Differentiable Planning with Indefinite Horizon

Daniel B. Dias[1], Leliane N. de Barros[1], Karina V. Delgado[2], Denis D. Mauá[1]

[1] Instituto de Matemática e Estatística – Universidade de São Paulo (IME-USP)
danielbpdias@gmail.com, leliane@ime.usp.br, ddm@ime.usp.br
[2] Escola de Artes, Ciências e Humanidades – Universidade de São Paulo (EACH-USP)
kvd@usp.br

Abstract. With the recent advances in automated planning based on deep-learning techniques, Deep Reactive Policies (DRPs) have been shown as a powerful framework to solve Markov Decision Processes (MDPs) with a certain degree of complexity, like MDPs with continuous action-state spaces and exogenous events. Some differentiable planning algorithms can learn these policies through policy-gradient techniques considering a finite horizon MDP. However, for certain domains, we do not know the ideal size of the horizon needed to find an optimal solution, even when we have a planning goal description, that can either be a simple reachability goal or a complex goal involving path optimization. This work aims to solve a continuous MDP through differentiable planning, considering the problem horizon as a hyperparameter that can be adjusted for a DRP training process. This preliminary investigation show that it is possible to find better policies by choosing a horizon that encompasses the planning goal.

Categories and Subject Descriptors: Computing methodologies/Artificial intelligence [**Planning and scheduling**]: Planning under uncertainty; Computing methodologies/Machine Learning [**Machine learning approaches**]: Markov decision processes

Keywords: continuous state and action planning, Markov decision processes, machine learning, differentiable planning

1. INTRODUÇÃO

O problema de planejamento probabilístico em espaços contínuos e estocásticos é considerado um grande desafio para pesquisadores da área de planejamento automático da IA. Uma forma de resolver esses problemas é modelá-los como Processos de Decisão Markovianos (do inglês, *Markov Decision Processes* - MDP) [Puterman 2014], com variáveis contínuas e tomada de decisão em tempo discreto, chamado de horizonte de decisões. Soluções conhecidas, em geral, fazem algum tipo de aproximação, por exemplo, os algoritmos de Programação Dinâmica Simbólica [Vianna et al. 2015; Sanner et al. 2011] raciocinam sobre uma abstração do espaço de estados com a limitação de resolverem apenas instâncias pequenas. Outras soluções fazem uma discretização do espaço de estados, como no caso do algoritmo UCT [Kocsis and Szepesvári 2006] que usa métodos de Monte-Carlo.

Uma outra forma de resolver problemas de planejamento em espaços contínuos e complexos é utilizar algoritmos baseados em técnicas modernas de aprendizado de máquina, que buscam explorar a capacidade das redes neurais serem bons aproximadores de funções para encontrar uma política/solução sem a necessidade de fazer alguma aproximação ou discretização a priori. Um exemplo clássico é o uso da técnica chamada de *gradiente de política* [Williams 1992; Sutton et al. 1999].

Um trabalho recente baseado em gradiente de política em conjunto com a modelagem de grafos de computação estocásticos [Schulman 2016] é o trabalho de [Wu et al. 2017], que resolve um problema de planejamento determinístico e contínuo utilizando gradientes de política e a técnica de gradiente descendente estocástico para encontrar valores aproximados de ações para este ambiente. Posteriormente, o trabalho de [Bueno et al. 2019; Bueno 2021] estendeu esta técnica para MDPs com estados e ações contínuas, treinando uma rede neural que representa uma política aproximada, chamada de política reativa profunda (DRP, do inglês *Deep Reactive Policy*). Por resolverem problemas de planejamento utilizando técnicas de diferenciação baseadas em gradiente, esses algoritmos são também chamados de algoritmos de ***planejamento diferenciável***. Uma das limitações do planejamento

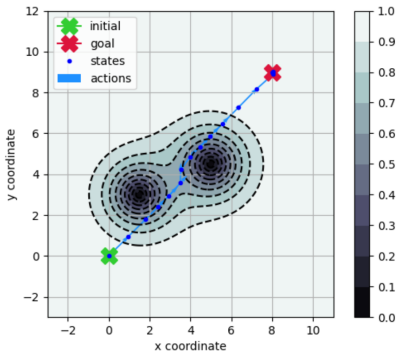

Fig. 1. Instância (Navigation.2) do domínio de planejamento contínuo *Navigation* (Figura extraída de [Bueno et al. 2019]).

diferenciável é que, devido a necessidade de construção do grafo de computação estocástico, é preciso definir a priori um horizonte fixo para o MDP em questão. No entanto, a definição de um horizonte muito pequeno pode fazer com que a meta de alcançabilidade não seja satisfeita; enquanto que com um horizonte maior é possível encontrar trajetórias melhores satisfazendo também a meta de caminho; por outro lado, com um horizonte muito grande, custos desnecessários podem ser adicionados às políticas encontradas.

[Domínio de navegação com meta de caminho] *O domínio Navigation [Faulwasser and Findeisen 2009] é um problema de planejamento de caminhos em 2 dimensões no qual o agente deve alcançar uma determinada região de destino (meta de alcançabilidade) se desviando de zonas de desaceleração ao longo do caminho (meta de caminho), as quais "atrasam" o seu movimento, fazendo com que o agente alcance seu destino em um número maior de passos. Dado o horizonte de tempo que o agente tem para agir, ele tenta traçar uma rota na qual possa ter o menor número de atrasos possível, enquanto tenta alcançar a localização destino. Uma instância desse domínio é chamada de Navigation.i, sendo i o número de zonas de desaceleração.*

Esse trabalho usa técnicas modernas de planejamento diferenciável para resolver problemas de planejamento no espaço contínuo e estocástico com horizonte indeterminado e com metas complexas de planejamento. O objetivo é analisar como diferentes valores de horizonte de um CSA-MDP estão relacionados com a satisfação da meta. Para isso, realizamos experimentos com o algoritmo TF-MDP [Bueno et al. 2019; Bueno 2021] para a instância Navigation.2 ilustrada na Figura 1.

2. FUNDAMENTOS TEÓRICOS

Na área de planejamento automático, um problema muito estudado é o planejamento probabilístico, onde o processo de tomada de decisão sequencial de um agente envolve um raciocínio sob incerteza. Isto é, ao executar uma ação o agente pode resultar em um conjunto de estados possíveis com incerteza sobre o custo (ou recompensa) acumulado ao longo de um dado horizonte. Um arcabouço comum utilizado para modelar esse problema é dado pelos Processos de Decisão Markovianos (MDPs, do inglês *Markov Decision Processes*) [Puterman 2014]. No contexto desse artigo, utilizaremos uma extensão de MDPs clássicos, chamados de CSA-MDPs (do inglês, *Continuous State-Action MDPs*).

Um CSA-MDP com tempo discreto e horizonte finito pode ser definido como em [Bueno et al. 2019; Bueno 2021] como uma tupla $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \Omega, \mathcal{T}, p_\omega, \mathcal{C}, \mathcal{H}, s_0)$, onde $\mathcal{S} \subseteq \mathbb{R}^n$ é o espaço de estados; $\mathcal{A} \subseteq \mathbb{R}^n$ é o espaço de ações; $\Omega \subseteq \mathbb{R}^m$ é o conjunto de eventos exógenos; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \Omega \times \mathcal{S} \to [0, 1]$ é a função de transição dada por uma função de densidade de probabilidade condicional $p(s'|s, a, \omega)$ sob o próximo estado s' dado o estado atual s, a ação a e o evento ω; p_ω é a probabilidade sob o conjunto de eventos exógenos; $\mathcal{C} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_{\geq 0}$ é a função custo que especifica o retorno imediato recebido

pelo estado atual s após aplicar a ação s; $\mathcal{H} = 0, 1, \ldots, H-1$ é o conjunto finito de passos de decisão; e s_0 é o estado inicial.

Os CSA-MDPs podem ser considerados MDPs fatorados [Boutilier et al. 1999], em que o estado s é representado por um vetor n-dimensional e essas dimensões são independentes dado o estado anterior e a ação. Nesta modelagem, a função de transição pode ser decomposta sobre o conjunto de funções de probabilidade das variáveis de estado: $p(s_{t+1}|s_t, a_t, \omega_t) = \prod_{j=1}^{n} p(s_{t+1}^j|s_t, a_t, \omega_t)$.

Uma **política determinística Markoviana** π, localizada no espaço de políticas Π, é definida pela função $\pi : \mathcal{S} \times \mathcal{H} \to \mathcal{A}$ onde uma ação pode ser obtida considerando um estado e um passo de decisão. Neste artigo para efeito de simplificação de notação, denotaremos $\pi(s, t)$ como $\pi(s_t)$. Por fim, a **função valor** $V^\pi(s)$ é considerada como o custo total esperado recebido pela política π do estado s para um horizonte finito H, *i.e.*:

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{H-1} C(s_t, \pi(s_t))\middle| s\right] \tag{1}$$

O objetivo de um agente em um CSP-MDP é encontrar uma **política ótima** π^* no espaço de políticas Π tal que:

$$\pi^* = \arg\min_{\pi \in \Pi} V^\pi(s_0) \tag{2}$$

2.1 Política Reativa Profunda e Grafo de Computação Estocástico

Uma Política Reativa Profunda (DRP, do inglês *Deep Reactive Policy*) [Bueno et al. 2019; Bueno 2021] é um modelo que tenta representar uma função que mapeia estados em ações. Usualmente ela pode ser representada como um modelo paramétrico construído a partir de uma rede neural *feed-forward*, redes que são boas aproximadoras universais de função [Hornik 1991; Leshno et al. 1993]. Os parâmetros desta função (os pesos da rede neural) são chamados de θ e a DRP em si é denotada como $\pi_\theta \approx \pi$.

Uma DRP, assim como uma rede neural, pode ser representada por um **grafo de computação estocástico** [Schulman et al. 2015; Schulman 2016]: um formalismo criado para representar uma estrutura de computação/modelo que mistura cálculos determinísticos com dados de variáveis aleatórias amostradas de distribuições cujos parâmetros dependem de resultados de computações anteriores. Essa representação pode ser utilizada para realizar cálculos eficientes de amostragem dos valores expressos por este grafo e uma estimação eficiente de gradientes dos cálculos expressos por este grafo.

O grafo de computação estocástico pode ser definido como um grafo dirigido acíclico $\mathcal{G} = (V, E)$ em que o conjunto de vértices $V = \Theta \cup D \cup S$ e E é o conjunto de arestas. Θ são os vértices de parâmetros de entrada, que são valores observáveis; D são os vértices determinísticos, que correspondem a funções dos vértices pais; e S são os vértices estocásticos, representando variáveis aleatórias condicionalmente distribuídas de acordo com uma função cujos parâmetros dependem dos vértices pais. Para o conjunto de arestas E, considera-se que uma aresta (u, v) pertence a ele se o vértice v ou sua função de probabilidade dependem do vértice u. Considerando v e w como arestas de grafo de computação estocástico $\mathcal{G}$, pode-se definir três propriedades neste grafo: $v \prec w$ diz que existe um caminho de dependência do vértice v ao w em $\mathcal{G}$; $v \prec^D w$ diz que existe um caminho de v a w que passa somente por vértices determinísticos e $v \prec^P w$ diz que existe um caminho de v a w que passa ao menos em um vértice estocástico. Em sua representação gráfica, vértices de entrada são denotados como elementos sem borda, vértices determinísticos com borda quadrada e vértices estocásticos com borda circular.

Pode se definir uma função objetivo $J : \Theta \to \mathbb{R}$ para estes grafos, dada por $J(\theta) = \mathbb{E}_{w \in S}\left[\sum_{c \in C} c(\theta)\right]$, em que $c \in C \subseteq D$ são vértices folhas conhecidos como vértices custo. É possível diferenciar J em relação à θ com algoritmos de diferenciação automática se não houver a propriedade $\theta \prec^P J(\theta)$.

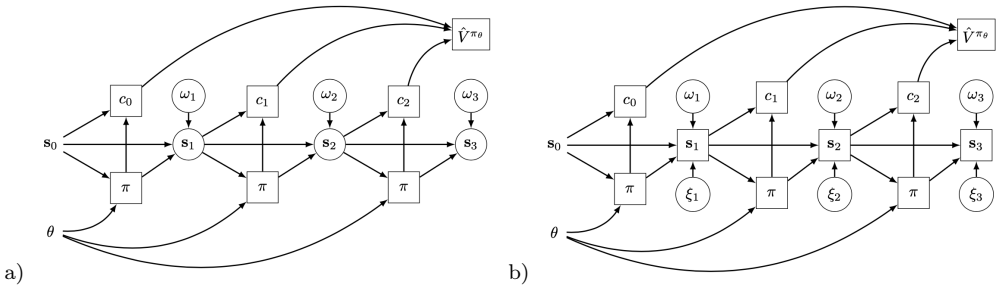

Fig. 2. Representação de um CSA-MDP como um grafo de computação estocástico: a) sem o truque de reparametrização; e b) com o truque de reparametrização (Figura extraída de [Bueno et al. 2019])

Na Figura 2.a pode-se ver a representação de um CSA-MDP como um grafo de computação estocástico (exemplo extraído de [Bueno et al. 2019]), em que: (i) os vértices de entrada são os vértices de estado inicial s_0 e parâmetros de política θ; (ii) os vértices estocásticos são as variáveis de estado $s_{t+1} \sim p(\cdot|s_t, a_t, \omega_t)$ e os eventos exógenos $\omega_t \sim p_\omega(\cdot)$; e (iii) os vértices determinísticos são as variáveis de ação $a_t = \pi_\theta(s_t)$ e os custos $c_t = C(s_t, a_t)$. O vértice de custo é a estimativa da função valor $\hat{V}^{\pi_\theta}$. Nesta figura pode-se notar que este CSA-MDP não pode ser diferenciado, pois existe ao menos um caminho de θ a $\hat{V}^{\pi_\theta}$ que passa por um vértice estocástico (ou seja, existe a propriedade $\theta \prec^P \hat{V}^{\pi_\theta}$). Para contornar esta limitação pode-se utilizar o **truque da reparametrização** [Kingma and Welling 2014], que permite amostrar uma variável aleatória $y \sim p(\cdot|\phi)$, transformando-a em uma função determinística $y = z(\xi; \phi)$ em que é amostrada uma variável auxiliar aleatória $\xi \sim p_\xi(\cdot)$ de uma distribuição p_ξ independente de y e ϕ. No caso dos CSA-MDPs, pode-se reparametrizar $s_{t+1} \sim p(\cdot|s_t, \pi_\theta(s_t))$ em $s_{t+1} = f(s_t, \pi_\theta, \xi_t)$, com $\xi_t \sim p(\cdot)$ amostrado como um ruído independente. Na Figura 2.b pode-se ver um CSA-MDP como um grafo de computação estocástico com o truque de reparametrização aplicado. Com isto, a propriedade $\theta \prec^D \hat{V}^{\pi_\theta}$ existe no grafo e a propriedade $\theta \prec^P \hat{V}^{\pi_\theta}$ não.

2.2 Planejamento Diferenciável

Tendo em vista que podemos tratar a busca de uma política ótima como um processo de otimização (Equação 2), uma maneira eficiente de se encontrar soluções aproximadas para CSA-MDPs é transformar uma instância de um problema em um grafo de computação estocástico, calcular seu gradiente e aplicar um processo de otimização deste grafo com um gradiente descendente estocástico. Nesta abordagem os algoritmos aprendem os parâmetros de entrada do grafo, tal qual um algoritmo de aprendizado de máquina ajusta e aprende os parâmetros do seu modelo. Estas técnicas de fato caracterizam a abordagem de planejamento chamada de planejamento diferenciável.

Em [Wu et al. 2017] foi proposto um algoritmo chamado de TF-PLAN para tomada de decisão sequencial em ambiente determinístico com espaços contínuos, no qual se estrutura um problema de decisão sequencial como um modelo representado por um grafo de computação estocástico em que os vértices de entrada são parâmetros que representam as ações a serem tomadas em cada instante de tempo.

Em [Bueno et al. 2019; Bueno 2021] foi proposta uma extensão do algoritmo TF-PLAN para MDPs, chamado de TF-MDP, em que uma Política Reativa Profunda é treinada em conjunto com um grafo de computação estocástico usando o truque de reparametrização. Além de considerar efeitos estocásticos, o TF-MDP permite que sejam resolvidos problemas para tomada de decisão sequencial em ambientes com estados e ações contínuas. Nesses algoritmos, a avaliação de política é feita executando-se o algoritmo TF-MDP por N vezes e calculando a média do valor esperado do estado inicial, isto é $V^\pi(s_0)$, que chamaremos de $custo_{medio}$ das N políticas encontradas pelo TF-MDP.

3. BUSCA PELO HORIZONTE IDEAL DE UM CSA-MDP COM HORIZONTE INDEFINIDO

Visto que os trabalhos atuais em planejamento diferenciável consideram apenas MDPs com horizonte fixo, este trabalho propõe o seguinte problema: *dado um CSA-MDP e a descrição da meta de planejamento, é possível definir o horizonte ideal para esse problema?*

A primeira modificação que faremos para diferenciar políticas ótimas geradas para diferentes horizontes é identificar aquelas que possuem baixo custo médio mas que não alcançam um estado meta. Para isso definimos um CSA-MDP orientado a meta com penalidade.

3.1 Processos de Decisão Markovianos com estado e ação contínuos orientados a meta com penalidade

Um **CSA-MDP orientado a meta com penalidade** (do inglês, *Goal-oriented CSA-MDP*) é definido pela tupla $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \Omega, \mathcal{T}, p_\omega, \mathcal{C}, \mathcal{H}, \mathcal{G}, s_0, k)$, em que $\mathcal{S}$, $\mathcal{A}$, Ω, $\mathcal{T}$, p_ω, $\mathcal{C}$, $\mathcal{H}$, e s_0 seguem a mesma definição de um CSA-MDP; $\mathcal{G} \subset \mathcal{S}$ é a meta de alcançabilidade dada por um conjunto de estados meta a serem alcançados pelo agente e k é uma penalidade que o agente recebe caso não termine em um estado meta no instante H (isto é, $s_H \notin \mathcal{G}$):

$$V^\pi(s_H) = \begin{cases} \hat{V}^\pi(s_H), & \text{se } s_H \in \mathcal{G} \\ \hat{V}^\pi(s_H) + k, & \text{caso contrário} \end{cases} \tag{3}$$

onde $\hat{V}^\pi(s_H)$ é custo total esperado de CSA-MDPs sem meta (Equação 1), e $k \geq V^{\pi^*}(s)$ para todo estado $s \in \mathcal{S}$ dada uma política ótima π^*.

3.2 Busca de horizonte para CSA-MDPs orientados a meta

Nesse trabalho assumimos que o horizonte ideal de um CSA-MDP $\mathcal{M}$ orientado a meta é desconhecido a priori, isto é, não se sabe a quantidade de passos de decisão necessária para se alcançar a meta. Como dito anteriormente, podemos fixar um horizonte que superestime o horizonte ideal, porém isso adicionaria custos desnecessários à política encontrada pelo TF-MDP. Por outro lado, se subestimamos o horizonte ideal, a política encontrada teria um custo maior devido a penalidade por não alcançarem um estado meta. Além disso, é importante notar que mesmo as políticas que satisfaçam a meta de alcançabilidade também podem ser melhoradas com um horizonte maior para satisfazerem a meta de caminho.

Chamaremos de H_g o horizonte de um CSA-MDP orientado a meta que quando resolvido pelo TF-MDP é capaz de devolver a política ótima de menor custo médio. Assim, esse trabalho investiga formas de encontrar o horizonte ideal H_g bem como sua relação com horizontes de políticas que satisfaçam a meta (de alcançabilidade e de caminho).

Para fazermos a busca por H_g utilizamos o TF-MDP com diferentes tamanhos de horizonte (inteiros entre H_{min} e H_{max}) e devolvemos a política de menor custo médio. Nesse processo consideramos o horizonte como um hiperparâmetro do algoritmo e como métrica de avaliação usamos a média do custo total esperado devolvido em N rodadas do algoritmo TF-MDP. Além disso, verificamos qual é o menor horizonte, chamado de H_{sat}, cuja política devolvida pelo TF-MDP possui uma taxa de satisfação de meta maior que 95% em Z simulações da política devolvida pelo TF-MDP.

O Algoritmo 1 realiza a busca descrita acima e devolve o valor do horizonte H_g, a política correspondente π_g bem como o valor de H_{sat}. O algoritmo recebe como entrada: (i) um CSA-MDP orientado a meta $\mathcal{M}$; (ii) um valor de horizonte mínimo H_{min}; (iii) um valor de horizonte máximo H_{max}; (iv) um inteiro N indicando o número de vezes que o algoritmo TF-MDP será executado para cada horizonte considerado; (v) um inteiro Z indicando o número de vezes que a política ótima para um dado horizonte é simulada para a realização do teste de meta e (vi) $\rho = (\alpha, epochs, batch)$, uma

tupla contendo a taxa de aprendizado α, o número de épocas de treinamento da DRP (*epochs*) e o número de amostras de trajetórias simuladas utilizadas para treinar a DRP (*batch*).

Algoritmo 1: TF-MDP com HORIZONTE INDEFINIDO

Entrada: $H_{min}, H_{max}, \mathcal{M}, N, Z, \rho$
início
  $custo_g := \infty$
  $H_g, H_{sat} := -\infty, -\infty$
  $\pi_g := \varnothing$
  $h := H_{min}$
  enquanto $h \leq H_{max}$ **faça**
    $custo_{medio}, \pi :=$ EXECUTATF-MDP-N-VEZES$(\mathcal{M}_h, N, \rho)$
    `// Caso as políticas encontradas não satisfaçam a meta, o` $custo_{medio}$ `é acrescido de uma penalidade.`
    se $custo_{medio} < custo_g$ **então**
      $custo_g := custo_{medio}$
      $H_g, \pi_g := h, \pi$
    fim
    se $H_{sat} \neq -\infty$ **então**
      $taxa_{meta} :=$ SIMULA-POLITICA$(\mathcal{M}_h, \pi, Z)$
      se $taxa_{meta} > 0.95$ **então**
        $H_{sat} := h$
      fim
    fim
    $h := h + 1$
  fim
  devolve H_g, H_{sat}, π_g
fim

4. EXPERIMENTOS

O Algoritmo 1 foi executado para N=10 na instância Navigation.2 (contendo 2 zonas de desaceleração) com os seguintes parâmetros: $\alpha = 0,001, epochs = 200, batch = 128, H_{min} = 5, H_{max} = 25, Z = 10$. Como arquitetura da DRP foi usada uma rede neural *feed-forward* com quatro camadas ocultas, de tamanhos 256, 128, 64, 32 e função de ativação ELU. Todos os experimentos foram executados em uma máquina com 16 GBs de memória e 6 processadores de 2,6 GHz em um ambiente Unix.

4.1 Domínio **Navigation**

Considere o domínio **Navigation** descrito na Seção 1. A posição do agente a cada instante t é dada pela variável de estado $p_t \in \mathbb{R}^2$ e o movimento do agente é dado pela variável de ação $a_t \in [-1, 1]^2$. A dinâmica de movimento do agente é dada pela função:

$$\lambda = \prod_j \frac{2}{1 + exp(-\alpha_j ||p_t - c_j)||_2)} - 1,$$

$$p_{t+1} \sim Normal(\mu = p_t + \lambda a_t, \sigma = \frac{\sigma_{max}}{\sqrt{2}} ||a_t||)$$

em que cada zona de desaceleração j é caracterizada por uma posição central c_j e uma constante de decaimento α_j, com seu efeito dependendo da distância euclidiana entre esta posição central e

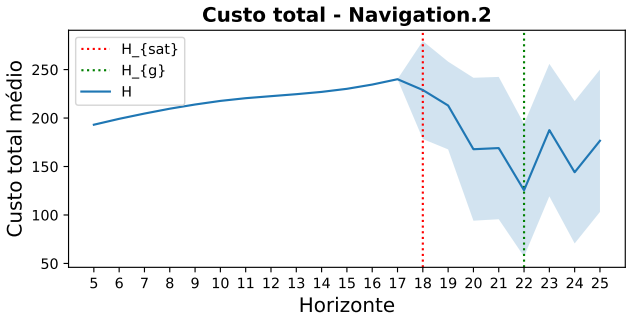

Fig. 3. Média e desvio padrão do custo de trajetória em cada horizonte para o domínio Navigation.

a posição do agente. O fator de desaceleração conjunto λ é dado pela multiplicação de cada zona de desaceleração independente. A função custo é a distância euclidiana da posição do agente até a posição da meta em cada passo de decisão do problema, ou seja: $C_t = ||p_t - g||_2$.

No último passo do problema, caso o agente não tenha alcançado a meta, ele receberá uma penalidade k. A penalidade k foi considerada como $k = 150$ para o *Navigation.2*. Os estados meta $s_t \in \mathcal{G}$ neste problema são aqueles nos quais a distância euclidiana da posição do agente p_t até a meta é menor ou igual a 1, ou seja, $||p_t - g||_2 \leq 1$.

Nos experimentos deste artigo o mapa de navegação é uma região contínua, limitada por um quadrado, com o agente iniciando na ponta inferior esquerda e a meta na ponta superior direita na instância *Navigation.2* que possui com duas zonas de desaceleração localizadas em pontos diferentes.

4.2 Resultados

A Figura 3 mostra a média e o desvio padrão do custo das políticas encontradas pelo TF-MDP para cada horizonte considerado pelo Algoritmo 1. Podemos observar que a política encontrada com horizonte 18 satisfaz o teste de meta (isto é, alcança a taxa mínima de sucesso definida pelo Algoritmo 1), mostrando também uma queda do custo total médio o que se justifica pelo fato dessas políticas não receberem penalidades. Note ainda que com valores maiores de horizonte, $H = 19, 20$ e 21, o agente tem mais "tempo" para se desviar das zonas de desaceleração e assim evitar que o seu movimento atrase, o que respectivamente causa uma diminuição ainda maior do custo total das políticas ótimas devolvidas pelo TF-MDP. No entanto, para horizontes maiores que $H = 22$, o custo total médio cresce, indicando que o aumento de passos de tempo adiciona custos desnecessários às políticas encontradas. Assim, os resultados mostram que $H_g = 22$ é o horizonte ideal para a instância Navigation2.

5. CONCLUSÃO

Dado um MDP com estados e ações continuas, uma limitação da solução baseada em planejamento diferenciável é a necessidade de construção do grafo de computação estocástico considerando um horizonte fixo. No entanto, sabemos que um horizonte muito pequeno pode fazer com que a meta de alcançabilidade de um MDP não seja satisfeita; enquanto que com um horizonte maior é possível encontrar trajetórias melhores satisfazendo também a meta de caminho do MDP. Por outro lado, um horizonte muito grande resulta em custos desnecessários.

Este trabalho propõe um estudo preliminar sobre como encontrar o horizonte ideal para CSA-MDPs orientados a meta. O algoritmo proposto se inspira no algoritmo de planejamento clássico GRAPHPLAN [Blum and Furst 1997], que cria um grafo de planejamento e incrementa o horizonte do problema até encontrar uma política que alcance a meta.

A ideia de encontrar o horizonte ideal para CSA-MDPs orientados a meta pode ser utilizada para expandir o uso de DRPs com planejamento online [Bueno 2021] ou mesmo buscar meios de adaptar esta modelagem de CSA-MDPs orientados a meta descritos em [Trevizan and Veloso 2012] para problemas de caminho estocásticos curtos (SSPs, do inglês *Shortest Stochastic Paths*) com domínios contínuos.

Em trabalhos futuros, pretendemos analisar diferentes domínios de planejamento bem como explorar outros tipos de metas complexas para investigar como os diferentes horizontes afetam o treinamento de DRPs e a política ótima encontrada. Outra proposta de trabalho é a construção de uma heurística para estimar o valor de horizonte H_{min} e assim encontrar o valor de H_g de maneira mais eficiente.

ACKNOWLEDGMENT

O presente trabalho foi realizado no Centro C4AI-USP com apoio da FAPESP (Processo 2019/07665-4) e da IBM.

REFERENCES

Blum, A. L. and Furst, M. L. Fast planning through planning graph analysis. *Artificial Intelligence* 90 (1): 281–300, Feb., 1997.

Boutilier, C., Dean, T., and Hanks, S. Decision-Theoretic Planning: Structural Assumptions and Computational Leverage. *Journal of Artificial Intelligence Research* vol. 11, pp. 1–94, July, 1999.

Bueno, T. P. *Planning in stochastic computation graphs: solving stochastic nonlinear problems with backpropagation.* Ph.D. thesis, Universidade de São Paulo, 2021.

Bueno, T. P., De Barros, L. N., Mauá, D. D., and Sanner, S. Deep Reactive Policies for Planning in Stochastic Nonlinear Domains. *Proceedings of the AAAI Conference on Artificial Intelligence* vol. 33, pp. 7530–7537, July, 2019.

Faulwasser, T. and Findeisen, R. Nonlinear Model Predictive Path-Following Control. In *Nonlinear Model Predictive Control: Towards New Challenging Applications*, L. Magni, D. M. Raimondo, and F. Allgöwer (Eds.). Lecture Notes in Control and Information Sciences. Springer, Berlin, Heidelberg, pp. 335–343, 2009.

Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Networks* 4 (2): 251–257, Jan., 1991.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes, 2014. arXiv:1312.6114 [cs, stat].

Kocsis, L. and Szepesvári, C. Bandit Based Monte-Carlo Planning. In *Machine Learning: ECML 2006*, J. Fürnkranz, T. Scheffer, and M. Spiliopoulou (Eds.). Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, pp. 282–293, 2006.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks* 6 (6): 861–867, Jan., 1993.

Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, 2014.

Sanner, S., Delgado, K. V., and de Barros, L. N. Symbolic dynamic programming for discrete and continuous state MDPs. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence.* UAI'11. AUAI Press, Arlington, Virginia, USA, pp. 643–652, 2011.

Schulman, J. *Optimizing Expectations: From Deep Reinforcement Learning to Stochastic Computation Graphs.* Ph.D. thesis, UC Berkeley, 2016.

Schulman, J., Heess, N., Weber, T., and Abbeel, P. Gradient Estimation Using Stochastic Computation Graphs. In *Advances in Neural Information Processing Systems.* Vol. 28. Curran Associates, Inc., 2015.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems.* Vol. 12. MIT Press, 1999.

Trevizan, F. and Veloso, M. Short-Sighted Stochastic Shortest Path Problems. *Proceedings of the International Conference on Automated Planning and Scheduling* vol. 22, pp. 288–296, May, 2012.

Vianna, L. G. R., de Barros, L. N., and Sanner, S. Real-time symbolic dynamic programming for hybrid MDPs. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence.* AAAI'15. AAAI Press, Austin, Texas, pp. 3402–3408, 2015.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8 (3): 229–256, May, 1992.

Wu, G., Say, B., and Sanner, S. Scalable Planning with Tensorflow for Hybrid Nonlinear Domains. In *Advances in Neural Information Processing Systems.* Vol. 30. Curran Associates, Inc., 2017.