Uma Ferramenta de Mineração de Textos como Forma de Apoio à Compreensão de Enunciados de Exercícios de Programação

Vinicius Hartmann Ferreira, Eliseo Berni Reategui

Programa de Pós-Graduação em Informática na Educação – Universidade Federal do Rio Grande do Sul (UFRGS)

Porto Alegre – RS – Brasil

{vinihf,eliseoreategui}@gmail.com

Abstract*. The introductory programming courses are known to exhibit high failure rate. Many are the difficulties faced by students in these disciplines, one being the understanding of the wording of the proposed exercises. This article describes an experiment in which the main concepts were identified from 123 set programming exercise with mining tool Sobek texts presenting them in the form of a graph. The realization of this experiment was to investigate the feasibility of using these graphs in order to provide students with a starting point for understanding the utterance. It concluded with the experiment that the statements that have wealth of information explicit generate graphs with more relevant concepts. However, it is believed to be the manipulation of graphs by students that will help them in understanding the statements.*

Resumo*. As disciplinas introdutórias de programação são conhecidas por apresentar alto índice de reprovação. Muitas são as dificuldades enfrentadas pelos estudantes nestas disciplinas, sendo uma delas a compreensão do enunciado dos exercícios propostos. Este artigo relata um experimento em que foram identificados os conceitos principais de 123 enunciados de exercícios de programação com a ferramenta de mineração de textos Sobek apresentando-os na forma de um grafo. A realização deste experimento teve como objetivo investigar a viabilidade de utilizar estes grafos com o intuito de fornecer aos estudantes um ponto de partida para a compreensão dos enunciados. Concluiu-se com o experimento que os enunciados que apresentam riqueza de informações explícitas geram grafos com mais conceitos relevantes. Porém, acredita-se que será a manipulação dos grafos pelos alunos que irá auxiliá-los na compreensão dos enunciados.*

1. Introdução

Aprender a programar não é fácil. Pesquisas internacionais indicam uma média global de aprovação em disciplinas introdutórias de algoritmos e de programação de apenas 67% [Caspersen 2007; Watson e Li 2014]. Dentre vários fatores que contribuem para este cenário, como o déficit no desenvolvimento do pensamento abstrato e do raciocínio lógico e mesmo aspectos técnicos próprios da tecnologia, se pode citar a inabilidade dos alunos em extrair informações dos enunciados de exercícios propostos em aula.

O fato de não compreender o enunciado de um exercício tem implicação na codificação e nos testes da solução construída. Devido à importância da leitura para as mais diversas áreas, sobretudo da compreensão do que se lê, pesquisas que utilizam as Tecnologias da Informação e Comunicação (TICs) para dar suporte a pessoas com dificuldades de leitura vem sendo desenvolvidas. Dentre as ferramentas propostas por estas pesquisas encontra-se o Sobek, que utiliza técnicas de mineração de textos para extrair os principais conceitos e as relações entre eles e representá-los na forma de um grafo. O Sobek se embasa no fato de que representações visuais de informação, como os mapas conceituais, apresentam resultados positivos no auxílio à compreensão de textos [Hessler e Reategui 2010].

Acredita-se que os grafos gerados pelo Sobek a partir de enunciados de exercícios de programação possam ser um ponto de partida para que os alunos, por meio da manipulação dos mesmos, sejam capazes de compreendê-los. Neste contexto, este artigo apresenta os resultados de um experimento que tem como objetivo investigar se é viável gerar estes grafos com a ferramenta Sobek com o intuito de fornecer ao aluno uma ideia inicial do conteúdo do enunciado. Este artigo está organizado de forma a apresentar os conceitos que fundamentam a pesquisa na Seção 2, a ferramenta Sobek na Seção 3, o experimento e seus resultados na Seção 4 e por fim as conclusões obtidas na Seção 5.

2. Aprendizagem de Algoritmos e Programação

Um algoritmo pode ser definido como uma sequência de passos dados para se chegar a solução de um problema. Podem ser citados como exemplos de algoritmos instruções para o preparo de um alimento, através dos passos descritos por uma receita, ou um guia sobre como trocar o pneu furado de um veículo. Este trabalho terá seu foco nos algoritmos computacionais, que pode ser definido como uma descrição lógica de um programa de computador que pode ser implementado em uma linguagem de programação [Raabe 2005].

Por programar entende-se a tradução de uma solução lógica para um problema, um algoritmo, em instruções compreendidas pela máquina através de linguagens de programação, adaptando a solução a sintaxe e semântica própria da linguagem utilizada. Com isto, pode-se concluir que um mesmo algoritmo pode ser implementado em uma ou mais linguagens de programação.

Dada a importância da programação para os cursos da área de informática, e mesmo para áreas afim, a disciplina com foco em introduzir os conceitos de algoritmos e programação está presente, em grande maioria dos cursos, no primeiro semestre dos cursos. Estas disciplinas assumem diferentes nomes (Algoritmos e Programação, Lógica de algoritmos, Programação I, entre outros), porém compartilham ementas similares, nas quais constam assuntos como: (i) representação de dados; (ii) operações; (iii) entrada e saída de dados; (iv) desvios condicionais; (v) laços de repetição; (vi) dados compostos; e (vii) modularização [Raabe 2005].

2.1 Dificuldades de Aprendizagem

A disciplina introdutória de algoritmos e programação é conhecida por ter altas taxas de reprovação e ser considerada uma das disciplinas mais difíceis dos cursos da área de informática, porém, são poucos os trabalhos que apresentam dados quantitativas que demonstrem isso. Em pesquisas realizadas por Bennedsen e Caspersen (2007) e Watson

e Li (2014) com instituições de nível superior que ofertam cursos da área de Informática foi possível concluir que a média de alunos que são aprovados na disciplina é de 67%, embora existam grandes variações entre alunos que são aprovados, reprovados ou que abandonaram o curso. Os pesquisadores também puderam verificar que turmas formadas por menos de 30 alunos, consideradas turmas pequenas, possuem maior taxa de aprovação. Além disso, concluíram também que o paradigma de programação escolhido pelo professor (procedural ou orientado a objetos) e a linguagem de programação utilizada não influenciaram as taxas de aprovação e que nos últimos 60 anos este número não aumentou de forma relevante.

Outra pesquisa, realizada duas vezes na Universidade de Otago (Nova Zelândia) com turmas introdutórias de programação de 250 alunos, permitiu identificar quais eram as principais dificuldades dos estudantes durante a disciplina, considerando aspectos técnicos da programação e aspectos externos ao ato de programar. Em ambas as execuções do experimento concluiu-se que os alunos possuem dificuldade em compreender os enunciados dos exercícios apresentados e que possuem dificuldade em trabalhar com o conceito de estruturas de dados compostas, conhecidas como vetores ou arrays. É interessante destacar que o conteúdo relacionado a vetores é trabalhado também em disciplinas de matemática nos cursos, o que pode traçar um paralelo entre o desempenho nas duas disciplinas [Robins, Haden e Garner 2006].

Uma síntese que categoriza as principais dificuldades enfrentadas na aprendizagem de algoritmos e programação foi elaborada por Raabe (2005) com base na literatura e em percepções sobre turmas desta disciplina na Universidade do Vale do Itajaí (UNIVALI). Nesta síntese as dificuldades são categorizadas em problemas de natureza didática, cognitiva e afetiva. Na categoria didática encontram-se problemas relacionados à atuação do professor, de sua relação com os alunos e também da forma com que a disciplina é organizada. Na categoria cognitiva estão problemas relacionados ao mau desenvolvimento de funções cognitivas para resolução de problemas algorítmicos, normalmente oriundos do ensino fundamental e médio. E na categoria afetiva encontram-se problemas relacionados ao estado afetivo do aluno que podem ter influência em sua aprendizagem. Nesta categorização os problemas afetivos foram classificados de acordo com a frequência com que ocorrem [Raabe 2005].

A forma como as disciplinas introdutórias de algoritmos são conduzidas varia para instituições e professores. Podem ser encontradas dentro desta disciplina diferentes práticas pedagógicas, porém a mais comum é a de resolução de problemas. Nesta prática são disponibilizadas listas de exercícios que devem ser resolvidos pelo aluno desenvolvendo algoritmos e implementando-os através de uma linguagem de programação.

Diante do enunciado de um exercício o aluno pode adotar diferentes estratégias para solucioná-lo, porém existem três passos que invariavelmente serão seguidos:

1. Abstração dos dados;
2. Elaboração da solução; e
3. Validação da solução desenvolvida.

O primeiro destes três passos envolve habilidades que vão muito além do conhecimento da sintaxe de uma linguagem de programação. A abstração dos dados envolve a boa leitura, a compreensão do que se leu e a inferência das informações

pertinentes. É neste passo que o aluno define o que deve fazer, com o que deve fazer e qual é o resultado esperado. Entra neste assunto também o contexto do problema proposto, pois a familiaridade que o aluno tem com o problema que deve resolver tem impacto direto no resultado do seu programa [Zanini 2013].

2.2 Enunciados de Exercícios de Programação

É muito comum encontrar alunos com dificuldade de compreensão de textos, incluindo-se nesta situação os enunciados de exercícios de programação. Isto pode ocorrer pelo fato de o aluno ter lacunas em sua habilidade de leitura e também pela falta de afinidade ou significância do problema [Zanini 2013]. Caso o aluno não compreenda o que deve fazer a partir de um enunciado as etapas de elaboração e validação da solução construída para um problema estarão comprometidas.

Com o intuito de investigar se o contexto e a estrutura dos enunciados de exercícios de programação influenciam no desempenho dos alunos nas disciplinas introdutórias de programação Zanini (2013) categorizou os enunciados quanto a sua estrutura em estruturados e não estruturados. Um enunciado estruturado é aquele que se encontra em um contexto específico (como de jogos ou matemática), apresenta indícios do processo de resolução de forma explícita e exemplos. Caso alguma destas características não seja encontrada o enunciado é classificado como não estruturado.

Ao analisar 428 enunciados retirados de 12 livros que contém exercícios de programação e estão presentes nas ementas de pelo menos 9% das universidades pesquisadas, Zanini (2013) concluiu que 64,72% destes enunciados não são contextualizados, 60,51% não apresentam indícios da resolução de forma explícita e 89,49% não apresentam exemplos de resolução. Ou seja, a grande maioria dos enunciados não é estruturado. E este fato pode contribuir para o baixo desempenho dos alunos nas disciplinas introdutórias, visto que no experimento realizado por Zanini (2013) comprovou-se que os alunos obtém melhor aproveitamento em enunciados estruturados do que em não estruturados.

3. A Ferramenta Sobek

Compreender o significado de um texto, seja ele um pequeno aviso ou um livro inteiro, é fundamental para o sucesso acadêmico e profissional. Dificuldades com a leitura podem ter implicações negativas na vida de um estudante, conforme relatam Korhonen, Linnanmaki e Aunio (2014) quando apontam a relação que há entre as dificuldades de aprendizagem, dentre elas a dificuldade de leitura, o bem-estar e o abandono escolar.

Pesquisas que envolvem as TICs vem sendo desenvolvidas com o intuito de dar suporte aos estudantes em atividades que envolvam leitura e, sobretudo compreensão de texto. Warschauer (2006) reforça a importância destas pesquisas quando afirma que a utilização de computadores em atividades que envolvam comunicação, seja ela oral ou escrita, potencializa o desenvolvimento destas habilidades pelos estudantes.

Dentre os trabalhos desenvolvidos, cita-se a ferramenta de mineração de textos denominada Sobek [Macedo *et al*. 2011]. O Sobek extrai os principais conceitos de um texto e os apresenta na forma de um grafo (Figura 1). Os conceitos extraídos tem potencial para auxiliar alunos na identificação do tema principal do texto e a representação gráfica tem potencial para auxiliá-los na compreensão da mensagem central do texto [Chang, Sung, e Chen 2002].

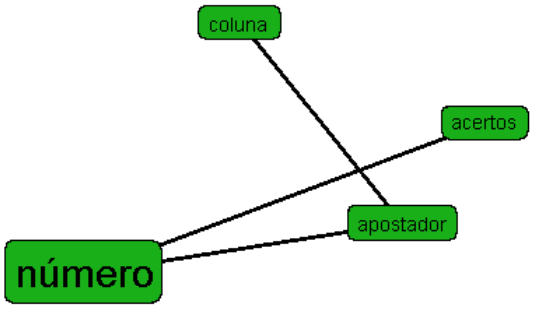

Figura 1. Grafo gerado pelo Sobek.

Os grafos gerados pelo Sobek apresentam os conceitos principais do texto utilizado como fonte como nodos e as relações entre estes conceitos na forma de arestas. Os conceitos que aparecem com mais frequência no texto são destacados com formato maior em relação aos outros, como o conceito "número" no grafo apresentado na Figura 1. Os grafos gerados não são estáticos, podendo o usuário adicionar ou remover conceitos e relações.

O Sobek já foi utilizado como suporte na sumarização de textos [Reategui, Klemann e Finco 2012], no suporte para a avaliação de textos produzidos de forma colaborativa [Macedo *et al.* 2011] e no suporte à compreensão da leitura [Hessler e Reategui 2010]. Assim como nos demais, o trabalho de Hessler e Reategui (2010) apresenta resultados relevantes na utilização do Sobek em atividades de leitura, demonstrando o potencial da ferramenta neste contexto, podendo ser explorado ainda mais quando combinado com diferentes práticas pedagógicas ou com o desenvolvimento de novas funcionalidades focadas na compreensão de texto.

4. O Experimento de Mineração de Textos com Enunciados

Para atingir o objetivo proposto neste artigo foi realizado um experimento de mineração de textos com enunciados de exercícios de uma disciplina introdutória de programação de um curso superior de tecnologia em Análise e Desenvolvimento de Sistemas e a ferramenta Sobek. Para este experimento foram selecionados todos os exercícios propostos para os alunos em um semestre da disciplina, totalizando 123 exercícios que tratavam sobre diferentes assuntos (Figura 2).

Figura 2. Assuntos tratados pelos exercícios apresentados aos alunos.

Dos 123 exercícios selecionados nenhum possuía o enunciado estruturado, refletindo o cenário descrito por Zanini (2013). Os exercícios se dividiam entre o contexto matemático, de jogos e de manipulação de palavras. Considerando um editor de texto com a fonte Times New Roman e o tamanho 12, o tamanho do enunciado dos exercícios avaliados variou entre 1 linha e meia e 8 linhas, com exceção do enunciado de dois trabalhos que possuíam mais de 20 linhas.

O experimento consistiu em gerar e analisar os grafos obtidos através da ferramenta Sobek para cada um dos enunciados. Levando em consideração o tamanho dos enunciados, configurou-se a ferramenta Sobek para extrair como conceito os termos que se repetissem pelo menos 3 vezes, limitou-se o grafo com o máximo de 15 conceitos e utilizou-se a lista de *stop words* padrão da ferramenta.

Dos grafos extraídos apenas 15,47% (19) apresentaram mais de 1 conceito. Este trabalho, portanto, analisou estes 19 grafos, tendo em vista que os demais apresentaram apenas 1 conceito, que não acrescenta muito na compreensão do problema, ou não foi possível gerar o grafo com o Sobek devido ao pequeno tamanho do enunciado.

É importante destacar que o número de conceitos extraídos está diretamente relacionado à quantidade de informações explícitas que ele contém, visto que o algoritmo do Sobek se baseia na frequência com que algum termo aparece no texto. Através do experimento foi possível constatar que enunciados muito pequenos, de no máximo 2 linhas e meia, e não estruturados, os mais comuns nas disciplinas, não geram grafos ou, se geram, apresentam apenas um conceito. Com isso, se pode concluir que a extração de grafos a partir de enunciados não estruturados e com poucas informações explícitas não se constitui como recurso válido para a compreensão dos enunciados, como o enunciado apresentado no Quadro 1.

Quadro 1. Enunciado do qual não foi possível extrair o grafo.

Construa um algoritmo que receba como entrada três valores e os imprima em ordem crescente.

Percebeu-se que em grafos menores (2 ou 3 termos) os termos extraídos apresentam, em sua maioria, um padrão do qual se pode destacar o assunto que o

problema trata e uma variável relacionada. O grafo apresentado na Figura 3 foi extraído a partir do enunciado apresentado no Quadro 2. Neste grafo é possível identificar que o tema central do exercício é um estacionamento e que uma das variáveis para resolução é o tempo em que o veículo ficou no estacionamento, representado pelo termo "minutos".

Quadro 2. Enunciado de problema sobre valor do pagamento de um estacionamento.

Elabore um algoritmo para o controle de pagamento de um **estacionamento**. Com base na tabela abaixo, informe qual será o valor pago por um motorista ao sair do **estacionamento**. Para isto, considere que o motorista informa ao sistema o tempo em **minutos** que ficou no **estacionamento**.

Tempo	Valor
Até 15 **minutos**	Grátis
Até 60 **minutos**	R$ 5.40
Acima de 60 **minutos**	R$ 6.20 + R$ 1.20 por hora excedente.

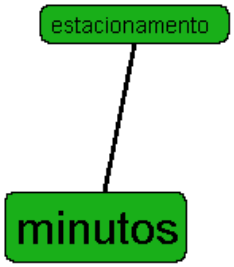

Figura 3. Grafo construído a partir do enunciado do Quadro 1.

Ao analisar os grafos maiores (com mais de 3 conceitos), percebe-se que o padrão apresentado nos grafos menores permanece. É possível identificar o tema central do enunciado e também variáveis que precisam ser observadas para a solução do exercício. O grafo (Figura 4) gerado a partir do enunciado apresentado no Quadro 3 permite identificar que o problema trata sobre vendas em uma empresa e que o corretor está relacionado às vendas e à comissão. Neste caso é possível concluir que os termos centrais do enunciado foram realmente extraídos, e mais do que isso, estão relacionados.

Quadro 3. Enunciado de problema sobre vendas e comissão de corretores.

Uma **empresa** de **vendas** tem três corretores. A **empresa** paga ao **corretor** uma **comissão** calculada de acordo com o valor de suas **vendas**. Se o **valor da venda** de um **corretor** for maior que R$ 50.000.00 a **comissão** será de 12% do valor vendido. Se o **valor da venda** do **corretor** estiver entre R$ 30.000.00 e R$ 50.000.00 (incluindo extremos) a **comissão** será de 9.5%. Em qualquer outro caso, a **comissão** será de 7%. Escreva um algoritmo que gere um relatório contendo nome**, valor da venda** e **comissão** de cada um dos corretores. O relatório deve mostrar também o total de vendas da **empresa**.

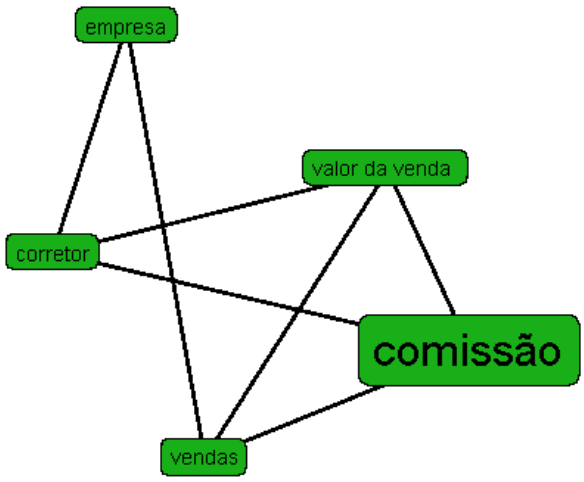

Figura 4. Grafo construído a partir do enunciado do Quadro 3.

Nem todos os grafos extraídos a partir dos enunciados apresentam termos válidos para a compreensão. Conforme pode ser visto no grafo na Figura 5 gerado a partir do enunciado apresentado no Quadro 4, por ser um enunciado com poucas informações explícitas, de difícil compreensão, os termos extraídos não estão totalmente relacionados e apresentam termos irrelevantes como “calcule” e “SB”.

Quadro 4. Enunciado de problema sobre vendas e comissão de corretores.

Faça um algoritmo que: a) Obtenha o **valor para a variável** HT (horas trabalhadas no mês); b) Obtenha o **valor para a variável** VH (valor hora trabalhada): c) Obtenha o **valor para a variável** PD (percentual de **desconto**); d) **Calcule** o **salário** bruto => **SB** = HT * VH; e) **Calcule** o total de **desconto** => TD = (PD/100)***SB**; f) **Calcule** o **salário** líquido => SL = **SB** – TD; g) Apresente os valores de: Horas trabalhadas, **Salário** Bruto, **Desconto**, **Salário** Liquido.

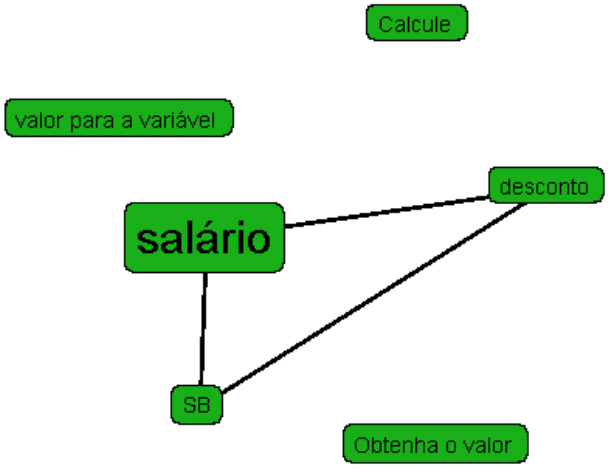

Figura 5. Grafo gerado a partir do enunciado de cálculo de salário de funcionários.

5. Conclusão

Este artigo apresentou os resultados de um experimento no qual foram construídos grafos de forma automática a partir de enunciados de exercícios de programação com uma ferramenta de mineração de textos. Este experimento foi realizado com o intuito de identificar e analisar se é viável utilizar os grafos gerados pelo Sobek como ponto inicial na identificação do conteúdo dos enunciados. Tendo em vista que os estudantes apresentam dificuldades em identificar os principais elementos de um texto, uma ferramenta que os extrai de forma automática pode tornar-se um recurso relevante.

Este trabalho apresenta como uma de suas conclusões o fato de que devido a maioria dos exercícios não ser estruturado, ou seja, não apresentar informações de forma explícita, exemplos de resolução e contextos bem definido, o número de grafos relevantes foi pequeno (apenas 19 dentro de um universo de 123 problemas). Isto leva ao questionamento acerca da disponibilização de listas imensas com exercícios superficiais para os alunos em contraste com listas de poucos exercícios bem descritos e contextualizados.

Concluiu-se também que, embora em menor número, os grafos que foram construídos apresentaram informações relevantes para a compreensão do enunciado. Dentro deste aspecto, foi possível visualizar que os enunciados com pouco mais de 2 linhas e meia apresentaram grafos com menos informações do que os enunciados maiores e com mais informações explícitas. Além disso, para além da ideia central do enunciado, foi possível identificar variáveis pertinentes ao problema, no sentido computacional da palavra, nos grafos.

É importante ressaltar que a representação visual das informações é um recurso comprovadamente útil para auxiliar na compreensão textual. Porém, é a manipulação destes recursos visuais pelos alunos que transforma a representação visual em um recurso relevante no processo de compreensão de textos. Partindo disto, pode-se concluir que os grafos gerados, ao apresentarem a ideia central do enunciado, se constituem como um importante ponto de partida para a compreensão dos mesmos.

Por mais que um grafo não contenha todos os conceitos presentes no enunciado, ele apresenta ao aluno sua ideia central, algo que talvez ele sozinho não conseguisse identificar. Partindo deste grafo inicial o aluno irá organizar seu pensamento acerca do enunciado adicionando e removendo conceitos e relações ao grafo. E é neste ponto que os grafos gerados pelo Sobek apresentam potencial como recurso para a dar suporte a compreensão dos enunciados de exercícios de programação.

Referências

Bennedsen, J. e Caspersen, M. E. (2007). Failure rates in introductory programming. In *SIGCSE Bulletin*, v. 39, n. 2, p. 32-36.

Chang, K. E., Sung, Y. T. e Chen, I. D. (2002). The effect of concept mapping to enhance text comprehension and summarization. In *The Journal of Experimental Education*, v. 71, n. 1, p. 5-23.

Hessler, J. C. e Reategui, E. B. (2010). Um método para apoio à leitura baseado no uso de uma ferramenta de mineração de texto. In *RENOTE - Revista Novas Tecnologias na Educação*, v. 8, n. 3.

Korhonen, J., Linnanmaki, K. e Aunio, P. (2014). Learning difficulties, academic well-being and educational dropout: a person-centred approach. In *Learning and Individual Differences*, v. 31, n. 1, p. 1-10.

Macedo, A. L., Azevedo, B., Behar, P. A. e Reategui, E. B. (2011). Acompanhamento da interação e produção textual coletiva através de mineração de textos. In *Informática na Educação: Teoria e Prática*, v. 14, n. 2.

Raabe, A. L. A. Uma proposta de arquitetura de sistema tutor inteligente baseada na teoria das experiências de aprendizagem mediadas. 2005. 155 f. Tese (Doutorado em Informática na Educação) – Programa de Pós-Graduação em Informática na Educação, UFRGS, Porto Alegre, 2005.

Reategui, E. B., Klemann, M. e Finco, M. (2012). "Using a text-mining tool to support text summarization". In: IEEE International Conference on Advanced Learning Technologies, 2012, Rome, Italy. Proceedings...**,** Rome: IEEE, 2012.

Robins, A., Haden, P. e Garner, S. (2006). "Problem Distributions in a CS1 Course". In: Australasian Computing Education Conference, 8, 2006, Hobart, Australia. Proceedings... Hobart, Australia, 2006.

Warschauer, M. (2006) "Literacy and technology: bridging the divide", In: Cyberlines 2: Languages and cultures of the Internet, Editado por Gibbs, D. e Lrause, K. L. John Wiley & Sons ltd., England.

Watson, C. e Li, F. W. B. (2014). "Failure rates in introductory programming revisited". In: Innovation and Technology in Computer Science Education, 19, 2014, Uppsala, Sweden. Proceedings**...** New York: ACM, 2014.

Zanini, A. S. Avaliação da influência dos enunciados na resolução de problemas de programação introdutória. 2013. 126 f. Dissertação (Mestrado em Computação Aplicada) – Programa de Pós-Graduação em Computação Aplicada, UNIVALI, Itajaí, 2013.