering, LaTiCE 2013, pages 229–232, 2013.

[63] E. Soloway and James C. Spohrer. Studying the Novice Programmer. L. Erlbaum Associates Inc., 1 edition, 1988.

[64] J. A Stone and T. K Clark. The impact of problem-oriented animated learning modules in a cs1-style course. In SIGCSE'11 - Proceedings of the 42nd ACM Technical Symposium on Computer Science Education, pages 51–56, 2011.

[65] L. A. Sudol-DeLyser, M. Stehlik, and S. Carver. Learning looping: From natural language to worked examples. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 7315 LNCS:710–711, 2012.

[66] H. Tamada, A. Ogino, and H. Ueda. A framework for programming process measurement and compiling error interpretation for novice programmers. In Proceedings - Joint Conference of the 21st International Workshop on Software Measurement, IWSM 2011 and the 6th International Conference on Software Process and Product Measurement, MENSURA 2011, pages 233–238, 2011.

[67] H Tsukamoto, H Nagumo, Y Takemura, and N Nitta. Change of students' motivation in an introductory programming course for non-computing majors. In Proceedings of the 12th IEEE International Conference on Advanced Learning Technologies, ICALT 2012, pages 124–125, 2012.

[68] E. Vrachnos and A. Jimoyiannis. Design and evaluation of a web-based dynamic algorithm visualization environment for novices. In Procedia Computer Science, volume 27, pages 229–239, 2013.

[69] M. Wirth and J. McCuaig. Making programs with the raspberry pi. In Proceedings of WCCCE 2014: The 19th Western Canadian Conference on Computing Education - In-Cooperation with ACM SIGCSE, 2014.

[70] P. Wu. Practice and experience in the application of problem-based learning in computer programming course. In ICEIT 2010 - 2010 International Conference on Educational and Information Technology, Proceedings, volume 1, pages V1170–V1172, 2010.

[71] S. Xinogalos. Programming techniques and environments in a technology management department. In ACM International Conference Proceeding Series, pages 136–141, 2012.

[72] A. Yadin. Reducing the dropout rate in an introductory programming course. ACM Inroads, 2(4):71–76, 2011.

[73] Y. P Yang, L. Y Zhang, C. Y Wu, and L Cao. Teaching method and practice about the course of program language design in college. In ICETC 2010 - 2010 2nd International Conference on Education Technology and Computer, volume 2, pages V2275–V2278, 2010.

[74] V. C. O. Aureliano and P. C. A. R. Tedesco. Ensino-aprendizagem de Programação para Iniciantes: uma Revisão Sistemática da Literatura focada no SBIE e WIE. In Anais do 23º Simpósio Brasileiro de Informática na Educação, SBIE 2012, pages 1-10, 2012.

[75] A. Vihavainen, J. Airaksinen and C. Watson. A Systematic Review of Approaches for Teaching Introductory Programming and Their Influence on Success. In Proceeding of The International Computing Education Research Conference, ICER, pages 19-26, 2014.