RODRIGO CESAR BONINI

DESCOBERTA E REUSO DE POLÍTICAS PARCIAIS
PROBABILÍSTICAS NO APRENDIZADO POR REFORÇO

São Paulo
2018

RODRIGO CESAR BONINI

DESCOBERTA E REUSO DE POLÍTICAS PARCIAIS
PROBABILÍSTICAS NO APRENDIZADO POR REFORÇO

Dissertação apresentada à Escola
Politécnica da Universidade de São Paulo
como requisito para obtenção do Título de
Mestre em Ciências.

São Paulo
2018

RODRIGO CESAR BONINI

DESCOBERTA E REUSO DE POLÍTICAS PARCIAIS PROBABILÍSTICAS NO APRENDIZADO POR REFORÇO

Dissertação apresentada à Escola
Politécnica da Universidade de São Paulo
como requisito para obtenção do Título de
Mestre em Ciências.

Área de Concentração:
Engenharia de Computação

Orientador:
Profa. Dra. Anna Helena Reali Costa

São Paulo
2018

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

Catálogo-na-publicação

Bonini, Rodrigo Cesar

Descoberta e Reuso de Políticas Parciais Probabilísticas no Aprendizado por Reforço / R. C. Bonini -- São Paulo, 2018.

72 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Inteligência Artificial 2.Aprendizado de Máquina 3.Aprendizado por Reforço 4.Transferência de Conhecimento 5.Políticas Parciais I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

EPÍGRAFE

"É impossível um homem aprender aquilo que ele acha que sabe."

Epictetus

"A ciência nunca resolve um problema sem criar pelo menos outros dez"

George Bernard Shaw

"Nasça para perder, viva para vencer"

Lemmy Kilmister

AGRADECIMENTOS

Primeiramente, eu agradeço à minha orientadora Anna Helena Reali Costa, por todo o suporte e encorajamento que foram essenciais durante esses anos.

Gostaria também de expressar minha gratidão pela minha família, pela força nos momentos difíceis que passamos nos últimos anos.

Também agradeço todos os membros do LTI (Laboratório de Técnicas Inteligentes) por todas as discussões, conversas e trabalhos que tivemos em conjunto.

Meu agradecimento especial para meus colegas Felipe Leno da Silva, Ruben Glatt, Ricardo Jacomini, Juan Perafan, Igor Lima, Luis Ludescher, Andre Jalbut, Vinicius Carvalho e Amir Saad por todo o suporte e pelos bons momentos vividos nesses anos.

Também merecem minha gratidão os membros do grupo do CEST (Centro de Estudos Sociedade e Tecnologia) pelo apoio, em especial o professor Edison Spina por acreditar no meu trabalho e dedicação.

Gostaria também de agradecer aos professores Edith Ranzini, Italo Vega, Daniel Gatti e Satoshi Nagayama, que sempre confiaram em mim, me indicaram para o mestrado e me incentivaram a estudar e a nunca desistir.

E por fim, a todos de outros lugares que estiveram envolvidos em algum momento desta pesquisa.

RESUMO

O aprendizado por reforço é uma técnica bem sucedida, porém lenta, para treinar agentes autônomos. Algumas soluções baseadas em políticas parciais podem ser usadas para acelerar o aprendizado e para transferir comportamentos aprendidos entre tarefas encapsulando uma política parcial. No entanto, geralmente essas políticas parciais são específicas para uma única tarefa, não levam em consideração recursos semelhantes entre tarefas e podem não corresponder exatamente a um comportamento ideal quando transferidas para outra tarefa diferente. A transferência descuidada pode fornecer más soluções para o agente, dificultando o processo de aprendizagem. Sendo assim, este trabalho propõe uma maneira de descobrir e reutilizar de modo probabilístico políticas parciais orientadas a objetos aprendidas, a fim de permitir melhores escolhas de atuação para o agente em múltiplas tarefas diferentes. A avaliação experimental mostra que a proposta é capaz de aprender e reutilizar com sucesso políticas parciais em diferentes tarefas.

Palavras-chave: Inteligência Artificial, Aprendizado de Máquina, Aprendizado por Reforço, Transferência de Conhecimento, Políticas Parciais, Processos de Decisão de Markov

ABSTRACT

Reinforcement Learning is a successful yet slow technique to train autonomous agents. Option-based solutions can be used to accelerate learning and to transfer learned behaviors across tasks by encapsulating a partial policy. However, commonly these options are specific for a single task, do not take in account similar features between tasks and may not correspond exactly to an optimal behavior when transferred to another task. Therefore, careless transfer might provide bad options to the agent, hampering the learning process. This work proposes a way to discover and reuse learned object-oriented options in a probabilistic way in order to enable better actuation choices to the agent in multiple different tasks. The experimental evaluation show that the proposal is able to learn and successfully reuse options across different tasks.

Keywords: Artificial Intelligence, Machine Learning, Reinforcement Learning, Transfer Learning, Partial Policies, Markov Decision Process

LISTA DE FIGURAS

2.1	Um agente interage com o ambiente através de uma sequência de ações escolhidas em um estado observado, com o objetivo de maximizar um sinal de recompensa. Para cada ação a realizada no estado s_t há uma recompensa r_{t+1} após o agente atingir determinado estado s_{t+1} . Adaptado de (SUTTON; BARTO, 1998).	30
2.2	Várias métricas diferentes para mensurar a TL são possíveis. Este gráfico esboça os benefícios das métricas de <i>jumpstart</i> , <i>performance assintótica</i> , <i>tempo p/ threshold</i> e <i>total reward</i> . (Adaptado de: (TAYLOR; STONE, 2009).	34
2.3	<i>TaxiWorld</i> . Neste ambiente, o taxi deve apanhar os passageiros e deixá-los em seu destino (G). À direita está a descrição do estado representado à esquerda na figura. Adaptado de (DIUK; COHEN; LITTMAN, 2008).	35
3.1	<i>Block Dude</i> com objetos rotulados (numerados), a saída em na parte superior esquerda e o agente à direita. Adaptado de (TOPIN et al., 2015).	39
3.2	Visão geral do processo utilizado no PPB (TOPIN et al., 2015). Soluções iniciais são dadas para as tarefas descritas como OO-MDPs. Então o PPB realiza os processos de abstração e descoberta de políticas parciais determinísticas e abstratas, formando um conjunto de tais políticas, as quais são utilizadas durante o aprendizado de uma nova tarefa, realizando um processo de abstração do estado concreto observado na tarefa alvo, seguido da definição da ação concreta indicada pela política parcial abstrata para o estado abstrato correspondente ao estado concreto observado; esta ação é aplicada na tarefa alvo.	48
3.3	Processo de abstração utilizado no PPB (TOPIN et al., 2015). Supondo que o valor Q do taxi mover-se para a direita no estado OO-MDP 1 seja 0,75 e que no estado do OO-MDP 2 incluído no mesmo estado abstrato seja 0,25, o valor Q do estado abstrato correspondentes a ambos estados concretos será a média desses 2 valores, o que neste caso seria 0,5.	49

4.1	<i>Goldmine</i> . Neste ambiente, o mineiro deve apanhar pedaços de ouro. O episódio termina somente quando ele coleta todos os pedaços. Adaptado de (DIUK; COHEN; LITTMAN, 2008).	54
4.2	Visão geral do processo utilizado pela proposta PRDO (BONINI et al., 2018): inicialmente PRDO utiliza o algoritmo PPB para descoberta de políticas parciais determinísticas e abstratas das tarefas de origem. PRDO combina estas políticas abstratas em uma política probabilística e, usando uma função de <i>grounding</i> dos estados abstratos da política probabilística para os estados da tarefa de destino, transforma em uma política π_o que norteará a exploração do agente na busca pela solução na tarefa de destino.	55
4.3	Exemplo da combinação efetuada: $\pi_1(s)$ e $\pi_2(s)$ são políticas parciais determinísticas previamente descobertas e $\pi_o(a s)$ é a política probabilística resultante da combinação de π_1 e π_2	56
5.1	Tarefas de origem, onde o círculo é o agente e o quadrado em amarelo (G) é o objetivo para cada tarefa. Adaptado de (BONINI et al., 2017).	60
5.2	Tarefas alvo, onde o círculo é o agente e o quadrado em amarelo (G) é o objetivo para cada tarefa. Adaptado de (BONINI et al., 2017).	60
5.3	A recompensa média por 1000 episódios durante o processo de aprendizagem (BONINI et al., 2017).	61
5.4	Exemplo de tarefa no GridWorld. O agente é representado pela circunferência cinza, o objetivo pelo quadrado amarelo (G) e os obstáculos (muros) estão em preto. Adaptado de (BONINI et al., 2018).	62
5.5	Média de passos para atingir o objetivo em 100 episódios durante 20 execuções do processo de aprendizagem nas tarefas destino do GridWorld (BONINI et al., 2018).	63
5.6	No <i>Taxiworld</i> , o agente é representado pelo carro (táxi), o objetivo pelo quadrado em amarelo contendo (G), os obstáculos (muros) estão em preto e os passageiros que o agente pode pegar estão em vermelho (P). O agente deve pegar todos os passageiros antes de chegar ao objetivo. Adaptado de (BONINI et al., 2018).	64
5.7	A quantidade média de passos até atingir o objetivo durante 10000 episódios durante 20 execuções no processo de treinamento na tarefa destino do <i>TaxiWorld</i> . 4 passageiros nas tarefas origem e 2 passageiros na tarefa destino (BONINI et al., 2018).	66

5.8	A quantidade média de passos até atingir o objetivo durante 10000 episódios durante 20 execuções no processo de treinamento na tarefa destino do <i>TaxiWorld</i> . 2 passageiros nas tarefas origem, 4 passageiros na tarefa destino (BONINI et al., 2018).	66
-----	--	----

LISTA DE TABELAS

3.1 Lista de abreviações usadas na tabela 3.2. 51

3.2 Trabalhos relacionados contendo métodos de TL e RL, classificados em termos de quatro dimensões: Diferenças de tarefas permitidas; tarefas de origem; transferência de conhecimento e métodos de aprendizagem. Os símbolos usados estão apresentados na tabela 3.1 52

LISTA DE ABREVIACES

RL	<i>Reinforcement Learning</i>
TL	<i>Transfer Learning</i>
MDP	<i>Markov Decision Process</i>
OO-MDP	<i>Object-Oriented MDP</i>
TD	<i>Temporal Difference</i>
CL	<i>Classification</i>
DP	<i>Dynamic Programming</i>
PPB	<i>Portable PolicyBlocks</i>
GCG	<i>Great Common Generalization</i>
PRDO	<i>Probabilistic Reuse of Discovered Options</i>
PB	<i>PolicyBlocks</i>

LISTA DE SÍMBOLOS

S	Conjunto de estados.
A	Conjunto de ações.
T	Função de transição de estados.
R	Função de recompensa.
s	Um estado.
a	Uma ação.
r	Um sinal de recompensa.
t	Um dado instante de tempo.
s_t	Os subscritos referem-se a intervalos de tempo, salvo indicação em caso contrários, e.g., s_t é o estado atual no instante t .
a_t	Os subscritos referem-se a intervalos de tempo, salvo indicação em caso contrários, e.g., a_t é a ação atual no instante t .
π	Uma política.
π^*	Uma política ótima.
Q	Função Q-table.
α	Taxa de aprendizado.
γ	Fator de desconto.
$\mathcal{I}$	Conjunto de estados iniciais da política parcial.
β	Condição de término da política parcial.
n	Número predefinido de passos.
o	Política parcial.
C	Um conjunto de classes.

C_i	Uma classe.
$Att(C_i)$	Um conjunto de atributos que compõem uma classe C_i .
b_j	Um atributo.
$Dom(C_i.b_j)$	Domínio do atributo b_j .
K	Conjunto de objetos.
k_i	Um objeto.
G	Estado destino (meta).
Ω	Um domínio.
b^0	Estado inicial.
ψ	Uma tarefa.
ψ_s	Uma tarefa de origem.
Ψ_s	Conjunto de tarefas de origem.
ψ_t	Uma tarefa de destino (alvo).
mrg	Combinação de correspondências entre políticas (<i>merge</i>).
L	Conjunto de políticas (soluções).
O	Conjunto de políticas parciais (<i>options</i>).
π_o	Política da política parcial.
s_{ab}	Estado abstrato.
$f(s)$	Função que mapeia estado abstrato para estado concreto (abstração e <i>grounding</i>).
i	Número desejado de políticas parciais.
c	Política parcial candidata.
π_M	Política mista (combinada).
$\pi_o(a s)$	Probabilidade da ação a ser escolhida pela política parcial única $\langle \pi_o, S, n \rangle$ no estado s .

$ A $	Tamanho do conjunto de ações.
$ O(a s) $	Número de políticas parciais em O que seleciona a no estado s .
$ O_s $	Número de políticas parciais definidas para o estado s .

SUMÁRIO

1	Introdução	25
1.1	Objetivos	26
1.2	Organização do Documento	26
1.3	Contribuições	27
2	Conceitos Básicos	29
2.1	Aprendizado por Reforço	29
2.2	Políticas Parciais	31
2.3	Transferência de Conhecimento	33
2.4	Object-Oriented MDPs	34
2.5	Conclusão do Capítulo	35
3	Formalização do Problema e Estado da Arte	37
3.1	O Problema	37
3.2	Trabalhos Relacionados	39
3.2.1	Métodos de TL com mesmo espaço de estados e ações	41
3.2.2	Aprendizado Hierárquico	44
3.2.3	Múltiplas Tarefas de Origem	45
3.3	Conclusão do Capítulo	50
4	Proposta	53
5	Experimentos e Resultados	59
5.1	Experimento 1 – Transferência em um Mesmo Espaço de Estados	59
5.2	Experimento 2 – Uso Controlado de Objetos	61
5.3	Experimento 3 – Tarefas Mais Gerais no OO-MDP	63
5.4	Conclusão do Capítulo	66
6	Conclusão e Trabalhos Futuros	69
	Referências	71

1 INTRODUÇÃO

A Inteligência Artificial tem como um de seus objetivos investigar como construir agentes inteligentes de modo que sejam capazes de realizar as mais diversas tarefas da forma mais autônoma e eficaz possível.

Uma solução muito utilizada para desenvolver tais agentes é dar-lhes a capacidade de aprender por experimentação e aprender como realizar a tarefa desejada atuando diretamente no ambiente, sem a necessidade de um especialista e de informação prévia. Este tipo de solução é denominada Aprendizado por Reforço (RL - *Reinforcement Learning*) (SUTTON; BARTO, 1998), que consiste em escolher e aplicar uma ação que afeta o ambiente, observando o quanto essa ação foi vantajosa. Um agente pode aprender a agir otimamente no ambiente executando este procedimento múltiplas vezes, visando maximizar as vantagens obtidas com sua atuação.

O RL pode ser entendido como um método de programação de agentes através do oferecimento de recompensas e punições, sem a necessidade de especificar como uma tarefa deve ser realizada.

Apesar dos progressos significativos, muitas técnicas de RL aprendem lentamente através de interações do tipo tentativa e erro. Para o RL ser útil em problemas reais, o processo de aprendizado deve ser acelerado, caso contrário, o tempo demandado até que o agente aprenda a agir adequadamente será proibitivo.

Uma das formas de acelerar este processo é aproveitar o conhecimento de tarefas já aprendidas no passado através da generalização (SONI; SINGH, 2006; KOGA; SILVA; COSTA, 2015) e da Transferência de Conhecimento (TL - *Transfer Learning*) (TAYLOR; WHITESON; STONE, 2007; PAN; YANG, 2010).

A TL tem como objetivo transferir conhecimento através de várias tarefas, problemas e agentes. A experiência e o conhecimento podem ser adquiridos de uma tarefa já realizada ou parcialmente explorada previamente, com a finalidade de ajudar a melhorar o aprendizado em outras tarefas semelhantes e relacionadas, porém diferentes daquela já aprendida.

Outra forma de reaproveitar conhecimento é através do reúso de políticas parciais (SUTTON; PRECUP; SINGH, 1999; KONIDARIS; BARTO, 2007), que são soluções parciais em tarefas de RL. Uma política parcial descreve um conjunto de ações que um agente pode realizar para atingir determinado subobjetivo em uma tarefa. O uso de políticas parciais fornece métodos com alternativas factíveis para a resolução de um problema de RL, pois permite que os agentes possuam um novo nível de abstração, de

habilidades e de ações possíveis a serem realizadas, já que deste modo pode-se reutilizar e combinar soluções parciais em diferentes tarefas e problemas, demandando menos tempo de aprendizado e provendo mais informação para os agentes.

Entretanto, os trabalhos anteriores apenas apresentam propostas dependentes de um humano especialista para identificar partes da solução ou apenas servem para algum problema específico em somente um tipo de tarefa ou problema, não sendo totalmente efetivas quando as tarefas são grandes ou diferentes,

A investigação deve buscar respostas para algumas perguntas que foram respondidas apenas parcialmente pelo estado da arte, entre elas: (i) Como abstrair apropriadamente o conhecimento parcial adquirido em um aprendizado? (ii) Como reutilizar esse conhecimento? (iii) O reuso de políticas parciais estocásticas é realmente efetivo no aprendizado de novas tarefas?

1.1 Objetivos

Esta pesquisa investiga e especifica um novo arcabouço que integra métodos existentes de descoberta de políticas parciais, visando a reutilização e melhor combinação de políticas parciais de tarefas previamente aprendidas em tarefas e problemas RL, e assim acelerar o processo de aprendizado atual dos algoritmos de TL para RL.

Em especial, o objetivo desta pesquisa é contribuir com uma técnica que permita o reuso **probabilístico** de políticas parciais com uma melhor combinação probabilística de políticas parciais, e depois dessa combinação aplicá-las e compará-las com técnicas de RL que foram utilizadas para acelerar o aprendizado entre tarefas.

1.2 Organização do Documento

Este texto foi estruturado da seguinte forma:

- **Capítulo 2 – Conceitos Básicos:** onde constarão conceitos básicos como RL, políticas parciais, TL e OO-MDP.
- **Capítulo 3 – Formalização do Problema e Estado da Arte:** onde será contextualizado o problema e será discutido o que se tem feito em trabalhos relacionados.
- **Capítulo 4 – Proposta:** onde será apresentada a proposta.
- **Capítulo 5 – Resultados:** onde serão mostrados resultados obtidos.
- **Capítulo 6 – Conclusão e Trabalhos Futuros:** onde serão mostrados conclusões, próximos passos e trabalhos futuros.

1.3 Contribuições

No decorrer deste mestrado, inicialmente gerou-se artigos com experimentos exploratórios com uma abordagem que utiliza políticas parciais em problemas multiobjetivo, que apesar de terem produzido resultados interessantes, serviram apenas como base para a proposta, já que foi verificado que o uso de tarefas multiobjetivo não é um fator determinante para que o aprendizado seja acelerado. Essas publicações encontram-se sumarizados abaixo:

1. BONINI, R. C.; SILVA, F. L. da; COSTA, A. H. R. Learning options in multiobjective reinforcement learning. In: *AAAI-17 Student Paper*, 2017. p. (4708–4709).

Nesse artigo foi investigado o uso do algoritmo *PolicyBlocks* para descoberta de políticas parciais para uma tarefa com múltiplos objetivos; essas políticas foram reutilizadas na mesma tarefa com o intuito de acelerar o aprendizado.

2. BONINI, R. C.; SILVA, F. L. da; SPINA, E.; COSTA, A. H. R. Using options to accelerate learning of new tasks according to human preferences. In: *AAAI-17 Workshop Human-Machine Collaborative Learning*, 2017. p. (1–8).

Nesse artigo foi investigado o uso do algoritmo *PolicyBlocks* para descoberta de políticas parciais para uma tarefa com múltiplos objetivos; porém aqui essas políticas foram reutilizadas em uma tarefa diferente também multiobjetivo, com o intuito de acelerar o aprendizado respeitando preferências de diferentes perfis de usuários.

Após esses experimentos iniciais, foi verificado que para realizar a transferência de conhecimento de forma efetiva, não importa se as tarefas possuem múltiplos objetivos, e sim a maneira como as soluções são descobertas e reutilizadas. Sendo assim, as contribuições mais significativas desta dissertação serão descritas a seguir. Apesar das abordagens bem sucedidas de trabalhos anteriores descritos ao longo deste documento, nenhuma delas fornece simultaneamente:

- **Um método automático de descoberta de políticas parciais em múltiplas tarefas contendo objetos (OO-MDPs).** - Foram estudados e analisados alguns algoritmos de descoberta de políticas parciais, e foi verificado que para realizar a TL entre tarefas é necessário que a técnica seja capaz de generalizar espaço de estados, por exemplo generalizando em relação a objetos.
- **Uma combinação das políticas parciais descobertas de modo a otimizar o aprendizado.** - Diversas técnicas descobrem várias políticas parciais, mas

essas políticas normalmente só funcionam para o mesmo espaço de estados, são usadas separadamente ou necessitam ser especificadas por humanos.

- **Uma reutilização probabilística de políticas previamente aprendidas e combinadas, controlando o nível de exploração do agente.** - Alguns trabalhos combinam políticas probabilisticamente e controlam o nível de exploração do agente, porém isso é feito manualmente por humanos (sem descoberta automática), além de que algumas ações podem ficar com probabilidade nula de serem escolhidas.

Nossa proposta foi divulgada nas seguintes publicações:

1. BONINI, R. C.; SILVA, F. L. da; GLATT, R.; COSTA, A. H. R. Transferring probabilistic options in reinforcement learning. In: *AAMAS Workshop in Transfer in Reinforcement Learning*, 2017.

Nesse artigo, foi investigado o uso do algoritmo *PolicyBlocks* para descoberta de políticas parciais para **múltiplas** tarefas de origem, onde essas políticas foram reutilizadas probabilisticamente, com o intuito de acelerar o aprendizado, em várias tarefas com configurações diferentes, mas com o mesmo espaço de estados.

2. BONINI, R. C.; SILVA, F. L. da; GLATT, R.; SPINA, E.; COSTA, A. H. R. A framework to discover and reuse object-oriented options in reinforcement learning. In: *BRACIS*, 2018.

Nesse artigo, foi investigado o uso do algoritmo *Portable PolicyBlocks*, uma extensão do algoritmo *PolicyBlocks* para descoberta de políticas parciais orientadas a objetos para múltiplas tarefas de origem. Essas políticas foram combinadas e reutilizadas probabilisticamente em uma tarefa diferente com o intuito de acelerar o aprendizado.

2 CONCEITOS BÁSICOS

Os conceitos básicos necessários para a compreensão deste projeto estão descritos a seguir.

2.1 Aprendizado por Reforço

Para um agente utilizar RL, é necessária uma descrição de seus sensores e atuadores para interações com o ambiente (através de estados e ações), onde os sensores fornecem uma observação que indica o estado do ambiente e atuadores executam as ações.

Além disso, deve-se definir também uma função de recompensa, e é aprendida uma função de valor, que determina uma política. A política é um conjunto de instruções que dita a ação a ser executada pelo agente em cada estado possível. Por sua vez, a função de recompensa codifica o quão boa é uma particular circunstância em que o agente se encontra, ou seja, indica a qualidade da atuação executada. Por fim, a função de valor representa o quão bom o estado atual do ambiente é. Esta função é geralmente baseada na esperança de recompensas futuras e na recompensa atual.

O objetivo do RL é aprender uma política ótima, que é um mapeamento da ação que o agente deve realizar em cada estado para obter a maior recompensa cumulativa esperada possível. Como a recompensa representa indiretamente o objetivo, a política ótima deve levar o agente a realizar a tarefa desejada da melhor forma possível.

Um dos modelos mais tradicionais para um problema de RL com um único agente é o Processo de Decisão de Markov (MDP - *Markov Decision Process*) (PUTERMAN, 1994), ilustrado na figura 2.1.

Formalmente, um MDP é descrito pela quádrupla $\langle S, A, T, R \rangle$, onde:

- S é o conjunto de possíveis **estados** do ambiente.
- A é o conjunto finito de **ações** que podem ser executadas pelo agente.
- $T : S \times A \times S \rightarrow [0, 1]$ é a **função de transição de estados**, onde $T(s_t, a_t, s_{t+1})$ define a probabilidade de uma transição do estado s_t para s_{t+1} após a execução da ação a_t no instante de tempo t .
- $R : S \times A \times S \rightarrow \mathbb{R}$ é a função de **recompensa** que mapeia a atuação do agente para uma recompensa numérica. Ou seja, é uma função de recompensa que indica a recompensa recebida após a execução de uma ação.

Através da função de recompensa R o agente percebe qual o comportamento desejável para determinada tarefa, uma vez que se pode definir um valor arbitrário para ser atribuído quando o agente chegar no estado que contém o objetivo. Alternativamente, pode-se atribuir também um valor de recompensa inversamente proporcional à distância do agente ao estado objetivo, caso o espaço do problema permita definir esta distância.

A cada ciclo de percepção-ação, como ilustra a figura 2.1, o agente irá observar em qual estado ele está neste momento e escolher uma ação condizente com o estado atual, então o próximo estado será definido através da função de transição. Para modelar o problema como um MDP, é necessário que a função de transição obedeça à propriedade ou condição de Markov (SUTTON; BARTO, 1998), ou seja, que dependa apenas do estado atual, não do histórico de atuação do agente.

Para escolher qual a ação apropriada para cada momento, o agente deve computar uma política π que indique qual é a ação mais apropriada para cada estado possível. A política pode ser determinística $\pi : S \rightarrow A$ (para cada estado, há um mapeamento direto para a ação mais apropriada) ou probabilística $\pi : S \times A \rightarrow [0, 1]$ (cada ação tem uma probabilidade de ser executada em cada estado).

Figura 2.1 – Um agente interage com o ambiente através de uma sequência de ações escolhidas em um estado observado, com o objetivo de maximizar um sinal de recompensa. Para cada ação a realizada no estado s_t há uma recompensa r_{t+1} após o agente atingir determinado estado s_{t+1} . Adaptado de (SUTTON; BARTO, 1998).

Uma política ótima π^* é uma função que escolhe sempre uma ação que maximize as recompensas futuras em qualquer estado, portanto, resolver um MDP é computar uma política ótima determinística.

Uma forma de resolver o MDP através de interações com o ambiente é por exemplo com o algoritmo *Q-learning* (WATKINS; DAYAN, 1992). Este algoritmo avalia a qualidade de cada par estado/ação de acordo com o valor esperado de recompensas

a longo prazo, avaliada na variável Q . A cada passo, o agente escolhe uma ação e atualiza a função de qualidade de acordo com a recompensa observada:

$$Q_{t+1}(s_t, a_t) \leftarrow (1 - \alpha)Q_t(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q_t(s_{t+1}, a)], \quad (2.1)$$

onde r_{t+1} é a recompensa observada após executar a ação a_t no estado s_t , α é a taxa de aprendizado ($0 < \alpha \leq 1$), que indica quão rápido será o aprendizado, e γ é um fator de desconto ($0 < \gamma \leq 1$), que codifica o horizonte em que recompensas são relevantes. A ação mais apropriada para cada estado é escolhida de acordo com os valores de Q , e, após um certo número de interações com o ambiente, o valor estabiliza em Q^* e o agente terá aprendido a política ótima, dada por:

$$\pi^*(s_t) = \operatorname{argmax}_a Q^*(s_t, a). \quad (2.2)$$

No entanto, o *Q-learning* padrão pode ser ineficiente em ambientes com grandes espaços de estados, pois o agente pode precisar de várias etapas para alcançar a solução ótima.

2.2 Políticas Parciais

Políticas parciais (SUTTON; PRECUP; SINGH, 1999) possibilitam um modo de formular ações de alto nível em um MDP, podendo facilmente ser incorporadas em uma infinidade de diferentes algoritmos de RL. Uma política parcial descreve uma sequência de ações que um agente pode realizar para atingir determinado subobjetivo em um problema MDP.

Conforme definido por (SUTTON; PRECUP; SINGH, 1999), uma política parcial para um MDP é uma sequência condicional de ações primitivas definidas como uma tripla-tupla $\langle \pi, \mathcal{I}, \beta \rangle$, consistindo de uma política ($\pi : S \rightarrow A$), um conjunto de estados iniciais ($\mathcal{I} \subseteq S$), e uma condição de término ($\beta : S \rightarrow [0, 1]$). O conjunto inicial $\mathcal{I}$ é um subconjunto de estados nos quais as políticas parciais podem começar a ser executadas. Por exemplo, uma política parcial está disponível para ser executada no estado s_t se $s_t \in \mathcal{I}$.

Se uma condição inicial suficiente para disparar a execução de uma política parcial for satisfeita e o agente selecioná-la, a política interna dessa política parcial será seguida até que sua condição de término β seja satisfeita. A condição de término β é uma função de probabilidade em estados que define a probabilidade com que o agente encerra a execução de uma política parcial quando está em determinado estado. Alternativamente, o término também pode se dar em um limite de tempo, que termina de forma determinística quando a execução da política π da política parcial

atinge um certo número de passos (BERNSTEIN, 1999). Assim, o agente também pode interromper a execução da política parcial após um número predefinido de passos efetuados numa tarefa, o que significa que β gera uma probabilidade de 1 após um número predefinido de passos n , isto é, a política parcial neste caso seria definida como $\langle \pi, \mathcal{I}, n \rangle$.

Quando uma política parcial termina, o agente tem a chance de selecionar outra política parcial, e este processo se repete ao longo do tempo, até atingir um estado final (objetivo da tarefa) ou um número predefinido de passos. As ações primitivas são um caso especial de política parcial que sempre duram exatamente 1 passo. No caso de um problema discreto de navegação robótica, uma ação primitiva seria mover-se apenas um passo para norte, sul, leste ou oeste, por exemplo.

Conforme definido em (SUTTON; PRECUP; SINGH, 1999), usa-se $Q(s, o)$ para ser o retorno esperado dado que o agente começa no estado s e executa a política parcial o . Assim, o conceito de uma função de ação-valor Q generaliza para uma função de valor de política parcial. Quando uma política parcial termina, o valor de Q é atualizado de acordo com o valor máximo da política parcial no estado resultante e a recompensa cumulativa descontada obtida durante a execução:

$$Q(s_t, o_t) \leftarrow (1 - \alpha)Q(s_t, o_t) + \alpha \left[r_o + \gamma^n \max_{o \in O_{s_{t+n}}} Q(s_{t+n}, o) \right], \quad (2.3)$$

onde $r_o = r_{t+1} + \gamma r_{t+2} + \dots + \gamma^{n-1} r_{t+n}$ e n é o tamanho da política parcial o .

Portanto, quando uma política parcial termina, seu valor esperado é atualizado em uma tabela Q que pode ser usada normalmente para derivar uma política.

Políticas parciais são tipicamente utilizadas como ações de alto nível, servindo para atingir subobjetivos em um MDP e terminando deterministicamente quando um estado objetivo ou um limite de tempo é alcançado.

Uma das vantagens do uso desse paradigma é que os estados iniciais das políticas parciais não precisam ser sequenciais/adjacentes. Com isso, a idéia central por trás do uso de políticas parciais em RL vem do simples fato de que a probabilidade de um agente se comportar de uma certa maneira deve ser proporcional à frequência com que esse comportamento foi bem sucedido no passado (THRUN; SCHWARTZ, 1995; BOWLING; VELOSO, 1998; BERNSTEIN, 1999; MCGOVERN; BARTO, 2001).

Muitos dos algoritmos desenvolvidos até hoje possuem como foco políticas parciais através de um especialista ensinando e dando dicas a um agente, sendo realizados assim poucos testes atribuindo autonomia ao agente (TOPIN et al., 2015). Alguns deles também assumem que todas as políticas usam o mesmo espaço de estados

(THRUN; SCHWARTZ, 1995; BERNSTEIN, 1999) ou requerem um mapeamento manual através dos diferentes espaços de estados (KONIDARIS; BARTO, 2007, 2009), sem contar o fato de que os algoritmos não são gerais, sendo muitas vezes específicos para determinada tarefa e apenas tarefas contendo um único objetivo. Desta forma, essas limitações restringiram a aplicabilidade das políticas parciais aprendidas.

Em (KONIDARIS; BARTO, 2007) foi apresentado que as políticas parciais podem ser transferidas com sucesso entre tarefas que dividem o mesmo espaço do agente, e que também melhoram significativamente o desempenho em experimentos de RL com apenas um agente (principalmente após muitas interações).

Essa política parcial aprendida reflete comportamentos que muitas vezes podem ser reutilizados em tarefas similares ou até mesmo em outros problemas. Este reuso se dá pela TL, descrita na próxima seção.

2.3 Transferência de Conhecimento

A TL baseia-se em utilizar conhecimento adquirido em tarefas aprendidas para acelerar o aprendizado em uma nova tarefa similar às antigas (realizadas previamente) (TAYLOR; STONE, 2009). Para isso, alguns fatores devem ser tratados para a utilização de TL: (i) Definir **o que** transferir; (ii) Definir **como** transferir; (iii) Definir **quando** transferir (TAYLOR; WHITESON; STONE, 2007).

A ideia por trás da TL foca em generalização, decorrente da abstração de problemas e soluções. Esta abordagem pode funcionar como a aprendizagem humana, onde o conhecimento prévio e parcial pode ser usado para acelerar a aprendizagem de uma nova tarefa. Por exemplo, é relativamente mais fácil aprender como andar de motocicleta sabendo previamente como andar de bicicleta.

Em TL, o conhecimento de uma ou mais tarefas de origem é usado para aprender uma ou mais tarefas alvo mais rapidamente. A tarefa de origem representa a tarefa na qual ocorrerá o treinamento do agente (no exemplo acima, na bicicleta), da qual será extraído todo o conhecimento para ser transferido para a tarefa alvo (no exemplo acima, andar de motocicleta), que representa a tarefa que o agente deve aprender, seu objetivo ou meta.

As tarefas também podem ter: (i) diferentes espaços de estados definidos pelo conjunto de todos os estados possíveis dentro do escopo da tarefa; (ii) diferentes espaços de ações definidos pelo conjunto de todas as ações disponíveis dentro do escopo da tarefa; (iii) ou diferenças nas funções de recompensa e transição.

TL tem objetivos voltados a cinco benefícios básicos de sua aplicação em problemas de RL (TAYLOR; STONE, 2009), conforme ilustra a figura 2.2:

1- *Jumpstart*: O desempenho inicial de um agente numa tarefa alvo deve ser melhorado (acelerado) tendo como objetivo central a TL para acelerar um início de aprendizado mais rápido.

2- *Asymptotic Performance*: A performance assintótica mede o aumento no desempenho final alcançado por um agente em determinada tarefa alvo utilizando TL em relação ao aprendizado não utilizando a TL.

3- *Total Reward*: A recompensa total acumulada por um agente (i. e. a área sob a curva de aprendizagem) deve ser melhorada se for utilizado o paradigma de TL comparando-se com sua não utilização.

4- *Transfer Ratio*: A razão da recompensa total acumulada pela TL e a recompensa total acumulada sem utilizar TL deve ser maior que 1.

5- *Time to Threshold*: O tempo de aprendizado necessário pelo agente para atingir um nível de desempenho pré-determinado (*threshold*) deve ser diminuído via TL.

Figura 2.2 – Várias métricas diferentes para mensurar a TL são possíveis. Este gráfico esboça os benefícios das métricas de *jumpstart*, *performance assintótica*, *tempo p/ threshold* e *total reward*. (Adaptado de: (TAYLOR; STONE, 2009).

2.4 Object-Oriented MDPs

O uso de descrições de tarefas que permitem uma abstração pode ajudar a acelerar o processo de aprendizagem. A representação orientada a objetos (DIUK; COHEN; LITTMAN, 2008) permite oportunidades de generalização e agrupamento por meio de uma descrição de tarefa fornecida intuitivamente. Um *Object-Oriented MDPs* (OOMDP) é definido como $\langle C, Att, Dom, K, A, T, R \rangle$ no qual o espaço de estados é abstraído através da descrição de um conjunto de classes $C = \{C_1, \dots, C_c\}$, onde cada classe C_i é composta por um conjunto de atributos denotados como $Att(C_i) = \{C_i.b_1, \dots, C_i.b_b\}$. Cada atributo b_j possui um domínio, $Dom(C_i.b_j)$, especificando o conjunto de valores

que esse atributo pode assumir; $K = \{k_1, \dots, k_k\}$ é o conjunto de objetos que existem em um ambiente, onde cada objeto k_i é uma instância de uma classe $C_i = C(k_i)$, então k_i é descrito pelo conjunto de atributos de sua classe, $k_i :: Att(C(k_i))$; A é o conjunto de ações, T é a função transição de estados e R é a função de recompensa. O estado do MDP é agora observado como a união de todos os estados dos objetos $s = \cup_{k \in K} k.state$, onde um estado do objeto é o conjunto de valores assumidos por cada um dos seus atributos em um determinado momento, $k_i.state = (\prod_{b \in Att(C(k_i))} k_i.b)$. O estado de um objeto a qualquer momento é o produto cartesiano dos valores atuais para cada atributo de sua classe.

A figura 2.3 esboça a representação orientada a objetos de um estado s . Neste ambiente, o taxi deve apanhar os passageiros e deixá-los em seu destino (G) (DIETTERICH, 2000; DIUK; COHEN; LITTMAN, 2008). As classes de objetos são $C = \{taxi, passageiro, muro\}$ e os objetos em si são instâncias das classes (e.g. *pass1*, *muro2*, etc).

Figura 2.3 – *TaxiWorld*. Neste ambiente, o taxi deve apanhar os passageiros e deixá-los em seu destino (G). À direita está a descrição do estado representado à esquerda na figura. Adaptado de (DIUK; COHEN; LITTMAN, 2008).

A representação orientada a objetos de um ambiente simplifica a representação de políticas e modelos de transição, abstraindo estados semelhantes. Se os espaços de estado de diferentes problemas ou tarefas de RL compartilharem um conjunto comum de objetos, o conhecimento poderá então ser transferido entre esses problemas (SILVA; COSTA, 2017; SILVA; TAYLOR; COSTA, 2018; GLATT; SILVA; COSTA, 2016).

2.5 Conclusão do Capítulo

O principal problema ao aplicar RL é que os algoritmos clássicos aprendem muito lentamente através de interações do tipo tentativa e erro, levando muito tempo para

convergir, pois as abordagens clássicas de RL precisam de muitas etapas para explorar o espaço de estados das tarefas.

Muitas soluções foram desenvolvidas para minimizar esses problemas, como as de TL e políticas parciais. O sistema descrito em (SUTTON; PRECUP; SINGH, 1999), no qual estamos especialmente interessados, propôs uma abordagem conceitual inicial para lidar com os problemas de escalabilidade em RL com o uso de políticas parciais.

As políticas parciais oferecem uma maneira de propor ações de alto nível que encapsulam sequências de ações realizadas por agentes, podendo ser facilmente incorporadas em vários algoritmos RL diferentes, e assim auxiliando a acelerar seu processo de aprendizagem. Uma política parcial pode resolver um subobjetivo em uma tarefa RL e pode conter um subconjunto de ações ótimas para determinados estados onde o agente está. Por exemplo, uma política parcial em um problema de navegação em uma sala poderia conter algumas ou todas as ações necessárias para um robô se mover em direção a uma porta, destrancá-la e abri-la.

Também foram desenvolvidas soluções de TL detalhadas em (TAYLOR; STONE, 2009), que permitem reutilizar os conhecimentos adquiridos em tarefas anteriores, generalizar e transferir conhecimento entre problemas, agentes e tarefas, acelerando assim a aprendizagem em problemas RL. Essas técnicas também foram utilizadas com as políticas parciais, sendo usado para reutilizar o conhecimento parcial de tarefas anteriormente aprendidas (TOPIN et al., 2015).

No entanto, aprender a descobrir quais conhecimentos parciais são úteis para um agente em um determinado problema e saber qual parte transferir desse conhecimento não é uma tarefa trivial, uma vez que o problema e as tarefas podem variar. Embora TL e políticas parciais tenham sido usadas de várias maneiras na RL, não há uma resposta padrão para muitos aspectos que devem ser definidos para: (i) descobrir e usar as políticas parciais; (ii) além de como e quando transferir seu conhecimento em problemas RL.

Portanto, esta pesquisa pretende investigar e especificar um novo uso para os métodos existentes de descoberta de políticas parciais, visando a reutilização e melhor combinação de políticas parciais de tarefas previamente aprendidas em tarefas RL, objetivando assim, acelerar o processo de aprendizado atual dos algoritmos RL e TL.

3 FORMALIZAÇÃO DO PROBLEMA E ESTADO DA ARTE

Neste capítulo serão descritos alguns problemas de RL e TL a serem tratados. Também constam alguns trabalhos que já foram feitos de forma a minimizar esses problemas, algumas lacunas nessas soluções e nossas propostas. Na seção 3.1 provemos algumas definições e restrições para definir o problema e o escopo deste trabalho. Na seção 3.2 são descritos trabalhos relacionados ao problema. Um resumo dos métodos e trabalhos descritos também pode ser encontrado nas tabelas 3.2 e 3.1 ao final deste capítulo.

3.1 O Problema

Este trabalho está interessado no problema geral de RL, ou seja, dadas as interações de um agente dentro de um MDP, o agente deve buscar a política ótima. No capítulo anterior, foram apresentadas as definições de RL e TL. Aqui será especificado o tipo de problema que estamos lidando neste trabalho. Os MDPs a serem resolvidos possuem as seguintes características:

- **Ambiente estacionário:** As dinâmicas do ambiente não mudam com o tempo; a função de transição T e a função de recompensa R permanecem sem mudança desde o começo até o fim do processo de aprendizagem em uma tarefa.
- **Episódico:** As tarefas possuem um estado objetivo G que, quando é atingido, faz com que a tarefa termine. Um episódio é um período de tempo entre o momento em que a tarefa começa, quando o agente está num estado inicial, e o momento que ela se encerra, quando o agente alcança um estado objetivo. Alternativamente, um episódio também pode terminar quando o agente atinge um determinado tempo limite sem ter alcançado o estado objetivo.
- **Observação total:** O agente tem visibilidade total do estado atual, ou seja, ele sabe exatamente em qual estado está.

Foi adotado o termo **tarefa** para referir-se a um ambiente descrito por um OO-MDP (DIUK; COHEN; LITTMAN, 2008). Conforme definido na seção 2.4, OO-MDPs representam ambientes por um **conjunto de objetos** $K = \{k_1, \dots, k_i\}$, onde cada objeto pertence a uma classe de objetos e tem um valor atribuído a um conjunto de atributos que estão associados à classe de objeto. $|K|$ indica a quantidade de objetos que uma

tarefa possui. O estado de um objeto em determinado momento é o conjunto atual de valores para cada atributo de sua classe. Da mesma forma, o estado s de um ambiente é definido como a união dos valores de atributo de todos os objetos no domínio em um determinado instante de tempo, $s = U_{k \in K} k.state$.

A figura 3.1 ilustra o *Block Dude*, contendo 3 blocos de objetos numerados, com as classes de objeto *wall*, *block*, *agent* e *exit*. Cada classe possui os atributos de posição x e y ; a classe agente possui um atributo de orientação (direita ou esquerda); e a classe de blocos possui um atributo *sendo carregado* que possui um valor *sim* ou *não*.

Neste domínio, o agente pode se mover para a direita, esquerda, pegar um bloco adjacente na direção que ele estiver virado (se a célula acima estiver vazia), soltar um bloco na direção que estiver virado (se essa célula estiver vazia), ou escalar um único bloco ou muro (se a célula acima dele estiver vazia). O agente tem como objetivo empilhar os blocos em uma configuração que garanta acesso à saída, seu objetivo. É importante notar que não importa qual bloco esteja em uma determinada posição, já que todos blocos são da mesma classe de objeto e dessa forma se 2 blocos forem trocados de posição, com cada um assumindo a posição anterior do outro, isso representará o mesmo estado s .

Define-se aqui **domínio** Ω um conjunto de tarefas que compartilham elementos da tupla $\langle C, Att, Dom, A \rangle$, onde C é o conjunto de classes, Att o conjunto de atributos e seus respectivos domínios Dom , e A é o conjunto de ações permitidas em cada tarefa. O OO-MDP provê uma representação estruturada que permite uma abstração do espaço de estados focando-se em objetos e atributos que são relevantes para determinada tarefa. Em outras palavras, considera-se como domínio Ω o tipo de cenário ou ambiente onde o agente terá tarefas a serem resolvidas.

Uma tarefa ψ aqui é definida pela tupla $\langle \Omega, S, K, R, T, b^0, G \rangle$, onde Ω é o domínio onde a tarefa será realizada, S é o espaço de estados, K é o conjunto de objetos, R é a função de recompensa dessa tarefa, T é a função de transição, b^0 a distribuição do estado inicial e G é o estado objetivo. Assume-se que tratam-se de tarefas episódicas, com estados finais absorventes. Um episódio começa colocando-se o agente em uma posição no ambiente dada por b^0 . Cada episódio termina quando o agente chega no estado final G ou quando executa um número máximo de passos.

Neste trabalho, o foco está na TL entre diferentes tarefas, dentro do mesmo domínio. Isso significa que são elegíveis tarefas com os mesmos espaços de ações e com objetos da mesma classe, mas com diferentes quantidades de objetos, diferentes estados iniciais e estados objetivos, funções de recompensa e de transição.

A representação OO-MDP de um ambiente simplifica a representação de políticas e funções de transição permitindo uma abstração sobre estados que são idênticos

Figura 3.1 – *Block Dude* com objetos rotulados (numerados), a saída em na parte superior esquerda e o agente à direita. Adaptado de (TOPIN et al., 2015).

para os propósitos de uma escolha de um agente. Se os espaços de estados de duas tarefas de RL diferentes possuem um conjunto de objetos em comum, então a TL pode potencialmente ocorrer entre esses dois problemas (DIUK; COHEN; LITTMAN, 2008).

Sendo assim, o problema de TL que queremos resolver é descrito da seguinte forma:

Problema Há um conjunto de tarefas de origem Ψ_s contendo as tarefas $\psi_1, \psi_2, \dots, \psi_n$, todas pertencendo ao mesmo domínio Ω , as quais o agente deve resolver usando RL. O agente recebe uma nova tarefa alvo ψ_t , do mesmo Ω , e com T e R_t desconhecidos. O objetivo do agente é resolver ψ_t com RL, utilizando conhecimento prévio adquirido de tarefas de origem, mostrando um desempenho melhor se comparada com a de resolução de ψ_t sem nenhum conhecimento prévio.

A principal melhoria que esse trabalho visa é no *jumpstart*, que indica que o agente tem um desempenho inicial melhor que um agente sem TL no início do aprendizado.

Porém, há também a possibilidade dos métodos de TL de fato piorarem o aprendizado ao invés de o acelerarem. Este caso é chamado de *transferência negativa*. Um dos principais desafios dos métodos de TL é fugir da transferência negativa. Ela ocorre quando as tarefas de origem e alvo são pouco relacionadas e o agente tem um maior trabalho para aprender a resolver a tarefa alvo do que se nada tivesse sido transferido.

3.2 Trabalhos Relacionados

Esta seção examina pesquisas relacionadas ao problema de TL. Aqui constam alguns termos e notações utilizados pelo estado da arte, e ao fim desta seção encontram-se as tabelas 3.1 e 3.2 contendo a relação dos mesmos com os trabalhos realizados na área. A TL busca generalizar soluções através de tarefas para acelerar o aprendizado

e essa abordagem vem ganhando atenção no universo de RL (TORREY; SHAVLIK, 2009; TAYLOR; STONE, 2009; PAN; YANG, 2010; LAZARIC, 2012).

Quando pesquisando sobre TL, existem algumas dimensões importantes que os algoritmos diferem, cada dimensão destacando diferentes aspectos do problema. As principais dimensões segundo (TAYLOR; STONE, 2009) são:

1. **Diferenças Permitidas de Tarefas** - quais as diferenças permitidas entre as tarefas de origem e alvo? Diversos trabalhos focaram somente na TL entre tarefas com o mesmo espaço de estado e ações, enquanto outros permitem que eles sejam diferentes. A possibilidade de que a TL ocorra entre tarefas de origem e alvo pouco similares dá ao agente mais flexibilidade.
2. **Tarefas de Origem** - O algoritmo de TL permite apenas uma tarefa de origem ou múltiplas? Alguns algoritmos levam em conta somente uma tarefa de origem, normalmente escolhida por um humano, enquanto outros se preocupam em agrupar conhecimento de várias tarefas de origem. Se várias tarefas de origem são permitidas, elas precisam também prover algum mecanismo de selecionar quais são as mais relevantes.
3. **Conhecimento Transferido** - Que tipo de informação é transferida entre a tarefa de origem e a tarefa alvo? Esse item se preocupa com como o conhecimento é codificado e representado para ser transferido. Os dois tipos mais comuns de conhecimento transferido são funções de valor e políticas. Outras abordagens populares consistem em transferir políticas parciais, as quais podem ser utilizadas para resolver partes de uma tarefa alvo.
4. **Métodos de Aprendizado** - que tipo de algoritmos são aplicáveis ao agente?

Diferenças Permitidas de Tarefas classificam os métodos de TL com relação as diferenças que podem existir entre as tarefas de origem e alvo. Os métodos mais simples permitem a transferência entre MDPs com diferentes funções de transição (T), funções de recompensa (R) ou distribuições de probabilidade de estado inicial (b^0). Nesses casos, o espaço de estados e ações são os mesmos entre as tarefas de origem e alvo.

Outros métodos permitem uma transferência mais ampla, com diferentes espaços de estados e ações, dadas algumas restrições. Métodos que usam representação relacional podem permitir que o número de objetos varie ou alguma restrição que os estados e ações sejam descritos pelas mesmas características. Diferentes espaços de estados e ações também podem ser alcançados assumindo-se que há uma função de mapeamento entre as tarefas de origem e alvo.

O dimensionamento de **tarefas de origem** preocupa-se com relação ao número de tarefas de origem usadas e como elas são selecionadas. A maioria dos métodos de TL assume que um humano tenha escolhido uma tarefa de origem e essa única tarefa será usada para TL (h).

Alguns outros métodos permitem que o conhecimento venha de múltiplas tarefas de origem; todas elas podem ser utilizadas para TL (all) ou somente um subconjunto das mais importantes também pode ser gerado (bib).

Os métodos de TL possuem diferentes tipos de **conhecimento transferido** entre as tarefas de origem e alvo. A transferência do valor aprendido (Q) é popular, usualmente sendo usado para inicializar a função da tarefa alvo. Políticas (π) também são transferidas, e uma abordagem bastante proeminente é transferir políticas parciais (π_p). A transferência de políticas também pode ser abstrata (π_{abs}) ou estocástica (π^{stoch}).

O tipo de TL diretamente afeta quais são os **métodos de aprendizado** permitidos para o agente. Se as funções de valor são transferidas (como Q , por exemplo), é mais interessante que o agente use um método de diferença temporal (TD). Outros métodos utilizam diferentes tipos de aprendizado como: aprendizado hierárquico (H); classificação (CL); ou programação dinâmica (DP); ao invés de aprendizado *online*.

As seções a seguir analisam alguns métodos TL para cada uma dessas dimensões.

3.2.1 Métodos de TL com mesmo espaço de estados e ações

As soluções de TL com mesmo espaço de estados e ações (TAYLOR et al., 2014) permitem reutilizar os conhecimentos adquiridos em tarefas anteriores, generalizando e transferindo conhecimento entre tarefas e agentes, acelerando assim a aprendizagem em problemas RL.

Começamos examinando métodos para TL entre tarefas com o mesmo conjunto de estados e ações. As tarefas de origem e alvo podem assim diferir com relação à função de transição T , de recompensa R , estado inicial b^0 ou estado objetivo G .

Um dos trabalhos mais antigos de TL com RL realizou TL mudando apenas a função de transição T (SELFRIDGE; SUTTON; BARTO, 1985). O agente primeiro aprendia como balançar um leve e longo mastro (tarefa de origem). Uma vez aprendido isso, a tarefa foi mudada para uma mais difícil (mudando-se T), com um mastro mais pesado e mais curto (tarefa alvo). O trabalho utilizou uma rede neural para aproximar a função Q . O tempo total gasto para aprender a sequência das duas tarefas e a reutilização dos pesos da rede neural aprendidos na primeira tarefa foi menor do que aprender somente a tarefa mais difícil diretamente (sem TL).

O trabalho proposto por (ASADA et al., 1996) apresenta uma ideia de aprender através de missões fáceis, provendo ao agente um conjunto de tarefas com dificuldade crescente. Esse trabalho aplica *Q-Learning* em um robô que necessita marcar um gol. O conjunto de tarefas precisa ser construído por um ser humano, que move o estado inicial (muda b^0) cada vez mais para longe do objetivo. Com o uso de TL, o agente aprende como atingir a meta mais rapidamente do que se tivesse que aprender a tarefa inteira diretamente. Como somente o estado inicial é mudado e o espaço de estados e espaço de ações permanecem os mesmos, a função Q é definida sobre o mesmo espaço e seus valores também são os mesmos para todas as tarefas.

Outro tipo conhecido de abordagem, é resolver tarefas através de soluções parciais. As primeiras abordagens conhecidas (AMAREL, 1968; ANZAI; SIMON, 1979) criavam subobjetivos examinando e encontrando soluções através do valor dos estados para problemas anteriores. Porém, esse tipo de abordagem depende de um treinamento muito longo e possui um custo computacional grande.

Uma abordagem geral adotada para a descoberta de políticas parciais é através da examinação de partes em comum ao longo de amostras de políticas (THRUN; SCHWARTZ, 1995; PICKETT; BARTO, 2002). Ao invés de focar-se em subobjetivos, essas técnicas focam em correspondências (partes em comum entre soluções), onde muitas das políticas parciais descobertas não possuem relação com nenhum subobjetivo ou problema em específico, podendo ser utilizadas para resolver várias tarefas diferentes que possuam algo em comum.

O trabalho de (BOWLING; VELOSO, 1998) discutiu o uso de políticas de problemas já resolvidos para novas tarefas similares, dando um valor limite para a subotimalidade de uma política anteriormente aprendida sobre um subproblema, o que de fato pode ser um ponto positivo se utilizarmos esse fator para examinar os trabalhos que verificam partes em comum (THRUN; SCHWARTZ, 1995; PICKETT; BARTO, 2002).

De maneira similar, (BERNSTEIN, 1999) utiliza soluções passadas combinadas probabilisticamente, onde são coletadas diversas soluções ótimas criadas por humanos e são analisadas quais as ações mais frequentes para cada estado visitado em cada solução (política). Quanto mais uma ação aparecer para determinado estado, maior sua probabilidade de ser escolhida pelo agente na fase de aprendizado. Entretanto, apesar de a combinação probabilística ser um ponto muito importante e prover um ganho interessante, esse tipo de solução fica muito preso e específico a um problema, pois uma vez que se muda muito uma tarefa, as políticas parciais aprendidas nem sempre farão sentido para a nova tarefa. Além disso, as políticas parciais iniciais precisam ser especificadas por um humano, não sendo aprendidas automaticamente.

Alguns trabalhos na descoberta de políticas parciais também utilizam classificação e agrupamento no espaço de estados para descobrir políticas que contém caminhos críticos e importantes. Alguns destes trabalhos baseiam-se em teorias de grafos para gerar políticas parciais de forma automática. Como por exemplo, particionamento de grafos locais (ŞİMŞEK; WOLFE; BARTO, 2005), classificação (MANNOR et al., 2004), descoberta e estrangulamento de regiões (MCGOVERN; BARTO, 2001), este último ocorrendo através de mudanças de b^0 e do reuso de várias de políticas em conjunto com políticas especificadas por humanos (dando foco em determinadas regiões, com humanos manipulando propositalmente os estados iniciais e função de recompensa do agente).

No entanto, cada um desses métodos requer análise prévia do espaço de estados por um humano especialista, o que significa que eles produzem políticas que não são necessariamente transferíveis para quaisquer problemas de diferentes espaços de estados ou funções de recompensa, uma vez que os humanos precisam definir previamente como será realizada a classificação sendo que nem sempre saberão detalhes sobre os problemas.

De modo a obter-se algoritmos de descoberta de políticas parciais mais gerais e mais independentes de tarefas anteriores, (PICKETT; BARTO, 2002) propõe um método de pontuação de políticas para determinada tarefa e realizam a união entre políticas para criar as políticas parciais, pegando suas partes de interesse em comum. Com isso, quanto maior (mais diversa) for a tarefa, e quanto mais tempo demorar o processo de treinamento (geração das políticas iniciais), mais efetivas serão as políticas parciais descobertas pelo algoritmo.

O algoritmo *PolicyBlocks*, proposto por (PICKETT; BARTO, 2002), é decomposto em um processo de três passos: (1) Em primeiro lugar, ele gera um conjunto de políticas parciais **candidatas** ao encontrar onde as soluções das amostras correspondem, em outras palavras, aqui ocorre o **processo de união** entre as soluções parciais, ($mrq(\pi_1 \cap \pi_2 \cap \dots, \pi_n)$). (2) Depois disso, ocorre a **pontuação** entre as soluções candidatas com base na correspondência ocorrida, onde quanto mais ações em comum para os estados existirem na política parcial candidata, maior será sua pontuação e, em seguida, escolhe a política parcial de pontuação mais alta. (3) Finalmente, ocorre a **subtração** desta política parcial do conjunto de políticas iniciais L , visando evitar redundância na geração das próximas políticas parciais. Então, ao final, obtém-se o conjunto de políticas parciais O que foram aprendidas com o algoritmo e que podem então ser usadas para acelerar a aprendizagem na mesma ou em novas tarefas similares.

Entretanto, por ser um algoritmo geral demais, o *PolicyBlocks* (PICKETT; BARTO, 2002) necessita de um longo treinamento inicial para prover as políticas iniciais e dependendo do tempo gasto para obter essas soluções iniciais, não há garantia de que as ações aprendidas pelo agente são necessariamente as melhores possíveis. Além disso, o aprendizado só pode ser reaproveitado no mesmo espaço de estados, uma vez que as soluções combinadas procuram partes em comum (ações em comum) para os mesmos estados, e se eles diferirem, não haverá correspondência.

3.2.2 Aprendizado Hierárquico

Para trazer mais flexibilidade para a TL, a aplicação de alguns tipos de abstração sobre o conhecimento faz com que ele seja mais fácil de ser generalizado e assim mais fácil de ser transferido.

(DIGNEY, 1998) propôs um método hierárquico no qual os estados mais frequentemente visitados ou que possuíssem um maior valor gradiente pelo valor dos estados se tornariam subobjetivos de um problema.

Um dos modos mais citados até a data presente de se realizar TL é através do uso de políticas parciais (SUTTON; PRECUP; SINGH, 1999). Esse trabalho explora o uso de abstração temporal para melhorar o desempenho no RL. Políticas parciais são a generalização de ações primitivas na forma de ações temporariamente estendidas. Políticas parciais são construídas de modo a prover partes de soluções anteriores para o agente, ao invés de apenas prover ações individuais. Existem diversos métodos que realizam o uso de políticas parciais para TL.

Em (ASADI; HUBER, 2007), por exemplo, foi feito o uso de políticas parciais e abstração de estados em TL. A arquitetura de aprendizado proposta possui dois níveis: um nível de **decisão** e um de **avaliação**. O agente automaticamente detecta subobjetivos (normalmente identificados por estados que são mais frequentemente visitados) para aprender políticas parciais e atingir esses subobjetivos no nível de decisão. O nível de decisão também tem uma representação compacta de estado e usa *Q-Learning*.

O nível de avaliação mantém totalmente o espaço de estados concreto e o espaço de ações, os quais são usados para encontrar discrepâncias durante o nível de decisão e para aumentar sua complexidade se necessário. A política no nível de decisão está dentro de um limite fixo da política ótima e, após o agente aprender na tarefa de origem, as políticas parciais aprendidas e a representação do nível de decisão são transferidas para a tarefa alvo.

Outros trabalhos consideram a reestruturação do problema em um novo espaço de estados que é mais simples de realizar a TL (KONIDARIS; BARTO, 2007, 2009).

Esses métodos executam a geração de políticas parciais automaticamente através da aprendizagem em uma representação do problema no espaço do agente onde os espaços do agente usam características deíticas definidas em relação à posição do agente ou estado (permitindo que o agente se refira por exemplo a "parede mais próxima").

3.2.3 Múltiplas Tarefas de Origem

A maioria dos trabalhos em TL, como os citados nas seções anteriores, assumem que uma única tarefa de origem tenha sido aprendida e que essa tarefa é escolhida por um humano especialista, assumindo assim que o agente pode utilizá-la para transferência, inclusive de forma prejudicial, produzindo *negative transfer*. Contudo, alguns algoritmos de TL permitem que o agente aprenda múltiplas tarefas de origem e use o conhecimento adquirido em todas (ou algumas delas) em novas tarefas. Usando esse conhecimento de múltiplas tarefas (não provido de humanos) pode ser um desafio, especialmente porque somente a solução de algumas tarefas de origem podem ser interessantes e bastante relacionadas com a tarefa alvo, enquanto que outras tarefas não.

Em (FERNANDEZ; VELOSO, 2006), foi introduzida a ideia de utilizar-se uma biblioteca de políticas, a qual contém políticas providas de tarefas de origem, onde é importante reforçar que uma política somente é incluída na biblioteca se for suficientemente diferente das outras que já estão dentro da biblioteca. Em seguida, quando aprendendo uma tarefa alvo, a usabilidade de cada política da biblioteca é estimada pela recompensa média recebida após a utilização de cada uma delas, onde a mais útil é transferida mais frequentemente. Ou seja, essa abordagem utiliza uma biblioteca de políticas, que armazena as políticas mais diferentes possível para identificar políticas que são básicas (fundamentais) em um problema e provêm a maior vantagem ao resolver uma nova tarefa, visando reutilizar probabilisticamente um conjunto de políticas determinísticas passadas que solucionam diferentes tarefas dentro do mesmo problema. Com isso, pode-se verificar que esse ponto de reutilizar as políticas probabilisticamente pode prover um ganho também se verificado com relação à políticas parciais.

Em (MARTÍN; GEFFNER, 2004) foi explorada uma abordagem que cria políticas generalizadas, as quais são políticas que, baseadas no número de instâncias de problemas resolvidos, são adequadas para resolver qualquer tarefa em um problema. Uma política generalizada é um conjunto simples de regras que generaliza soluções de um número de instâncias resolvidas. Porém, somente a generalização realizada

dessa maneira não é o suficiente para saber quais soluções de fato serão mais eficientes quando transferidas para uma nova tarefa.

Políticas generalizadas também foram exploradas em (SILVA; PEREIRA; COSTA, 2012). Os autores propuseram um algoritmo que lida com um conjunto de tarefas de origem em conjunto, como se todas as tarefas de origem formassem um único MDP. O algoritmo então encontra a melhor solução que maximiza o retorno médio de todas as tarefas, construindo uma política estocástica abstrata. O método proposto usa a pesquisa de políticas executando uma iteração de política baseada em gradiente, mas esse algoritmo é baseado em modelo e adequado apenas para aprendizado offline.

Uma estrutura proposta em (KOGA; SILVA; COSTA, 2015) explora a idéia de que a generalização de problemas estreitamente relacionados, mas resolvidos, pode produzir políticas que fornecem boas decisões em estados de uma nova tarefa não resolvida, porque evita assim a possibilidade de um mal desempenho de uma política conhecida em uma nova tarefa, indicando que políticas abstratas e não determinísticas podem oferecer uma orientação efetiva aos agentes. Esses benefícios acontecem na estratégia de exploração.

Outro ramo de trabalhos que um grande número de pesquisadores investigaram consiste em métodos para aprender políticas parciais em diversas tarefas através da identificação de subobjetivos nelas.

Alguns autores (BUTZ; SWARUP; GOLDBERG, 2004; TEMBO et al., 2014) tentaram utilizar políticas parciais para TL através da descoberta dessas políticas parciais com foco de pontos de referência para guiar os agentes mais rapidamente do que treinando-os a aprender uma tarefa por completo. Esses métodos podem permitir a TL se duas tarefas compartilharem subobjetivos e espaços de estados em comum. No entanto, em problemas e tarefas onde não existe um subobjetivo claro ou um ponto de referência, é preferível encontrar pontos comuns e realizar uma combinação (*merge*) entre eles ao invés disso, conforme realizado em (THRUN; SCHWARTZ, 1995; PICKETT; BARTO, 2002). Mas esse tipo de abordagem depende do espaço de estados e não leva em conta similaridades entre objetos e habilidades que podem ser reaproveitadas pelos agentes em tarefas parecidas.

Uma abordagem que utiliza políticas parciais separadamente para diversas tarefas de origem diferentes foi proposta em (MACGLASHAN, 2013). Especificamente, essas políticas parciais são chamadas de "transferíveis", pois são políticas inteiras especificadas por humanos ou aprendidas separadamente e independentemente para cada tarefa e sendo assim podem ser utilizadas, adaptadas e combinadas futuramente de acordo com o design do problema em tarefas maiores, se tornando soluções "parciais" nessas tarefas maiores. Entretanto, as soluções iniciais e a ordem com a que as

tarefas serão aprendidas necessitam ser especificadas por um especialista, não havendo nenhum mapeamento automático entre as diferentes tarefas, funções (R, T) e objetos K . Esse trabalho usou como base a idéia proposta em (MADDEN; HOWLEY, 2004), que tem como foco principal aprender-se tarefas mais simples primeiro e ir aumentando o grau de dificuldade delas, criando uma espécie de um currículo para o agente, onde o mesmo vai adquirindo habilidades e aprimorando seu currículo. Porém, nem sempre usar somente o tamanho ou a complexidade das tarefas é a melhor métrica para se transferir habilidades, uma vez que pode-se aprender algo útil em tarefas mais complexas primeiro e transferir para tarefas menores depois.

Por isso, como em (PICKETT; BARTO, 2002) as tarefas necessitam ser aprendidas no mesmo espaço de estados para que sejam extraídas as partes em comum e é necessário um longo treinamento inicial para prover as políticas subótimas, foi desenvolvido uma extensão dele para trabalhar também com objetos. Essa extensão, chamada de *Portable Policy Blocks* - PPB em (TOPIN et al., 2015) serve para extrair partes em comum com relação à objetos e a habilidades do agente, e não somente com relação ao espaço de estados. Outro ponto positivo é que ela é simétrica, funcionando tanto para a transferência de conhecimento de tarefas mais simples para mais complexas, como o inverso, por conta de seu processo de generalização.

Apesar do sucesso das abordagens iniciais para o uso de políticas parciais em uma única tarefa (SUBRAMANIAN; ISBELL; THOMAZ, 2011; MACGLASHAN, 2013), aprender automaticamente políticas parciais sem supervisão humana e mapeá-las entre diferentes tarefas ainda são problemas difíceis. O algoritmo PPB propõe uma maneira de mapear entre tarefas e um método de pontuação para avaliar a descoberta de políticas parciais antes de armazená-las e reutilizá-las (TOPIN et al., 2015).

O algoritmo original *PolicyBlocks* (PICKETT; BARTO, 2002) precisava aprender por um longo tempo para descobrir políticas parciais úteis apenas em um único espaço de estados. Por esse motivo, esse método não é apropriado para transferir políticas parciais entre tarefas com diferentes espaços de estado. Dado um conjunto de soluções ótimas em algumas tarefas de origem, o algoritmo PPB encontra pontos comuns entre as soluções, construindo um conjunto de políticas parciais com base nisso. Para generalizar o espaço de estados das tarefas, é utilizada uma representação orientada a objetos. Uma visão geral do funcionamento da abordagem utilizada pelo PPB pode ser visualizada na figura 3.2.

Dado um conjunto de políticas (soluções) L para as tarefas de origem $\psi \in \Psi$ e um número i de políticas parciais desejadas, o algoritmo PPB pode ser visto como um processo de 6 passos: **(1)** gera um conjunto de todos os subconjuntos possíveis de L (powerset), gerando apenas triplas, pois conforme evidenciado em (PICKETT;

Figura 3.2 – Visão geral do processo utilizado no PPB (TOPIN et al., 2015). Soluções iniciais são dadas para as tarefas descritas como OO-MDPs. Então o PPB realiza os processos de abstração e descoberta de políticas parciais determinísticas e abstratas, formando um conjunto de tais políticas, as quais são utilizadas durante o aprendizado de uma nova tarefa, realizando um processo de abstração do estado concreto observado na tarefa alvo, seguido da definição da ação concreta indicada pela política parcial abstrata para o estado abstrato correspondente ao estado concreto observado; esta ação é aplicada na tarefa alvo.

BARTO, 2002), a combinação com mais de 3 elementos não provê ganho e eficiência para a técnica; **(2)** para cada subconjunto gerado, abstrai a grande generalização comum (GCG - *Great Common Generalization*), que é o conjunto máximo de objetos que estão presentes em todas as tarefas de origem (por exemplo 1 taxi, 1 bola, 1 hotel e 1 muro) no exemplo da figura 3.2), gerando uma política abstrata mapeada através das tarefas de origem. No exemplo com 2 tarefas com objetos diferentes, deve-se observar a primeira delas (OO-MDP 1) e verificar todas as possíveis distribuições de objetos no mapa (espaço de estados). Para cada estado, olhar a posição que os objetos estão, e procurar todas as possíveis distribuições na segunda tarefa (OO-MDP 2) que tenham os objetos exatamente na mesma posição. No caso da figura 3.2, procurar estado por estado em todo o espaço de estados no OO-MDP 2 e achar os estados que tenham um taxi, uma bola e um hotel na mesma posição que o OO-MDP 1 que está sendo avaliado. Assim, cada conjunto de estados que tiver esses objetos na mesma posição, irá formar um estado abstrato s_{ab} , que é um conjunto de estados concretos inclusos em sua definição; **(3)** mescla em ordem crescente de quantidade de objetos (primeiro pega-se a tarefa com menos objetos, depois a segunda menor e etc) todas as políticas contidas em cada subconjunto de tarefas em políticas parciais candidatas abstratas. Compara a ação indicada pela política abstrata com as ações sugeridas pelas políticas concretas iniciais. **(4)** Após a comparação, **pontua** todas as candidatas a política parcial de acordo com sua similaridade com as soluções L , verificando quantos pares estado-ação são iguais na política abstrata e nas tarefas de origem; **(5)** **subtrai** a política parcial candidata de maior pontuação do conjunto inicial de tarefas de origem

para evitar redundância nas próximas políticas parciais a serem descobertas; e (6) adiciona a melhor política parcial candidata c ao conjunto de políticas parciais O . Esse processo é repetido até que o algoritmo encontre i políticas parciais ou até não haver mais políticas em L . Após o final do algoritmo, as i políticas parciais abstratas de melhor avaliação serão retornadas. As políticas parciais resultantes são determinísticas. Ao utilizar essas políticas descobertas na tarefa destino, deve-se realizar a abstração dos estados concretos observados. Esse processo é o mesmo da fase inicial de abstração. Primeiro converte-se o estado pra abstrato eliminando os objetos a mais, e então o algoritmo lê qual a ação sugerida para a lista de estados, para então aplicá-la na tarefa destino (no estado concreto observado). Contudo, esse processo pode ser um pouco lento, pois o PPB guarda na memória todos os valores Q calculados no aprendizado das tarefas de origem para determinar a melhor ação na política parcial determinística, tirando a média entre os valores Q para selecionar a ação que tem o valor Q maior.

Figura 3.3 – Processo de abstração utilizado no PPB (TOPIN et al., 2015). Supondo que o valor Q do taxi mover-se para a direita no estado OO-MDP 1 seja 0,75 e que no estado do OO-MDP 2 incluído no mesmo estado abstrato seja 0,25, o valor Q do estado abstrato correspondentes a ambos estados concretos será a média desses 2 valores, o que neste caso seria 0,5.

As políticas parciais abstratas retornadas podem então ser reutilizadas em uma nova tarefa. Para isso, as políticas parciais são adicionadas ao conjunto de ações A da tarefa destino como uma possível ação a ser escolhida no processo de decisão. Quando o agente seleciona a política parcial, a política abstrata será executada encontrando um estado abstrato que tenha o maior número possível de objetos em comum com o estado atual observado na nova tarefa.

Mesmo permitindo a reutilização de políticas parciais aprendidas transferindo-se várias políticas parciais para uma nova tarefa, a abordagem PPB pode até resultar em transferência negativa, pois se o agente for praticamente obrigado a seguir a política descoberta até encontrar um estado indefinido, ela nem sempre poderá ser a melhor.

Uma abordagem para reutilização probabilística de políticas parciais foi proposta por Bernstein (BERNSTEIN, 1999). Esse trabalho segue a intuição de que a probabilidade de um agente se comportar de certa maneira deve ser proporcional à frequência com que esse comportamento foi bem-sucedido no passado. Assim, quando uma política parcial está sendo executada (reutilizada), a probabilidade de que uma ação seja escolhida em um determinado estado está relacionada ao número de políticas (ou políticas parciais) nas quais a ação é escolhida naquele estado, em tarefas prévias. Dado um conjunto de políticas estocásticas de tarefas prévias $\pi_1, \pi_2, \dots, \pi_m$, a probabilidade de que uma ação seja escolhida para reuso em um determinado estado (concreto) observado na nova tarefa é dada por π_M sendo a *política combinada de decisão* formada pela média da probabilidade de cada política escolher certa ação a e um determinado estado s :

$$\pi_M(a|s) = \frac{1}{m}(\pi_1(a|s) + \pi_2(a|s) + \dots + \pi_m(a|s)). \quad (3.1)$$

3.3 Conclusão do Capítulo

O aprendizado com relação a objetos permite ao agente desenvolver habilidades que podem ser facilmente transferidas, mas exige considerações especiais de projeto e mais tempo de aprendizado adicional do que a aprendizagem de políticas parciais de outras maneiras, como por exemplo considerando somente o espaço de estados. Sendo assim, enquanto objetos podem ser benéficos para o aprendizado e para a transferência do conhecimento, não resolvem o problema de descobrir e escolher novas políticas parciais de forma efetiva e rápida, além de que não há nenhuma combinação entre essas políticas para se saber se determinada solução descoberta é boa ou não. Um ponto positivo é o fato de poder-se aprender diversas tarefas e transferir conhecimento adquirido de todas essas tarefas para uma nova.

Apesar de haver técnicas que abstraem objetos da mesma classe, e possibilitem a TL, o agente pode acabar ficando viciado em soluções ruins previamente aprendidas, podendo chegar em uma solução até pior do que sem o uso de políticas parciais e de TL, uma vez que não leva em consideração determinadas soluções e características de um problema. E também, se o agente confrontar-se com problemas mais complexos e reais, que são maiores, com múltiplos objetivos ou quando o agente não tem um conhecimento sobre todo o MDP, acaba sendo inviável uma representação apenas da tabela de estados da política aprendida.

Assim, argumenta-se que as políticas parciais devem ser utilizadas probabilisticamente. Acredita-se também que as soluções de políticas parciais obtidas para múltiplos

tiplas tarefas origem e orientadas a objetos podem ser mais facilmente usadas para acelerar a aprendizagem e transferir comportamento aprendido nessas tarefas.

Por fim, o capítulo é finalizado com a tabela 3.1 que faz uma listagem comparativa dos trabalhos relacionados. Os símbolos usados em tal tabela estão descritos na tabela 3.1.

Tabela 3.1 – Lista de abreviações usadas na tabela 3.2.

Diferenças de Tarefas Permitidas	
b^0	A distribuição inicial da probabilidade do estado pode diferir
K	Objetos (conjuntos constantes) podem diferir
Ω	problemas podem diferir
R	Função de recompensa ou estado objetivo podem diferir
T	Função de Transição pode diferir
map	Qualquer coisa pode diferir, havendo mapeamento
Tarefas de Origem	
all	Todas tarefas de origem são usadas
h	Somente uma tarefa de origem selecionada por um humano é permitida
bib	Biblioteca contendo um subconjunto de todas as tarefas de origem
Transferência de Conhecimento (TL)	
Q	Função de valor-ação
π_{abs}^{stoch}	Política estocástica abstrata
π_p	Política parcial
π_p^{stoch}	Política parcial estocástica
Métodos de Aprendizado	
Batch	Aprendizado Batch
CL	Classificação
DP	Programação Dinâmica
H	Aprendizado Hierárquico
TD	Diferença Temporal

Tabela 3.2 – Trabalhos relacionados contendo métodos de TL e RL, classificados em termos de quatro dimensões: Diferenças de tarefas permitidas; tarefas de origem; transferência de conhecimento e métodos de aprendizagem. Os símbolos usados estão apresentados na tabela 3.1

Trabalho Relacionado	Diferenças de Tarefas Permitidas	Tarefas de Origem	TL	Métodos de Aprendizagem
<i>Mesmo Espaço de Estados e Ações</i>				
Selfridge et al. (1985)	T	h	Q	TD
Thrun et al. (1995)	b^0	h	π_p	TD
Asada et al. (1996)	b^0	h	Q	TD
Bowling and Veloso (1998)	b^0, T	bib	π	TD
Bernstein (1999)	b^0, T	h	π_p	TD
McGovern and Barto (2001)	R, b^0	all	Q, π_p	CL,TD
Pickett and Barto (2002)	b^0, T	bib	π_p	DP,TD
<i>Aprendizado Hierárquico</i>				
Asadi and Huber (2007)	R	h	π_p	TD,H
Konidaris et al. (2012)	Ω	h	π_p	TD,H
<i>Múltiplas Tarefas de Origem</i>				
Fernandez and Veloso (2006)	R, b^0	bib	π	TD
Konidaris and Barto(2007)	map	all	π_p	TD,H
Subramanian et al. (2011)	R, b^0	bib	Q, π_p	TD,H
Silva et al. (2012)	Ω	all	π_{abs}^{stoch}	TD
MacGlashan (2013)	K	bib,h	π_p	TD,H
Topin et al. (2015)	map	all	π_p^{abs}	TD,H
Koga et al. (2015)	Ω	bib	π_{abs}	TD
Silva et al. (2015)	map	bib	π	TD
<i>Métodos Propostos neste trabalho</i>				
Multiobjective Options (2017)	b^0, C, R	all	π_p	TD,H
Probabilistic Reuse of Discovered Options (2018)	map	all	π_p^{stoch}	TD,H

4 PROPOSTA

Este trabalho propõe um método para gerar e combinar de forma probabilística políticas parciais aprendidas para diferentes tarefas previamente resolvidas, disponibilizando tais políticas a um agente aprendiz de uma nova tarefa, visando acelerar seu aprendizado nesta nova tarefa similar às anteriormente resolvidas.

(FERNANDEZ; VELOSO, 2006) propuseram o uso de uma biblioteca de políticas aprendidas previamente em tarefas similares. Tais políticas são determinísticas e completas e as tarefas, episódicas. A finalidade é reusar de forma probabilística as políticas da biblioteca, visando acelerar o novo aprendizado. Deste trabalho veio a inspiração de usar um conjunto de políticas previamente aprendidas em tarefas similares no aprendizado de uma nova tarefa, para, assim, direcionar a busca da solução em partes mais prováveis de serem úteis do espaço de soluções.

Já (KOGA; SILVA; COSTA, 2015) exploram a idéia de que a generalização de soluções de tarefas previamente resolvidas e diretamente relacionadas à nova tarefa podem produzir políticas estocásticas que melhor direcionam o aprendizado na nova tarefa alvo. (BERNSTEIN, 1999) propõe um método para combinar e reusar políticas previamente aprendidas em uma política probabilística por uma quantidade de passos n . Estes dois trabalhos inspiraram o uso de transferência de uma única política probabilística para agilizar o aprendizado da nova tarefa.

Em (KONIDARIS; BARTO, 2007; KONIDARIS; SCHEIDWASSER; BARTO, 2012) foi discutido também sobre a portabilidade das políticas aprendidas, tanto em relação ao espaço do agente, como do ponto de vista do espaço do problema. (SILVA; PEREIRA; COSTA, 2012) propõem o uso de políticas probabilísticas em uma representação relacional para a transferência de políticas entre tarefas. Estes trabalhos inspiraram a busca por representações mais poderosas para a transferência de conhecimento entre as tarefas.

Finalmente, (SILVA; COSTA, 2018) e (TOPIN et al., 2015) usam uma representação de OO-MDP para efetuarem a transferência e aqui esta modelagem foi adotada. Ainda, (TOPIN et al., 2015) propõem um método de descoberta de políticas parciais, o que também foi aqui adotado.

Enquanto que soluções parciais com relação ao ponto de vista do de objetos são muito portáteis pois funcionam independente de problema e cenário, elas podem acabar sendo não tão interessantes se não houver muitas relações entre ações e objetos disponíveis no problema, já que serão tarefas diferentes. Por exemplo no *Goldmine*

(figura 4.1), de nada adianta o mineiro saber o que fazer quando estiver em um estado adjacente a uma célula que contém ouro, se na nova tarefa sua missão não for coletar ouro ou se determinado estado que ele já aprendeu não existir.

Figura 4.1 – *Goldmine*. Neste ambiente, o mineiro deve apanhar pedaços de ouro. O episódio termina somente quando ele coleta todos os pedaços. Adaptado de (DIUK; COHEN; LITTMAN, 2008).

Ao mesmo tempo, soluções que usam somente o ponto de vista do problema são muito específicas e restringem a transferência de conhecimento entre tarefas que tenham funções de transição e metas muito diferentes.

Sendo assim, o argumento nesta dissertação é que:

- o uso de políticas parciais pode ser melhor do que de políticas completas para a transferência de conhecimento pois as políticas parciais representam partes da solução, encapsulando comportamentos necessários para resolver muitos problemas relacionados, além de não ser necessário aprender soluções inteiras que normalmente são mais custosas;
- o uso de políticas probabilísticas para transferência de conhecimento são melhores do que o de políticas determinísticas, pois elas oferecem mais opções de escolha ao agente, permitindo melhor adaptação no aprendizado de uma nova e desconhecida tarefa;
- o uso de representações mais poderosas, como o OO-MDP orientado a objetos, permite um mapeamento entre tarefas de forma mais fácil e intuitiva;
- o uso de abstração na representação do conhecimento a ser transferido permite maior abrangência e facilidade no seu uso.

O desafio de integrar em um único arcabouço a transferência de políticas parciais probabilísticas e abstratas de várias tarefas de origem para uma tarefa de destino,

visando acelerar o aprendizado, foi pouco investigado na literatura. Apesar das abordagens bem-sucedidas descritas no capítulo anterior, nenhuma delas fornece simultaneamente: (1) um método automático de descoberta de políticas em múltiplas tarefas de origem orientadas a objetos; (2) o uso de soluções abstratas para transferência; (3) uma combinação probabilística das soluções aprendidas; (4) a reutilização das políticas probabilísticas e abstratas previamente aprendidas em novas tarefas visando controlar a exploração do agente e agilizar o novo aprendizado.

Por isso, este trabalho contribui com um arcabouço denominado PRDO (*Probabilistic Reuse of Discovered Options*) visando combinar políticas parciais aprendidas de diferentes tarefas, com o intuito de fornecer um bom conjunto de ações de decisão ao agente que está aprendendo uma nova tarefa, sem tirar a possibilidade do agente visitar estados ainda inexplorados e evitando que ele fique preso em determinadas regiões específicas na tarefa destino.

A finalidade do PRDO é aprender políticas parciais para múltiplas tarefas de origem (onde os algoritmos *PolicyBlocks* e sua extensão PPB, por exemplo, podem ser usados por serem gerais e gravarem partes em comum entre tarefas e pelo fato de o segundo trabalhar com relação a objetos), combiná-las em uma política abstrata probabilística, transferí-la e instanciá-la no espaço de estados da nova tarefa de destino.

A proposta PRDO é esboçada na figura 4.2 e descrita no Algoritmo 1.

Figura 4.2 – Visão geral do processo utilizado pela proposta PRDO (BONINI et al., 2018): inicialmente PRDO utiliza o algoritmo PPB para descoberta de políticas parciais determinísticas e abstratas das tarefas de origem. PRDO combina estas políticas abstratas em uma política probabilística e, usando uma função de *grounding* dos estados abstratos da política probabilística para os estados da tarefa de destino, transforma em uma política π_o que norteará a exploração do agente na busca pela solução na tarefa de destino.

Como entrada PRDO requer um conjunto de tarefas de origem Ψ_s e suas respectivas soluções, as quais podem ser aprendidas ou informadas por humanos. Requer

		a_1	a_2	a_3			a_1	a_2	a_3
$\pi_1 =$	s_1	1	0	0	$\pi_2 =$	s_1	0	1	0
	s_2	1	0	0		s_2	1	0	0

Figura 4.3 – Exemplo da combinação efetuada: $\pi_1(s)$ e $\pi_2(s)$ são políticas parciais determinísticas previamente descobertas e $\pi_o(a|s)$ é a política probabilística resultante da combinação de π_1 e π_2 .

ainda um número i de políticas parciais a serem descobertas, um número de passos n que indica por quantos passos as políticas parciais descobertas serão executadas na tarefa de destino e uma tarefa de destino ψ_t .

Inicialmente o algoritmo define como vazio o conjunto de políticas parciais O no passo 1. Então, nos passos 2-4, executa um algoritmo de descoberta de políticas parciais. Foi utilizado o algoritmo PPB (TOPIN et al., 2015), mas qualquer outro algoritmo de descoberta de políticas parciais também pode ser usado aqui. Este algoritmo retorna i políticas parciais abstratas para as tarefas origem, incluindo todas elas em O . Depois disso, as políticas das políticas parciais descobertas são combinadas no passo 5 em uma política probabilística abstrata π_o^{ab} , conforme ilustra o exemplo da figura 4.3. As políticas parciais são combinadas de uma forma que π_o escolhe ações com uma probabilidade de acordo com o número de vezes que elas aparecem nas políticas parciais anteriores e com o número de ações disponíveis no domínio; quanto mais a ação aparece, maior é sua probabilidade de escolha. Considere duas políticas em O , π_1 e π_2 , que determinam $\pi_1(s_1) = a_1$ e $\pi_1(s_2) = a_1$, $\pi_2(s_1) = a_2$ e $\pi_2(s_2) = a_1$. Como $|A| = 3$, com $A = \{a_1, a_2, a_3\}$, e como $|O| = 2$, considera-se que cada ação inicialmente tem uma probabilidade de $1/(|O| + |A|)$ ser executada, caso nenhuma política em O a recomende. Observe que $\pi_o(a_3|s_1) = 1/5$, $\pi_o(a_2|s_2) = \pi_o(a_3|s_2) = 1/5$. Entretanto, quando uma política em O recomenda uma ação para aquele estado, a probabilidade daquela ação ser recomendada aumenta proporcionalmente sua chance de ser recomendada. Assim, como somente π_1 recomendou a_1 para s_1 , $\pi_o(a_1|s_1) = 2/5$. Similarmente, como π_1 e π_2 recomendaram a_1 para s_2 , $\pi_o(a_1|s_2) = 3/5$.

Em seguida, no passo 6, a política probabilística π_o^{ab} é instanciada no espaço de estados da tarefa de destino ψ_t , resultando na política probabilística concreta $\pi_o(a|s)$, com $a \in A_{\psi_t}$ e $s \in S_{\psi_t}$.

O processo de aprendizado de ψ_t é então executado nos passos 7-9, com uso da política política probabilística concreta $\pi_o(a|s)$. Aqui, usa-se a estratégia ϵ -greedy para a seleção das ações de treino, sendo que com probabilidade ϵ , $\pi_o(a|s)$ é escolhida; a ação de fato executada depende da escolha probabilística feita para o estado observado s . Com isso, uma tendência de escolha maior é dada para ações que tiveram sucesso na solução de tarefas semelhantes.

Algoritmo 1 PRDO

▷ Dados de Entrada: Conjunto $Sol : \langle \psi_s, \pi_s \rangle$ de tarefas de origem ψ_s e suas respectivas soluções π_s , número desejado de políticas parciais i , número de passos n para a política transferida ser executada e a tarefa de destino ψ_t

- 1: $O \leftarrow \emptyset$ //conjunto vazio de políticas parciais
- 2: **para cada** tarefa de origem $\psi_s \in \Psi_s$ **faça**
- 3: $O \leftarrow Descobre\ as\ Políticas\ Parciais(\psi_t, i, Sol)$ //p. ex. usando PPB
- 4: **fim para**
- 5: $\pi_o^{ab} \leftarrow Combina\ probabilisticamente\ (O)$
- 6: $\pi_o \leftarrow Grounding\ (\psi_t, \pi_o^{ab})$
- 7: **para** H episódios **faça**
- 8: $\pi_t \leftarrow Aprende\ (< \pi_o, n, \psi_t >)$
- 9: **fim para**
- 10: **return** π_t //solução da tarefa de destino

5 EXPERIMENTOS E RESULTADOS

Para avaliar o arcabouço proposto foram executados alguns experimentos. Todos foram realizados em Java, implementados usando BURLAP (MACGLASHAN, 2015). Este capítulo descreve os principais experimentos realizados. Os resultados desses e outros experimentos foram publicados em artigos de conferências (BONINI; SILVA; COSTA, 2017), (BONINI et al., 2017), (BONINI et al., 2017), (BONINI et al., 2018).

Assume-se que as tarefas de origem e de destino são estacionárias e discretas, e que todas pertencem ao mesmo domínio.

Inicialmente foram investigadas técnicas de descoberta de políticas parciais para tarefas multiobjetivo, que deram origem às publicações (BONINI; SILVA; COSTA, 2017; BONINI et al., 2017). Porém, como o foco do trabalho consiste em avaliar a transferência de conhecimento, excluiu-se a característica multiobjetivo das tarefas para que os resultados não fossem influenciados por outros fatores. Outro fator que foi verificado nessas publicações e em (BONINI et al., 2017), descrito no primeiro experimento, é que o algoritmo utilizado para a descoberta de políticas parciais *PolicyBlocks*, funciona somente para o mesmo espaço de estados, e sendo assim não é adequado para TL. Com isso, decidiu-se utilizar sua extensão PPB a fim de gerar políticas parciais mais eficientes.

5.1 Experimento 1 – Transferência em um Mesmo Espaço de Estados

Os experimentos dessa fase foram realizados em um *Gridworld* 11x11. As tarefas de destino e de origem possuem os mesmos espaços de ação e de estados. Portanto, a abstração em objetos não foi aqui realizada. O objetivo foi avaliar o uso de políticas probabilísticas para a transferência.

O agente começa em um estado não-terminal aleatório e deve realizar várias tarefas de origem (uma tarefa por vez) com uma posição de objetivo (estado final) diferente a ser alcançada em cada uma delas. O agente tem que aprender a alcançar o estado objetivo o mais rápido possível de forma independente em 6 tarefas de origem (figura 5.1). Foi executado o algoritmo *Q-learning* durante 1000 episódios, fornecendo políticas ótimas para cada tarefa de origem.

Como usou-se modelagem MDP (e não OO-MDP), o algoritmo adotado para descoberta de políticas parciais foi o *PolicyBlocks – PB* (PICKETT; BARTO, 2002), que atua com as tarefas no mesmo espaço de ações e de estados. O PB foi aplicado

nas tarefas de origem e suas soluções, extraindo 3 políticas parciais para cada tarefa. Depois disso, essas políticas parciais são usadas probabilisticamente e avaliadas em 6 tarefas alvo diferentes (figura 5.2).

Figura 5.1 – Tarefas de origem, onde o círculo é o agente e o quadrado em amarelo (G) é o objetivo para cada tarefa. Adaptado de (BONINI et al., 2017).

Figura 5.2 – Tarefas alvo, onde o círculo é o agente e o quadrado em amarelo (G) é o objetivo para cada tarefa. Adaptado de (BONINI et al., 2017).

O conjunto de ações disponível é dado por $A = \{\text{norte}, \text{sul}, \text{leste}, \text{oeste}\}$. Os episódios terminam quando o agente atinge o estado objetivo, resultando em uma recompensa de +1. A recompensa é 0 para qualquer outro estado. Também foi adotada a taxa de aprendizado $\alpha = 0,2$ e a taxa de desconto $\gamma = 0,9$.

Para avaliar a eficácia relativa ao reuso probabilístico das políticas parciais aprendidas, foram realizados 1000 episódios de aprendizagem na tarefa de destino, usando o algoritmo Q-Learning sem uso de políticas parciais e outros 1000 episódios de aprendizado na tarefa de destino utilizando PRDO sobre as políticas parciais aprendidas. O reuso se dá com o parâmetro $n = 1$ (PRDO 1-passo), fazendo com que os valores das ações na política combinada sejam atualizados a cada passo.

A figura 5.3 mostra a recompensa média em 1000 repetições da experiência. O PRDO superou o *Q-learning* padrão, aprendendo mais rápido no início da aprendizagem, conseguindo uma recompensa média de 0,30 após apenas 10 episódios.

O *Q-learning* padrão sem políticas parciais não conseguiu resultados semelhantes mesmo após 1000 episódios de aprendizagem.

Essa diferença entre a recompensa cumulativa média indica que PRDO oferece uma aceleração principalmente no início processo de aprendizagem, melhorando consideravelmente o desempenho nos episódios iniciais de aprendizagem, ao mesmo tempo que fornece decisões alternativas ao agente, já que o mesmo não é 100% forçado a escolher uma solução devido às ações serem escolhidas probabilisticamente.

A principal contribuição deste experimento é a conclusão de que as políticas parciais probabilísticas proporcionam aos agentes boas alternativas baseadas no conhecimento prévio. Entretanto, esse experimento foi realizado em um cenário simples, apenas mudando-se as posições iniciais do agente e do objetivo. Se o espaço de es-

Figura 5.3 – A recompensa média por 1000 episódios durante o processo de aprendizagem (BONINI et al., 2017).

tados for diferente, a abordagem de descoberta de políticas parciais PB não funciona adequadamente, pois não haverá correspondência de estado entre as tarefas, já que alguns estados não serão iguais. Com isso, foi verificado que sua extensão PPB poderia ser mais efetiva para a TL por usar a modelagem OO-MDP e descobrir políticas abstratas. Os benefícios do uso de PPB são mostrados nos experimentos 2 e 3, que geraram a publicação (BONINI et al., 2018).

5.2 Experimento 2 – Uso Controlado de Objetos

Os experimentos dessa fase foram realizados em um *Gridworld* 11x11, no qual o agente começa em um estado não-terminal aleatório e deve realizar várias tarefas de origem (uma tarefa por vez) com uma posição de objetivo (estado final) diferente a ser alcançada em cada uma delas. Aqui, usa-se a modelagem OO-MDP, porém as classes de objetos não mudam entre as tarefas, apenas muda-se a quantidade de objetos de cada classe. Os muros não variam de posição, apenas pode-se ter todos os muros (conforme na figura 5.4) ou uma quantidade menor de muros, com alguns deles não presentes. Como a quantidade de objetos varia, o espaço de estados S e a função de transição T também variam.

Em primeiro lugar, o agente tem que aprender a alcançar o estado objetivo o mais rápido possível em 10 tarefas de origem (figura 5.4) e, após obter as soluções iniciais, as políticas parciais são descobertas e armazenadas usando o PPB. Os experimentos foram realizados em 10 diferentes tarefas de origem e 1 tarefa alvo, para 20 testes (configurações diferentes).

Em cada um dos 20 testes, foi executado o algoritmo *Q-learning* por 100 episódios para cada tarefa de origem, fornecendo políticas ótimas para cada tarefa de origem e passando-as para o algoritmo PPB, que extrai 3 políticas parciais para todas as tarefas de origem. Em seguida, essas políticas parciais que foram armazenadas em O e combinadas probabilisticamente em π_o , usando o processo descrito no capítulo anterior. Esse processo foi avaliado em 1 tarefa alvo diferente para cada tentativa (figura 5.4).

Optou-se pelo algoritmo PPB para beneficiar o arcabouço do uso de objetos (utilizados aqui, mas mais explorados no experimento seguinte).

Figura 5.4 – Exemplo de tarefa no GridWorld. O agente é representado pela circunferência cinza, o objetivo pelo quadrado amarelo (G) e os obstáculos (muros) estão em preto. Adaptado de (BONINI et al., 2018).

O conjunto de ações disponível é composto por $A = \{norte, sul, leste, oeste\}$. Os episódios terminam quando o agente atinge o estado objetivo, resultando em uma recompensa de +1; para qualquer outro estado a recompensa é 0. Também foi adotada a taxa de aprendizado $\alpha = 0,2$ e a taxa de desconto $\gamma = 0,9$.

A fim de avaliar a eficácia relativa do reuso probabilístico das políticas parciais aprendidas, foram executados 100 episódios de aprendizagem na tarefa alvo e verificada a quantidade média de passos médios para atingir o objetivo nas 20 execuções.

A figura 5.5 mostra a quantidade média de passos para atingir a meta nas 20 tentativas do experimento no Gridworld, que tem tarefas um pouco semelhantes. O PRDO com políticas parciais de 5 passos superou o Q-Learning, o PPB padrão (sem combinação probabilística das políticas) e as políticas parciais de 1 passo, aprendendo mais rápido no início do processo de aprendizagem. Isso aconteceu porque as políticas parciais de 5 passos fornecem ao agente um menor tempo de exploração, sendo mais direto do que as políticas parciais de 1 passo e do que o PPB. Como as tarefas são relativamente similares (origem e destino), mostra-se vantajoso seguir por mais tempo a política transferida. Também é possível observar que todos os 3 casos for-

necem conhecimento prévio, o que é melhor do que apenas aprender do zero usando Q-Learning.

Figura 5.5 – Média de passos para atingir o objetivo em 100 episódios durante 20 execuções do processo de aprendizagem nas tarefas destino do GridWorld (BONINI et al., 2018).

A principal contribuição deste experimento é a conclusão de que não só se confirma que as políticas parciais probabilísticas proporcionam aos agentes boas alternativas baseadas no conhecimento prévio, mas também que é possível a transferência entre tarefas que alterem o número de objetos entre si (alterando, consequentemente, a função de transição e o espaço de estados das mesmas).

5.3 Experimento 3 – Tarefas Mais Gerais no OO-MDP

No terceiro experimento, foi testado o PRDO em uma tarefa mais complexa. O teste foi em um ambiente contendo um taxi, onde passageiros (objetos) devem ser apanhados antes do taxi atingir um estado objetivo (*TaxiWorld*).

O agente (táxi) começa em um estado não terminal aleatório e deve executar 10 tarefas de origem (uma tarefa por vez) com uma posição de objetivo diferente a ser alcançada em cada uma delas, com passageiros para pegar em diferentes posições (ver figura 5.6). Os muros não variam de posição, apenas pode-se ter todos os muros (conforme na figura 5.6) ou menos muros, com alguns deles não presentes. Assim, como no experimento anterior, aqui as classes de objetos não mudam, apenas muda-

se a quantidade de objetos de cada classe. Como a quantidade de objetos varia, o espaço de estados S e a função de transição T também variam.

Na primeira avaliação, o número de passageiros disponíveis nas primeiras tarefas de origem (figura 5.7) é de 4 e na tarefa de destino é de 2, para verificar um cenário ao aprender tarefas mais difíceis e transferi-las para tarefas mais fáceis.

Na segunda avaliação (figura 5.8), o número de passageiros disponíveis nas tarefas de origem é 2 e na tarefa de destino é 4, como sugerido em (MADDEN; HOWLEY, 2004; MACGLASHAN, 2013; SILVA; COSTA, 2018), aprendendo em tarefas mais fáceis e transferindo para tarefas mais difíceis.

Essas avaliações diferentes foram feitas para verificar a simetria da estrutura, que funciona de tarefas maiores para uma tarefa menor e de tarefas menores para uma maior por conta do processo de generalização utilizado. O conjunto de ações disponível nessas tarefas é $A = \{norte, sul, leste, oeste, pickup, dropoff\}$. O agente pode pegar (*pickup*) um passageiro quando está na mesma posição que ele e pode deixar um passageiro (*dropoff*) em qualquer posição, e os episódios terminam apenas quando o agente alcança o estado de objetivo (meta) depois de pegar todos os passageiros, resultando em uma recompensa de +1 com desconto de $\gamma = 0,9$ e, caso contrário, a recompensa é 0 para qualquer passo. No caso de se mover contra uma parede, o agente permanece na mesma posição.

Figura 5.6 – No *Taxiworld*, o agente é representado pelo carro (táxi), o objetivo pelo quadrado em amarelo contendo (G), os obstáculos (muros) estão em preto e os passageiros que o agente pode pegar estão em vermelho (P). O agente deve pegar todos os passageiros antes de chegar ao objetivo. Adaptado de (BONINI et al., 2018).

Aqui, como no primeiro experimento, o agente precisa atingir a posição de meta o mais rápido possível, pegando todos os passageiros antes de chegar ao objetivo.

Em cada um dos 20 testes, assim como no experimento anterior, para extrair as políticas parciais, executa-se inicialmente o algoritmo Q-learning por 1000 episódios para cada tarefa de origem, fornecendo políticas ótimas para cada uma das tarefas

de origem e passando para PPB extrair 3 políticas parciais para todas as tarefa de origem.

Depois disso, essas políticas parciais que foram adicionadas em O e combinadas em π_o usando o processo conforme descreve a figura 4.3 e avaliadas em uma tarefa de destino ψ_t .

Foram executados 10000 episódios de aprendizado para todos os casos na tarefa de destino e foi contabilizada a quantidade média de passos para atingir o objetivo dentre 20 execuções. Devido ao custo computacional, cada episódio foi limitado a 10000 etapas (*steps*).

A figura 5.7 mostra a quantidade média de passos para chegar ao objetivo em 20 execuções do experimento no *TaxiWorld* de tarefas maiores para uma menor e a figura 5.8 mostra os resultados de tarefas menores para uma maior. Em ambos os casos, o PRDO com políticas parciais de 1-passo foi o melhor, superando o Q-Learning Padrão, PRDO com políticas parciais de 5-passos e as políticas parciais de PPB.

Isso ocorreu porque as políticas parciais de 1-passo fornecem ao agente mais possibilidades de exploração que as de 5-passos e do que o PPB padrão, o que é esperado em uma tarefa um pouco mais diferente e mais difícil. Todos os algoritmos novamente foram melhores que o Q-Learning padrão. É importante observar também que uma exploração gananciosa nem sempre é a melhor solução, porque os passageiros (objetos) mudam de posição (podem iniciar em qualquer posição). Por isso, até mesmo em tarefas um pouco semelhantes, quando a complexidade da tarefa é alterada, essas abordagens (PPB e políticas parciais de 5 ou mais passos) não funcionam muito bem se não tiverem soluções iniciais muito boas e permitirem atualizações em suas políticas, já que forçam o agente a seguir políticas não adequadas por um determinado número de passos maior do que o necessário.

Normalmente, no mundo real, as tarefas são diferentes e mais complexas, então a idéia de usar políticas parciais muito longas de n -passos pode fornecer transferência negativa de conhecimento, ao invés de ajudar o agente. Assim, foi verificado que o parâmetro n deve ser balanceado e alterado de acordo com o problema, talvez avaliando alguma forma de similaridade entre as tarefas anteriores.

Figura 5.7 – A quantidade média de passos até atingir o objetivo durante 10000 episódios durante 20 execuções no processo de treinamento na tarefa destino do *TaxiWorld*. 4 passageiros nas tarefas origem e 2 passageiros na tarefa destino (BONINI et al., 2018).

Figura 5.8 – A quantidade média de passos até atingir o objetivo durante 10000 episódios durante 20 execuções no processo de treinamento na tarefa destino do *TaxiWorld*. 2 passageiros nas tarefas origem, 4 passageiros na tarefa destino (BONINI et al., 2018).

5.4 Conclusão do Capítulo

Durante o desenvolvimento deste trabalho adotaram-se diversos benefícios providos por trabalhos prévios na literatura. A proposta de (MACGLASHAN, 2013) foi muito

interessante, pois o aprendizado das políticas parciais ocorre em múltiplas tarefas de origem e com isso, pode-se obter soluções mais variadas e interessantes para realizar a TL do que somente em uma tarefa. O algoritmo de descoberta automática e transferência de políticas parciais proposto em (PICKETT; BARTO, 2002) e sua extensão PPB (TOPIN et al., 2015) são bem gerais e fornecem boas soluções parciais iniciais, além de o segundo também prover um modo eficiente de mapear objetos entre tarefas distintas e domínios distintos. As ideias propostas em (KONIDARIS; BARTO, 2007; KONIDARIS; SCHEIDWASSER; BARTO, 2012) também são muito interessantes pois criam uma abstração de utilização de políticas parciais tanto no ponto de vista do problema quanto do ponto de vista do agente, dando um passo inicial importante para a representação orientada à objetos proposta em (DIUK; COHEN; LITTMAN, 2008). Além disso, para o agente não ficar viciado em soluções que podem produzir transferência negativa, aplica-se o conceito de probabilidade sobre soluções passadas (BERNSTEIN, 1999; FERNANDEZ; VELOSO, 2006; KOGA; SILVA; COSTA, 2015; GLATT; SILVA; COSTA, 2017).

Sendo assim, uma das principais contribuições do arcabouço PRDO é que ele proporciona aos agentes boas alternativas baseadas em seu conhecimento prévio obtido de múltiplas tarefas (MACGLASHAN, 2013), de forma automática (TOPIN et al., 2015); realiza a transferência desse conhecimento através de um mapeamento entre objetos (TOPIN et al., 2015) e por fim reutiliza esse conhecimento transferido em uma nova tarefa de maneira probabilística (BERNSTEIN, 1999; FERNANDEZ; VELOSO, 2006; KOGA; SILVA; COSTA, 2015). Os experimentos com o algoritmo PRDO no *Gridworld* e no *Taxiworld* mostraram que essa abordagem provou-se promissora tanto para acelerar a aprendizagem quanto para guiar os agentes para boas soluções em tarefas RL.

Como extensão, seria interessante unir o conhecimento de humanos com o conhecimento dos agentes antes de realizar a TL, durante sua realização e após sua realização. Dessa forma, com a interferência humana ela poderá ser mais efetiva. Assim, pode-se avaliar/classificar as políticas parciais antes de combiná-las ou transferi-las, comparar políticas parciais probabilísticas com políticas não determinísticas (KOGA; SILVA; COSTA, 2015) e com o conhecimento provido de especialistas humanos (PENG et al., 2016). Finalmente, também é necessário avaliar essas abordagens em problemas mais complexos e realistas, como alguns de *Smart Grid*, Navegação, *Flow Shop*, Mídias Sociais, Jogos e etc.

6 CONCLUSÃO E TRABALHOS FUTUROS

Uma das principais contribuições deste trabalho é o fato de as políticas parciais combinadas probabilisticamente proporcionam aos agentes boas alternativas baseadas no conhecimento prévio.

Outro ponto chave deste trabalho é a reutilização probabilística de políticas parciais orientadas a objetos, uma vez que assim não é necessário que as tarefas sejam as mesmas, ou possuam o mesmo espaço de estados, basta que elas compartilhem objetos e atributos em comum.

Os experimentos no problema *GridWorld* e no problema *TaxiWorld* mostram que a estrutura proposta é promissora tanto para acelerar o aprendizado quanto para orientar o agente com boas soluções em problemas de RL.

No futuro, uma boa maneira de estender essa abordagem é avaliar melhor as soluções iniciais abstraídas pela abordagem PPB. Uma maneira possível de fazer isso seria utilizar a mesma idéia do processo da figura 4.3 proposta neste trabalho, ao invés de se usar a média da soma dos Q-Values para criar os estados abstratos, já que o procedimento usado em PPB verifica todos os estados onde cada objeto pode estar e isso é lento e bastante custoso para abstrair apenas uma média de soma dos Q-Values. Por isso, uma abordagem probabilística poderia evitar ou melhorar se utilizar de toda essa varredura.. Além disso, um parâmetro ou métrica de similaridade, parecido com a GCG, entre as tarefas para definir o número de etapas n automaticamente ao reutilizar as políticas também seria interessante.

Outra possibilidade é avaliar a abordagem em tarefas mais complexas, como tarefas contínuas (não estacionárias) (KONIDARIS; BARTO, 2009; SUBRAMANIAN; ISBELL; THOMAZ, 2011; BRUNSKILL; LI, 2014) onde (BERNSTEIN, 1999) obteve desempenho ruim, produzindo até *negative transfer*; e multiobjetivo (BONINI et al., 2017; ROIJERS et al., 2014; SILVA; COSTA, 2015), onde ainda não foi avaliada. Um estudo inicial gerando um reuso ponderado de políticas parciais em tarefas multiobjetivo apenas com o algoritmo PB foi realizado em (BONINI; SILVA; COSTA, 2017; BONINI et al., 2017), porém essa abordagem só foi verificada sem a extensão do algoritmo com uso de objetos (PPB), ficando dependente do espaço de estados na hora de realizar a transferência de conhecimento entre as tarefas. Por isso, uma possível extensão dessa abordagem para trabalhar com objetos, além do reuso probabilístico de soluções também seria interessante.

Muitos outros métodos de descoberta de políticas parciais também podem ser testados e comparados, e acreditamos que eles se encaixariam muito bem em nossa estrutura, basta que sejam adaptados ou que já sejam específicos para TL. Por fim, como adaptar PRDO para permitir a transferência de políticas parciais aprendidas entre diferentes problemas ainda é uma questão em aberto.

REFERÊNCIAS

- AMAREL, S. On representations of problems of reasoning about actions. *Machine intelligence*, v. 3, n. 3, p. 131–171, 1968.
- ANZAI, Y.; SIMON, H. A. The theory of learning by doing. *Psychological review*, American Psychological Association, v. 86, n. 2, p. 124, 1979.
- ASADA, M.; NODA, S.; TAWARATSUMIDA, S.; HOSODA, K. Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine learning*, Springer, v. 23, n. 2, p. 279–303, 1996.
- ASADI, M.; HUBER, M. Effective control knowledge transfer through learning skill and representation hierarchies. In: *IJCAI*. [S.l.: s.n.], 2007. v. 7, p. 2054–2059.
- BERNSTEIN, D. S. *Reusing old policies to accelerate learning on new MDPs*. [S.l.], 1999.
- BONINI, R. C.; SILVA, F. L. da; COSTA, A. H. R. Learning options in multiobjective reinforcement learning. In: *AAAI-17 Student Paper*. [S.l.: s.n.], 2017. p. (4708–4709).
- BONINI, R. C.; SILVA, F. L. da; GLATT, R.; COSTA, A. H. R. Transferring probabilistic options in reinforcement learning. In: *AAMAS Workshop in Transfer in Reinforcement Learning*. [S.l.: s.n.], 2017.
- BONINI, R. C.; SILVA, F. L. da; GLATT, R.; SPINA, E.; COSTA, A. H. R. A framework to discover and reuse object-oriented options in reinforcement learning. In: *BRACIS*. [S.l.: s.n.], 2018. p. (Accepted Paper).
- BONINI, R. C.; SILVA, F. L. da; SPINA, E.; COSTA, A. H. R. Using options to accelerate learning of new tasks according to human preferences. In: *AAAI-17 Workshop Human-Machine Collaborative Learning*. [S.l.: s.n.], 2017. p. (1–8).
- BOWLING, M.; VELOSO, M. Reusing learned policies between similar problems. In: CITESEER. *AI-98 Workshop on New Trends in Robotics*. [S.l.], 1998.
- BRUNSKILL, E.; LI, L. Pac-inspired option discovery in lifelong reinforcement learning. In: *ICML*. [S.l.: s.n.], 2014. p. 316–324.
- BUTZ, M. V.; SWARUP, S.; GOLDBERG, D. E. Effective online detection of task-independent landmarks. *Urbana*, v. 51, p. 61801, 2004.
- DIETTERICH, T. G. Hierarchical reinforcement learning with the maxq value function decomposition. *J. Artif. Intell. Res.(JAIR)*, v. 13, p. 227–303, 2000.
- DIGNEY, B. L. Learning hierarchical control structures for multiple tasks and changing environments. In: *Proceedings of the fifth international conference on simulation of adaptive behavior on from animals to animats*. [S.l.: s.n.], 1998. v. 5, p. 321–330.

- DIUK, C.; COHEN, A.; LITTMAN, M. L. An Object-oriented Representation for Efficient Reinforcement Learning. In: *ICML*. [S.l.: s.n.], 2008. p. 240–247.
- FERNANDEZ, F.; VELOSO, M. Probabilistic policy reuse in a reinforcement learning agent. In: *AAMAS*. [S.l.: s.n.], 2006. p. 720–727.
- GLATT, R.; SILVA, F. L. da; COSTA, A. H. R. Towards knowledge transfer in deep reinforcement learning. In: *IEEE. BRACIS*. [S.l.], 2016. p. 91–96.
- GLATT, R.; SILVA, F. L. da; COSTA, A. H. R. Case-based policy inference for transfer in reinforcement learning. In: *ECML Workshop on Scaling-Up Reinforcement Learning*. [S.l.: s.n.], 2017. p. 1–8.
- KOGA, M. L.; SILVA, V. F. da; COSTA, A. H. R. Stochastic abstract policies: Generalizing knowledge to improve reinforcement learning. *IEEE Transactions on Cybernetics*, IEEE, p. 77–88, 2015.
- KONIDARIS, G.; BARTO, A. G. Building portable options: Skill transfer in reinforcement learning. In: *IJCAI*. [S.l.: s.n.], 2007. p. 895–900.
- KONIDARIS, G.; BARTO, A. G. Skill discovery in continuous reinforcement learning domains using skill chaining. In: *NIPS*. [S.l.: s.n.], 2009. p. 1015–1023.
- KONIDARIS, G.; SCHEIDWASSER, I.; BARTO, A. G. Transfer in reinforcement learning via shared features. *Journal of Machine Learning Research*, v. 13, n. May, p. 1333–1371, 2012.
- LAZARIC, A. Transfer in reinforcement learning: a framework and a survey. *Reinforcement Learning*, Springer, v. 12, p. 143–173, 2012.
- MACGLASHAN, J. *Multi-source Option-based Policy Transfer*. Tese (Doutorado), Catonsville, MD, USA, 2013.
- MACGLASHAN, J. *Brown-UMBC Reinforcement Learning and Planning (BURLAP)*, <http://burlap.cs.brown.edu/index.html>. 2015.
- MADDEN, M. G.; HOWLEY, T. Transfer of experience between reinforcement learning environments with progressive difficulty. *Artificial Intelligence Review*, Springer, v. 21, n. 3-4, p. 375–398, 2004.
- MANNOR, S.; MENACHE, I.; HOZE, A.; KLEIN, U. Dynamic abstraction in reinforcement learning via clustering. In: *ACM. ICML*. [S.l.], 2004. p. 71.
- MARTÍN, M.; GEFFNER, H. Learning generalized policies from planning examples using concept languages. *Applied Intelligence*, Springer, v. 20, n. 1, p. 9–19, 2004.
- MCGOVERN, A.; BARTO, A. G. Automatic discovery of subgoals in reinforcement learning using diverse density. In: *ICML*. [S.l.]: Morgan Kaufmann, 2001. p. 361–368.
- PAN, S. J.; YANG, Q. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, IEEE, v. 22, n. 10, p. 1345–1359, 2010.

- PENG, B.; MACGLASHAN, J.; LOFTIN, R.; LITTMAN, M. L.; ROBERTS, D. L.; TAYLOR, M. E. A need for speed: Adapting agent action speed to improve task learning from non-expert humans. In: *AAMAS*. [S.l.: s.n.], 2016. p. 957–965.
- PICKETT, M.; BARTO, A. G. Policyblocks: An algorithm for creating useful macro-actions in reinforcement learning. In: *ICML*. [S.l.: s.n.], 2002. p. 506–513.
- PUTERMAN, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. 1st. ed. New York, NY, USA: John Wiley & Sons, Inc., 1994.
- ROIJERS, D. M.; VAMPLEW, P.; WHITESON, S.; DAZELEY, R. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research (JAIR)*, v. 48, p. 67–113, 2014.
- SELFRIDGE, O. G.; SUTTON, R. S.; BARTO, A. G. Training and tracking in robotics. In: *IJCAI*. [S.l.: s.n.], 1985. p. 670–672.
- SILVA, F. L. da; COSTA, A. H. R. Multi-objective reinforcement learning through reward weighting. *IJCAI Workshop on Synergies between Multiagent Systems, Machine Learning and Complex Systems*, v. 1, p. 25–36, 2015.
- SILVA, F. L. da; COSTA, A. H. R. Towards Zero-Shot Autonomous Inter-Task Mapping through Object-Oriented Task Description. In: *AAMAS Workshop in Transfer in Reinforcement Learning*. [S.l.: s.n.], 2017.
- SILVA, F. L. da; COSTA, A. H. R. Object-Oriented Curriculum Generation for Reinforcement Learning. In: *AAMAS*. [S.l.: s.n.], 2018. p. 1026–1034.
- SILVA, F. L. da; TAYLOR, M. E.; COSTA, A. H. R. Autonomously Reusing Knowledge in Multiagent Reinforcement Learning. In: *IJCAI*. [S.l.: s.n.], 2018. p. 5487–5493.
- SILVA, V. F. da; PEREIRA, F. A.; COSTA, A. H. R. Finding memoryless probabilistic relational policies for inter-task reuse. In: SPRINGER. *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*. [S.l.], 2012. p. 107–116.
- ŞİMŞEK, Ö.; WOLFE, A. P.; BARTO, A. G. Identifying useful subgoals in reinforcement learning by local graph partitioning. In: ACM. *ICML*. [S.l.], 2005. p. 816–823.
- SONI, V.; SINGH, S. Using homomorphisms to transfer options across continuous reinforcement learning domains. In: *AAAI*. [S.l.: s.n.], 2006. v. 6, p. 494–499.
- SUBRAMANIAN, K.; ISBELL, C.; THOMAZ, A. Learning options through human interaction. In: CITESEER. *IJCAI*. [S.l.], 2011.
- SUTTON, R. S.; BARTO, A. G. *Reinforcement learning: An introduction*. 1st. ed. Cambridge, MA, USA: MIT Press, 1998.
- SUTTON, R. S.; PRECUP, D.; SINGH, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, Elsevier, p. 181–211, 1999.

- TAYLOR, M. E. et al. Reinforcement learning agents providing advice in complex video games. *Connection Science*, v. 26, n. 1, p. 45–63, 2014.
- TAYLOR, M. E.; STONE, P. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, v. 10, p. 1633–1685, 2009.
- TAYLOR, M. E.; WHITESON, S.; STONE, P. Transfer via inter-task mappings in policy search reinforcement learning. In: ACM. *AAMAS*. [S.l.], 2007.
- TEMBO, T.; TOPIN, N.; BISHOFF, M.; SQUIRE, S.; MACGLASHAN, J.; CARIGNAN, R.; HALTMEYER, N. et al. Discovering subgoals in complex domains. In: *2014 AAAI Fall Symposium Series*. [S.l.: s.n.], 2014.
- THRUN, S.; SCHWARTZ, A. Finding structure in reinforcement learning. In: *NIPS*. [S.l.]: MIT Press, 1995. p. 385–392.
- TOPIN, N.; HALTMEYER, N.; SQUIRE, S.; WINDER, J.; MACGLASHAN, J. et al. Portable option discovery for automated learning transfer in object-oriented markov decision processes. In: AAAI PRESS. *IJCAI*. [S.l.], 2015. p. 3856–3864.
- TORREY, L.; SHAVLIK, J. Transfer learning. *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, v. 1, p. 242, 2009.
- WATKINS, C. J.; DAYAN, P. Q-learning. *Machine learning*, Springer Netherlands, v. 8, n. 3, p. 279–292, 1992.