AutoML e transferência de aprendizado por reforço entre problemas de otimização combinatória

Gleice Kelly Barbosa Souza
RAI - Robotics and Artificial Intelligence
Centro de Ciências Exatas e Tecnológicas
Universidade Federal do Recôncavo da Bahia
Cruz das Almas - BA, Brasil
kelly.189@hotmail.com

André Luiz Carvalho Ottoni
RAI - Robotics and Artificial Intelligence
Centro de Ciências Exatas e Tecnológicas
Universidade Federal do Recôncavo da Bahia
Cruz das Almas - BA, Brasil
andre.ottoni@ufrb.edu.br

***Resumo* — O Aprendizado por Reforço (AR) é uma das linhas do Aprendizado de Máquina e que pode ser aplicado às mais diversas situações. A Otimização Combinatória, por exemplo, é uma relevante área de aplicação do AR. Destaca-se que, na literatura recente, tem ocorrido uma alta movimentação para realizar o desenvolvimento de sistemas de AR para resolução de problemas conhecidos de Otimização Combinatória, como o Problema do Caixeiro Viajante (PCV) e o *Sequential Ordering Problem* (SOP). No entanto, um desafio em aberto nessa área é a automatização da transferência de aprendizado entre o PCV e o SOP. Neste sentido, o objetivo deste trabalho foi propor um sistema de transferência de aprendizado com AutoML para otimização do SOP a partir de bases de conhecimento geradas com instâncias assimétricas do PCV. Essa abordagem foi denominada de Auto_TL_RL. Os resultados obtidos pelo Auto_TL_RL mostram que, para a maioria das instâncias analisadas (aproximadamente 78%), o sistema proposto alcançou os melhores resultados em comparação aos resultados obtidos sem transferência aprendizado.**

***Palavras-chaves* — Aprendizado por Reforço, AutoML, Sequential Ordering Problem, Problema do Caixeiro Viajante, Transferência de Aprendizado.**

I. Introdução

O Aprendizado por Reforço (AR) ou *Reinforcement Learning* (RL), é uma das vertentes do Aprendizado de Máquina, do inglês *Machine Learning* (ML) no qual, o agente aprende a medida que interage com o ambiente. Nesse sentido, o agente não sabe antecipadamente quais ações deve realizar e o mesmo deve aprender a partir de suas interações e retornos recebidos no ambiente [1]–[3]. Uma recente área de pesquisa estudada em conjunto com o AR é o AutoML ou *Automated Machine Learning*. Assim, quando AutoML é aplicado ao AR, a abordagem pode ser denominada de AutoRL (*Automated Reinforcement Learning*) [4]. O AutoRL é um sistema inteligente desenvolvido para selecionar as condições apropriadas para realizar o aprendizado por reforço antes do início do aprendizado [5]. A utilização do AutoRL visa reduzir a necessidade do conhecimento requerido por parte do usuário, bem como, reduzir o custo computacional requerido durante o aprendizado [5], [6]. Estas condições iniciais podem ser definidas através do *metalearning*, no qual o sistema vai utilizar suas experiências prévias na realização das atividades futuras [6]–[8].

Uma das possibilidades de aprimoramento de abordagens do AutoRL é com o uso da transferência de aprendizado ou *Transfer Learning* (TL). A TL, visa acelerar o aprendizado fornecendo autonomia ao sistema [9]. Dessa forma, o objetivo é transferir conhecimentos entre tarefas de domínios distintos e proporcionar diversos benefícios, como: melhoria no desempenho do agente da tarefa de destino, melhoria na recompensa total do agente e a redução do tempo necessário para realizar o aprendizado [9], [10]. A *Transfer Reinforcement Learning*, possui importantes aplicações em diversos trabalhos. Dentre eles, cita-se: Navegação autônoma [11], Agente autônomo multitarefas [12] e Sistemas multiagentes [13].

Nesse sentido, pode-se notar que é possível encontrar diversas situações onde a transferência de aprendizado por reforço é utilizada. No entanto, a literatura ainda carece de estudos e aplicações de TL automatizada entre problemas de otimização combinatória, que é um campo muito abrangente e que conta com diversas aplicações do AR [14]–[17].

Nesse aspecto, em [18], os autores realizam a transferência de aprendizado entre problemas da otimização combinatória. No estudo de [18], os mesmos realizam a transferência da matriz de aprendizado entre os problemas estudados, sendo eles: Problema do Caixeiro Viajante (PCV) e *Sequential Ordering Problem* (SOP). A abordagem adotada tem como objetivo gerar as matrizes de aprendizado utilizando problemas do PCV assimétrico que sejam relacionados ao SOP e reutilizar as matrizes de aprendizado geradas para o aprendizado do SOP. Neste sentido, os autores verificaram o impacto causado pela transferência de aprendizado do PCV para o SOP na distância final obtida e no tempo computacional requerido durante o aprendizado do SOP. No entanto, a transferência de aprendizado proposta por [18] não foi automatizada em um sistema AutoRL.

O trabalho de [19] realiza a implementação de um sistema de aprendizado de máquina automatizado para o problema do caixeiro viajante, tanto simétrico quanto assimétrico. No

estudo realizado por [19], os autores visam otimizar os parâmetros aplicados durante o processo de aprendizagem realizado com o aprendizado por reforço utilizando a metodologia de superfície de resposta. Nesse sentido, são ajustados os valores da taxa de aprendizado e do fator de desconto, em seguida, os parâmetros ajustados são adotados de forma automatizada para o problema estudado. Assim, os autores buscavam analisar os impactos nos resultados de distância final através do ajuste dos parâmetros a cada instância específica analisada. Entretanto, os autores de [19] não investigaram a transferência de aprendizado automatizada entre os problemas de otimização combinatória.

Dessa forma, a principal contribuição deste trabalho é a proposta da abordagem Auto_TL_RL (*Automated - Transfer Learning - Reinforcement Learning*). Esse algoritmo de AutoRL possibilita a realização da transferência de aprendizado automatizada entre dois problemas da otimização combinatória tradicionais na literatura: PCV e SOP. Neste sentido, o sistema proposto realiza a transferência de aprendizado a partir do domínio fonte (Problema do Caixeiro Viajante) para o domínio objetivo (*Sequencial Ordering Problema*). Para isso, são utilizadas instâncias da biblioteca TSPLIB [20].

Este artigo está subdividido em seções. A Seção 2 apresenta os conceitos teóricos deste trabalho. A Seção 3 descreve a metodologia adotada neste trabalho. Já na Seção 4, são discutidos os resultados. Por fim, na Seção 5, são apresentadas as conclusões.

II. Fundamentação Teórica

A. Aprendizado por Reforço

O Aprendizado por Reforço (AR) baseia-se no método de tentativas e erros [21]. Dessa forma, o agente deve se adaptar ao ambiente em que foi colocado. Para isto, a partir de suas percepções do ambiente o agente irá realizar ações [22]. Neste sentido, a cada ação executada, o agente irá receber reforços que irão indicar o sucesso ou o fracasso da ação realizada [23]. Assim, o agente tomador de decisões aprende sobre o ambiente em que está inserido, fazendo com que o mesmo procure executar ações que irão lhe fornecer as melhores recompensas [3], [24].

1) Algoritmo SARSA: O SARSA é um dos algoritmos mais conhecidos do AR. Este algoritmo trata-se de uma modificação do algoritmo Q-learning [25] e seu nome foi atribuído por [26]. O SARSA utiliza o método de aprendizagem Q. Na aprendizagem Q, o agente aprende uma função de ação-valor, esta função irá fornecer ao agente um reforço para cada ação tomada em um determinado estado. Esta função pode ser representada por *Q(s,a)* [24].

O algoritmo SARSA de forma procedural foi exibido em [27] e é apresentado no Algoritmo 1.

Neste algoritmo, inicialmente são definidos os parâmetros (linha 1). Em seguida, ocorre a inicialização da matriz Q (linha 2), definição do estado inicial (linha 3) e da ação inicial (linha 4). Na sequência, inicia-se um *loop* no qual a ação é executada (linha 6), uma recompensa é selecionada (linha 7), um novo estado e uma nova ação são definidos (linhas 8 e 9). Por fim, o estado atual e a ação atual são atualizados (linhas 11 e 12) com o estado e ação definidos anteriormente. Este ciclo será executado até o critério de parada ser satisfeito.

Algoritmo 1: Algoritmo SARSA

1 Defina os parâmetros: α, γ e ϵ
2 Em cada ***s,a*** faça ***Q(s,a) = 0***
3 Observe o estado ***s***
4 Selecione a ação ***a*** utilizando a política ϵ-greedy
5 **repita**
6 Execute a ação ***a***
7 Receba a recompensa imediata ***r(s,a)***
8 Observe o novo estado ***s'***
9 Selecione a ação ***a*** utilizando a política ϵ-greedy
10 $Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r(s_t, a_t) + \gamma Q(s', a') - Q(s_t, a_t)]$
11 ***s = s'***
12 ***a = a'***
13 **até** *o critério de parada ser satisfeito*;

A matriz de aprendizado do algoritmo SARSA é atualizada de acordo com a Equação (1) [24]:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r(s_t, a_t) + \gamma Q(s', a') - Q(s_t, a_t)], \quad (1)$$

em que,

- Q é a matriz de aprendizado;
- $r(s_t, a_t)$ é o reforço recebido por realizar a ação a_t no estado s_t;
- α é a taxa de aprendizado;
- γ é o fator de desconto;
- s_t é o estado atual;
- a_t é a ação realizada no estado atual;
- s' é o estado futuro;
- a' é a ação que será realizada no estado futuro.

B. Otimização Combinatória

Os problemas de Otimização Combinatória tem como objetivo maximizar ou minimizar uma função objetivo que é definida em um determinado domínio finito que classifica a solução do problema como ótima [21], [28]. Sucintamente, estes problemas podem ser caracterizados da seguinte maneira [21], [29], [30]:

- Variáveis de decisão: são os critérios que serão manipulados em busca da solução ótima.
- Função objetivo: função que contém as variáveis de decisão que serão alteradas durante a busca pela melhor solução para o problema. Vale salientar que todo problema de otimização combinatória possui no mínimo uma função objetivo.
- Restrições: são condições impostas ao problema para assegurar que a solução encontrada seja viável. É importante ressaltar que não é obrigatório que um problema possua restrições, neste caso, considera-se que todas as soluções são viáveis.

A literatura contém diversos problemas de otimização, dentre eles [1], [28], [30]:

- Problema do Caixeiro Viajante;
- Problema da Mochila;
- Problema da Atribuição Quadrática.

C. Problema do Caixeiro Viajante

O Problema do Caixeiro Viajante (PCV) é um dos problemas mais conhecidos dentre os problemas de otimização combinatória [31], [32]. Este problema tem como objetivo determinar um caminho hamiltoniano que passe por todos os vértices de um grafo. Além disso, o problema tem ainda como objetivo, que o caminho hamiltoniano encontrado seja o de menor custo [28], [31], [33].

O PCV é composto por um conjunto de cidades, onde o caixeiro deve visitar todas as cidades do conjunto e ao final do percurso o caixeiro deve retornar para a cidade inicial [32]. Como restrição, cada cidade do conjunto deve ser visitada apenas uma vez, com exceção da cidade final que deve ser a mesma cidade de início [31]–[33].

O PCV pode ser classificado de diversas maneiras, dentre elas, como PCV simétrico ou PCV assimétrico. No PCV simétrico, o custo de se deslocar do nó A para o nó B é o mesmo de realizar o deslocamento do nó B para o nó A. Já no caso do PCV assimétrico, o custo varia de acordo com o sentido de deslocamento adotado [31], [34]. Dessa forma, o PCV simétrico pode ser representado por um grafo com arestas bidirecionais, enquanto que o PCV assimétrico pode ser representado por um grafo com arestas direcionais [31].

O problema do caixeiro viajante possui diversas aplicações, [28] e [33] apresentam algumas delas:

- Sequenciamento de tarefas;
- Perfuração de placas de circuito impresso;
- Análise de estruturas de cristais;
- Manipulação de itens em estoque;
- Otimização do movimento de ferramentas de corte;
- Roteamento de entrega postal.

1) Formulação: Existem diversas formulações possíveis para o problema do caixeiro viajante, algumas delas são apresentadas em [35] e em [28]. A formulação apresentada a seguir foi apresentada por [35], conforme Equações (2) a (6):

$$Minimizar \sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}x_{ij} \tag{2}$$

sujeito a:

$$\sum_{i=1}^{N} x_{ij} = 1 \qquad (\forall j = 1, ..., N) \tag{3}$$

$$\sum_{j=1}^{N} x_{ij} = 1 \qquad (\forall i = 1, ..., N) \tag{4}$$

$$x_{ij} \in \{0, 1\} \qquad (\forall i, j = 1, ..., N) \tag{5}$$

$$X = x_{ij} \in S \qquad (\forall i, j = 1, ..., N) \tag{6}$$

A função objetivo do PCV é representada pela Equação (2). Na qual, c_{ij} representa o custo entre as cidades *i* e *j*. x_{ij} representa se o arco (*i*, *j*) faz parte da solução do problema. Caso x_{ij} seja igual a 1, o arco faz parte da solução. Caso x_{ij} seja igual a 0, o arco não pertence a solução do problema. Além disso, as Equações (3) e (4) asseguram que cada cidade será visitada somente uma vez. Já a Equação (5), representa a restrição de que o valor de x_{ij} será sempre binário. Finalmente, a Equação (6) garante que não serão formadas sub-rotas na solução do problema [32], [33], [35], [36].

D. Sequential Ordering Problem

O *Sequential Ordering Problem* (SOP) é uma variação do Problema do Caixeiro Viajante assimétrico, com a adição de restrições de precedência [37], [38]. Dessa forma, assim como no PCV assimétrico, o SOP tem como objetivo encontrar um caminho hamiltoniano de custo mínimo em um grafo direcionado [37].

No SOP, c_{ij} também representa o custo do arco. Quando $c_{ij} \geq 0$, este valor irá representar o custo de se deslocar da cidade *i* para a cidade *j*. Quando c_{ij} = -1, este valor indicará que há uma restrição de precedência. Neste caso, a cidade *j* deve preceder a cidade *i* [37].

No estudo de [17], os autores apresentam uma formulação para o SOP baseada na formulação do PCV de [35]. A qual acrescenta a seguinte restrição na formulação do PCV apresentada anteriormente, conforme Equação (7):

$$c_{ij} \geq 0 \vee c_{ij} = -1 \wedge c_{ji} \geq 0 (\forall i, j = 1, \cdots, N) \tag{7}$$

Esta nova restrição visa satisfazer a restrição de ordem de precedência existente no SOP.

E. TSPLIB

A TSPLIB[1] (*Traveling Salesman Problem Library*) é uma base de dados aberta que fornece informações de diversos problemas de otimização [1], [20]. Dentre estes dados, estão os dados de problemas como [20]:

- *Symmetric traveling salesman problem* (TSP);
- *Asymmetric traveling salesman problem* (ATSP);
- *Sequential ordering problem* (SOP).

Além disso, vale salientar que além dos dados, a biblioteca fornece ainda a solução ótima conhecida para cada problema disponibilizado [20].

F. AutoML e Transferência de Aprendizado

O AutoML (*Automated Machine Learning*) foi desenvolvido com o objetivo principal de reduzir o esforço humano requerido durante o ajuste das configurações do aprendizado [39]. Dessa forma, o AutoML é uma técnica utilizada para definir parâmetros e algoritmos que serão adotados durante a

[1]http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

aprendizagem com o objetivo de obter melhores resultados, tendo em vista que as configurações são realizadas de acordo com o problema a ser estudado [40], [41].

Neste sentido, devido às melhorias observadas pelo uso do AutoML e pela crescente necessidade de sistemas de aprendizado cada vez mais robustos a fim de conseguir atender as grandes quantidades de dados que surgem constantemente, esta técnica vem sendo utilizada também para automatizar outras etapas do processo de aprendizagem [39], [41], [42].

O meta-aprendizado é um dos tópicos relacionados ao AutoML e refere-se a capacidade do sistema aprender a aprender [43]. Nesse sentido, o objetivo do meta-aprendizado é reaproveitar as experiências prévias em tarefas que serão realizadas futuramente [7], [43]. Dessa forma, o sistema não precisa realizar o aprendizado do zero e pode adaptar-se a situação atual a partir das suas experiências prévias [44]. Logo, ocorrerá uma redução dos esforços empregados na realização das próximas tarefas se comparado ao esforço utilizado para executar as tarefas anteriores [6], [45]. Vale salientar que, quanto maior for a semelhança entre as tarefas (as tarefas já realizadas e as tarefas a serem realizadas), maiores são as chances de obtenção de bons resultados com a reutilização de conhecimentos obtidos em tarefas anteriores [8], [45].

A transferência de aprendizado ou *Transfer Learning*, refere-se a transferência de conhecimentos entre problemas semelhantes. Esta abordagem torna possível que ocorra uma redução do tempo gasto durante o treinamento do problema objetivo, uma melhor aprendizagem e consequentemente, melhores resultados finais [46], [47].

Em [10], os autores discutem sobre a transferência de aprendizado entre problemas do AR. Os autores ressaltam sobre a transferência direta da matriz de aprendizado Q entre tarefas, em que, a matriz de aprendizado da tarefa de destino é inicializada com os valores advindos da matriz de aprendizado da tarefa de origem. Essa transferência entre domínios pode ser representada pela Equação [10]:

$$\forall s, \forall a, Q_{objetivo}(s,a) = Q_{fonte}(s,a) \tag{8}$$

III. Metodologia

O desenvolvimento deste trabalho foi guiado pela metodologia que consistiu nas seguintes etapas:

1) Desenvolvimento do sistema de AutoRL para transferência de aprendizado;
2) Realização dos experimentos;
3) Análise dos resultados obtidos.

A. Modelagem do Sistema de Aprendizado por Reforço

A modelagem do sistema de AR adotada neste trabalho foi inspirada em [18] e [19], a qual considera que:

- Estados: são os vértices da instância;
- Ações: são os vértices a serem visitados;
- Reforço: retorno dado ao agente ao executar uma ação, o reforço segue a regra apresentada na Equação (9):

$$R = -c_{ij}, \tag{9}$$

em que, *R* é o reforço e c_{ij} refere-se ao deslocamento do nó *i* para o nó *j*. Ressalta-se que, para atender as restrições de precedência do SOP, utilizou-se a mesma abordagem de [18], o qual propõe o algoritmo RLSOP que verifica se o nó de destino possui alguma restrição de precedência. Caso exista restrição, um novo nó de destino é selecionado e a verificação é realizada novamente.

B. Algoritmo de Auto_TL_RL Proposto

O fluxograma apresentado na Figura 1, apresenta o fluxo de funcionamento do algoritmo proposto. Este algoritmo foi denominado de Auto_TL_RL. Inicialmente, uma instância do SOP é selecionada e caso já exista uma base de dados o SOP é executado. Caso contrário, a base de dados é gerada a partir de uma instância do ATSP, essa base de dados é armazenada e por fim, o SOP é executado.

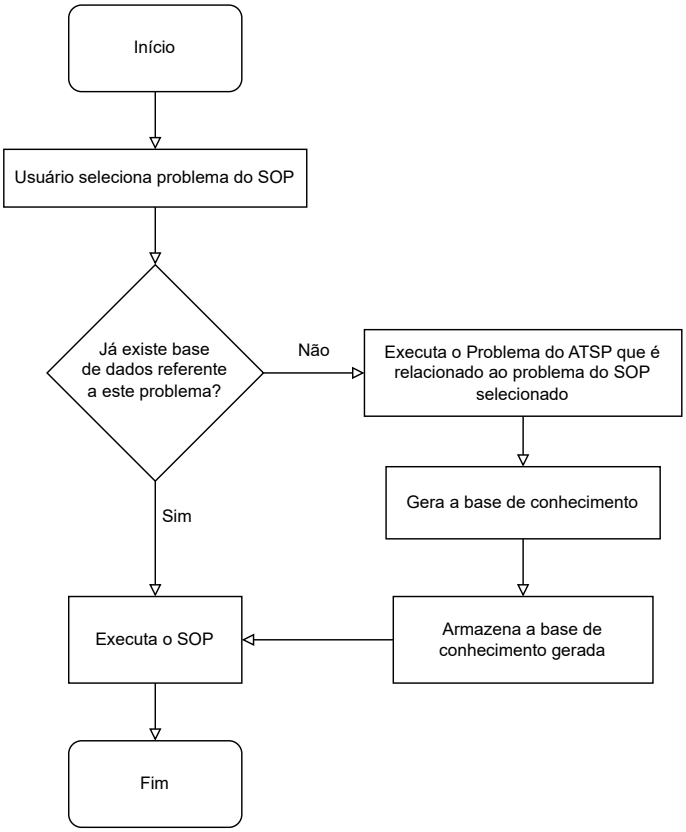

Figura 1. Fluxograma que representa o funcionamento do algoritmo de Auto_TL_RL proposto.

Já o Algoritmo 2, representa de maneira procedural o algoritmo de Auto_TL_RL proposto, o qual foi desenvolvido utilizando a linguagem R. Vale destacar que o algoritmo proposto utiliza o algoritmo SARSA para execução do PCV assimétrico e do SOP.

Inicialmente, o usuário deve selecionar qual instância do SOP será executada (linha 1) e a partir disto o tamanho da instância é extraído (linha 2). Na sequência, deve ser definido qual o valor do parâmetro ϵ-greedy e a quantidade de episódios que serão executados durante o aprendizado (linhas 3 e 4). Em

Algoritmo 2: Algoritmo Auto_TL_RL

```
1  Defina a instância do SOP que será executada
2  Extraia o tamanho da instância do SOP
3  Defina o parâmetro: ϵ
4  Defina o número de episódios: 1000
                    Etapa 1
5  Verificação de existência da base de conhecimento
                    Etapa 2
6  Defina a instância do ATSP que será executada
7  Extraia o tamanho da instância do ATSP
8  Defina os parâmetros: α e γ
9  Defina o número de épocas: 5
10 para cada α_t em α faça
11     para cada γ_t em γ faça
12         para cada epoca ate numeroEpocas faça
13             SARSA(α, γ, tamanhoDaInstancia)
14         fim
15     fim
16 fim
17 Armazena a base de dados gerada
                    Etapa 3
18 Defina os parâmetros: α e γ
19 Defina o número de épocas: 10
20 para cada epoca ate numeroEpocas faça
21     SARSA(α, γ, tamanhoDaInstancia)
22 fim
```

seguida, a execução do algoritmo é dividida em três etapas. Na primeira etapa, é realizada uma verificação de existência da base de dados relacionada ao problema selecionado (linha 5). Já a segunda etapa, refere-se a etapa de geração da base de conhecimentos (linhas 6 à 17). Por fim, na terceira etapa, é onde ocorrerá a execução do SOP (linhas 18 à 22).

A partir dos resultados obtidos na primeira etapa, pode-se seguir dois caminhos. A primeira opção será utilizada em caso de já existência de uma base de dados relacionada a instância do SOP selecionada. Nesse caso, a segunda etapa não é executada e o algoritmo avança direto para terceira etapa utilizando a base de conhecimento já existente. Nesta etapa, inicialmente, define-se os valores dos parâmetros de taxa de aprendizado e fator de desconto (linha 18) e o número de épocas (linha 19). Por fim, o SOP é executado utilizando o algoritmo SARSA (linhas 20 à 22).

Já o segundo caminho, será adotado quando não houver uma base de dados preexistente para a instância do SOP selecionada. Assim, primeiro será executada a segunda etapa, na qual inicialmente deverá ser definida uma instância do ATSP e o seu tamanho é extraído (linhas 6 e 7). Além disso, define-se os valores dos parâmetros de taxa de aprendizado e fator de desconto (linha 8) e o número de épocas (linha 9). Na sequência, a base de dados é gerada utilizando a instância do ATSP selecionada (linhas 10 à 16) e ao final da execução a base gerada é armazenada para ser reutilizada em aplicações futuras (linha 17). Por fim, a terceira etapa finalmente é executada utilizando a base de dados que foi gerada na etapa anterior.

C. Experimentos

Os experimentos foram realizados adotando instâncias do SOP e do ATSP disponibilizadas pela biblioteca TSPLIB. Ressalta-se que, para permitir a transferência de aprendizado, foi necessário considerar instâncias SOP que são derivadas dos problemas ATSP originais da TSPLIB. Assim, as instâncias ATSP e SOP selecionadas podem ser vistas nas Tabelas I e II, respectivamente.

Tabela I
INSTÂNCIAS ATSP.

Problema	Nós	Ótimo
br17	17	39
p43	43	5620
ry48p	48	14422
ft53	53	6905
ft70	70	38673

Tabela II
INSTÂNCIAS SOP.

Problema	Nós	Ótimo
br17.10	18	55
br17.12	18	55
p43.1	44	28140
p43.2	44	28480
p43.3	44	28835
p43.4	44	83005
ry48p.1	49	14422
ry48p.2	49	16074
ry48p.3	49	19490
ry48p.4	49	31446
ft53.1	54	7531
ft53.2	54	8026
ft53.3	54	10262
ft53.4	54	14425
ft70.1	70	39313
ft70.2	70	40101
ft70.3	70	42535
ft70.4	70	53530

Além disso, foram definidos ainda quais os parâmetros, a quantidade de épocas e a quantidade de episódios seriam utilizados no algoritmo de Auto_TL_RL proposto (Algoritmo 2) durante os experimentos. Tratando-se das configurações da segunda etapa, estas são as mesmas apresentadas em [19] e podem ser conferidas na Tabela III. Quanto às condições de aprendizagem da etapa final, seguiu-se a mesma estrutura. Entretanto, algumas adaptações foram realizadas para uma melhor adequação aos cenários de experimento, as quais também estão apresentadas na Tabela III.

IV. RESULTADOS

Nesta seção, serão apresentados os resultados obtidos com este trabalho.

Tabela III
CONFIGURAÇÕES UTILIZADAS DURANTE O EXPERIMENTO.

Parâmetro	**Segunda etapa**		**Etapa final**	
	Quantidade	**Valores**	**Quantidade**	**Valores**
α	8	0,01; 0,15; 0,30; 0,45; 0,60; 0,75; 0,90; 0,99	1	0,75
γ	8	0,01; 0,15; 0,30; 0,45; 0,60; 0,75; 0,90; 0,99	1	0,15
ϵ	1	0,01	1	0,01
Combinações	$8 \times 8 \times 1 = 64$	-	$1 \times 1 \times 1 = 1$	-
Épocas por Combinação	5	-	10	-
Episódios por Época	1000	-	1000	-
Episódios por Combinação	$5 \times 1000 = 5000$	-	$10 \times 1000 = 10000$	-
Total de Épocas	$5 \times 64 = 320$	-	10	-
Total de Episódios	$1000 \times 320 = 320000$	-	$10 \times 1000 = 10000$	-

A. Resultados do Auto_TL_RL

Inicialmente, para fins de comparação, as instâncias selecionadas do SOP foram executadas sem utilizar a transferência de aprendizado. Na sequência, as bases de conhecimento foram geradas utilizando as instâncias ATSP no momento da execução dos problemas br17.10, p43.1, ry48p.1, ft53.1 e ft70.1. Dessa forma, no momento da execução destas instâncias, todas as etapas do algoritmo proposto foram executadas.

Para as demais instâncias, como já havia uma base de conhecimento preexistente, somente a primeira e a terceira etapa do algoritmo foram executadas. A Tabela IV, apresenta todos os resultados obtidos durante a execução dos experimentos.

Tabela IV
RESULTADOS DA EXECUÇÃO DAS INSTÂNCIAS SOP SEM O ALGORITMO DE AUTO_TL_RL E COM O ALGORITMO DE AUTO_TL_RL.

ATSP	**SOP**	**Ótimo**	**Sem Auto_TL_RL**	**Com Auto_TL_RL**
br17	br17.10	55	57	**55**
	br17.12	55	**57**	**57**
p43	p43.1	28140	28765	**28715**
	p43.2	28480	29265	**29170**
	p43.3	28835	29545	**29535**
	p43.4	83005	84110	**83985**
ry48p	ry48p.1	14422	18154	**17922**
	ry48p.2	16074	18549	**18459**
	ry48p.3	19490	**22789**	22853
	ry48p.4	31446	38235	**37679**
ft53	ft53.1	7531	**8852**	9056
	ft53.2	8026	9839	**9588**
	ft53.3	10262	12598	**12594**
	ft53.4	14425	17650	**16935**
ft70	ft70.1	39313	43460	**42707**
	ft70.2	40101	44841	**44499**
	ft70.3	42535	**48015**	48311
	ft70.4	53530	60049	**59275**

Com base nos resultados apresentados na Tabela IV, é possível observar que a distância final alcançada pelo sistema de transferência de aprendizado proposto apresentou melhores resultados em várias das instâncias testadas, mais precisamente em 14 das 18 instâncias. Dentre estes resultados, destaca-se os valores de distância final obtidos nas instâncias ft70.4, ft70.1, ft53.4 e ry48p.4, sendo que nestas quatro instâncias houve uma diferença superior a 500 unidades nos valores obtidos ao realizar o aprendizado com o sistema de Auto_TL_RL e sem o sistema de Auto_TL_RL. Com isto, pode-se notar que ao utilizar o sistema de Auto_TL_RL proposto foi possível obter valores de distância final mais próximos aos apresentados como ótimo pela TSPLIB, se comparados aos resultados obtidos quando não foi utilizado o sistema de Auto_TL_RL.

B. Comparação com outros trabalhos

A Tabela V apresenta uma comparação deste trabalho com outros dois estudos recentes da literatura. Nesta, destaca-se a principal contribuição deste trabalho, ou seja, a proposta de automatização da transferência de aprendizado por reforço entre PCV e SOP com AutoML. Nota-se que diferente dos dois trabalhos anteriores analisados, foram abordadas as duas técnicas descritas (transferência de aprendizado e aprendizado automatizado) para realizar o estudo no problema do SOP. Já no que se refere aos algoritmos, é possível notar que todos os trabalhos utilizaram algoritmos clássicos do Aprendizado por Reforço. Não menos importante, destaca-se ainda que todos os trabalhos utilizaram a biblioteca TSPLIB como fonte de dados para os estudos, variando somente os problemas analisados.

Tabela V
TABELA DE COMPARAÇÕES ENTRE O TRABALHO PROPOSTO, O TRABALHO APRESENTADO EM [19] (I) E O TRABALHO APRESENTADO POR [18] (II).

		Proposto	**I**	**II**
Técnicas	Transfer Learning	✓	-	✓
	AutoML	✓	✓	-
Algoritmos	SARSA	✓	-	✓
	Q-learning	-	✓	-
Biblioteca	TSPLIB	✓	✓	✓
Problemas	SOP	✓	-	✓
	TSP	-	✓	-
	ATSP	✓	✓	✓

V. CONCLUSÃO

Este trabalho teve como objetivo propor um sistema de transferência de aprendizado automatizado utilizando Aprendizado por Reforço para ser aplicado no *Sequential Ordering Problem*. O sistema proposto, conta com uma funcionalidade que possibilita a identificação da presença ou ausência de uma base de conhecimento para um determinado problema. Quando a base já existe, o problema é executado diretamente. Porém, quando uma base de conhecimento prévia não está disponível, uma base de conhecimento é gerada e armazenada para

possíveis reutilizações em situações futuras. Para isto, utilizou-se conceitos discutidos em [18] e [19], como transferência de aprendizado e aprendizado automatizado.

A transferência de aprendizado realizada pelo algoritmo Auto_TL_RL, proporcionou o obtenção de melhores resultados em 14 das 18 instâncias da TSPLIB analisadas. O que demonstra a sua eficiência e importância se comparado aos métodos tradicionais de aprendizado. Além disso, é importante ressaltar ainda sobre a importância da transferência automatizada, dado que todo o esforço humano e computacional requerido para realizar este estudo foi reduzido por conta da utilização de um sistema de aprendizado automatizado.

Em trabalhos futuros, tem-se o objetivo de conduzir o experimento com diferentes instâncias. Ademais, pretende-se aprimorar o sistema para viabilizar a aplicação dos parâmetros empregados durante a etapa de geração da base de conhecimento que demonstraram os melhores resultados.

VI. Agradecimentos

Os autores agradecem ao apoio da Universidade Federal do Recôncavo da Bahia.

Referências

[1] R. A. d. C. Bianchi, "Uso de heurísticas para a aceleração do aprendizado por reforço." Ph.D. dissertation, Universidade de São Paulo, 2004.

[2] J. P. Q. d. Santos, J. D. de Melo, A. D. D. Neto, and D. Aloise, "Reactive search strategies using reinforcement learning, local search algorithms and variable neighborhood search," *Expert Systems with Applications*, vol. 41, no. 10, pp. 4939–4949, 2014.

[3] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[4] J. Parker-Holder, R. Rajan, X. Song, A. Biedenkapp, Y. Miao, T. Eimer, B. Zhang, V. Nguyen, R. Calandra, A. Faust, F. Hutter, and M. Lindauer, "Automated Reinforcement Learning (AutoRL): A Survey and Open Problems," *Journal of Artificial Intelligence Research*, vol. 74, pp. 517–568, June 2022.

[5] R. R. Afshar, Y. Zhang, J. Vanschoren, and U. Kaymak, "Automated reinforcement learning: An overview," 2022.

[6] P. Brazdil, C. G. Carrier, C. Soares, and R. Vilalta, *Metalearning: Applications to data mining*. Springer Science & Business Media, 2008.

[7] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, vol. 28, pp. 2962–2970.

[8] L. Tuggener, M. Amirian, K. Rombach, S. Lorwald, A. Varlet, C. Westermann, and T. Stadelmann, "Automated Machine Learning in Practice: State of the Art and Recent Results," in *2019 6th Swiss Conference on Data Science (SDS)*, June 2019, pp. 31–36.

[9] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *Journal of Machine Learning Research*, vol. 10, no. 56, pp. 1633–1685, 2009.

[10] J. L. Carroll and T. S. Peterson, "Fixed vs. dynamic subtransfer in reinforcement learning," in *International Conference on Machine Learning and Applications*, 2002.

[11] A. Balakrishnan, J. Lee, A. Gaurav, K. Czarnecki, and S. Sedwards, "Transfer reinforcement learning for autonomous driving," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 31, pp. 1–26, 2021.

[12] E. Parisotto, J. L. Ba, and R. Salakhutdinov, "Actor-mimic: Deep multitask and transfer reinforcement learning," 2016.

[13] Y. Hou, Y.-S. Ong, L. Feng, and J. M. Zurada, "An evolutionary transfer reinforcement learning framework for multiagent systems," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 601–615, 2017.

[14] J. P. Q. d. Santos, "Uma implementação paralela híbrida para o problema do caixeiro viajante usando algoritmos genéticos, grasp e aprendizagem por reforço," Master's thesis, Universidade Federal do Rio Grande do Norte, 2009.

[15] A. L. C. Ottoni, E. G. Nepomuceno, and M. S. d. Oliveira, "Análise do desempenho do aprendizado por reforço na solução do problema da mochila multidimensional," *Revista Brasileira de Computação Aplicada*, vol. 9, no. 3, pp. 56–70, 2017.

[16] M. M. Alipour, S. N. Razavi, M. R. Feizi Derakhshi, and M. A. Balafar, "A hybrid algorithm using a genetic algorithm and multiagent reinforcement learning heuristic to solve the traveling salesman problem," *Neural Computing and Applications*, vol. 30, no. 9, p. 2935–2951, Nov. 2018.

[17] A. L. C. Ottoni, E. G. Nepomuceno, M. S. de Oliveira, and D. C. R. de Oliveira, "Tuning of Reinforcement Learning Parameters Applied to SOP Using the Scott–Knott Method," *Soft Computing*, vol. 24, no. 6, p. 4441–4453, Mar. 2020.

[18] A. L. C. Ottoni, M. S. de Oliveira, D. C. R. de Oliveira, and E. G. Nepomuceno, "Transferência de aprendizado por reforço em problemas de otimização combinatória," in *Anais do Congresso Brasileiro de Automática 2020*. SBA, Dec. 2020.

[19] G. K. B. Souza and A. L. C. Ottoni, "AutoRL-TSP-RSM: sistema de aprendizado por reforço automatizado com metodologia de superfície de resposta para o problema do caixeiro viajante," *Revista Brasileira de Computação Aplicada*, vol. 13, no. 3, pp. 86–100, Nov. 2021.

[20] G. Reinelt, "Tsplib95," *University Heidelberg*, 1995.

[21] J. P. Q. d. Santos, "Estratégias de busca reativa utilizando aprendizagem por reforço e algoritmos de busca local," Ph.D. dissertation, UFRN, 2014.

[22] A. L. C. Bazzan, "Contribuições de aprendizado por reforço em escolha de rota e controle semafórico," *Estudos Avançados*, vol. 35, no. 101, pp. 95–110, Abr. 2021.

[23] P. Bartmeyer, A. A. S. Leão, F. Toledo, and L. T. de Oliveira, "Aprendizado por reforço aplicado ao problema de empacotamento de peças irregulares em faixas," in *Anais do Simpósio Brasileiro de Pesquisa Operacional, 2021, João Pessoa*, 2021, anais eletrônicos... Campinas, Galoá, 2021.

[24] S. J. Russell and P. Norvig, *Inteligência artificial*. Campus, 2013.

[25] A. L. Ottoni, E. Nepomuceno, M. Oliveira, L. Cordeiro, and R. Lamperti, "Análise da influência da taxa de aprendizado e do fator de desconto sobre o desempenho dos algoritmos q-learning e sarsa: aplicação do aprendizado por reforço na navegação autônoma," *Revista Brasileira de Computação Aplicada*, vol. 8, no. 2, pp. 44–59, Set. 2016.

[26] R. S. Sutton, "Generalization in reinforcement learning: Successful examples using sparse coarse coding," in *Advances in neural information processing systems*, 1996, pp. 1038–1044.

[27] A. L. C. Ottoni, E. G. Nepomuceno, L. B. Felix, and M. S. d. Oliveira, "Modelo híbrido de aprendizado por reforço e lógica fuzzy aplicado ao problema do caixeiro viajante com reabastecimento," in *XXI Congresso Brasileiro de Automática*, vol. 21, 2016.

[28] M. C. Goldbarg and H. P. L. Luna, *Otimização combinatória e programação linear: modelos e algoritmos*, Elsevier, Ed. Elsevier, 2005.

[29] C. P. d. Almeida, "Transgenética computacional aplicada a problemas de otimização combinatória com múltiplos objetivos," Ph.D. dissertation, Universidade Tecnológica Federal do Paraná, 2012.

[30] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: a methodological tour d'horizon," 2018.

[31] O. R. Pedro, "Uma abordagem de busca tabu para o problema do caixeiro viajante com coleta de prêmios," Master's thesis, Universidade Federal de Minas Gerais, 2013.

[32] S. A. de Araujo, D. A. de Carvalho, E. dos Santos Teixeira, M. Scalabrin, and G. W. Carlos, "Problema do caixeiro viajante hierárquico: Uma aplicação no controle de formigas em uma indústria de papel e celulose," in *Anais do Simpósio Brasileiro de Pesquisa Operacional, 2021, João Pessoa*, 2021, anais eletrônicos... Campinas, Galoá, 2021.

[33] A. Vitor, "Uma proposta de algoritmo genético híbrido para o problema do caixeiro viajante," Ph.D. dissertation, Universidade Federal do Paraná, 2015.

[34] A. L. C. Ottoni, E. G. Nepomuceno, and M. S. d. de Oliveira, "Aprendizado por reforço na solução do problema do caixeiro viajante assimétrico: Uma comparação entre os algoritmos q-learning e sarsa," in *Simpósio de Mecânica Computacional*, 2016.

[35] L. Bodin, B. Golden, A. Assad, and M. Ball, "Routing and Scheduling of Vehicles and Crews – The State of the Art," *Computers and Operations Research*, vol. 10, no. 2, pp. 63–211, 1983.

[36] A. L. C. Ottoni, "Análise de sensibilidade dos parâmetros do aprendizado por reforço na solução do problema do caixeiro viajante," Master's thesis, Universidade Federal de São João del-Rei, 2016.

[37] L. M. Gambardella and M. Dorigo, "An ant colony system hybridized with a new local search for the sequential ordering problem," *INFORMS Journal on Computing*, vol. 12, no. 3, p. 237–255, Aug. 2000.

[38] L. Libralesso, A. Bouhassoun, H. Cambazard, and V. Jost, "Tree search algorithms for the sequential ordering problem," *ArXiv*, vol. abs/1911.12427, 2019.

[39] F. Majidi, M. Openja, F. Khomh, and H. Li, "An empirical study on the usage of automated machine learning tools," in *2022 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2022, pp. 59–70.

[40] C. A. d. S. Barreto, A. M. d. P. Canuto, J. C. Xavier-Júnior, A. Feitosa-Neto, D. F. A. Lima, and R. R. F. d. Costa, "PBIL AutoEns: uma Ferramenta de Aprendizado de Máquina Automatizado Integrada à Plataforma Weka / PBIL AutoEns: an Automated Machine Learning Tool integrated to the Weka ML Platform," *Brazilian Journal of Development*, vol. 5, no. 12, p. 29226–29242, Dec. 2019.

[41] K. Chauhan, S. Jani, D. Thakkar, R. Dave, J. Bhatia, S. Tanwar, and M. S. Obaidat, "Automated machine learning: The new wave of machine learning," in *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, 2020, pp. 205–212.

[42] R. S. Olson and J. H. Moore, "TPOT: A tree-based pipeline optimization tool for automating machine learning," in *Workshop on automatic machine learning*. PMLR, 2016, pp. 66–74.

[43] Y. Li, J. Wu, and T. Deng, "Meta-GNAS: Meta-reinforcement learning for graph neural architecture search," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106300, 2023.

[44] R. G. Mantovani, A. L. D. Rossi, E. Alcobaça, J. Vanschoren, and A. C. P. L. F. de Carvalho, "A meta-learning recommender system for hyperparameter tuning: Predicting when tuning improves SVM classifiers," *Information Sciences*, vol. 501, pp. 193–221, 2019.

[45] F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., *Automated Machine Learning: Methods, Systems, Challenges*. Springer, 2019.

[46] R. R. Vianna, J. L. Marin, and J. M. de Seixas, "Transferência de conhecimento para filtragem online de partículas baseada em calorimetria de altas energias," in *Anais do 15 Congresso Brasileiro de Inteligência Computacional*. Joinville, SC: SBIC, 2021, pp. 1–7.

[47] R. T. da Silva and H. S. Lopes, "A transfer learning approach for the tattoo detection problem," in *Anais do 15 Congresso Brasileiro de Inteligência Computacional*. Joinville, SC: SBIC, 2021, pp. 1–8.