Análise de modelo para a tomada de decisões pedagógicas em um ambiente voltado ao aprendizado de algoritmos

Djefferson S. S. Maranhão[1], **Antonio Carlos Raposo**[1], **Carlos de Salles Soares Neto**[1]

[1]Departamento de Informática – Universidade Federal do Maranhão (UFMA)
São Luís – MA – Brasil

{djefferson.maranhao, antoniocarlosraposo93}@gmail.com, csalles@deinf.ufma.br

Abstract. *This paper analyzes qualitatively the use of POMDP as a model for making pedagogical decisions in an environment focused on learning algorithms. The issue to be faced is the sequencing of computational problems in intelligent tutoring environments, since it is common practice to propose computational problems in introductory algorithm courses. The sequencing of activities was seen as a sequential decision process under uncertainty, since the tutor does not know exactly what knowledge state the student is currently in. The main contribution of this work was to show that the model can be applied satisfactorily to the intelligent tutoring process, considering that students were guided by the paths that offered the greatest gains in knowledge.*

Resumo. *O presente trabalho analisa de forma qualitativa o emprego de POMDP como modelo para a tomada de decisões pedagógicas em um ambiente voltado ao aprendizado de algoritmos. A questão a ser enfrentada é o sequenciamento de problemas computacionais em ambientes de tutoria inteligente, tendo em vista ser uma prática comum a proposição de problemas computacionais em cursos introdutórios de algoritmos. O sequenciamento das atividades foi encarado como um processo de decisão sequencial sob incerteza, vez que o tutor não sabe exatamente em qual estado de conhecimento o aluno está atualmente. A principal contribuição deste trabalho foi mostrar que o modelo pode ser aplicado de forma satisfatória ao processo de tutoria inteligente, tendo em vista que os alunos foram guiados pelos caminhos que ofereciam os maiores ganhos de conhecimento.*

1. Introdução

O aprendizado de algoritmos é um tema multifacetado e complexo. Em geral, os alunos demonstram pouca habilidade em interpretar a descrição de um problema e decompô-lo em instâncias menores [McCracken et al. 2001]. Também apresentam dificuldades em rastrear, ler e entender trechos de código e não são capazes de compreender princípios e rotinas básicas de programação [Lister et al. 2004].

Adicionalmente, o aprendizado de programação possui etapas bem definidas, que são pré-requisitos umas das outras. De forma ilustrativa, primeiro aprende-se o conceito de declaração e inicialização de variáveis. Depois, passa-se para o estudo de operadores aritméticos, de comparação e lógicos. Por fim, compreende-se o conceito de estruturas de decisão, o qual é necessário para a compreensão de laços de repetição.

DOI: 10.5753/cbie.sbie.2020.1223

Uma prática comum no aprendizado de algoritmos é a resolução de problemas computacionais do mundo real. Para isso, o professor precisa selecionar um conjunto de atividades, organizando-as de acordo com o conceito abordado e o nível de dificuldade. Há uma ampla diversidade de ferramentas que apoiam o professor nesse processo de escolha das atividades, tais como o URI Online [Bez et al. 2014] e o UVA Online Judge [Revilla et al. 2008].

Embora sejam muito úteis enquanto repositórios de problemas, essas ferramentas não oferecem um suporte à navegação inteligente. Assim, o professor deve definir a ordem apropriada para a resolução das atividades, para que os alunos não se sintam desmotivados. Todavia, o esforço exigido por essa tarefa pode se tornar impraticável, levando-se em consideração que cada aluno aprende em um ritmo particular.

Ademais, esse trabalho de organização das atividades de forma personalizada pode ser encarado como um processo de tomada de decisão sequencial, em que o professor é um agente/tutor inteligente responsável por monitorar as soluções submetidas pelos alunos e por propor novas ações de ensino de forma sequencial, levando em consideração as respostas observadas anteriormente.

Nesse ambiente, o tutor inteligente não consegue aferir com exatidão o estado de conhecimento do aluno, mas é capaz de fazer uma série de suposições com base nas respostas submetidas pelos alunos. Esse processo de ensino no qual a tomada de decisões ocorre em meio a incertezas quanto ao estado de conhecimento do aluno pode ser modelado por meio de Processos de Decisão de Markov Parcialmente Observáveis [Woolf 2010] [Wang 2018].

Os estudos acerca da aplicação dessa técnica à tutoria inteligente começaram ainda na década de 90. Nos primeiros anos, ela foi usada para modelar estados mentais e para encontrar as melhores maneiras de se ensinar certos conceitos. Os trabalhos mais atuais, todavia, caracterizam-se pela aplicação da técnica para otimizar e personalizar o ensino, mas variam conforme as definições de estados, ações, observações, etc.

O objetivo deste trabalho é analisar qualitativamente o emprego de Processos de Decisão de Markov Parcialmente Observáveis como modelo para a tomada de decisões pedagógicas em um ambiente voltado ao aprendizado de algoritmos. Esse modelo tem se mostrado promissor para a construção de tutores inteligentes e recomendadores de conteúdo educacional que levam em conta o aprendizado atual do aluno.

O artigo está organizado como segue: a Seção 2 apresenta a fundamentação teórica acerca do Processo de Decisão de Markov Parcialmente Observável e do algoritmo de iteração de valor baseada em pontos; a Seção 3 apresenta trabalhos relacionados a esta pesquisa; a Seção 4 descreve o modelo proposto; a Seção 5 detalha o experimento realizado; e a Seção 6 apresenta a conclusão do trabalho.

2. Processo de Decisão de Markov Parcialmente Observável

O presente tópico faz uma revisão dos Processos de Decisão de Markov (MDP, do inglês Markov Decision Processes) e dos Processos de Decisão de Markov Parcialmente Observáveis (POMDP, do inglês *Partially Observable Markov Decision Processes*). Também será apresentada uma breve explanação sobre o algoritmo de Iteração de Valor Baseado em Pontos (PBVI, do inglês *Point Based Iteration Value*).

Um MDP é um modelo probabilístico comumente utilizado para representar problemas de decisão sequencial. Esse modelo consiste em um processo estocástico discreto no tempo em que um agente atua em um ambiente que lhe é visível e busca alcançar recompensas cada vez mais promissoras. Dessa forma, um MDP modela um agente atuando em um ambiente em que o estado pode ser observado.

Um MDP é uma tupla $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$ em que: $\mathcal{S}$ é um espaço de estados discreto e finito; $\mathcal{A}$ é um espaço de ações discreto e finito; $\mathcal{T}\big(s' \mid s, a\big)$ é uma função de probabilidades de transição, isto é, a probabilidade de transitar do estado s para o estado s' ao tomar a ação a; $\mathcal{R}(s, a)\big)$ é uma função de recompensa de valor real, que define as recompensas recebidas por tomar a ação a no estado s; γ é um fator de desconto [Van Otterlo and Wiering 2012] [Kolobov 2013].

Um POMDP é uma extensão do modelo MDP para representar processos de decisão que envolvem estado oculto e incerteza nos efeitos das ações. Essa estrutura matemática é capaz de modelar um agente que não tem acesso direto ao estado atual, mas que pode observá-lo por meio de sensores com um certo nível de ruído. Por conta dessa informação imprecisa, o agente armazena um conjunto de crenças sobre os possíveis estados.

Formalmente, um POMDP é uma tupla $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma, b_0 \rangle$, em que: $\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}$, são os mesmos que em um MDP; Ω é um conjunto finito e discreto de observações; $\mathcal{O}$ é a probabilidade de observar o após executar a e alcançar o estado s; b_0 define o estado de crença inicial, isto é, a crença do agente com relação ao seu estado inicial [Ten Pas 2012][Van Otterlo and Wiering 2012].

Um estado de crença descreve a distribuição de probabilidade com relação aos estados, provendo uma estatística suficiente para um determinado histórico. O próximo estado de crença, denominado $b' = \tau\big(b, a, o\big)$, que incorpora a ação a e a observação o prévias e o estado de crença atual b, é atualizado pela Equação 1:

$$b\big(s'\big) = \frac{\mathcal{O}\big(o \mid s', a\big) \sum_{s \in S} b\big(s\big) \mathcal{T}\big(s' \mid s, a\big)}{\Pr\big(o \mid b, a\big)} \tag{1}$$

$$\Pr\big(o \mid b, a\big) = \sum_{s \in S} b\big(s\big) \sum_{s' \in S} \mathcal{T}\big(s' \mid s, a\big) \mathcal{O}\big(o \mid s', a\big) \tag{2}$$

Por fim, o objetivo do modelo POMDP é encontrar uma política $\pi : \mathcal{B} \rightarrow \mathcal{A}$, mapeando crenças $b \in \mathcal{B}$ para o espaço de ação $a \in \mathcal{A}$ [Somani et al. 2013], que maximize algum aspecto do fluxo de recompensas. Para um horizonte infinito, o valor de uma política π para uma crença b é a recompensa total descontada que o agente receberá ao executar π, calculada pela Equação 3:

$$V_\pi\big(b\big) = \mathbb{E}\Bigg[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}\Big(s_t, \pi\big(b_t\big)\Big) \mid b_0 = b \Bigg] \tag{3}$$

2.1. Funções de Valor

Uma função de valor V para um espaço de crenças pode ser representada como um conjunto finito de vetores de $|\mathcal{S}|$ dimensões, conhecidos como vetores α, isto é, $V =$

$\{\alpha_1, \alpha_2, \cdots, \alpha_m\}$. Assim, V é contínua, linear por partes e convexa. Essa propriedade geométrica simplifica o processo de busca de uma solução, vez que possibilita a construção de uma sequência de estimativas da função de valor de forma iterativa [Smallwood and Sondik 1973]. A Figura 1 mostra como um espaço de crença pode ser dividido em regiões, cada uma com um vetor associado.

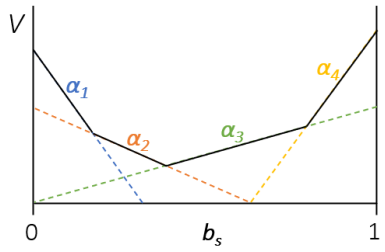

Figura 1. Distribuição de vetores no espaço de crenças.

Uma política com relação ao espaço de crenças é definida pela associação de uma ação a a cada vetor α, de tal forma que $\alpha \cdot b = \sum_s \alpha(s)b(s)$ representa o valor de se realizar uma ação a no estado de crença b e seguir com a política posteriormente. A política pode ser calculada por meio da Equação 4:

$$\pi_V(b) = \text{argmax}_{a:\alpha_a \in V} \ \alpha_a \cdot b \tag{4}$$

Uma política ótima pode ser calculada pela função de valor ótima, que pode ser computada por meio de uma série de iterações. Dada uma função de valor inicial e um conjunto de vetores, pode-se calcular vetores de etapas mais distantes e atualizar o valor de cada função de valor. O processo é repetido até que um ponto de equilíbrio seja alcançado. A etapa de iteração (atualização de Bellman) é calculada pela Equação 5, em que $r_a = \mathcal{R}(s, a)$ é uma representação vetorial da função de recompensa.

$$V_{n+1}(b) = \max_{a \in \mathcal{A}} \left[b \cdot r_a + \gamma \sum_{o \in \Omega} \Pr\left(o \mid a, b\right) V_n\left(\tau\left(b, a, o\right)\right) \right] \tag{5}$$

Uma função de valor pode ser definida usando outra representação, como um mapeamento direto entre estados e valores de crença. Por essa representação, usa-se o operador de *backup* H (Equação 7), para calcular uma atualização da função de valor.

$$Q_V(b, a) = b \cdot r_a + \gamma \sum_{o \in \Omega} \Pr\left(o \mid a, b\right) V_n\left(\tau\left(b, a, o\right)\right) \tag{6}$$

$$HV_n = \max_{a \in \mathcal{A}} Q_V(b, a) \tag{7}$$

De forma mais geral, a expressão que representa a operação de *backup* de valor é dada na Equação 8:

$$V_{n+1} = HV_n \tag{8}$$

2.2. Iteração de Valor Baseada em Pontos (PBVI, do inglês *Point Based Value Iteration*)

Calcular todo o espaço de crenças de um agente é uma tarefa muito difícil, pois exige muito poder computacional. Uma abordagem mais eficiente consiste em computar apenas um conjunto limitado de crenças de forma amostral, permitindo que o agente interaja aleatoriamente com o ambiente. Em vez de computar todos os pontos de crença no espaço de estados, apenas um pequeno conjunto finito de pontos alcançáveis é calculado.

Um método baseado em pontos para o cálculo de modelos POMDP é o algoritmo PBVI [Pineau et al. 2003]. O PBVI aproxima a função de valor selecionando um pequeno conjunto finito de pontos de crença representativos e rastreando o valor apenas para esses pontos. O PBVI também divide o espaço de crenças em regiões de forma iterativa, mas não calcula cada vetor da mesma maneira.

Em vez disso, o algoritmo usa um conjunto finito de pontos de crença $B = \{b_0, b_1, \cdots, b_q\}$ do espaço de estado e inicia um vetor α separado para cada ponto de crença selecionado. Em seguida, ele atualiza repetidamente o valor desse vetor até convergir. Conhecendo o valor dos vetores, a função de valor de uma crença pode ser calculada da mesma maneira que no algoritmo de iteração de valor, usando a expressão 9.

$$V_{n+1} = \tilde{H}_{PBVI} V_n \tag{9}$$

A Figura 2 mostra como os pontos de crença podem ser escolhidos no espaço de crenças. Somente os pontos de crença representados são usados para calcular os vetores correspondentes.

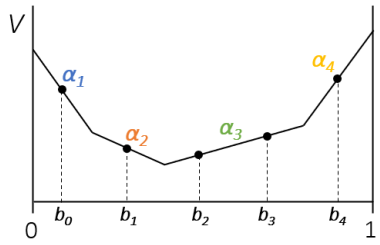

Figura 2. Função de valor usando um algoritmo PBVI.

Finalmente, como o algoritmo seleciona apenas crenças alcançáveis, em vez de crenças aleatórias, ele é capaz de superar vários outros métodos [Pineau et al. 2003].

3. Trabalhos Relacionados

Esta Seção apresenta alguns trabalhos que estão relacionados, ainda que de forma indireta, com a linha de pesquisa deste artigo. São apresentadas brevemente três abordagens relacionadas à aplicação de modelos POMDP em ambientes de tutoria inteligente.

No primeiro trabalho, [Theocharous et al. 2009] apresentaram um tutor inteligente que utiliza moedas virtuais para ensinar conceitos de ordenação. O sistema é modelado como um POMDP em que os estados são representados por variáveis físicas e as probabilidades de transição são definidas por variáveis sociais. O problema de dimensão

do espaço de estados é solucionado pela quebra do problema em soluções menores, cada qual representando a melhor forma de ensinar diferentes tipos de alunos.

No próximo trabalho, [Rafferty et al. 2011] demonstraram que o tempo de aprendizagem para a assimilação de conteúdos acadêmicos pode ser acelerado quando utilizadas técnicas baseadas em POMDP. No experimento realizado, os participantes foram divididos em dois grupos: um que recebia instruções baseadas em uma política fornecida por um modelo POMDP e um que recebia instruções aleatórias. Os autores concluíram que o tempo médio de ensino pode ser reduzido em aproximadamente 28% quando realizado por meio de um modelo POMDP.

Por último, [Zhang 2013] aborda estratégias de ensino para um STI que leciona um assunto composto por um conjunto de conceitos, os quais se relacionam entre si, formando uma espécie de cadeia de pré-requisitos. O autor compreende a estratégia de ensino como o ponto de partida pelo qual o tutor deverá iniciar o ensino, considerando o estado de conhecimento do aluno. Como esse estado não é totalmente observável pelo sistema tutor, os modelos POMDP são utilizados para auxiliar na tomada de decisão.

4. Modelo para a tomada de decisões pedagógicas

No presente tópico, serão discutidos cada um dos elementos que compõem o modelo POMDP $\mathcal{P} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma, b_0 \rangle$, com foco específico processo de ensino de algoritmos, em que tutor inteligente deve ser capaz de sequenciar um conjunto de problemas computacionais, $Q = \{q_1, q_2, q_3, \cdots, q_n\}$, de acordo com as interações dos alunos.

4.1. Espaço de Estados ($\mathcal{S}$)

Os estados de conhecimento serão codificados por meio de vetores $s = \{s_{q_1}, s_{q_2}, s_{q_3}, \cdots, s_{q_n}\}$, em que cada componente s_{q_i} pode assumir os valores 0 ou 1, para $1 \leq i \leq n$. O valor 0 denota que o aluno ainda não é capaz de responder a uma questão, enquanto que o valor 1 indica o contrário. Por exemplo, se estado inicial do aluno era $s_0 = \{0, 0, 0\}$ e ele foi capaz de acertar o problema q_1, é provável que o novo estado percebido pelo tutor seja $s_4 = \{1, 0, 0\}$.

4.2. Espaço de Ações ($\mathcal{A}$)

As ações indicam as recomendações de problemas computacionais realizadas pelo tutor durante o processo de ensino. Assim, deverá haver uma ação de recomendar, a_{q_i}, para cada questão $q_i \in Q$ existente na base de problemas.

4.3. Função de transição ($\mathcal{T}$)

Quando uma ação a_{q_i} é capturada em um estado $s \in \mathcal{S}$, existe uma probabilidade de transição para um novo estado s' que é expressa pela Equação 10, em que t indica um momento no tempo e s' é o novo estado no tempo $t + 1$:

$$\mathcal{T}\big(s' \mid s, a_{q_i}\big) = P\big(s_{t+1} = s' \mid s_t = s, a_t = a_{q_i}\big) \tag{10}$$

Assim, a função de transição será calculada pela seguinte expressão:

$$\mathcal{T}\big(s' | s, a_{q_i}\big) = \frac{\text{\# transições de } s \text{ para } s' \text{ quando } a_{q_i} \text{ é tomada em } s}{\text{\# } a_{q_i} \text{ é tomada em } s} \tag{11}$$

4.4. Função de Recompensa ($\mathcal{R}$)

O tutor conquista recompensas maiores quando um aluno é capaz de solucionar uma maior quantidade de problemas em um menor espaço de tempo. Ao solucionar um determinado problema, o estado de conhecimento do aluno transita de s_i para s_j, sendo distribuída uma recompensa no valor de $\mathcal{R}\big(s_j \mid \ s_i, a_{q_k}\big) = +100$. Por outro lado, quando o aluno submete uma resposta incorreta, fazendo com que o estado permaneça inalterado, o tutor recebe uma penalização no valor de $\mathcal{R}\big(s_i \mid \ s_i, a_{q_k}\big) = -50$.

4.5. Espaço de Observações (Ω)

As observações representam as percepções que o tutor consegue capturar das ações praticadas pelos alunos. Elas resultam do confronto das soluções submetidas pelos alunos aos problemas propostos pelo tutor contra uma bateria de casos de teste. Como consequência disso, o tutor poderá observar ou uma resposta correta, o_1, ou uma resposta incorreta o_2.

4.6. Função de Observação ($\mathcal{O}$)

Quando um estado s' é causado por uma ação a_{q_i}, existe uma probabilidade de se capturar uma observação o que é expressa pela Equação 12, em que t indica um momento no tempo e s' é o novo estado no tempo $t+1$:

$$\mathcal{O}\big(o \mid s', a_{q_i}\big) = P\big(o_{t+1} = o \mid s_{t+1} = s', a_t = a_{q_i}\big) \tag{12}$$

Assim, a função de observação será calculada pela seguinte expressão:

$$\mathcal{O}\big(o \mid s', a_{q_i}\big) = \frac{\text{\# } a_{q_i} \text{ é tomada, } s' \text{ é alcançado e } o \text{ é observado}}{\text{\# } a_{q_i} \text{ é tomada e } s' \text{ é alcançado}} \tag{13}$$

4.7. Fator de desconto (γ)

O fator de desconto será estipulado em 0.95, considerando um horizonte infinito de ações de ensino. Esse fator é responsável por determinar que as recompensas recebidas no futuro sejam descontadas de acordo com o tempo em que forem recebidas.

5. Experimento

Para a realização do experimento, utilizou-se a ferramenta PyPOMDP que possui módulos que permitem criar, implementar e executar modelos POMDP. Essa ferramenta possui suporte a uma gramática que facilita a representação de ambientes dinâmicos. Além disso, ela também oferece suporte ao algoritmo PBVI, cujo referencial teórico foi apresentado anteriormente. O propósito do experimento é aprofundar o entendimento quanto ao funcionamento interno do modelo POMDP, assim como em relação ao desempenho do algoritmo PBVI.

O experimento foi conduzido em três estágios: modelagem, simulação e análise de resultados. Para facilitar o entendimento, o estágio de modelagem levou em consideração apenas um conjunto de três questões, que resultaram em um total de oito estados. Além disso, foram necessárias três ações (uma para cada questão), e duas observações (resposta correta ou incorreta). As probabilidades de transição foram descritas na Figura 3, e as

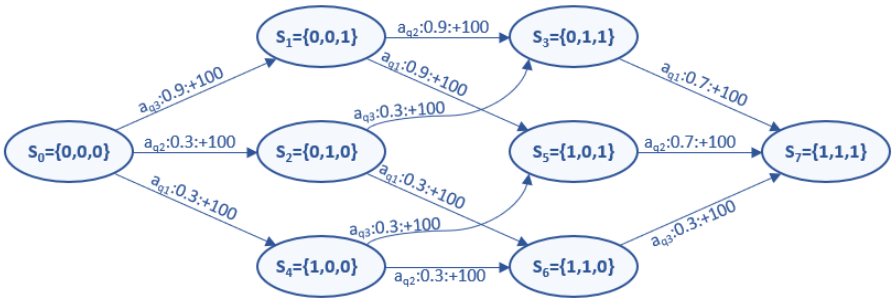

Figura 3. MDP para a resolução de três atividades sequenciais.

probabilidades de observação foram definidas de maneira uniforme. O estado de crença inicial foi estabelecido como sendo s_0, supondo-se que o aluno ainda não possui para responder nenhuma atividade.

Durante o estágio de simulação, o modelo proposto foi executado repetidas vezes para um horizonte de 3 etapas, até que houvesse a convergência do algoritmo PBVI. Ao longo das execuções, foram anotados o tempo total de execução e a recompensa descontada média para os seguintes processamentos: 1, 10, 100, 1000, 10000, 100000. Essa etapa também permitiu acompanhar a evolução dos estados de crença, em cada avanço no horizonte de planejamento.

Com relação aos resultados, os dados coletados apontaram que o algoritmo PBVI convergiu em apenas $1,021s$ para o modelo proposto, atingindo uma recompensa descontada média de $259,30$ (Fig. 4). Esse valor é compatível com o valor máximo esperado calculado de forma manual, $266,73$. Quanto ao tempo de processamento, constatou-se um aumento linear, proporcional ao número de simulações realizadas (Fig. 5).

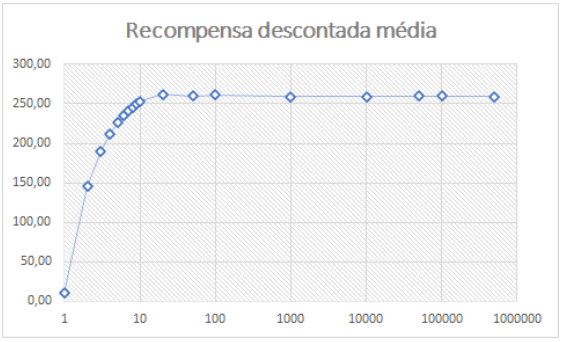

Figura 4. Recompensa descontada média

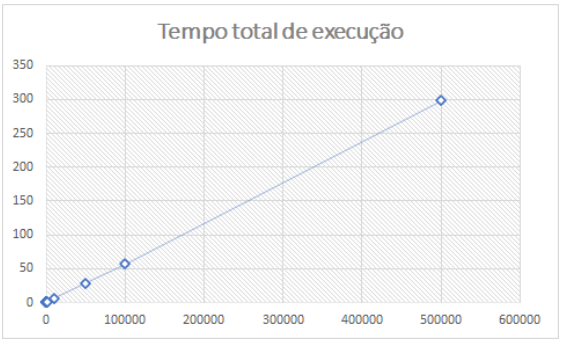

Figura 5. Tempo de Execução

No que diz respeito à evolução dos estados de crença, o experimento permitiu verificar que o algoritmo PBVI, para um total de 1000 execuções, foi capaz de convergir para estados de crença esperados, isto é, na direção dos caminhos $a_{q_3} \rightarrow a_{q_2} \rightarrow a_{q_1}$ e $a_{q_3} \rightarrow a_{q_1} \rightarrow a_{q_2}$, como mostram as distribuições de probabilidade das Figuras 6, 7, 8 e 9. O processo teve início no estado de crença inicial $b_0 = \{1, 0, 0, 0, 0, 0, 0, 0\}$. Após a primeira atualização, o estado de crença passou a ser $b = \{0.08, 0.14, 0.14, 0.01, 0.6, 0.01, 0.01, 0.01\}$, mostrando a crença de que o aluno adquiriu alguma habilidade, podendo agora estar nos estados s_1, s_2 e s_4.

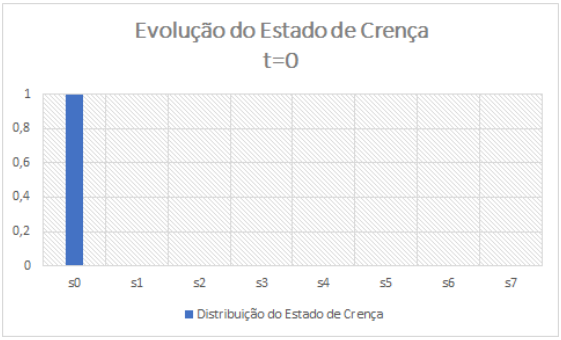

Figura 6. Estado de crença ($t = 0$)

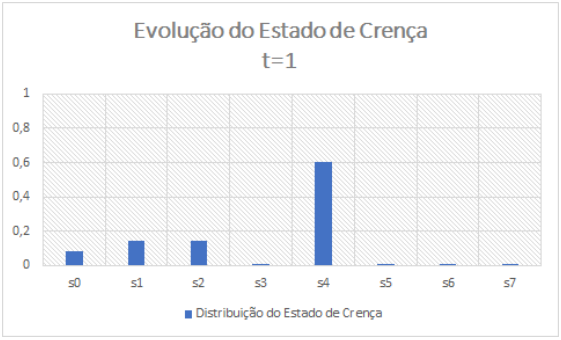

Figura 7. Estado de crença ($t = 1$)

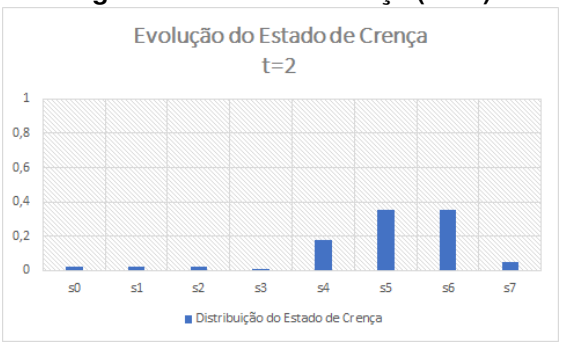

Figura 8. Estado de crença ($t = 2$)

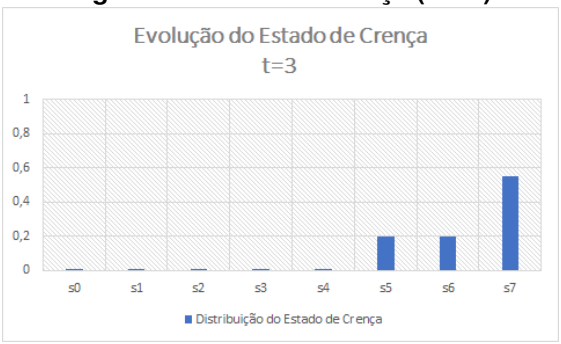

Figura 9. Estado de crença ($t = 3$)

6. Conclusão

O presente trabalho analisa, de forma qualitativa, o emprego de POMDP como modelo para a tomada de decisões pedagógicas em um ambiente voltado ao aprendizado de algoritmos. A questão enfrentada é o sequenciamento de problemas computacionais em ambientes de tutoria inteligente, tendo em vista que a proposição de problemas computacionais é uma prática comum em cursos introdutórios de algoritmos.

Neste contexto, o trabalho de organização das atividades de forma personalizada é encarado como um processo de tomada de decisão sequencial, em que um tutor inteligente é responsável por monitorar as soluções submetidas pelos alunos e por propor novas ações de ensino de forma sequencial, levando em consideração as respostas observadas anteriormente.

Assim, para analisar a aplicabilidade do modelo POMDP, simula-se o sequenciamento de um conjunto de três problemas computacionais, os quais são ofertados segundo probabilidades preestabelecidas. Os resultados mostram que a recompensa descontada média, para o algoritmo PBVI, converge rapidamente para um valor $(259, 30)$ que se aproxima do máximo esperado $(266, 73)$. Além disso, os estados de crença evoluem de forma natural para os caminhos que proporcionam as maiores recompensas.

A principal contribuição deste trabalho foi mostrar que o modelo POMDP pode ser aplicado de forma satisfatória ao processo de tomada de decisões pedagógicas, vez que se pode computar políticas de sequenciamento que proporcionam ganhos próximos do valor considerado ótimo. Em trabalhos futuros, a simulação será ampliada para um número maior de problemas computacionais, assim como para outros algoritmos, proporcionando uma base mais robusta para a comparação de resultados.

Referências

Bez, J. L., Tonin, N. A., and Rodegheri, P. R. (2014). Uri online judge academic: A tool for algorithms and programming classes. In *2014 9th International Conference on Computer Science Education*, pages 149–152.

Kolobov, A. (2013). *Scalable methods and expressive models for planning under uncertainty*. PhD thesis.

Lister, R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, J. E., Sanders, K., Seppälä, O., et al. (2004). A multi-national study of reading and tracing skills in novice programmers. *ACM SIGCSE Bulletin*, 36(4):119–150.

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B.-D., Laxer, C., Thomas, L., Utting, I., and Wilusz, T. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year cs students. In *Working group reports from ITiCSE on Innovation and technology in computer science education*, pages 125–180.

Pineau, J., Gordon, G., Thrun, S., et al. (2003). Point-based value iteration: An anytime algorithm for pomdps. In *IJCAI*, volume 3, pages 1025–1032.

Rafferty, A. N., Brunskill, E., Griffiths, T. L., and Shafto, P. (2011). Faster teaching by pomdp planning. In *International Conference on Artificial Intelligence in Education*, pages 280–287. Springer.

Revilla, M. A., Manzoor, S., and Liu, R. (2008). Competitive learning in informatics: The uva online judge experience. *Olympiads in Informatics*, 2(10):131–148.

Smallwood, R. D. and Sondik, E. J. (1973). The optimal control of partially observable markov processes over a finite horizon. *Operations research*, 21(5):1071–1088.

Somani, A., Ye, N., Hsu, D., and Lee, W. S. (2013). Despot: Online pomdp planning with regularization. In *Advances in neural information processing systems*, pages 1772–1780.

Ten Pas, A. (2012). Simulation based planning for partially observable markov decision processes with continuous observation spaces. Master's thesis, Faculty of Humanities and Sciences, Maastricht University.

Theocharous, G., Beckwith, R., Butko, N., and Philipose, M. (2009). Tractable pomdp planning algorithms for optimal teaching in "spais". In *IJCAI PAIR Workshop*.

Van Otterlo, M. and Wiering, M. (2012). Reinforcement learning and markov decision processes. In *Reinforcement Learning*, pages 3–42. Springer.

Wang, F. (2018). Reinforcement learning in a pomdp based intelligent tutoring system for optimizing teaching strategies. *International Journal of Information and Education Technology*, 8(8).

Woolf, B. P. (2010). *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. Morgan Kaufmann.

Zhang, P. (2013). *Using POMDP-based reinforcement learning for online optimization of teaching strategies in an intelligent tutoring system*. PhD thesis.