Inteligência Coletiva como ferramenta de apoio na construção de Loops Internos em Sistemas Tutores Inteligentes

Thyago Tenório[1,2], **Seiji Isotani**[1], **Ig Ibert Bittencourt**[2]

[1]Instituto de Ciências Matemáticas e de Computação (ICMC) – USP
Avenida Trabalhador São Carlense, 400 – 13566-590 – São Carlos – SP – Brasil

[2]Instituto de Computação (IC) – UFAL
Avenida Lourival Melo Mota, S/N – 57072-900 – Maceió – AL – Brasil

tm.thyago@usp.br, sisotani@icmc.usp.br, ig.ibert@ic.ufal.br

Abstract. *The Inner Loops present in the Intelligent Tutoring Systems (ITS) are responsible for providing feedbacks for each step of the student's interaction. However, for its correct functioning, it is necessary that the ITS has knowledge of the involved elements in the step by step resolution process of the activities. The representation of this knowledge in the current ITS is specific context-based or dependent of the system interface and their informations are registered by the professor, which increases your workload. In this sense, this article presents a modeling and proposes a process for the construction of the Inner Loops using the concept of Collective Intelligence, extracting the necessary informations from the step-by-step resolutions and students.*

Resumo. *Os Loops Internos presentes nos Sistemas Tutores Inteligentes (STIs) são responsáveis por fornecer feedbacks a cada passo de interação do estudante. Contudo, para seu correto funcionamento, é necessário que o STI tenha o conhecimento dos elementos envolvidos no processo de resolução passo a passo das atividades. A representação desse conhecimento nos STIs atuais é dependente de contexto ou vinculada a interface do sistema e o processo de cadastro é atrelado ao professor, gerando uma sobrecarga de trabalho. Nesse sentido, este artigo apresenta uma modelagem e propõe um processo para a construção dos Loops Internos utilizando o conceito de Inteligência Coletiva, extraindo as informações necessárias das resoluções passo a passo e dos estudantes.*

1. Introdução

Os Sistemas Tutores Inteligentes (STIs) se apresentam como uma ferramenta com enorme potencial do ponto de vista educacional. Eles são programas de computador que utilizam técnicas de Inteligência Artificial para representar o conhecimento [Polson and Richardson 2013] e fornecem instruções diretas personalizadas ou feedback aos alunos ao mesmo tempo em que estes executam uma tarefa [Psotka et al. 1988]. Eles proporcionam um ensino adaptado a cada aluno, se aproximando do comportamento do professor na sala de aula [Nkambou et al. 2010] e apresentam bons resultados de aprendizagem [VanLehn 2011].

Esses resultados estão relacionados ao nível de granularidade desse tipo de sistema. Diferente dos sistemas de tutoria tradicionais, os quais são limitados a prover feedbacks apenas para as respostas finais dos estudantes, os STIs trabalham em um nível

DOI: 10.5753/cbie.sbie.2020.1233

mais baixo de granularidade, permitindo que os estudantes entrem com todos os passos requeridos para resolver uma determinada atividade [Vanlehn 2006]. Assim, um STI tem potencial para apresentar feedbacks, dicas e explicações a cada passo no processo de resolução do estudante [VanLehn 2011].

O menor nível de granularidade é implementado e executado nos STIs através do Loop Interno, o qual é responsável por fornecer feedbacks para cada passo tomado pelo aluno na resolução de uma atividade [Vanlehn 2006]. Para viabilizar a sua construção, é necessário que o STI tenha o conhecimento dos elementos no processo de resolução passo a passo das atividades. No entanto, representar e instanciar esse conhecimento é uma tarefa complexa e onerosa ao professor. Por esse motivo, alguns STIs não possuem Loops Internos, enquanto outros implementam de forma simplificada [Vanlehn 2006]. Nos STIs que possuem Loops Internos, a representação do conhecimento é dependente do domínio de aplicação ou vinculada a interface do sistema e o processo de cadastro das informações é fortemente atrelado ao professor, o que gera uma sobrecarga no seu trabalho.

Nesse sentido, esse artigo apresenta uma modelagem de resolução passo a passo, independente de domínio de aplicação e da interface do sistema, e propõe a utilização de Inteligência Coletiva como ferramenta de apoio na construção e implementação dos Loops Internos, obtendo as informações necessárias a partir dos dados das interações dos estudantes no sistema e de informações colhidas diretamente dos estudantes, fazendo com que o processo de cadastro das informações não gere uma sobrecarga ao professor. Esse artigo está dividido em 6 seções. A Seção 2 apresenta alguns conceitos de background e trabalhos relacionados. Por sua vez, a Seção 3 mostra a definição do modelo construído enquanto que a Seção 4 traz o processo de obtenção das informações. Em seguida, a Seção 5 apresenta uma avaliação com a utilização do processo e algumas discussões. Finalmente, a Seção 6 apresenta as conclusões e alguns trabalhos futuros.

2. Background e Trabalhos Relacionados

O Loop Interno é o processo responsável por analisar a resolução passo a passo de um estudante e fornecer o apoio necessário para que este consiga concluir sua tarefa. Alguns exemplos de apoio são feedback mínimo, sugestões e dicas, comentários de um erro específico e revisão da solução [Vanlehn 2006]. No entanto, para que esse processo funcione corretamente, é necessário que o sistema tenha os conhecimentos envolvidos no processo de resolução passo a passo das atividades que serão desenvolvidas [Vanlehn 2006]. Assim, é necessária uma modelagem formal para representar esse conhecimento bem como um processo para a instanciação das informações necessárias.

Por ser uma tarefa complexa e onerosa ao professor, muitos sistemas não possuem Loops Internos, enquanto outros implementam de forma simplificada, limitando os feedbacks providos ao aluno. Um exemplo é o tutor em física [Vanlehn 2006], no qual o aluno apenas fornece uma resposta numérica ao sistema. Se o número estiver correto, o tutor diz isso e atribui o próximo problema. Se a resposta estiver incorreta e for um número que os autores sabem e só pode ser gerado com um erro específico, então o sistema aponta o erro e pede ao aluno que tente novamente. Esses sistemas de Loop único obtêm uma única evidência por tarefa e, inevitavelmente, se tornam bastante limitados, [Vanlehn 2006].

Por outro lado, alguns exemplos de STIs com Loops Internos são o Algebra Cognitive Tutor [Pane et al. 2014], Andes [Schulze et al. 2000] e o SQL-Tutor [Mitrovic 2003].

No entanto, a modelagem do conhecimento utilizada por esses sistemas são focadas em contextos específicos e a construção do conhecimento é dependente do professor. Trazendo uma proposta diferente, o PAT2Math [Seffrin et al. 2012] utiliza sistemas especialistas para gerar o conhecimento necessário acercas dos passos a serem executados pelo aluno [Seffrin et al. 2012]. Com isso, o conhecimento é construído independente do professor, porém sua aplicação ainda se mantém em um domínio específico. Uma ferramenta de autoria chamada CTAT [Aleven et al. 2006] permite a construção de STIs com Loops Internos independente de domínio, tendo sua modelagem do conhecimento baseada em grafos. No entanto, há a sobrecarga adicional no trabalho do professor, o qual precisa criar todo o grafo de conhecimento e essa modelagem é dependente da interface do CTAT.

Esse trabalho apresenta uma modelagem e um processo de construção das informações baseado em Inteligência Coletiva (IC). O Centro de Inteligência Coletiva do Instituto de Massachusettes de Tecnologia (MIT) define IC como um grupo de indivíduos fazendo coisas de forma coletiva que são inteligentes [Leimeister 2010]. A ideia é utilizar as interações e os próprios estudantes nos STIs, colhendo e agrupando as informações necessárias para o Loop Interno. Um exemplo que utiliza IC é o ASSISTments [Heffernan and Heffernan 2014], uma plataforma para "crowdsource" conteúdo educacional, tais como dicas, explicações, feedbacks imediatos [Razzaq et al. 2020], respostas comuns de erro, vídeos e novos problemas, utilizando o auxílio de milhares de professores, permitindo que se construa feedbacks em vários contextos e evite a sobrecarga de trabalho do professor. A abordagem proposta se diferencia em dois pontos principais: menor nível de granularidade e a utilização de estudantes ao invés de professores.

3. O modelo de Resolução Passo a Passo

Para essa modelagem, as atividades que consistem na execução de múltiplos passos, por parte dos alunos, será limitada a resolução de problemas. As únicas informações importantes são seu enunciado (EP) e a resposta correta (ACP). É possível haver respostas erradas, as quais formarão o conjunto de alternativas do problema (AP). Essa simplificação permite maior compatibilidade com os STIs. Em resumo, um problema P é definido como $P = (EP, AP, ACP)$. Ao longo desse artigo, usaremos o problema P_1- *"Qual o valor da expressão numérica X = 10 * 50 - 10 / 5"* para exemplificações.

Ao longo do processo de resolução do problema, os valores encontrados pelo aluno representam um estado E. De forma geral, um estado representa um ponto de parada do estudante ao longo do processo de encontrar a solução. Eles podem ser classificados em Estados Iniciais E_I (ponto de partida da resolução), Estados Intermediários ou Estados Finais E_F (ponto final da resolução). A partir de ACP, encontramos o E_F esperado para o problema, porém este nem sempre é alcançado pelo estudante. Como exemplo, temos que para P_1, o E_I é *"X = 10 * 50 - 10 / 5"* e o estado final esperado E_F é *"X = 498"*. Ao se encontrar em um estado, o estudante pode executar alguma ação A, a qual consiste em executar alguma operação com os valores contidos no estado.

Nesse contexto, um passo consiste na execução de uma ação, por parte do estudante, responsável por levar o estudante de um determinado estado de origem E_1 para um estado de destino E_2 com a finalidade de resolver o problema, sendo representado por $P(E_1, A) \rightarrow E_2$. Um passo somente retorna o estado E_2 resultante quando podemos aplicar a ação A no estado E_1. Assim, considerando o problema P_1, o estudante poderia

executar a ação A = "Multiplicar", o que o levaria ao estado E_2 com valor *"X = 500 - 10 / 5"*. Se, ao invés de executar a ação A, o estudante executasse a ação A_2 = "Dividir", o estado resultante seria E_2 com valor *"X = 10 * 50 - 2"*. Finalmente, o estudante poderia executar a ação A_3 = "Subtrair", o que o levaria ao estado E_2 com valor *"X = 10 * 40 / 5"*. Nota-se, com esse exemplo, que executar um determinado passo pode levar o estudante a um estado correto ou incorreto, a depender da validade da ação executada no estado atual.

Para resolver um determinado problema, o aluno deve executar múltiplos passos, de forma sequencial, até que atinja a solução E_F. Dessa maneira, uma resolução passo a passo R é o conjunto de passos P_i, tal que a aplicação sucessiva de P_i levará o estado inicial E_I em um estado final E_F, sendo representada por $R = \{P_1, P_2, ..., P_i\}$. Uma resolução R é considerada correta se o E_F obtido pela execução dos passos contidos em R a partir de E_I for igual ao E_F esperado para o problema e todas as ações A_i contidas nos passos P_i de R sejam consideradas válidas. Uma resolução pode ser considerada parcialmente correta se algum passo P_i contiver uma ação A_i considerada inválida e ainda assim a resolução R conseguir atingir o E_F esperado. Finalmente, caso o E_F obtido a partir de R seja diferente do E_F esperado, consideramos R como uma resolução incorreta.

Uma resolução R pode ser representada visualmente como apresentado na Figura 1. Cada vértice representa um estado da resolução (E_i) e cada aresta representa um passo P_i pertencente a resolução. A aresta é direcionada, isto é, a base da flecha indica o estado de origem do passo e a ponta indica o estado destino do passo, após aplicar a ação A. Considerando um problema P qualquer, é possível termos várias resoluções R_i, a depender do estado inicial E_I e dos passos P_i executados ao longo do processo de solução.

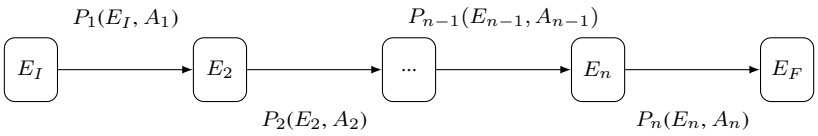

Figura 1. Representação Visual para uma Resolução R

3.1. O Grafo de Conhecimento do Problema

Ao trabalhar com resolução passo a passo, o sistema deve dar feedbacks ao estudante em cada passo de sua resolução. Para conseguir dar esse feedback, o ideal é que o sistema tenha o conhecimento de todas as possíveis resoluções do problema - e consequentemente, todos os estados, passos e ações possíveis - integradas, em uma espécie de rede de conhecimento. Para representá-la , **podemos modelar a rede através de um grafo de conhecimento (GC)**, definido como $GC = (V, A)$, sendo V o conjunto não vazio de todos os estados (E) possíveis e A o conjunto de todos os passos (P) possíveis.

Um exemplo de grafo de conhecimento GC (construído para o problema P_1), representado visualmente, pode ser visto na Figura 2. **Cada vértice (indicado por um retângulo com bordas arredondadas) representa um estado possível do problema**, com destaque para o estado inicial E_I, cuja linha de borda é tracejada e para os estados finais (E_{F_1}, E_{F_2}, E_{F_3} e E_{F_4}) cuja linha de borda é dupla. Por sua vez, **cada aresta (representada por uma flecha direcionada) representa um passo possível a ser executado**. Podemos notar que, diferentemente do que acontece com uma resolução, no grafo

de conhecimento um estado pode conter mais de um passo de entrada e/ou saída, como podemos ver nos estados E_I, E_2, E_3, E_4 e E_{F_3}.

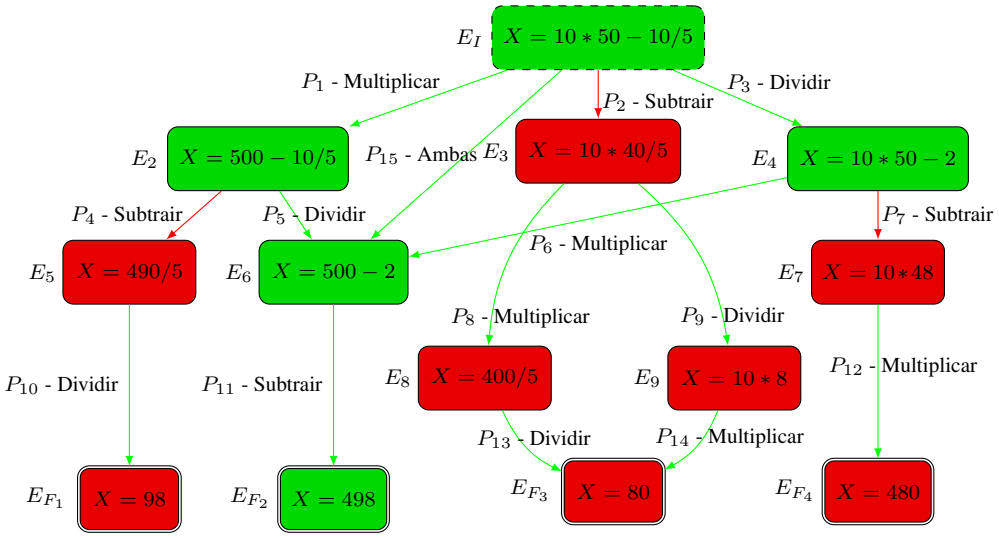

Figura 2. Exemplo de grafo de conhecimento para o problema P_1

Também podemos representar se um estado está correto ou incorreto e se um passo é válido ou inválido. Para isso, podemos **colorir os estados corretos em verde e os estados incorretos em vermelho**. Na Figura 2, temos que os estados E_I, E_2, E_4, E_6 e E_{F_2} são estados corretos enquanto que os estados E_3, E_5, E_7, E_8, E_9, E_{F_1}, E_{F_3} e E_{F_4} são considerados incorretos. De forma semelhante aos estados, os **passos com ações válidas são coloridos com verde e passos com ações inválidas são coloridos com vermelho**. Na Figura 2, os passos P_1, P_3, P_5, P_6, P_8, P_9, P_{10}, P_{11}, P_{12}, P_{13} e P_{14} são passos válidos, enquanto que os passos P_2, P_4 e P_7 são considerados inválidos.

No contexto do grafo de conhecimento, uma resolução é um caminho no grafo. Podemos representar **a resolução obtida através do grafo como uma sequência de vértices e arestas, de forma alternada, começando e terminando com vértice, sendo a origem em algum estado inicial (E_I) e término em algum estado final (E_F)**. Na Figura 2, uma resolução R_1 poderia ser a sequência E_I, P_1, E_2, P_5, E_6, P_{11}, E_{F_2}, enquanto que uma resolução R_2 poderia ser a sequência E_I, P_3, E_4, P_7, E_7, P_{12}, E_{F_4}. Ainda podemos destacar no grafo as resoluções corretas ou incorretas. Caso o caminho realizado por ela no grafo inclua somente vértices e arestas verdes, como no caso da R_1 então é uma resolução correta. Caso passe por algum estado ou vértice em vermelho mas encontre um estado final verde, a resolução é parcialmente correta. Em qualquer outro caso, a resolução é considerada incorreta, como, por exemplo, a R_2.

4. O processo de obtenção das informações

A Seção 3 apresentou um modelo para representar as informações necessárias para o funcionamento dos Loops Internos. No entanto, é necessário instanciar as informações (através de um processo) para cada problema cadastrado pelo professor (etapa 1). A etapa 2 consiste nos alunos respondendo-os utilizando resolução passo a passo e recebendo feedbacks pelo sistema, utilizando-os em seu processo de resolução e avaliando sua eficácia.

Na etapa 3, o sistema constrói/atualiza o GC a partir das resoluções e das avaliações que vão sendo obtidas, utilizando-as como ponto de aprendizagem para novos estados, passos e feedbacks. A Figura 3 apresenta o processo de forma resumida.

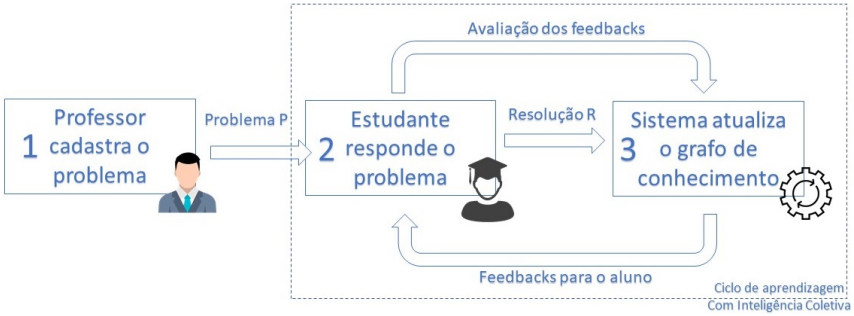

Figura 3. Resumo do processo de construção do grafo de conhecimento

Notamos que há um ciclo de aprendizagem incremental entre as etapas 2 e 3, uma vez que os feedbacks gerados pela etapa 3 são utilizados pelos alunos na etapa 2, o qual avalia sua eficácia. As novas resoluções e as avaliações dos feedbacks geradas pela etapa 2 são utilizadas para criar e melhorar os feedbacks existentes na etapa 3, os quais serão utilizados no processo de novas resoluções, fechando o ciclo. Esse é um processo de construção incremental utilizando IC dos alunos como núcleo. O Foco central nesse processo é como o sistema irá construir e atualizar o GC a partir das resoluções e das avaliações de feedbacks recebidas. Para isso, definimos e implementamos um algoritmo[1], chamado *updateGC*. De forma resumida, o algoritmo executa os seguintes passos:

- Recuperar o grafo de conhecimento atual do problema P. Caso não exista um grafo cadastrado para o problema, isto é, é a primeira resolução a ser cadastrada, é necessário inicializar um novo grafo de conhecimento vazio para o problema P.
- Se existir avaliações A, avaliá-las e atualizar os valores referentes aos elementos (estados ou passos) no grafo de conhecimento vinculados a avaliação.
- Recuperar os passos da resolução R e para cada passo S recuperado deve:
  - I. Recuperar o estado de origem do passo S no grafo. Caso esse estado não esteja cadastrado no grafo, deve-se criá-lo. Note que se S for o *primeiro passo* da resolução R, esse estado deve ser marcado como E_I no grafo.
  - II. Recuperar o estado de destino do passo S no grafo. Caso esse estado não esteja cadastrado no grafo, deve-se criá-lo. Marcar como E_F se S for o *último passo* da resolução R.
  - III. Recuperar o passo no grafo cujo estado de origem, estado de destino e ação seja iguais ao passo S. Caso não existe, deve-se criá-lo.
  - IV. Recuperar passos no grafo com a mesma origem e destino do passo S e marcá-los como passos equivalentes à S.
- Atualizar os valores referentes a corretude e validade no grafo de conhecimento dos elementos (estados e passos) envolvidos na resolução R.

O último passo tem como objetivo atualizar os valores de corretude dos estados e validade dos passos da resolução R no GC. Para estimar esses valores, definimos mecanismos **baseando-se na ideia que passos e estados corretos/incorretos estarão presentes em resoluções corretas/incorretas, respectivamente**. Para isso, consideremos C_R

[1]Disponível em https://bityli.com/mYvNm

o conjunto de resoluções do problema P, utilizadas para construir o GC. Consideremos C_{RC} o subconjunto de resoluções corretas e C_{RI} o subconjunto de resoluções incorretas. Definimos a função PE (Equação 1), que indica se um elemento (estado ou passo) está presente ou não em uma resolução. Definimos também a função PEC (Equação 2) que conta quantas vezes um determinado elemento está presente em um conjunto de resoluções.

$$\text{PE}(E, R) = \begin{cases} 1 & \text{se} \quad E \in R \\ 0 & \text{se} \quad E \notin R \end{cases} \qquad (1) \qquad PEC(E, C_R) = \sum_{i=1}^{|C_R|} PE(E, R_i) \qquad (2)$$

Com isso, podemos estimar a validade de um passo $V(P)$ ou corretude de um estado $C(E)$ como indicado na Equação 3. Em resumo, a equação conta quantas vezes um elemento apareceu nas resoluções corretas e subtrai quantas vezes ele apareceu em resoluções incorretas e finalmente divide pelo total de aparições no conjunto de resoluções. O intervalo da função $V(P)$ ou $C(E)$ é $[-1, 1]$.

$$V(P)\ ou\ C(E) = \frac{PEC(E, C_{RC}) - PEC(E, C_{RI})}{PEC(E, C_R)} \qquad (3)$$

A partir da Equação 3, criamos um mapeamento entre o valor dessas funções e algumas classes para esses elementos (sendo as classes Inválido (I), Fortemente Inválido (FI), Possivelmente Inválido (PI), Neutro(N), Possivelmente Válido (PV), Fortemente Válido (FV) e Válido (V) para os passos e as classes Correto (C), Incorreto (I) e Desconhecido (D) para os estados), conforme indicado na Figura 4.

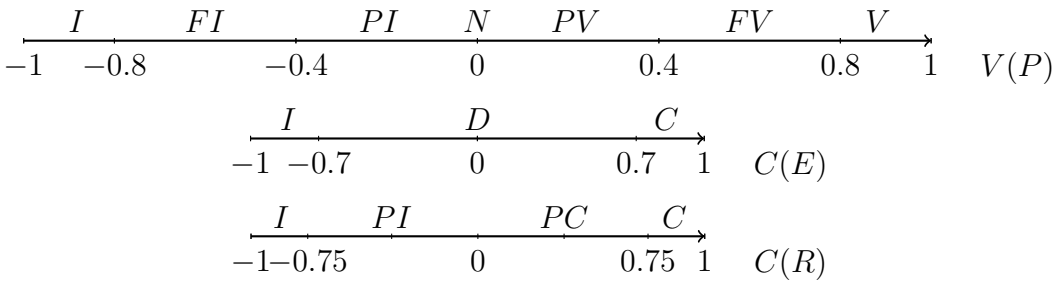

Figura 4. Mapeamento entre função e classes dos Elementos

Finalmente, para calcular a corretude de uma resolução $C(R)$, definimos a Equação 4, considerando C_E o conjunto de estados da resolução R e C_P o conjunto de passos da resolução R. Analisando a Equação 4 temos que a corretude de uma resolução R, ou $C(R)$, é a soma da média da corretude dos estados E contidos em R com a média da validade dos passos P contidos em R, dividido por 2 para normalizar os resultados, mantendo o intervalo em $C(R) = [-1, 1]$.

$$C(R) = \frac{1}{2 * |C_E|} * \sum_{i=1}^{|C_E|} C(E_i) + \frac{1}{2 * |C_P|} * \sum_{j=1}^{|C_P|} V(P_j) \qquad (4)$$

Também realizamos um mapeamento entre a função e as classes definidas para uma resolução (Incorreta (I), Parcialmente Incorreta (PI), Parcialmente Correta (PC) ou Correta (C)), apresentado na Figura 4.

5. Avaliação - Construindo um grafo de conhecimento

Para avaliar a proposta do trabalho, realizamos uma implementação simplificada de um STI com o domínio de expressões numéricas e disponibilizada no Amazon Mechanical Turk ao longo de 5 dias consecutivos. Na implementação, um usuário, escolhido aleatoriamente, em primeiro momento, recebia instruções rápidas sobre expressões numéricas e sobre a atividade que lhe era solicitada e indicava seu nível de conhecimento no assunto em uma escola de 1 a 10. A atividade consistia em responder de 1 a 3 problemas (nível de dificuldade crescente) que foram previamente cadastrados por um especialista, com limite de 10 minutos por problema. Os alunos então responderam cada problema utilizando resolução passo a passo, recebendo feedbacks ao final do processo e, a cada resposta, o sistema atualizava de forma assíncrona o GC.

No total, foram registradas 249 resoluções ($P_1 = 99$, $P_2 = 81$ e $P_3 = 69$). Das resoluções do problema P_1, 18 (18,2%) foram consideradas inválidas e 81 (81,8%) foram válidas. Dessas, 56 (69.1%) estavam corretas e 25 (30.9%) incorretas, sendo 12 respostas incorretas diferentes. Com relação ao problema P_2, 9 (11,1%) foram consideradas inválidas e 72 (88,9%) válidas. Dessas, 41 (56,9%) estavam corretas e 31 (43,1%) erradas, sendo 13 respostas diferentes. Finalmente, para o problema P_3, 14 (20,3%) resoluções eram inválidas enquanto 55 (79,7%) válidas. Dessas, apenas 18 (32,7%) estavam corretas enquanto que 37 (67,3%) estavam incorretas, sendo 29 respostas diferentes. A Tabela 1 apresenta um resumo dos dados das resoluções.

	Resoluções Tot / Invál / Vál	**Resoluções Cor / Incor**	**Tempo(s) Mín / Max / Avg**	**Conhe. Média**	**Qtd. Passos Mín / Max / Avg**
P_1	99 / 18 / 81	56 / 25 (12)	59 / 567 / 236.54	7.86	3 / 12 / 7
P_2	81 / 9 / 72	41 / 31 (13)	45 / 590 / 250.50	7.98	4 / 11 / 7
P_3	69 / 14 / 55	18 / 37 (29)	79 / 593 / 375.13	8.03	4 / 15 / 8.34

Tabela 1. Resumo dos dados das resoluções cadastradas

Podemos observar na Tabela 1 que quanto maior a dificuldade do problema, maior é a taxa de erro, levando a mais resoluções diferentes. Com isso, a dificuldade do problema está relacionada com o tamanho de seu GC, o que leva também a uma maior sobrecarga do professor. Esses achados são confirmados na Tabela 2, a qual apresenta os dados dos GCs gerados, e indica que o número de estados e passos aumentou significativamente com a dificuldade. Mesmo problemas relativamente simples apresentam GCs complexos.

Nesse sentido, por mais que o professor se esforce no cadastro das informações, **haverão resoluções indicadas pelo estudante, cujos passos, ações ou estados são desconhecidos pelo sistema** e, consequentemente, este fica impossibilitado de dar feedback ao aluno. Com uso de inteligência coletiva, o sistema pode atualizar as informações a medida que novas resoluções vão sendo adicionadas, permitindo a construção dos GCs mais complexos. Para avaliar os resultados do modelo proposto, um especialista analisou cada GC gerado. A Tabela 3 apresenta os resultados obtidos.

	Estados Tot / Ini / Int / Fin	**Estados Cor / Incor / Des**	**Passos Tot**	**Passos I / FI / PI / N / PV / FV / V**
P_1	122 / 19 / 90 / 13	42 / 74 / 6	165	82 / 0 / 0 / 1 / 0 / 6 / 76
P_2	123 / 19 / 90 / 14	27 / 81 / 15	182	98 / 1 / 0 / 5 / 6 / 10 / 62
P_3	240 / 16 / 192 / 32	37 / 188 / 15	282	207 / 2 / 2 / 5 / 4 / 3 / 59

Tabela 2. Resumo dos dados dos Grafos de Conhecimento gerados

		Especialista - Estado			**Especialista - Passos**	
	Estado	**Correto**	**Incorreto**	**Passo**	**Válido**	**Inválido**
P_1	**Correto**	34	8	**PV+FV+V**	79	3
	Incorreto	1	73	**PI+FI+I**	0	82
	Desconhecido	6	0	**Neutro**	1	0
P_2	**Correto**	22	5	**PV+FV+V**	70	8
	Incorreto	8	73	**PI+FI+I**	16	83
	Desconhecido	15	0	**Neutro**	5	0
P_3	**Correto**	36	1	**PV+FV+V**	64	2
	Incorreto	50	138	**PI+FI+I**	61	150
	Desconhecido	15	0	**Neutro**	5	0

Tabela 3. Matriz de Confusão com dados Modelo x Especialista

Analisando os dados da Tabela 3, temos que o modelo apresentou uma taxa de acerto para P_1 de 87,70% (107 de 122) para os estados e 97,57% (161 de 165) para os passos. Com relação a P_2, a taxa foi de 77,24% (95 de 123) e 84,07% (153 de 182), respectivamente. Finalmente, as taxas para P_3 foram 72,50% (174 de 240) e 75,89% (214 de 282). Podemos observar que a precisão de acerto da solução proposta para os estados e passos é alta. Ao observar os estados onde houveram divergências, vimos que a grande maioria se referia ao uso incorreto de parênteses. Isso pode ter ocorrido por falta de atenção na avaliação dos estados. Um outro fato observado na Tabela 3 é a alta taxa de falso negativos para os passos (8,79% em P_2 e 21,63% em P_3). Ao analisar as classificações, percebemos que havia uma diferença no processo realizado pelos estudantes e o especialista. Os estudantes avaliaram os passos olhando todo o contexto do problema e assim, classificavam todos os passos como inválidos a partir de um determinado erro. O especialista analisou cada passo individualmente, classificando-os como corretos. Esse é um ponto que precisa ser ajustado para definir quando um passo é inválido.

6. Conclusões e Trabalhos Futuros

Esse artigo apresentou uma modelagem de resolução passo a passo que buscou representar os principais conceitos envolvidos nos Loops Internos dos STIs de forma simplificada, independente de domínio e implementações. Além da modelagem, foi apresentado um processo para a obtenção das informações necessárias a partir dos estudantes, não sobrecarregando o professor, mantendo o grafo sempre mais atualizado e permitindo novos caminhos de resoluções não previamente planejados pelo professor. Os resultados indicaram que é possível construir o GC a partir dos estudantes com boa precisão.

Como trabalhos futuros, pretendemos realizar a integração do modelo e do processo em um STI real, aplicando-os em outros domínios e obtendo resultados mais

próximos de uma aplicação na prática. Além disso, pretendemos expandir o modelo e o processo para incluir outras informações como dicas, explicações, justificativas e feedbacks de erro específico, que poderão ser utilizadas pelo sistema para auxiliar ainda mais os estudantes no processo de resolução passo a passo. A ideia é construir tais informações utilizando os estudantes com o suporte e o apoio do professor no processo.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

Referências

Aleven, V., McLaren, B. M., Sewall, J., and Koedinger, K. R. (2006). The cognitive tutor authoring tools (ctat): preliminary evaluation of efficiency gains. In *International Conference on Intelligent Tutoring Systems*, pages 61–70. Springer.

Heffernan, N. T. and Heffernan, C. L. (2014). The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497.

Leimeister, J. M. (2010). Collective intelligence. *Business & Information Systems Engineering*, 2(4):245–248.

Mitrovic, A. (2003). An intelligent sql tutor on the web. *International Journal of Artificial Intelligence in Education*, 13(2-4):173–197.

Nkambou, R., Mizoguchi, R., and Bourdeau, J. (2010). *Advances in intelligent tutoring systems*, volume 308. Springer Science & Business Media.

Pane, J. F., Griffin, B. A., McCaffrey, D. F., and Karam, R. (2014). Effectiveness of cognitive tutor algebra i at scale. *Educational Evaluation and Policy Analysis*, 36(2):127–144.

Polson, M. C. and Richardson, J. J. (2013). *Foundations of intelligent tutoring systems*. Psychology Press.

Psotka, J., Massey, L. D., and Mutter, S. A. (1988). *Intelligent tutoring systems: Lessons learned*. Psychology Press.

Razzaq, R., Ostrow, K. S., and Heffernan, N. T. (2020). Effect of immediate feedback on math achievement at the high school level. In *Artificial Intelligence in Education*, pages 263–267.

Schulze, K. G., Shelby, R. N., Treacy, D. J., Wintersgill, M. C., Vanlehn, K., and Gertner, A. (2000). Andes: An intelligent tutor for classical physics. *Journal of Electronic Publishing*, 6(1).

Seffrin, H., Rubi, G., and Jaques, P. (2012). O modelo cognitivo do sistema tutor inteligente pat2math. In *Simpósio Brasileiro de Informática na Educação-SBIE*.

Vanlehn, K. (2006). The behavior of tutoring systems. *International journal of artificial intelligence in education*, 16(3):227–265.

VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4):197–221.