Compromissos arbitrários entre custo e probabilidade à meta e algoritmo de busca heurística em planejamento probabilístico sob o critério GUBS

Gabriel Nunes Crispino

Dissertação apresentada ao
Instituto de Matemática e Estatística
da Universidade de São Paulo
para obtenção do título de
Mestre em Ciências

Programa: Ciência da Computação
Orientadora: Profª. Drª. Karina Valdivia Delgado

Durante o desenvolvimento deste trabalho o autor recebeu auxílio financeiro da CAPES

São Paulo
Fevereiro de 2023

Compromissos arbitrários entre custo e probabilidade à meta e algoritmo de busca heurística em planejamento probabilístico sob o critério GUBS

Gabriel Nunes Crispino

Esta versão da dissertação contém
as correções e alterações sugeridas
pela Comissão Julgadora durante a
defesa da versão original do trabalho,
realizada em 13 de Fevereiro de 2023.

Uma cópia da versão original está
disponível no Instituto de Matemática e
Estatística da Universidade de São Paulo.

Comissão julgadora:

Profª. Drª. Karina Valdivia Delgado (orientadora) – IME-USP
Prof. Dr. Felipe Werndl Trevizan – ANU
Prof. Dr. André Grahl Pereira – UFRGS

Agradecimentos

À professora Karina por toda a atenção, confiança e ensinamentos concedidos durante o trabalho desenvolvido por esses anos. Sua orientação foi essencial para o desenvolvimento e contínua melhora do trabalho como um todo.

Ao professor Valdinei pelo apoio desde o início do trabalho, ajudando como se eu fosse seu orientando. Sua disposição e paciência me ajudaram muito a entender melhor o critério GUBS e assuntos relacionados.

Aos meus pais, Nicolau e Gláucia, e à minha irmã Rafaela, por todo o amor e carinho que me completam como pessoa. Sou muito grato pela companhia, apoio e por tudo mais que aprendo com vocês.

Ao restante da minha família, também, por sempre serem presentes como pessoas importantes na minha vida.

À minha companheira Lilya pelo amor e apoio durante todos esses anos. Teu carinho, compreensão e parceria me ajudaram a passar por todos os obstáculos que encontrei no caminho.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), pelo apoio financeiro no presente trabalho (Código de Financiamento 001).

Resumo

Gabriel Nunes Crispino. **Compromissos arbitrários entre custo e probabilidade à meta e algoritmo de busca heurística em planejamento probabilístico sob o critério GUBS**. Dissertação (Mestrado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2023.

Processos de Decisão Markovianos de Caminho Estocástico mais Curto (SSP-MDPs) são utilizados para modelar problemas de decisões sequenciais probabilísticas em que o objetivo é minimizar o custo acumulado esperado para a meta. No entanto, na presença de estados a partir dos quais não se pode alcançar a meta (*dead ends*), o critério convencional de SSP-MDPs, que minimiza o custo acumulado esperado, pode deixar de ser bem-definido. Critérios lexicográficos podem resolver isso ao definir uma preferência sobre políticas que alcançam a meta com a máxima probabilidade possível. Outros critérios podem ao invés disso realizar um compromisso entre alguma medida de custo e probabilidade à meta. No entanto, ambas abordagens podem escolher políticas que podem não representar a escolha de um tomador de decisão realístico. O critério GUBS, que combina priorização de metas sobre históricos com Teoria da Utilidade Esperada para realizar compromissos entre custos e probabilidade à meta, foi proposto para resolver esses problemas. O critério eGUBS, que é um caso particular do GUBS quando uma função de utilidade exponencial é utilizada, foi depois também proposto, juntamente com o eGUBS-VI, um algoritmo baseado em iteração de valor que constitui o primeiro método para resolver SSP-MDPs sob o critério GUBS de maneira ótima. Esta dissertação contém uma comparação entre o critério GUBS e outros critérios para resolver SSP-MDPs na presença de *dead ends*, e introduz definições e resultados teóricos que mostram que o GUBS é o único desses critérios que não apenas possibilita a realização de compromissos entre grandes custos acumulados sobre pequenas perdas em probabilidade à meta, mas que também garante compromissos arbitrários que podem ser configurados a partir dos seus parâmetros sem conhecimento prévio do problema a ser resolvido. Além disso, esta dissertação introduz o eGUBS-AO*, um algoritmo ótimo de busca heurística para resolver o critério eGUBS. Resultados indicam que, quando uma boa função heurística é disponível ou quando o espaço de estados do problema é muito grande, o eGUBS-AO* pode ter um desempenho melhor que o eGUBS-VI ao realizar uma busca eficiente. Em outros casos, a abordagem mais simples do eGUBS-VI pode trazer melhores resultados.

Palavras-chave: Processos de Decisão Markovianos. Caminho Estocástico mais Curto. Tomada de decisão sequencial. Planejamento probabilístico.

Abstract

Gabriel Nunes Crispino. **Arbitrary trade-offs between cost and probability-to-goal and heuristic search algorithm in probabilistic planning under the GUBS criterion**. Thesis (Master's). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2023.

Stochastic Shortest Path Markov Decision Processes (SSP-MDPs) are used to model probabilistic sequential decision problems where the objective is to minimize the expected accumulated cost to goal. However, in the presence of dead ends, the conventional criterion for SSP-MDPs, which minimizes the expected accumulated cost, can become ill-defined. Lexicographic criteria can solve this by preferring policies that reach the goal with the highest possible probability. Other criteria can instead make a trade-off between some cost measure and probability-to-goal. However, both of these approaches can lead to policies that might not represent the choice of a real decision-maker. The GUBS criterion, which combines goal prioritization over histories with Expected Utility Theory to make trade-offs between costs and probability-to-goal, was proposed to address these problems. The eGUBS criterion, which is a particular case of GUBS when an exponential utility function is used, was later proposed, along with eGUBS-VI, a VI-based algorithm that is the first method to optimally solve SSP-MDPs under the GUBS criterion. This dissertation contains a comparison between GUBS and other related criteria for solving SSP-MDPs in the presence of dead-end states, and introduces theoretical definitions and results that show that it is the only criterion between all related criteria that not only allows for a trade-off between a large accumulated cost and a small loss in probability-to-goal, but also guarantees arbitrary trade-offs that can be tuned from its parameters without previous knowledge of the problem being solved. Also, this dissertation introduces eGUBS-AO*, an optimal heuristic search algorithm for solving the eGUBS criterion. Results indicate that, when there is a good heuristic function available or when the state space is too large, eGUBS-AO* can perform better than eGUBS-VI by doing an efficient search. In other cases, eGUBS-VI's simpler approach might have better results.

Keywords: Markov Decision Processes. Stochastic Shortest Path. Sequential decision making. Probabilistic planning.

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Sumário

Capítulo 1

Introdução

Processos de Decisão Markovianos (*Markov Decision Processes* – MDPs) (Puterman, 1994) são utilizados para modelar problemas em que um agente interage com um ambiente por meio de ações com resultados estocásticos. No caso de problemas de MDPs de Caminho Estocástico mais Curto (*Stochastic Shortest Path* MDPs – SSP-MDPs) (Bertsekas, 1995; Mausam e Kolobov, 2012), o objetivo é minimizar o custo esperado para a meta. No entanto, em problemas em que não existem caminhos que alcancem uma meta com probabilidade 1, esse critério se torna mal-definido. Isso acontece quando existem determinados estados tal que não seja possível alcançar uma meta a partir deles. Esses estados são chamados de *dead ends*.

Por essa razão, novos modelos ou adaptações do critério padrão para resolver SSP-MDPs precisam ser definidos para esse caso particular. É possível, por exemplo, utilizar um fator de desconto $\gamma \in (0, 1)$ (Teichteil-Königsbuch *et al.*, 2011) ou uma penalidade finita D para desistir do processo (Mausam e Kolobov, 2012) para que o critério de custo acumulado esperado seja bem definido nesses problemas; o critério MAXPROB (Kolobov, D. S. Weld *et al.*, 2011), para maximizar a probabilidade de alcançar a meta (probabilidade à meta); ou outros critérios lexicográficos que minimizam o custo acumulado esperado à meta considerando apenas caminhos que maximizam a probabilidade à meta, como iSSPUDEs (Kolobov, D. Weld *et al.*, 2012), S^3Ps (Teichteil-Königsbuch, 2012) e o critério MCMP (F. W. Trevizan *et al.*, 2017). Apesar desses modelos serem bem definidos e resolverem SSP-MDPs na presença de *dead ends*, uma pergunta que pode ser feita é se algumas decisões que podem ser realizadas por meio deles seriam aceitáveis por um tomador de decisão realístico.

Para ilustrar esses problemas, considere o exemplo de uma pessoa que precisa pegar um voo em um determinado horário. Isso pode ser considerado um problema de raciocínio sob incerteza, já que certas ações que o agente (a pessoa em questão) pode tomar com esse objetivo são estocásticas. Por exemplo, ao esperar em um ponto de ônibus para pegar um ônibus para o aeroporto, não se pode definir deterministicamente o tempo que o próximo ônibus demorará para chegar, ou até mesmo o tempo que levará para a pessoa chegar no aeroporto considerando qualquer que seja o meio de transporte que será utilizado para esse fim. Critérios que consideram políticas que maximizam a probabilidade à meta (como MAXPROB (Kolobov, D. S. Weld *et al.*, 2011) e critérios lexicográficos como

iSSPUDE (Kolobov, D. Weld *et al.*, 2012), S^3P (Teichteil-Königsbuch, 2012) e MCMP (F. W. Trevizan *et al.*, 2017)) não resultariam em decisões realísticas para esse problema, já que haveria uma grande chance de tais decisões fazerem com que o agente saia para o aeroporto muito cedo (ou até mesmo imediatamente no momento do planejamento) ao tentar maximizar a probabilidade de chegar lá, ao invés de sair com um intervalo de tempo mais realístico. Outro problema dessa abordagem lexicográfica é que ela não permite que compromissos sejam feitos entre políticas que levam a custos grandes e outras que não maximizam a probabilidade à meta, mas que levam a custos consideravelmente menores. Ao longo desse trabalho esse tipo de compromisso será referido como *compromisso infinito-infinitesimal*. No caso do exemplo dado, um possível caso em que esse tipo de compromisso não seria realizado seria se o agente preferisse uma política que tem como ação pegar um táxi para o aeroporto pagando um valor arbitrariamente grande por isso, comparada com outra que leva o agente a pegar um ônibus com um custo arbitrariamente menor, desde que a probabilidade à meta da primeira opção seja maior que a segunda. A primeira opção seria a política ótima mesmo que essa diferença entre as probabilidades seja infinitesimal, e o tempo de chegada de ambas seja similar.

Os outros critérios mencionados (uso de um fator de desconto ou de uma penalidade finita) podem realizar compromissos ao ter seus parâmetros ajustados para refletir em decisões mais realistas se comparadas apenas à escolha de políticas que maximizam a probabilidade à meta. No entanto, para alguns problemas esses modelos podem dar preferência igual ou superior a políticas que não alcançam metas se comparadas a políticas que alcançam, o que é uma propriedade indesejada. Uma possível solução dependente do SSP-MDP é formular um problema de decisão multiobjetivo que leva em conta custo e probabilidade à meta e considerar que o tomador de decisão ou participa em um processo de elicitação de preferências ou escolhe de uma fronteira Pareto-ótima (Kuo e Freire, 2021).

Baseado nos problemas desses critérios presentes na literatura, o critério GUBS (*Goals with Utility-Based Semantics*) foi proposto (Freire e Delgado, 2017). GUBS combina priorização de metas sobre históricos com Teoria da Utilidade Esperada. O critério eGUBS, um caso particular do GUBS em que uma função de utilidade exponencial é utilizada juntamente com um fator de risco negativo, foi proposto em Freire, Delgado e Reis, 2019. No mesmo trabalho, o algoritmo eGUBS-VI, um algoritmo baseado em iteração de valor, foi introduzido para resolver SSP-MDPs de maneira ótima sob o critério eGUBS.

Na seguinte seção, serão apresentados os objetivos do presente trabalho.

1.1 Objetivos

O presente trabalho tem como principais objetivos realizar uma revisão com adições de propriedades teóricas no critério GUBS e na sua relação com outros critérios para resolver SSP-MDPs com *dead ends*; e propor o eGUBS-AO*, um algoritmo de busca heurística desenvolvido a partir de uma adaptação não trivial do algoritmo AO* (Martelli e Montanari, 1973; Martelli e Montanari, 1978; Nilsson, 1982), que resolve SSP-MDPs sob o critério eGUBS.

Sobre as propriedades teóricas tratadas no presente trabalho, nele é contida uma

comparação teórica entre o critério GUBS e outros critérios relacionados baseada nas preferências deles entre pares de políticas. Para isso, são introduzidos dois conceitos principais: primeiro, uma definição formal do que decisões em SSP-MDPs que permitem compromissos infinito-infinitesimais são; e segundo, uma propriedade da preferência entre pares de políticas em um critério. Essa propriedade, nomeada de propriedade de priorização de probabilidade à meta α-forte, pode ser garantida por um critério se a razão entre valores de probabilidade à meta de todos pares de políticas estão limitados por um α, para todo SSP-MDPs, em que $0 \leq \alpha \leq 1$. Entre outros resultados, o presente trabalho demonstra que GUBS é o único critério entre todos os analisados que não apenas permite compromissos infinito-infinitesimais, como também garante a propriedade de priorização de probabilidade à meta α-forte para um α arbitrário, tal que $0 \leq \alpha \leq 1$.

Como objetivos secundários, são definidos:

- Avaliar o desempenho do eGUBS-AO* comparado com o eGUBS-VI em diferentes domínios de SSP-MDPs com *dead ends*;
- Desenvolver e executar experimentos para comparar na prática o critério GUBS com outros critérios presentes na literatura com o mesmo propósito;
- Estender o arcabouço teórico referente às propriedades dos critérios GUBS e eGUBS presente na literatura com a definição da propriedade de priorização de probabilidade à meta α-forte, tal como a introdução de resultados teóricos advindos dessa definição.

1.2 Organização do Trabalho

No Capítulo 2 são apresentados os conceitos que constituem a base teórica de MDPs, SSP-MDPs, e algoritmos para solucioná-los. No Capítulo 3 é realizada uma revisão e discussão de critérios presentes na literatura para resolver SSP-MDPs com *dead ends*, e no Capítulo 4 são apresentados conceitos para analisar esses critérios com relação à priorização de metas que cada um deles oferece. Nos capítulos 5 e 6, são apresentados os critérios GUBS e eGUBS em detalhes, respectivamente. O Capítulo 7 contém a definição dos algoritmos que resolvem o critério eGUBS de maneira exata: eGUBS-VI (já existente) e eGUBS-AO* (proposto no presente trabalho). O Capítulo 8 apresenta os experimentos realizados neste trabalho e, por fim, no Capítulo 9 estão presentes as suas considerações finais.

Capítulo 2

Processos de Decisão Markovianos

Na área de planejamento probabilístico, o formalismo mais utilizado para modelar a interação de um agente com o ambiente é o de Processos de Decisão Markovianos (*Markov Decision Processes* - MDPs). A partir desse modelo, é definido o processo de tomada de decisão sequencial de um agente, de maneira que em determinado passo ele está em um estado, e pode tomar uma ação dentro de um conjunto de ações. O estado do agente no próximo passo é definido por uma função de transição, dado o estado atual e a ação escolhida no passo atual. A seguinte seção contém a definição formal de MDPs.

2.1 Definição Formal

Definição 1 (Processos de Decisão Markovianos). *Um MDP pode ser definido formalmente como uma tupla $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, c \rangle$ (Puterman, 1994), em que:*

- *$\mathcal{S}$ é o conjunto de estados em que o agente pode estar em cada passo de tempo;*
- *$\mathcal{A}$ é o conjunto de ações que podem ser executadas em cada estado;*
- *$P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ é a função de transição. $P(s, a, s') = \Pr(s_{t+1} = s' | s_t = s, a_t = a)$ indica a probabilidade do agente estar em um estado s' no próximo passo, dado que a ação a é executada no estado s no passo atual;*
- *$c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ é a função de custo, que define o custo associado ao executar a ação a estando no estado s.*

Dessa maneira, a interação do agente com o ambiente, considerando cada passo de tempo $t \in \{0, 1, ..., T\}$, pode ser resumida em um histórico $h = \{\langle s_1, a_1, c_1 \rangle, \langle s_2, a_2, c_2 \rangle, ..., \langle s_{T-1}, a_{T-1}, c_{T-1} \rangle, s_T\}$, no qual $s_t \in \mathcal{S}$ é o estado em que o agente se encontra no passo t, $a_t \in \mathcal{A}$ é a ação escolhida em s_t, e $c_t = c(s_t, a_t)$. A partir disso, o agente vai para o próximo estado s_{t+1} com probabilidade $P(s_t, a_t, s_{t+1})$. Define-se $\mathcal{H} = (\mathcal{S} \times \mathcal{A} \times \mathbb{R}_{>0})^* \times \mathcal{S}$ como o conjunto de todos os históricos.

Em geral, o **horizonte** de um MDP pode ser classificado de três maneiras diferentes:

é considerado **finito** o horizonte de um processo que tem um número fixo de passos até que seja finalizado. Nesse caso, T é um número finito. Outra possibilidade é que esse processo aconteça infinitamente, caso em que $T \to \infty$, no qual o MDP é classificado como de horizonte **infinito**. O terceiro caso é o de horizonte **indeterminado**. Esse caso acontece quando se sabe que o processo finalizará assim que o agente atingir um estado meta, mas não é definido em exatamente quantos passos isso se realizará.

A solução de um MDP é uma **política**. Uma política π pode assumir diferentes descrições como as seguintes:

- Estacionária ($\pi : \mathcal{S} \to \mathcal{A}$): um mapeamento de estados s_t em ações $a_t = \pi(s_t)$;
- Não markoviana ($\pi : \mathcal{H} \to \mathcal{A}$): um mapeamento de históricos H_t no passo t em ações $a_t = \pi(H_t)$; e
- Política aumentada de custo acumulado: ($\pi : \mathcal{S} \times \mathbb{R} \to \mathcal{A}$): uma política não markoviana que mapeia todos estados s_t e custos acumulados C_t no passo t em ações $a_t = \pi(s_t, C_t)$.

Classifica-se como **política ótima** uma política que otimiza alguma função objetivo. A solução ótima de um MDP é uma política ótima.

Na área de planejamento probabilístico, um caso comum é o de se encontrar políticas ótimas em processos de horizonte indeterminado sob o problema de **Caminho Estocástico mais Curto** (*Stochastic Shortest Path* – SSP). Nesse caso, assume-se que existe um conjunto de estados absorvedores metas (ou terminais). A seção seguinte contém uma discussão dessa classe de problemas em mais detalhes.

2.2 Caminho Estocástico mais Curto

Formalmente, um MDP de Caminho Estocástico mais Curto (SSP-MDP) é definido como o seguinte:

Definição 2 (MDP de Caminho Estocástico mais Curto). *Um SSP-MDP (Bertsekas, 1995) é uma tupla $\mathcal{M} = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle$, em que:*

- *$\mathcal{S}$, $\mathcal{A}$ e P são definidos como na Definição 1;*
- *$s_0 \in \mathcal{S}$ é o estado inicial;*
- *$c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_{>0}$ é a função de custo, que define um custo $c(s, a)$ ao tomar uma ação a em um estado s;*
- *$\mathcal{G} \subset \mathcal{S}$ é o conjunto de estados meta absorvedores, ou seja, estados $g \in \mathcal{G}$ tal que $c(g, a) = 0$ e $P(g, a, g) = 1, \forall a \in \mathcal{A}$. Além disso, $c(s, a) > 0, \forall s \in \mathcal{S} \setminus \mathcal{G}$.*

Um SSP-MDP modela um processo discreto em que a cada passo de tempo $t \in \{0, 1, ...\}$ um agente observa um estado s_t, toma uma ação a_t, paga um custo $c_t > 0$ e transita para um estado s_{t+1} até que um estado meta em $\mathcal{G}$ seja atingido, momento no qual o processo acaba. A cada tempo t, um histórico aleatório h_t é definido como é feito na Seção 2.1, logo após a Definição 1. A solução de um SSP-MDP é uma política estacionária π.

O objetivo de algoritmos para SSP-MDPs é encontrar uma política ótima sob algum critério especificado. Esses critérios dependem da existência de políticas próprias. Uma política π é própria se o processo alcança um estado meta e finaliza com probabilidade 1 quando ela é seguida. Formalmente, isso pode ser descrito pela seguinte equação:

$$\lim_{t\to\infty} \Pr(s_t \in \mathcal{G} \mid \pi, s_0) = 1.$$

Uma política própria pode não existir se *dead ends* existirem. Formalmente, um estado s_{DE} é um *dead end* se, para toda política $\pi \in \Pi$, em que Π é o conjunto de todas políticas de interesse (estacionárias, não markovianas ou políticas aumentadas de custo acumulado), a seguinte equação é verdadeira:

$$\lim_{t\to\infty} \Pr(s_t \in \mathcal{G} \mid \pi, s_0 = s_{DE}) = 0.$$

Pode-se definir três classes diferentes de SSP-MDPs: SSP-MDP convencionais (sem *dead ends*), SSP-MDPs com *dead ends* evitáveis, e SSP-MDPs com *dead ends* inevitáveis (Kolobov, D. Weld *et al.*, 2012). Em SSP-MDPs convencionais, *dead ends* não existem, o que significa que um estado meta pode ser alcançado com probabilidade 1 a partir de qualquer estado $s \in \mathcal{S}$. Em SSP-MDPs com *dead ends* evitáveis, apesar desses estados existirem no problema, também existe uma política própria enraizada no estado inicial s_0. Em SSP-MDPs com *dead ends* inevitáveis, não existem políticas próprias partindo de s_0. Isso significa que existem *dead ends* nesse caso particular, e que por sua vez esses *dead ends* são inevitáveis.

Essas três classes de SSP-MDPs são relacionadas por uma hierarquia. De maneira similar, qualquer critério ou algoritmo para SSP-MDPs com *dead ends* inevitáveis pode ser utilizado para resolver SSP-MDPs com *dead ends* evitáveis. Por fim, qualquer critério ou algoritmo para SSP-MDPs com *dead ends* evitáveis pode ser utilizado para resolver SSP-MDPs convencionais (Kolobov, D. Weld *et al.*, 2012).

Tradicionalmente, SSP-MDPs convencionais e SSP-MDPs com *dead ends* evitáveis consideram o mesmo critério de otimalidade: o custo acumulado esperado. Nesse critério, uma política π é avaliada simplesmente pela seguinte função valor:

$$V^{\pi}(s) = \lim_{T\to\infty} \mathbb{E}\left[\sum_{t=0}^{T-1} c_t \mid \pi, s_0 = s\right]. \tag{2.1}$$

A partir da Equação 2.1, pode-se derivar o seguinte sistema de equações lineares para avaliar uma política (Puterman, 1994):

$$V^{\pi}(s) = \begin{cases} 0, & \text{se } s \in \mathcal{G} \\ c(s,\pi(s)) + \sum_{s'\in\mathcal{S}} [P(s,\pi(s),s')V^{\pi}(s')], & \text{caso contrário.} \end{cases} \tag{2.2}$$

De maneira análoga, a função valor **ótima**, denotada por V^*, pode ser definida em

termos do seguinte sistema de equações de Bellman (Bellman *et al.*, 1957):

$$V^*(s) = \begin{cases} 0, & \text{se } s \in \mathcal{G} \\ \min_{a \in \mathcal{A}} \left\{ c(s,a) + \sum_{s' \in \mathcal{S}} \left[P(s,a,s') V^*(s') \right] \right\}, & \text{caso contrário.} \end{cases} \tag{2.3}$$

Pode-se também, ao invés de avaliar um estado s, considerar um par estado-ação (s,a). Dessa maneira, denota-se por $Q^*(s,a)$ a qualidade ótima da ação a em um estado s:

$$Q^*(s,a) = \begin{cases} 0, & \text{se } s \in \mathcal{G} \\ c(s,a) + \sum_{s' \in \mathcal{S}} \left[P(s,a,s') V^*(s') \right], & \text{caso contrário.} \end{cases} \tag{2.4}$$

Assim, a política ótima π^* pode ser obtida em um estado s obtendo ação que minimiza $Q^*(s,a)$:

$$\begin{aligned} \pi^*(s) &= \arg\min_{a \in \mathcal{A}} \left\{ c(s,a) + \sum_{s' \in \mathcal{S}} \left[P(s,a,s') V^*(s') \right] \right\} \\ \pi^*(s) &= \arg\min_{a \in \mathcal{A}} \{ Q^*(s,a) \}. \end{aligned} \tag{2.5}$$

A política ótima π^* pode também ser definida como a política que minimiza a função valor V^π, i.e.

$$V^{\pi^*}(s) \leq V^\pi(s) \quad \forall \pi \in \Pi \text{ e } \forall s \in \mathcal{S}.$$

Se π é uma política própria, o valor $V^\pi(s)$ é finito, para todo $s \in \mathcal{S}$. Por outro lado, se $c(s,a) > 0$ para todo $s \notin \mathcal{G}$, então históricos que não alcançam a meta geram custos acumulados infinitos e qualquer política imprópria π' tem um valor infinito, ou seja, $V^{\pi'}(s) = \infty, \forall s \in \mathcal{S}$. Portanto, qualquer política ótima π^* para os critérios de SSP-MDPs convencionais ou de SSP-MDPs com *dead ends* evitáveis é uma política própria.

Note que a medida de custo acumulado esperado não é bem-definida para SSP-MDPs com *dead ends* inevitáveis, já que nesse caso não existem políticas próprias, e o valor de qualquer política imprópria diverge para o infinito. O capítulo 3 contém uma discussão sobre diferentes critérios para resolver problemas como esse.

Um exemplo de SSP-MDP, representado como um hipergrafo, pode ser visualizado na Figura 2.1. Nesse tipo de representação, os estados do SSP-MDP são os nós, e as ações, as hiperarestas do grafo. Ao seguir uma ação, a probabilidade do agente ir para um novo estado é marcada em cada ramo da sua hiperaresta. No caso da ação ser determinística, essa probabilidade é omitida. O problema exemplificado na figura tem 3 estados: s_1, s_2 e s_3, sendo s_3 o estado meta. Em s_1, a ação a pode ser tomada, levando o agente deterministicamente a s_2, ou a ação b é escolhida, levando o agente ao estado meta com probabilidade P, ou fazendo-o permanecer em s_1 com probabilidade $1 - P$. Em s_2, a ação a leva o agente à meta, e a ação b faz o agente permanecer em s_2, ambas com probabilidade 1. O custo é unitário (igual a 1) para qualquer ação em qualquer estado s, exceto quando s é um estado meta, caso em que o custo é 0.

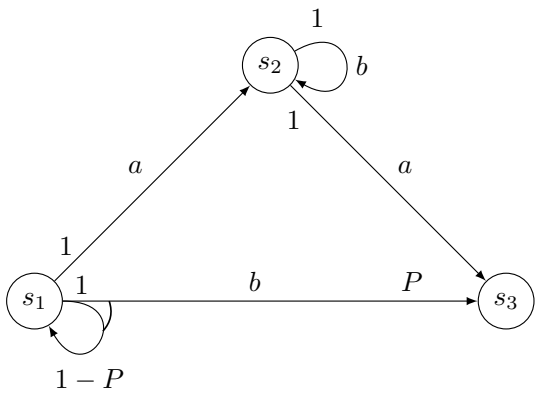


Figura 2.1: *Exemplo ilustrativo de um SSP-MDP. Nesse caso, o agente em s_1 pode ir diretamente para a meta com probabilidade P ao tomar a ação b, correndo o risco de continuar onde está com probabilidade $1 - P$, ou seguir o caminho determinístico que leva a s_2, executando a ação a. Em s_2, a ação a leva à meta e a ação b faz o agente permanecer em s_2. O custo de tomar qualquer ação é 1, exceto em estados meta, nos quais é 0.*

No SSP-MDP da Figura 2.1, existem duas políticas possíveis: $\pi_1(\cdot) = a$ e $\pi_2(\cdot) = b$. Considerando $P = 0.8$, $V^{\pi}(s_3) = 0$ para qualquer política $\pi \in \Pi$, já que s_3 é meta. $V^{\pi_1}(s_1)$ é dado por:

$$V^{\pi_1}(s_2) = c(s_2, a) + 1 \times V^{\pi_1}(s_2) = 1 + 1 \times 0 = 1.$$

Por outro lado, $V^{\pi_2}(s_1)$ não é bem-definido, já que nesse caso o agente nunca sai de s_2, o que faz o custo acumulado esperado ser infinito ($V^{\pi_2}(s_1) = \infty$).

Seja $V^{\pi_1}(s_1)$ dado por:

$$V^{\pi_1}(s_1) = c(s_1, a) + 1 \times V^{\pi_1}(s_2) = 1 + 1 \times 1 = 2,$$

e $V^{\pi_2}(s_1)$, por:

$$\begin{aligned} V^{\pi_2}(s_1) &= c(s_1, a) + [0.8V^{\pi_2}(s_3) + 0.2V^{\pi_2}(s_1)] \\ V^{\pi_2}(s_1) &= 1 + (0.8 \times 0 + 0.2V^{\pi_2}(s_1)) \\ V^{\pi_2}(s_1) &= 1 + 0.2V^{\pi_2}(s_1) \\ 0.8V^{\pi_2}(s_1) &= 1 \\ V^{\pi_2}(s_1) &= 1.25. \end{aligned}$$

A política ótima nesse SSP-MDP em s_1 é $\pi^* = \pi_2$, pois $V^{\pi_2}(s_1) < V^{\pi_1}(s_1)$.

As definições e equações apresentadas na presente seção fornecem a base comum para algoritmos que encontram a solução ótima para SSP-MDPs. Nas seções seguintes, serão apresentados alguns desses algoritmos.

2.3 Algoritmos Síncronos

Utilizando-se das equações apresentadas na seção anterior, podem-se utilizar métodos para calcular uma solução para um SSP-MDP. Algoritmos propostos com esse fim inicialmente consistem em, em cada iteração, calcular uma versão da função valor baseada na versão calculada na iteração anterior, depois de passar por todo o conjunto de estados a serem processados nessa iteração. Esses algoritmos são comumente chamados de **síncronos**, e serão apresentados na presente seção. Na seção a seguir, serão expostos algoritmos que calculam a versão atual da função valor a partir de valores calculados já nessa mesma iteração. São esses os algoritmos **assíncronos**.

2.3.1 Iteração de Valor

A partir da Equação 2.3, pode-se calcular o valor ótimo da tomada de decisão de um agente em um estado s, ao escolher a ação a que minimiza a soma do custo de executar a em s e do valor esperado do próximo estado, assumindo que o agente age de maneira ótima a seguir.

Mesmo que não se saiba o valor ótimo do próximo estado, é possível inicializar V com valores arbitrários e, iterativamente, atualizar o valor de cada estado na iteração i, representado pela função V_i, utilizando a seguinte equação de Bellman:

$$V_i(s) \leftarrow \min_{a \in \mathcal{A}} \left\{ c(s,a) + \sum_{s' \in S} \left[P(s,a,s') V_{i-1}(s') \right] \right\}. \tag{2.6}$$

A cada iteração, V_i se aproxima de V^* (ou seja, $\lim_{i \to \infty} V_i = V^*$) (Puterman, 1994). É chamado de resíduo da iteração i a diferença máxima entre $V_i(s)$ e $V_{i-1}(s)$ $\forall s \in \mathcal{S}$, aqui denotada por $\|V_i - V_{i-1}\|_\infty$ (Mausam e Kolobov, 2012). Como V_i se aproxima de V^* a cada iteração, conforme isso acontece, o resíduo diminui, até que no limite atinja 0 ($\lim_{i \to \infty} \|V_i - V_{i-1}\|_\infty = 0$).

A intuição do algoritmo de **Iteração de Valor** (*Value Iteration* - VI) (Bellman *et al.*, 1957) consiste em, por repetidas iterações, calcular a Equação 2.6 para cada estado, até que o resíduo seja menor que um parâmetro, representado como ϵ. Por isso, pode-se dizer que o algoritmo VI é ϵ-consistente. No Algoritmo 1, o pseudocódigo do algoritmo de Iteração de Valor pode ser visualizado. O laço da linha 8 é executado enquanto o resíduo atual não seja menor que ϵ. Em cada iteração, a qualidade de cada par estado-ação (s, a) é calculada na linha 12, de maneira que, para cada estado s, $V(s)$ é o valor da menor qualidade em s (linha 14), e $\pi(s)$ é a ação a que minimiza essa qualidade (linha 15).

2.3.2 Iteração de Política

O algoritmo de **Iteração de Política** (*Policy Iteration* - PI) (Howard, 1960) realiza um processo semelhante ao algoritmo de Iteração de Valor. No entanto, a diferença é que nesse algoritmo a iteração é feita no espaço de políticas. Isso pode ser útil pois é possível que a política ótima seja encontrada antes que V convirja para V^*_ϵ (a função valor de saída do Algoritmo 1). A abordagem em PI consiste principalmente na alternância entre dois passos: o passo de **avaliação de política**, e o de **melhoria de política**. Como visto na Seção 2.1,

Algoritmo 1: IteraçãoDeValor

Entrada: SSP-MDP $\mathcal{M}_s = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, \epsilon$
Saída: $V^*_\epsilon, \pi^*_\epsilon$

início
  Seja $V : \mathcal{S} \to \mathbb{R}$ e $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$
  para cada $s \in \mathcal{S}$ **faça**
    $V(s) \leftarrow 0$
    $Q(s, \cdot) \leftarrow 0$
  fim
  $res \leftarrow \infty$
  enquanto $res > \epsilon$ **faça**
    $V' \leftarrow V$
    para cada $s \in \mathcal{S}$ **faça**
      para cada $a \in \mathcal{A}$ **faça**
        $Q(s,a) = c(s,a) + \sum_{s' \in \mathcal{S}} [P(s,a,s')V(s')]$
      fim
      $V(s) = \min_{a \in \mathcal{A}} \{Q(s,a)\}$
      $\pi(s) = \arg\min_{a \in \mathcal{A}} \{Q(s,a)\}$
    fim
    $res \leftarrow \|V - V'\|_\infty$
  fim
  $V^*_\epsilon \leftarrow V$
  $\pi^*_\epsilon \leftarrow \pi$
  retorna $V^*_\epsilon, \pi^*_\epsilon$
fim

uma política pode ser avaliada pelo sistema de equações lineares definido na Equação 2.2. Dessa maneira, o passo de avaliação de política pode ser realizado utilizando um método de solução de sistemas de equações lineares. Outra maneira de realizar esse passo é resolver esse sistema de maneira iterativa, realizando o cálculo da seguinte expressão repetidas vezes, tal que a função valor inicial V^π_0 é definida arbitrariamente:

$$V^\pi_i(s) \leftarrow c(s, \pi(s)) + \sum_{s' \in \mathcal{S}} \left[P(s, \pi(s), s') V^\pi_{i-1}(s') \right]. \tag{2.7}$$

O algoritmo PI, assim como VI, é ϵ-consistente.

O passo de melhoria de política consiste em, a partir da função valor V^π de uma política π, melhorar π ao verificar se alguma ação melhora o seu valor, minimizando a seguinte equação de Bellman:

$$\pi(s) = \arg\min_{a \in \mathcal{A}} \left\{ c(s,a) + \sum_{s' \in \mathcal{S}} \left[P(s,a,s') V^\pi(s') \right] \right\}. \tag{2.8}$$

Se esse passo não encontrar uma política melhor, o seu resultado será a mesma política que a atual. Nesse caso, isso significa que não é possível encontrar uma política melhorada,

o que indica que a política ótima foi encontrada.

O pseudocódigo do algoritmo de Iteração de Política pode ser visto no Algoritmo 2. A cada iteração do laço da linha 3, a política π é avaliada (linha 4), e a política é melhorada em cada estado na linha 7. Se a política arbitrária inicial do algoritmo (linha 2) for própria, PI converge para a política ótima.

Algoritmo 2: IteraçãoDePolítica

Entrada: SSP-MDP $\mathcal{M}_s = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle$
Saída: V^*, π^*

1 **início**
2 Inicialize $\pi : \mathcal{S} \to \mathcal{A}$ com uma política própria arbitrária
3 **repita**
4 Avalie a política π e salve o valor em $V : \mathcal{S} \to \mathbb{R}$
5 $\pi' \leftarrow \pi$
   ▷ `Melhoria de política`
6 **para cada** $s \in \mathcal{S}$ **faça**
7 $\pi(s) = \arg\min_{a \in \mathcal{A}} \left\{ c(s,a) + \sum_{s' \in \mathcal{S}} [P(s,a,s')V(s')] \right\}$
8 **fim**
9 **até** $\pi = \pi'$;
10 $V^* \leftarrow V$
11 $\pi^* \leftarrow \pi$
12 **retorna** V^*, π^*
13 **fim**

2.4 Algoritmos Assíncronos

Os algoritmos apresentados na seção anterior fornecem uma base para a solução de SSP-MDPs, mas dificilmente são adequados em situações reais, pois em domínios grandes pode ser impraticável fazer passagens no espaço de estados por completo. Nem sempre é necessário que todos os estados tenham seus valores atualizados com a mesma precisão, demandando um número reduzido de atualizações. Além disso, seguindo uma política ótima a partir de um estado inicial s_0, é comum que um subconjunto considerável dos estados não seja visitado, o que pode dispensar o cálculo do valor dos estados desse conjunto.

Assim, algoritmos mais eficientes, classificados como assíncronos, foram propostos como alternativas que normalmente atacam esse tipo de abordagem em SSP-MDPs, para reduzir consideravelmente o número médio de atualizações por estado. Nas seguintes subseções serão analisados dois algoritmos de busca heurística, AO* (Martelli e Montanari, 1973; Martelli e Montanari, 1978; Nilsson, 1982) e LAO* (Hansen e Zilberstein, 2001). Esses algoritmos têm como objetivo solucionar SSP-MDPs sob um custo computacional reduzido dependendo do problema, se comparados com os algoritmos síncronos já vistos.

2.4.1 AO*

Como demonstrado na Figura 2.1, um MDP pode ser representado como um hipergrafo, ao considerar seus estados como nós e as transições entre esses estados como hiperarestas (isto é, arestas que ligam um nó a k outros nós). Um grafo com essas características também pode ser representado como um grafo AND/OR (Nilsson, 1982).

Com o algoritmo AO* (Martelli e Montanari, 1973; Martelli e Montanari, 1978; Nilsson, 1982), é possível encontrar a solução de um grafo AND/OR acíclico utilizando busca heurística. No caso de SSP-MDPs, se o problema puder ser representado por um hipergrafo que não contém ciclos, então AO* pode ser aplicado para computar a sua política ótima (Hansen e Zilberstein, 2001). A busca do AO* é iniciada a partir de um estado inicial s_0, dado como entrada para o algoritmo, e é guiada por uma função heurística $h : \mathcal{S} \to \mathbb{R}$, utilizada para inicializar os valores de estados visitados pela primeira vez. Dependendo do problema, o subconjunto de $\mathcal{S}$ avaliado pelo algoritmo pode ser consideravelmente menor que o conjunto completo de estados, o que pode resultar em uma substancial melhora de desempenho.

O Algoritmo 3 contém o pseudocódigo do AO*, e funciona da seguinte maneira: primeiro, os dois grafos G e G' são criados, ambos inicialmente contendo apenas s_0 (linha 4). G é mantido como o grafo de melhor solução parcial, e G' como grafo explícito (um grafo que contém todos nós e arestas visitados pelo algoritmo). Além disso, considere a função $\mathcal{S}_{solv}(s)$ a função que tem o valor de 1 se o estado s já foi resolvido ou de 0, caso contrário (definida na linha 3). Enquanto o estado inicial ainda não tiver sido resolvido, um desses estados é escolhido e expandido (linhas 6–18). O passo de expansão do estado s consiste em adicionar os seus sucessores ao grafo G', e inicializar o valor deles de acordo com h. Para o caso em que o estado sucessor seja meta, ele é inicializado com 0.

Depois, é feita a atualização dos valores de determinados estados da seguinte maneira: um conjunto Z, contendo apenas s, é criado (linha 19). Em cada iteração do laço da linha 20, é retirado um estado s' de Z, tal que não existem estados em Z que podem ser alcançados ao seguir as ações que minimizam o custo esperado a partir de s'. O custo ótimo desse estado é então atualizado na linha 23. Se todos estados sucessores de s' seguindo ações marcadas em G já tiverem sido resolvidos, s' também é marcado como resolvido (linha 25). Se s' já foi resolvido, ou se houve uma melhora no seu valor atual, então todos os estados predecessores de s' em G' seguindo ações marcadas em G são adicionados a Z, para que seus valores também sejam atualizados. O laço continua até que Z esteja vazio, e, quando finalizado, o algoritmo atualiza o grafo G adicionando s, e marcando as ações que originam o melhor valor para cada estado. O laço da linha 5, por sua vez, é executado enquanto o estado inicial s_0 não tiver sido marcado como resolvido. Quando isso acontece, o laço é finalizado e o algoritmo termina a sua execução.

Para que o AO* garanta a otimalidade da política parcial encontrada a partir de s_0, é necessário que a heurística utilizada seja **admissível**. Isso significa que a estimativa do valor definido por ela para um estado s deve ser menor ou igual que $V^*(s)$.

Algoritmo 3: AO*

Entrada: SSP-MDP $\mathcal{M}_s = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, h$
Saída: G^*

início
  Seja $V : \mathcal{S} \to \mathbb{R}$
  Seja $S_{solv} : \mathcal{S} \to \{0, 1\}$
  $G \leftarrow \{s_0\}, G' \leftarrow \{s_0\}$
  enquanto $S_{solv}(s_0) = 0$ **faça**
    ▷ `Expansão de` s`:`
    $s \leftarrow$ escolha um estado $s \in G'$ tal que s ainda não foi expandido e $s \notin \mathcal{G}$
    $S_{solv}(s) \leftarrow 1$
    $\mathcal{S}' \leftarrow \{s' \in \mathcal{S} \mid \exists a \in \mathcal{A}$ e $P(s, a, s') > 0\}$ ▷ `Estados sucessores de` s
    para cada $s' \in \mathcal{S}'$ **faça**
      $G' \leftarrow G' \cup \{s'\}$
      se $s' \in \mathcal{G}$ **então**
        $V(s') \leftarrow 0$
        $S_{solv}(s') \leftarrow 1$
      senão
        $V(s') \leftarrow h(s')$
        $S_{solv}(s) \leftarrow 0$
      fim
    fim
    ▷ `Atualização de` V `e` G`:`
    $Z \leftarrow \{s\}$
    enquanto Z *não está vazio* **faça**
      Retire um estado s' de Z tal que não existem estados em Z que podem ser alcançados ao seguir ações marcadas a partir de s'
      $v_{old} \leftarrow V(s')$
      $V(s') \leftarrow \min_{a \in \mathcal{A}} \{c(s', a) + \sum_{s'' \in \mathcal{S}} [P(s', a, s'')V(s'')]\}$
      se $S_{solv}(s'') = 1, \forall s'' \in \mathcal{S}_{succ}(s')$ **então**
        $S_{solv}(s') \leftarrow 1$
      fim
      se $V(s') < v_{old}, \forall s'' \in \mathcal{S}_{succ}(s')$ *ou* $S_{solv}(s') = 1$ **então**
        $Z \leftarrow Z \cup \{s'' \mid s''$ é um estado predecessor de s' em G' seguindo ações marcadas em $G\}$
      fim
    fim
    Atualize G com s e com as melhores ações para cada estado modificado no passo anterior
  fim
fim
retorna G

2.4.2 LAO*

O algoritmo LAO* (Hansen e Zilberstein, 2001), assim como o AO*, também utiliza busca heurística para encontrar a solução de um SSP-MDP representando-o como um hipergrafo. A melhoria do LAO*, no entanto, é que ele aceita SSP-MDPs cujos grafos admitem laços. Isso é possiblitado ao modificar a maneira que os custos são atualizados para cada estado processado, realizando passos de iteração de valor para estados em G até que o resíduo seja menor do que um valor ϵ.

O Algoritmo 4 contém o pseudocódigo do LAO*, e funciona da seguinte maneira: como no AO*, primeiro, os dois grafos G e G' (grafos de melhor solução parcial e explícito, respectivamente) são criados, ambos inicialmente contendo apenas s_0 (linha 3). É executado um laço na linha 4 enquanto G possui nós correspondentes a estados que ainda não foram expandidos e que não são metas. Também como no AO*, o passo de expansão é executado no início do laço, nas linhas 5–14.

O que difere o LAO* do AO* é como os valores de cada estado são atualizados. No primeiro, é executado o algoritmo de iteração de valor (linha 17) em Z, um conjunto que contém s e os estados que alcançam s em G' seguindo as melhores ações definidas em G, atualizando os valores desses estados e das melhores ações em G, caso elas mudem, até que o resíduo seja menor que o valor do parâmetro ϵ. Isso não é necessário no AO*, porque a condição de que o grafo seja acíclico em conjunto com o procedimento de marcação de estados como resolvidos é suficiente para que a solução ótima seja encontrada. Na linha 20, é feita uma iteração de valor novamente sobre os estados de G, para que se garanta que os estados nesse grafo tenham seus valores computados até convergência, já que ele foi recém-atualizado. Caso a melhor solução parcial não mude, o algoritmo termina e retorna G. Caso contrário, o laço da linha 4 é executado novamente, e o processo se repete.

Por fim, assim como no AO*, para que o LAO* garanta a otimalidade da política parcial encontrada a partir de s_0, é necessário que a heurística utilizada seja **admissível**.

Algoritmo 4: LAO*

Entrada: SSP-MDP $\mathcal{M}_s = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, h, \epsilon$
Saída: G^*

1 **início**
2 Seja $V : \mathcal{S} \to \mathbb{R}$
3 $G \leftarrow \{s_0\}, G' \leftarrow \{s_0\}$
4 **enquanto** G tiver algum nó que não pertence a $\mathcal{G}$ (não é meta) e ainda não foi expandido **faça**
   ▷ `Expansão de` s`:`
5 $s \leftarrow$ escolha um estado $s \in G'$ tal que s ainda não foi expandido e $s \notin \mathcal{G}$
6 $\mathcal{S}' \leftarrow \{s' \in \mathcal{S} \mid \exists a \in \mathcal{A}$ e $P(s, a, s') > 0\}$ ▷ `Estados sucessores de` s
7 **para cada** $s' \in \mathcal{S}'$ **faça**
8 $G' \leftarrow G' \cup \{s'\}$
9 **se** $s' \in \mathcal{G}$ **então**
10 $V(s') \leftarrow 0$
11 **senão**
12 $V(s') \leftarrow h(s')$
13 **fim**
14 **fim**
   ▷ `Atualização de` V `e` G`:`
15 $Z \leftarrow \{s\} \cup \{$
16 $s' \in \mathcal{S} \mid s'$ é ancestral de s em G' seguindo ações marcadas em $G\}$
17 $V \leftarrow$ IteraçãoDeValor$(\langle Z, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, \epsilon)$
18 Atualize G com s e com as melhores ações para cada estado modificado no passo anterior
19 **fim**
   ▷ `Teste de convergência`
20 $V \leftarrow$ IteraçãoDeValor$(\langle G, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, \epsilon)$
21 Atualize G com s e com as melhores ações para cada estado modificado no passo anterior
22 **se** G tiver sido modificado de maneira que ele tenha um estado que não foi expandido **então**
23 Volte para a linha 4 (passo de expansão)
24 **fim**
25 **fim**
26 **retorna** G

Capítulo 3

Critérios para SSP-MDPs com *Dead Ends* Inevitáveis

O critério convencional para SSP-MDPs pode não ser suficiente para resolver esses problemas na presença de *dead ends*. Por essa razão, critérios alternativos são propostos na literatura com o objetivo de tratar esses casos. As seguintes seções contém uma revisão de alguns desses critérios em mais detalhes. Primeiro, resumos de históricos (C_T, β_T) e a função de probabilidade à meta $P_G^\pi(s)$ serão definidos a seguir.

Um resumo $(C_T, \beta_T) \in \mathbb{R} \times \{0, 1\}$ é uma variável aleatória bidimensional para históricos finitos aleatórios $h_T = \{\langle s_0, a_0, c_0\rangle, \langle s_1, a_1, c_1\rangle, \ldots, s_T\}$ no passo de tempo T. Se $s_T \in \mathcal{G}$ então $C_T = \sum_{t=0}^{T-1} c(s_t, a_t)$ e $\beta_T = 1$, senão $C_T = \sum_{t=0}^{T-1} c(s_t, a_t)$ e $\beta_T = 0$. Ou seja, C_T é o custo acumulado de um histórico até o tempo T, e o valor de β_T indica se um estado meta foi atingido até T.

A função de probabilidade à meta $P_G^\pi(s)$ descreve a probabilidade de alcançar um estado meta ao seguir a política π a partir de um estado s. Ou seja,

$$P_G^\pi(s) = \lim_{t \to \infty} \Pr(s_t \in \mathcal{G} | \pi, s_0 = s). \tag{3.1}$$

3.1 MAXPROB

Um método válido para se escolher políticas em SSP-MDPs é o de maximizar a probabilidade à meta a partir de um estado específico. O critério MAXPROB (Kolobov, D. S. Weld *et al.*, 2011) tem como política ótima a política que maximiza a probabilidade de alcançar a meta a partir do estado inicial s_0. Ou seja, uma política π^* é ótima segundo o critério MAXPROB se:

$$\pi^* \in \arg\max_{\pi \in \Pi} P_G^\pi(s_0).$$

Apesar desse método ser bem-definido mesmo na presença de *dead ends*, considerar apenas a probabilidade de alcançar a meta pode ser uma estratégia muito simples. Por exemplo, duas políticas podem ter a mesma probabilidade à meta mas valores muito

diferentes para uma medida de custo. Pelo critério MAXPROB, ambas políticas teriam a mesma preferência apesar disso.

3.2 Critérios Lexicográficos

Ao invés de apenas maximizar a probabilidade à meta, pode-se tomar decisões a partir de critérios que seguem a preferência lexicográfica de minimizar alguma medida de custo considerando apenas políticas que maximizam a probabilidade à meta $P_G^\pi(s)$.

Ao longo deste trabalho, critérios que seguem esse tipo de preferência serão chamados de *critérios lexicográficos*. Primeiro, considere o conjunto de políticas Π^{MP} que maximizam a probabilidade à meta a partir do estado inicial. Ou seja, $\Pi^{MP} = \{\pi \mid P_G^\pi(s_0) = \max_{\pi' \in \Pi} P_G^{\pi'}(s_0)\}$. Segundo, seja $\bar{C}_L^\pi$ uma medida de custo qualquer bem-definida sobre distribuições de históricos. Políticas ótimas π^* sob critérios lexicográficos são dadas pela seguinte expressão:

$$\pi^* \in \arg\min_{\pi \in \Pi^{MP}} \bar{C}_L^\pi.$$

De maneira geral, a medida de custo $\bar{C}_L^\pi$ precisa considerar custos acumulados gastos durante o processo, mas ao mesmo tempo "encolher" custos acumulados infinitos originados por históricos que possuem *dead ends*. As próximas subseções contém a definição de dois critérios lexicográficos, tais como algumas de suas principais características.

3.2.1 *Stochastic Safest and Shortest Path* (S^3P)

O critério *Stochastic Safest and Shortest Path* (Caminho Estocástico mais Curto e Seguro, ou apenas S^3P) (Teichteil-Königsbuch, 2012) dá uma maior preferência a políticas que minimizam o custo acumulado esperado de históricos que alcançam a meta, dado que essas políticas maximizam probabilidade à meta.

De maneira mais formal, seja $\bar{C}_G^\pi(s)$ o custo acumulado esperado ao alcançar a meta considerando apenas históricos que alcançam metas ao seguir a política π a partir de estado s. Ou seja:

$$\bar{C}_G^\pi(s) = \lim_{T \to \infty} \mathbb{E}\left[\sum_{t=0}^{T-1} c_t \mid \pi, s_0 = s, \beta_T = 1\right].$$

O critério S^3P trata custos acumulados infinitos ao simplesmente ignorar o custo de históricos que não alcançam metas, já que todas as políticas em Π^{MP} possuem a mesma densidade de probabilidade de históricos que não alcançam a meta. Como denotado em F. W. Trevizan *et al.*, 2017, quando $c(s,a) > 0, \forall s \in \mathcal{S} \setminus \mathcal{G}$ e $a \in \mathcal{A}$, o critério S^3P é equivalente ao critério iSSPUDE (*Infinite-penalty SSP with Unavoidable Dead Ends* – SSP de Penalidade Infinita com *Dead Ends* inevitáveis) (Kolobov, D. Weld *et al.*, 2012).

3.2.2 *Min-Cost given Max-Prob (MCMP)*

Ao invés de considerar apenas históricos que alcançam a meta, o critério *Min-Cost given Max-Prob* (Custo Mínimo dada Probabilidade Máxima – MCMP) fornece uma alternativa

para resolver SSP-MDPs com *dead ends* ao transformar históricos que alcançam esse tipo de estado, também considerando apenas políticas que maximizam a probabilidade à meta. Tais históricos são transformados de maneira que apenas os passos até o momento que um *dead end* é atingido são considerados, o que faz com que a medida de custo desse critério não divirja.

Considere como $\psi : \mathcal{H} \to \mathcal{H}$ a função que transforma históricos da seguinte maneira:

$$\psi(h) = \begin{cases} \{s_i\}, & \text{se } |h| = 1 \text{ ou } P_G^\pi(s_i) = 0 \\ \{\langle s_i, a_i, c_i\rangle\} \cup \psi(\{\langle s_{i+1}, a_{i+1}, c_{i+1}\rangle, \langle s_{i+2}, a_{i+2}, c_{i+2}\rangle, \dots\}), & \text{caso contrário,} \end{cases} \tag{3.2}$$

tal que $h = \{\langle s_i, a_i, c_i\rangle, \langle s_{i+1}, a_{i+1}, c_{i+1}\rangle, \dots\}$.

Ou seja, ψ *poda* um histórico h no seu primeiro estado caso esse estado seja um *dead end*, ou caso ele já seja um histórico de um estado apenas. Caso contrário, ψ é aplicada recursivamente para o restante do histórico.

Seja $\bar{C}^\pi_{MCMP}(s)$ o custo esperado considerando históricos podados pela função ψ:

$$\bar{C}^\pi_{MCMP}(s) = \mathbb{E}\left[\sum_{t=0}^{|\psi(h)|} c_t \mid \pi, s_0 = s\right].$$

Diferentemente do critério S^3P, no MCMP os custos de históricos que não alcançam metas não são ignorados. Do contrário, seus custos são considerados até que um *dead end* seja alcançado. Novamente, isso faz sentido apenas porque todas as políticas em Π^{MP} possuem a mesma densidade de probabilidade de históricos que não alcançam a meta.

3.3 Critérios que Realizam Compromissos entre Custos e Metas

Uma alternativa para critérios lexicográficos são métodos que permitem um compromisso arbitrário entre uma medida finita de custo e probabilidade à meta. Uma abordagem ingênua seria combinar as funções P_G^π e $\bar{C}_L^\pi(s)$, atribuindo pesos para cada uma dessas duas funções de maneira arbitrária. No entanto, note que utilizar as funções $\bar{C}_G^\pi$ e $\bar{C}^\pi_{MCMP}$ faz sentido somente porque elas são utilizadas para comparar políticas com valores iguais de P_G^π.

Assim como em critérios lexicográficos, critérios que fazem compromissos entre custos e metas precisam "encolher" o custo infinito de históricos que não alcançam metas. Exemplos de critérios que seguem esse raciocínio são os critérios fSSPUDE (*Finite-penalty SSP with Unavoidable Dead Ends* - SSP de Penalidade Finita com *Dead Ends* inevitáveis) (Mausam e Kolobov, 2012), um critério que adiciona uma ação artificial para desistir do processo e ir diretamente para uma meta a partir de qualquer estado, pagando uma penalidade finita recebida como parâmetro pelo critério; e o critério de custo descontado

(Teichteil-Königsbuch *et al.*, 2011), que utiliza um fator de desconto para fazer com que o custo acumulado esperado de políticas com probabilidade à meta menor que 1 convirja. Para esses métodos, não existe a necessidade de primeiro maximizar a probabilidade à meta, e em seguida minimizar uma medida de custo definida. No caso deles, uma única medida de custo é definida, cada uma em conjunto com um parâmetro que faz o compromisso entre custo e probabilidade à meta.

3.3.1 Critério fSSPUDE

Uma maneira de considerar históricos que alcançam *dead ends* ao realizar um compromisso entre probabilidade à meta e custo é definir um SSP-MDP estendido com uma ação artificial q e uma penalidade finita D. Quando q é tomada, o processo transita para um estado meta artificial e paga um custo finito D. O custo D pode ser interpretado então como uma penalidade para desistir do processo e finalizá-lo.

É assim que o critério fSSPUDE (Kolobov, D. Weld *et al.*, 2012) é definido. Formalmente, $P(s,q,g) = 1$ e $c(s,q) = D$, $\forall s \in \mathcal{S}$ e um $g \in \mathcal{G}$. Nesse critério o objetivo é encontrar a política que minimiza o custo acumulado esperado do SSP-MDP resultante dessa alteração. Esse custo será aqui denotado como $\bar{C}_D^\pi$, tal que $\bar{C}_D^\pi = \lim_{T\to\infty} \mathbb{E}\left[\sum_{t=0}^{T-1} c(s_t,a_t) \,\middle|\, \pi, s_0\right]$. Então, o critério fSSPUDE fornece uma maneira de definir o quão indesejado é o evento de alcançar um estado *dead-end* ao escolher um valor apropriado para a penalidade D.

3.3.2 Critério de Custo Descontado

O uso de um fator de desconto $\gamma \in (0,1)$ é comumente observado em MDPs de horizonte infinito, porque o custo descontado $\bar{C}_\gamma^\pi = \lim_{T\to\infty} \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t c(s_t,a_t) \,\middle|\, \pi, s_0\right]$ converge para um valor finito nesse caso, mesmo para históricos infinitos (Teichteil-Königsbuch *et al.*, 2011). Por essa razão, é possível utilizar esse critério como um método bem-definido para resolver SSP-MDPs com *dead ends*.

3.3.3 MAXPROB como um Comportamento Limitante

Ambos critérios fSSPUDE e de custo descontado dependem respectivamente de uma penalidade finita D e um fator de desconto $\gamma \in (0,1)$. Uma questão interessante a ser feita é: no limite ($D \to \infty$ e $\gamma \to 1$) o critério MAXPROB é obtido? A resposta é sim (Freire e Delgado, 2017).

Finalmente, se existe uma política própria, os métodos S^{3}P, MCMP, custo descontado com $\gamma \to 1$ e fSSPUDE com $D \to \infty$ resultam na mesma política ótima que o critério convencional para SSP-MDPs.

Capítulo 4

Políticas que Priorizam Metas

Na última seção foram apresentados critérios que definem medidas de otimalidade em SSP-MDPs na presença de *dead ends* inevitáveis. Enquanto critérios que maximizam probabilidade à meta não possuem parâmetros, é razoável questionar o quão racionais são decisões tomadas sob esses critérios. Por outro lado, critérios que permitem compromissos arbitrários entre custo e metas o fazem a partir de uma escolha apropriada de valores para seus parâmetros, o que pode se fazer questionar a dificuldade de escolher esses valores apropriadamente. Na presente seção essas questões são discutidas com mais detalhes, e uma abordagem para definir um critério alternativo para SSP-MDPs com *dead ends* é apresentada.

Para começar, considere o SSP-MDP na Figura 4.1, em que $1 \ll \ell \ll L$ são custos arbitrários, $P \in (0, 1]$ é uma probabilidade arbitrária, $\delta \in (0, P)$ é um escalar arbitrário, s_g é um estado meta, e s_d um *dead end*.

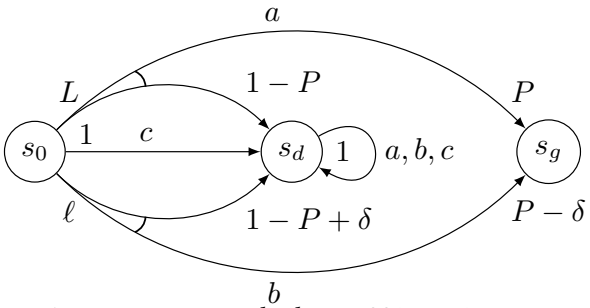


Figura 4.1: *Exemplo de um SSP-MDP.*

No estado inicial, o agente tem três ações para tomar: a, b e c. A ação a é a que tem a probabilidade à meta máxima P, mas que leva o agente a pagar um custo grande L. Essa seria a escolha ótima para os critérios MCMP e S^{3}P. A ação b tem probabilidade à meta $P - \delta$, um valor muito próximo de P, e por sua vez faz com que o agente pague ℓ, um custo muito menor que L. Uma questão importante é a de se maximizar probabilidade à meta é uma escolha racional quando $\delta \to 0$ e $L - \ell \to \infty$.

A ação c leva o agente diretamente para o *dead end* s_d ao pagar 1 de custo. Assim que o agente alcança s_d, ele paga 1 para os próximos passos de tempo. Outra pergunta que pode

ser feita é se é racional a escolha de c em s_0, mesmo quando $P \to 0$ e $\ell \to \infty$.

4.1 Quanto Pagar por um Aumento Infinitesimal de Probabilidade à Meta?

Apesar de critérios baseados em maximizar probabilidade à meta como MAXPROB, MCMP, e S^3P parecerem não ter grandes desvantagens, eles não permitem compromissos entre grandes custos acumulados e pequenas perdas em probabilidade à meta. (*i*) Considere uma pessoa planejando uma rota para chegar no aeroporto a tempo para pegar um voo. Para evitar uma situação imprevista como um engarrafamento ou um problema no carro, essa pessoa pode decidir sair de casa dias antes do voo. (*ii*) Considere um estudante estudando para uma prova. Se esse estudante acredita que mais horas de estudo aumentam a sua chance de passar na prova, então ele pode decidir passar qualquer tempo disponível estudando. (*iii*) Considere uma empresa que é responsável pela prevenção de acidentes de aviões. Se um de seus engenheiros acredita que mais horas de manutenção preventiva diminuem a chance de um acidente, ele pode decidir passar todo tempo disponível em manutenções para esse avião.

Agentes que seguem critérios baseados em maximizar probabilidade à meta podem escolher políticas que fazem o agente pagar custos arbitrariamente grandes em média, ao invés de políticas com custos muito menores com uma perda pequena em probabilidade à meta. Em outras palavras, nesses casos não existe uma maneira de realizar um compromisso entre políticas com grandes custos e outras que não maximizam a probabilidade à meta, mesmo que elas levem a valores muito menores sob alguma medida de custo. Então, critérios que maximizam probabilidade à meta não permitem a realização de compromissos infinito-infinitesimais.

Teorema 1 (**Falta de compromissos infinito-infinitesimais para critérios que maximizam probabilidade à meta**). *Considere um agente que toma decisões ao maximizar probabilidade à meta. Existe um SSP-MDP $\mathcal{M}$ em que tal agente paga um aumento arbitrariamente grande de custo acumulado ΔC para obter um aumento arbitrariamente pequeno de probabilidade à meta ΔP.*

Demonstração. Considere o SSP-MDP da Figura 4.1 e escolha valores $\delta < \Delta P$ e $L - \ell > \Delta C$. Além disso, seja π a política que toma a ação a em s_0 e π', a política que escolhe b. Sob qualquer critério que maximiza probabilidade à meta, o agente preferirá π ao invés de π', mesmo que a probabilidade à meta de π seja maior por no máximo ΔP, e o custo de π seja maior que o custo de π' por no mínimo ΔC. □

4.2 Até Quanto um Aumento de Probabilidade à Meta Compensa ter Custos Infinitos?

Na seção anterior, foi provado que maximizar probabilidade à meta não permite a realização de compromissos infinito-infinitesimais, porque a partir desse método um custo arbitrariamente grande pode ser pago em favor de um acréscimo infinitesimal

em probabilidade à meta. Se esse tipo de compromisso é desejável, pode-se perguntar para o quanto de acréscimo em probabilidade à meta se torna razoável pagar um custo arbitrariamente grande (infinito). A próxima definição mensura tal acréscimo de maneira relativa.

Considere a seguinte notação para políticas e históricos: $A \succ B$ significa que o tomador de decisões prefere A a B, $A \sim B$ significa que o tomador de decisões tem preferência igual para A e B, e $A \succeq B$ significa que o tomador de decisões tem preferência maior ou igual por A em relação a B.

Definição 3 (Priorização de probabilidade à meta α-forte). *Considerando $0 \leq \alpha \leq 1$, um critério de decisão tem priorização de probabilidade à meta α-forte se, para todos SSP-MDPs $\mathcal{M}$ e para todos pares de políticas $(\pi, \pi') \in \Pi \times \Pi$, a seguinte condição é verdadeira:*

$$\pi \succeq \pi' \implies \frac{P_G^{\pi}(s_0)}{P_G^{\pi'}(s_0)} \geq \alpha,$$

tal que $P_G^{\pi'}(s_0) > 0$. Se π^* é a política ótima, então para toda política $\pi' \in \Pi$ a seguinte equação é verdadeira:

$$\pi^* \succeq \pi' \implies \frac{P_G^{\pi^*}(s_0)}{P_G^{\pi'}(s_0)} \geq \alpha.$$

Note que $\frac{P_G^{\pi}(s_0)}{P_G^{\pi'}(s_0)} \geq \alpha$ é uma condição necessária para que $\pi \succeq \pi'$ seja verdadeiro.

Entre os seus possíveis tipos, existem dois casos especiais de priorização de probabilidade à meta α-forte: α-forte com $\alpha = 1$, chamada de **1-forte**, e α-forte com $\alpha \not> 0$, chamada de **apenas-0-forte**.

O seguinte resultado mostra que critérios que maximizam probabilidade à meta tem priorização de probabilidade à meta 1-forte.

Teorema 2 (Garantia de priorização de probabilidade à meta 1-forte para critérios lexicográficos e MAXPROB). *Critérios lexicográficos e o critério MAXPROB garantem priorização de probabilidade à meta 1-forte.*

Demonstração. Sob critérios lexicográficos e o critério MAXPROB, para uma política π ser preferível a outra política π' ($\pi \succeq \pi'$), $P_G^{\pi}(s_0)$ precisa ser maior ou igual que $P_G^{\pi'}(s_0)$. Isso significa que $\frac{P_G^{\pi}(s_0)}{P_G^{\pi'}(s_0)} \geq 1$ é uma condição necessária para $\pi \succeq \pi'$. Então, $\pi \succeq \pi' \implies \frac{P_G^{\pi}(s_0)}{P_G^{\pi'}(s_0)} \geq 1$, o que significa que esses critérios tem priorização de probabilidade à meta 1-forte. □

Considere novamente o SSP-MDP da Figura 4.1. Se o critério de decisão utilizado tem priorização de probabilidade à meta 1-forte, então a ação a é ótima independentemente dos valores de P, L, ℓ, e δ. Se o critério de decisão tem priorização de probabilidade à meta apenas-0-forte, a ação c pode ser ótima mesmo considerando o fato de ela nunca poder alcançar um estado meta. Se o critério de decisão tem priorização de probabilidade à meta α-forte com $0 < \alpha < 1$, a ação b pode ser ótima dependendo dos valores de P, L, ℓ, e δ e/ou dos parâmetros do critério.

Na Seção 3.3.3, foi demonstrado que os critérios de compromisso de custo e meta fSSPUDE e de custo descontado aproximam o critério MAXPROB no limite (quando $D \to \infty$ e $\gamma \to 1$, respectivamente). Pelo Teorema 2, se conclui que o critério MAXPROB tem priorização de probabilidade à meta 1-forte, o que significa que o α da condição $\pi \succeq \pi' \implies \frac{P_G^{\pi}(s_0)}{P_G^{\pi'}(s_0)} \geq \alpha$ da Definição 3 tende a 1 para os critérios fSSPUDE e de custo descontado, dado um SSP-MDP específico (novamente, quando $D \to \infty$ e $\gamma \to 1$, respectivamente). Pode-se perguntar se existe uma configuração dos parâmetros de compromisso (fator de desconto γ e penalidade finita D) que garante a priorização de probabilidade à meta α-forte para um valor de α e SSP-MDP $\mathcal{M}$ arbitrários. Os teoremas 3 e 4 mostram que isso não é possível. Na verdade, enquanto um α arbitrário pode ser garantido ao escolhê-lo para um SSP-MDP específico, não é possível escolher um $\alpha > 0$ que seja suficiente para qualquer SSP-MDP. Isso significa que uma política π que não alcança a meta ($P_G^{\pi}(s_0) = 0$) pode ser escolhida no lugar de qualquer outra política π' que tem alguma chance de alcançar a meta ($P_G^{\pi'}(s_0) > 0$).

Teorema 3 (Critério fSSPUDE não tem priorização de probabilidade à meta α-forte para $\alpha > 0$). *No critério fSSPUDE, dado um valor de α arbitrário tal que $\alpha > 0$, não existe uma penalidade finita D que garante priorização de probabilidade à meta α-forte.*

Demonstração. Considere o SSP-MDP da Figura 4.1 com $L \geq D$. Seja $\pi(s) = c$ e $\pi'(s) = a, \forall s \in \mathcal{S} \setminus \mathcal{G}$. Os valores de π e π' para o estado s_d são dados pela seguinte equação:

$$\bar{C}_D^{\pi}(s_d) = \bar{C}_D^{\pi'}(s_d) = \min\{\infty, D\} = D.$$

Então, o valor de $\bar{C}_D^{\pi}(s_0)$ é:

$$\bar{C}_D^{\pi}(s_0) = \min\{1 + \bar{C}_D^{\pi}(s_d), D\} = \min\{1 + D, D\} = D.$$

Além disso, o valor de $\bar{C}_D^{\pi'}(s_0)$ é:

$$\begin{aligned}
\bar{C}_D^{\pi'}(s_0) &= \min\{L + P\bar{C}_D^{\pi'}(s_g) + (1 - P)\bar{C}_D^{\pi'}(s_d), D\}, \\
\bar{C}_D^{\pi'}(s_0) &= \min\{L + D(1 - P), D\}, \\
\bar{C}_D^{\pi'}(s_0) &= D.
\end{aligned}$$

Logo, $\bar{C}_D^{\pi}(s_0) = \bar{C}_D^{\pi'}(s_0)$ e $\pi \sim \pi'$, mas $\frac{P_G^{\pi}(s_0)}{P_G^{\pi'}(s_0)} = 0$, o que completa a prova. □

Teorema 4 (Critério de custo descontado não tem priorização de probabilidade à meta α-forte para $\alpha > 0$). *No critério de custo descontado, dado um valor de α arbitrário tal que $\alpha > 0$, não existe um fator de desconto $\gamma < 1$ que garante priorização de probabilidade à meta α-forte.*

Demonstração. Considere o SSP-MDP da Figura 4.1 com $L \geq \frac{1}{1-\gamma}$. Seja $\pi(s) = c$ e $\pi'(s) =$

$a, \forall s \in \mathcal{S} \setminus \mathcal{G}$. Os valores de π e π' para o estado s_d são dados pela seguinte equação:

$$\bar{C}^{\pi}_{\gamma}(s_d) = \bar{C}^{\pi'}_{\gamma}(s_d) = 1 + \gamma \bar{C}^{\pi'}_{\gamma}(s_d),$$
$$\bar{C}^{\pi}_{\gamma}(s_d) = \bar{C}^{\pi'}_{\gamma}(s_d) = \frac{1}{1-\gamma}.$$

Então, o valor de $\bar{C}^{\pi}_{\gamma}(s_0)$ é:

$$\bar{C}^{\pi}_{\gamma}(s_0) = 1 + \gamma \bar{C}^{\pi}_{\gamma}(s_d),$$
$$\bar{C}^{\pi}_{\gamma}(s_0) = 1 + \frac{\gamma}{1-\gamma},$$
$$\bar{C}^{\pi}_{\gamma}(s_0) = \frac{1}{1-\gamma}.$$

Adicionalmente, o valor de $\bar{C}^{\pi'}_{\gamma}(s_0)$ é:

$$\bar{C}^{\pi'}_{\gamma}(s_0) = L + \gamma(P\bar{C}^{\pi'}_{\gamma}(s_g) + (1-P)\bar{C}^{\pi'}_{\gamma}(s_d)),$$
$$\bar{C}^{\pi'}_{\gamma}(s_0) = L + \gamma(1-P)\bar{C}^{\pi'}_{\gamma}(s_d),$$
$$\bar{C}^{\pi'}_{\gamma}(s_0) = L + (1-P)\frac{\gamma}{1-\gamma}.$$

Como $L > \frac{1}{1-\gamma}$:

$$\bar{C}^{\pi'}_{\gamma}(s_0) = L + (1-P)\frac{\gamma}{1-\gamma},$$
$$\bar{C}^{\pi'}_{\gamma}(s_0) > \frac{1}{1-\gamma} + (1-P)\frac{\gamma}{1-\gamma},$$
$$\bar{C}^{\pi'}_{\gamma}(s_0) > \frac{1}{1-\gamma},$$
$$\bar{C}^{\pi'}_{\gamma}(s_0) > \bar{C}^{\pi}_{\gamma}(s_0).$$

Logo, $\bar{C}^{\pi}_{\gamma}(s_0) < \bar{C}^{\pi'}_{\gamma}(s_0)$ e $\pi \succ \pi'$, mas $\frac{P^{\pi}_{G}(s_0)}{P^{\pi'}_{G}(s_0)} = 0$, o que completa a prova. □

4.3 Priorização de Metas sobre Históricos

Na seção passada a propriedade de priorização de probabilidade à meta α-forte foi definida. Essa propriedade considera características relacionadas a distribuições de políticas e, consequentemente, de históricos. A seguir, são definidas propriedades desejáveis sobre históricos.

A preferência de um tomador de decisões que prioriza metas sobre históricos pode ser definida como segue:

Definição 4 (Priorização de metas sobre históricos). [1] *Um tomador de decisões prioriza metas se ele apresenta a seguinte preferência:*

$$(C_T, 1) \succ (C'_T, 0), \forall C_T, C'_T \in \mathbb{R}_{>0} \cup \infty, \tag{4.1}$$

$$(C_T, 1) \succ (C'_T, 1), \text{ se } C_T < C'_T \; \forall C_T, C'_T \in \mathbb{R}_{>0}, \tag{4.2}$$

$$(C_T, 0) \sim (C'_T, 0), \forall C_T, C'_T \in \mathbb{R}_{>0} \cup \infty. \tag{4.3}$$

Essa definição indica que: (1) o tomador de decisões prefere um histórico que alcance a meta sobre históricos que não o fazem, independentemente do custo final; (2) se ambos históricos alcançam a meta, o tomador de decisões prefere o histórico com custo menor; e (3) se nenhum dos dois históricos alcançam a meta, então a preferência entre esses históricos é igual.

O próximo capítulo mostra como, se essa definição de priorização de metas sobre históricos for combinada com Teoria da Utilidade Esperada, um critério de decisão com priorização de probabilidade à meta α-forte para um valor arbitrário $0 \leq \alpha \leq 1$ pode ser obtido.

[1] Essa é uma versão modificada da Definição 1 em Freire e Delgado, 2017.

Capítulo 5

Critério GUBS (*Goals with Utility-Based Semantics*)

Apesar de que todos critérios apresentados até o momento tem formulações bem-definidas para resolver SSP-MDPs com *dead ends*, todos eles apresentam alguns problemas. Agentes que seguem os critérios lexicográficos S^3P e MCMP não realizam compromissos infinito-infinitesimais (Teorema 1). Os critérios fSSPUDE e de custo descontado fornecem alternativas para esse problema, mas como demonstrado nos teoremas 3 e 4, eles não garantem a propriedade de priorização de probabilidade à meta α-forte para $\alpha > 0$. Esses critérios apenas garantem a condição $\pi \succeq \pi' \implies \frac{P^{\pi}_{G}(s_0)}{P^{\pi'}_{G}(s_0)} \geq \alpha$ da Definição 3 para um SSP-MDP $\mathcal{M}$ específico, dado um valor de D ou γ.

O critério *Goals with Utility-Based Semantics* (Metas com Semântica Baseada em Utilidade – GUBS) (Freire e Delgado, 2017) foi proposto como um critério baseado em Teoria da Utilidade Esperada. Como é demonstrado ao longo do presente capítulo, esse critério garante priorização de metas como descrito na Definição 4, e a priorização de probabilidade à meta α-forte sob certas condições, para $0 \leq \alpha \leq 1$.

GUBS avalia um histórico baseado no seu custo acumulado e na condição de um estado meta ter sido alcançado ou não durante esse histórico. No critério GUBS, então, são atribuídos pesos para essas duas propriedades do histórico baseados, respectivamente, em uma função de utilidade u sobre custos acumulados C_T, e uma utilidade constante de alcançar metas K_g.

Definição 5 (Critério GUBS (Freire e Delgado, 2017)). *O critério GUBS avalia um histórico com a seguinte função de utilidade U:*

$$U(C_T, \beta_T) = u(C_T) + K_g \beta_T. \tag{5.1}$$

Um agente segue o critério GUBS se ele avalia uma política π sob a seguinte função valor:

$$V^{\pi}_{GUBS}(s) = \lim_{T \to \infty} \mathbb{E}[U(C_T, \beta_T) | \pi, s_0 = s]. \tag{5.2}$$

Uma política π^* é ótima sob o critério GUBS se ela maximiza a função a função V^{π}_{GUBS}, i.e.:

$$V^{\pi^*}_{GUBS}(s) \geq V^{\pi}_{GUBS}(s) \quad \forall \pi \in \Pi \text{ e } \forall s \in \mathcal{S}.$$

Note que uma maximização é utilizada nesse caso porque no critério GUBS valores de utilidade são considerados ao invés de custos, como é o caso do critério convencional para SSP-MDPs definido na Seção 2.2.

Para maior clareza, se o argumento s é omitido ao longo da presente seção quando as funções P^{π}_{G} e V^{π}_{GUBS} são referenciadas, significa que os valores de probabilidade à meta e utilidade esperada sendo referidos são os do estado inicial s_0. Ou seja, nesses casos, $P^{\pi}_{G} = P^{\pi}_{G}(s_0)$ e $V^{\pi}_{GUBS} = V^{\pi}_{GUBS}(s_0)$.

5.1 Priorização de Metas sobre Históricos e Compromisso Mínimo no Critério GUBS

O seguinte teorema fornece o resultado de que um agente seguindo o critério GUBS prioriza metas sobre históricos como descrito na Definição 4 se a função u e a constante K_g são escolhidas apropriadamente.

Teorema 5 (Condições do critério GUBS [1]). *Um agente que segue o critério GUBS prioriza metas sobre históricos como na Definição 4, se:*

1. $u : \mathbb{R} \cup \{\infty\} \rightarrow [U_{min}, U_{max}]$;
2. $u(C)$ *é estritamente decrescente em* C;
3. $K_g > U_{max} - U_{min}$;
4. $c(s, a) > 0 \; \forall s \notin \mathcal{G}$ *e* $\forall a \in \mathcal{A}$.

Demonstração. A condição $(C_T, 1) \succ (C'_T, 0)$ significa o mesmo que $U(C_T, 1) > U(C'_T, 0)$. Então, sob as condições do teorema, segue que:

$$\begin{aligned} U(C_T, 1) &= u(C_T) + K_g \\ &> U_{min} + U_{max} - U_{min} \\ &= U_{max} \\ &\geq U(C'_T, 0). \end{aligned}$$

A condição $(C_T, 1) \succ (C'_T, 1)$ quando $C_T < C'_T$ tem o mesmo significado que $U(C_T, 1) > U(C'_T, 1)$. Sob as condições do teorema, $u(C_T) > u(C'_T)$ é verdade. Então:

$$U(C_T, 1) = u(C_T) + K_g > u(C'_T) + K_g = U(C'_T, 1).$$

Finalmente, a condição $(C_T, 0) \sim (C'_T, 0)$ é verdadeira se $U(C_T, 0) = U(C'_T, 0)$. Isso leva ao

[1] Esse teorema é uma versão modificada do Teorema 6 definido em Freire e Delgado, 2017.

seguinte:

$$\begin{aligned} U(C_T, 0) &= U(C'_T, 0), \\ u(C_T) + 0 \times K_g &= u(C'_T) + 0 \times K_g, \\ u(C_T) &= u(C'_T). \end{aligned}$$

Então, $(C_T, 0) \sim (C'_T, 0)$ se $u(C_T) = u(C'_T)$. Já que C_T e C'_T são custos de históricos que não alcançam a meta, ambos são custos infinitos, porque $c(s, a) > 0 \; \forall s \notin \mathcal{G}$ e $\forall a \in \mathcal{A}$. Então, sob essas condições, $u(C_T) = u(C'_T) = U_{min}$. □

No critério GUBS, uma política π' tem preferência maior ou igual sobre outra política π ($\pi' \succeq \pi$) mesmo que $P_G^{\pi'} < P_G^{\pi}$, desde que o custo para alcançar a meta seguindo π' compense para a perda em probabilidade à meta. O seguinte teorema demonstra as condições necessárias para que $\pi' \succeq \pi$ seja verdade.

Teorema 6 (Compromisso mínimo no critério GUBS [2]). *Sejam π e π' duas políticas, tal que $P_G^{\pi} > P_G^{\pi'}$, então $\pi' \succeq \pi$ é garantido pelo critério GUBS somente se:*

$$\frac{P_G^{\pi'}}{P_G^{\pi}} \geq \frac{K_g}{U_{max} - U_{min} + K_g}. \tag{5.3}$$

Demonstração. Da Equação 5.2:

$$\begin{aligned} V_{GUBS}^{\pi} &= \lim_{T \to \infty} \mathbb{E}[u(C_T) + K_g \beta_T | \pi] \\ &= \lim_{T \to \infty} \mathbb{E}[u(C_T) | \pi] + K_g P_G^{\pi} \\ &\geq U_{min} + K_g P_G^{\pi}, \end{aligned}$$

e

$$\begin{aligned} V_{GUBS}^{\pi'} &= \lim_{T \to \infty} \mathbb{E}[u(C_T) + K_g \beta_T | \pi'] \\ &= \lim_{T \to \infty} \mathbb{E}[u(C_T) | \pi'] + K_g P_G^{\pi'} \\ &\leq P_G^{\pi'} U_{max} + (1 - P_G^{\pi'}) U_{min} + K_g P_G^{\pi'}, \end{aligned}$$

em que a última equação foi obtida ao considerar o melhor cenário restrito por P_G: quando um estado meta é alcançado, a melhor utilidade ($K_g + U_{max}$) é obtida e, se um estado meta não for alcançado, um custo infinito deve ser pago, o que resulta na pior utilidade (U_{min}).

[2] Esse teorema é uma pequena modificação do Teorema 5 definido em Freire e Delgado, 2017.

Pelo fato da condição $\pi' \succeq \pi$ ser o mesmo que $V^{\pi'}_{GUBS} \geq V^{\pi}_{GUBS}$, pode-se concluir que:

$$\begin{aligned} P^{\pi'}_G U_{max} + (1 - P^{\pi'}_G)U_{min} + K_g P^{\pi'}_G &\geq U_{min} + K_g P^{\pi}_G, \\ P^{\pi'}_G (U_{max} - U_{min} + K_g) &\geq K_g P^{\pi}_G, \\ \frac{P^{\pi'}_G}{P^{\pi}_G} &\geq \frac{K_g}{U_{max} - U_{min} + K_g}. \end{aligned}$$

□

O seguinte corolário indica que, sob a condição de que $K_g \geq U_{max} - U_{min}$, se $P^{\pi}_G = 1$, então $P^{\pi'}_G \geq \frac{1}{2}$ é a condição necessária para que $\pi' \succeq \pi$ seja verdade, mas o custo originado por π' precisa adicionalmente ser menor que o originado por π para compensar a diferença entre P^{π}_G e $P^{\pi'}_G$.

Corolário 1 (Compromisso mínimo no critério GUBS quando existe uma política própria [3]). *Considere um SSP-MDP e duas políticas π e π' tal que $P^{\pi}_G = 1$, então, sob o critério GUBS e considerando $K_g \geq U_{max} - U_{min}$, $\pi' \succeq \pi$ somente se $P^{\pi'}_G \geq \frac{1}{2}$.*

Demonstração. A partir do Teorema 6, segue que $\pi' \succeq \pi$ se:

$$\frac{P^{\pi'}_G}{P^{\pi}_G} \geq \frac{K_g}{U_{max} - U_{min} + K_g}.$$

Substituindo o valor de $P^{\pi}_G = 1$ e considerando $K_g \geq U_{max} - U_{min}$:

$$\begin{aligned} \frac{P^{\pi'}_G}{1} &\geq \frac{K_g}{K_g + K_g} = \frac{K_g}{2K_g}, \\ P^{\pi'}_G &\geq \frac{1}{2}, \end{aligned}$$

completando a prova. □

O próximo corolário, adicionalmente, mostra que, quando $K_g > 0$, uma política que tem probabilidade maior que 0 de chegar na meta sempre sera preferível sob o GUBS a outra política que tem probabilidade 0 de chegar na meta.

Corolário 2 (Preferência do critério GUBS considerando políticas com probabilidade à meta igual a 0). *Em um SSP-MDP sob o critério GUBS com $K_g > 0$ e duas políticas π e π' tal que $P^{\pi}_G > 0$, $P^{\pi'}_G = 0$, então $\pi \succ \pi'$.*

Demonstração. Pelo Teorema 6, segue que $\pi' \succeq \pi$ somente se:

$$\frac{P^{\pi'}_G}{P^{\pi}_G} \geq \frac{K_g}{U_{max} - U_{min} + K_g}.$$

[3] Esse corolário é uma pequena modificação do Corolário 1 definido em Freire e Delgado, 2017.

Substituindo o valor de $P_G^{\pi'} = 0$:

$$0 \geq \frac{K_g}{U_{max} - U_{min} + K_g}$$

assumindo $K_g > 0$ isso não pode ser verdade, já que $0 < \frac{K_g}{U_{max} - U_{min} + K_g}$. □

5.2 Critério GUBS e Priorização de Probabilidade à Meta α-forte

O critério GUBS tem priorização de probabilidade à meta α-forte para um $0 \leq \alpha \leq 1$ arbitrário se U_{max}, U_{min} e a constante K_g forem escolhidos apropriadamente. Isso é demonstrado formalmente pelo próximo corolário.

Corolário 3 (Priorização de probabilidade à meta α-forte para o critério GUBS). *Para um $0 \leq \alpha \leq 1$ arbitrário, o critério GUBS possui priorização de probabilidade à meta α-forte.*

Para $0 \leq \alpha < 1$, essa propriedade é mantida se a seguinte condição é verdadeira:

$$K_g \geq \frac{(U_{max} - U_{min})\alpha}{1 - \alpha}.$$

Para $\alpha = 1$, a propriedade é mantida quando $U_{max} = U_{min}$ e $K_g > 0$.

Demonstração. Para $0 \leq \alpha < 1$, considerando $K_g \geq \frac{(U_{max} - U_{min})\alpha}{1-\alpha}$ segue que:

$$K_g - K_g\alpha \geq (U_{max} - U_{min})\alpha,$$

$$K_g \geq (U_{max} - U_{min} + K_g)\alpha,$$

$$\frac{K_g}{U_{max} - U_{min} + K_g} \geq \alpha.$$

Pelo Teorema 6, a seguinte inequação é verdadeira:

$$\pi' \succeq \pi \implies \frac{P_G^{\pi'}}{P_G^{\pi}} \geq \frac{K_g}{U_{max} - U_{min} + K_g} \geq \alpha.$$

Então, o seguinte também é verdade:

$$\pi' \succeq \pi \implies \frac{P_G^{\pi'}}{P_G^{\pi}} \geq \alpha,$$

o que segue a Definição 3.

Para $\alpha = 1$, quando $U_{max} = U_{min}$ e $K_g > 0$, segue também que:

$$\pi' \succeq \pi \implies \frac{P_G^{\pi'}}{P_G^{\pi}} \geq \frac{K_g}{K_g} = 1.$$

□

Note que na última parte do Corolário 3, se $U_{max} = U_{min}$, então a segunda condição do Teorema 5 não é satisfeita, porque nesse caso u não é estritamente decrescente. Por essa razão, a propriedade de priorização sobre históricos não é mantida nesse caso.

A Tabela 5.1 resume como a priorização de probabilidade à meta α-forte é garantida para cada critério analisado neste trabalho (segunda coluna), assim como se cada um deles permite a realização de compromissos infinito-infinitesimais (terceira coluna). Note que o GUBS é o único critério que garante priorização de probabilidade à meta α-forte para $0 \leq \alpha \leq 1$ e permite a realização de compromissos infinito-infinitesimais.

	Priorização de probabilidade à meta α-forte	Falta de compromissos infinito-infinitesimais
MAXPROB (Kolobov, D. S. Weld *et al.*, 2011)	1-forte (Teorema 2)	Sim (Teorema 1)
Lexicográficos (Kolobov, D. Weld *et al.*, 2012; Teichteil-Königsbuch, 2012; F. W. Trevizan *et al.*, 2017)	1-forte (Teorema 2)	Sim (Teorema 1)
fSSPUDE (Kolobov, D. Weld *et al.*, 2012)	apenas-0-forte (Teorema 3)	Não
Custo descontado (Teichteil-Königsbuch *et al.*, 2011)	apenas-0-forte (Teorema 4)	Não
GUBS (Freire e Delgado, 2017)	α-forte, $0 \leq \alpha \leq 1$ (Corolário 3)	Não

Tabela 5.1: *Tipos de priorização de probabilidade à meta α-forte para cada critério e se eles podem realizar compromissos infinito-infinitesimais*

5.3 Exemplo Ilustrativo

Considere o SSP-MDP da Figura 4.1 e seja π a política que sempre toma a ação a, e π', a política que toma a ação b. Considere também os valores $P_G^{\pi} - P_G^{\pi'} = \delta < \Delta P = 10^{-5}$ e $L - \ell > \Delta C = 10^5$. Para essas condições serem mantidas, considere $\delta = 10^{-6}$ (tal que $\delta < \Delta P = 10^{-5}$), $L = 10^6 + 1$, e $\ell = 1$ (tal que $L - \ell = 10^6 > \Delta C = 10^5$).

O valor sob o critério GUBS para a política π em s_0 é então igual a:

$$V^{\pi}_{GUBS}(s_0) = P(u(c(s_0, a)) + K_g) + (1 - P)(u(\infty)),$$
$$V^{\pi}_{GUBS}(s_0) = P(u(10^6 + 1) + K_g) + (1 - P)(u(\infty)),$$

e o valor para a política π' é:

$$V^{\pi'}_{GUBS}(s_0) = (P - \delta)(u(c(s_0, b)) + K_g) + [1 - (P - \delta)]\,(u(\infty)),$$
$$V^{\pi'}_{GUBS}(s_0) = (P - 10^{-6})(u(1) + K_g) + \left[1 - (P - 10^{-6})\right](u(\infty)).$$

A política ótima π em s_0, $\pi^*_{s_0}$, é:

$$\pi^*_{s_0} = \arg\max_{\{\pi,\pi'\}}\{V^{\pi}_{GUBS}(s_0), V^{\pi'}_{GUBS}(s_0)\}.$$

O critério GUBS permite a escolha de qualquer função de utilidade sobre custo u. Para esse exemplo considere a função de utilidade exponencial modificada $u(C_T)$, definida como:

$$u(C_T) = \begin{cases} 0, \text{se } C_T = \infty \\ e^{-0.4C_T}, \text{ senão,} \end{cases}$$

Note que a função u satisaz a primeira e segunda condições do Teorema 5, em que $U_{min} = 0$ e $U_{max} = 1$ (quando C_T é respectivamente ∞ e 0). Além disso, para garantir que o critério GUBS priorize metas sobre históricos, o valor de K_g precisa ser escolhido tal que $K_g > U_{max} - U_{min} = 1$.

Utilizando essa função de utilidade, π^* é dada por:

$$\begin{aligned}\pi^* &= \arg\max\{P(u(10^6 + 1) + K_g) + (1 - P)u(\infty), \\ &\qquad (P - 10^{-6})(u(1) + K_g) + \left[1 - (P - 10^{-6})\right] u(\infty)\} \\ &= \arg\max\{P(e^{-0.4\times(10^6+1)} + K_g) + (1 - P) \times 0, \\ &\qquad (P - 10^{-6})(e^{-0.4} + K_g) + \left[1 - (P - 10^{-6})\right] \times 0\} \\ &= \arg\max\{P(e^{-0.4\times(10^6+1)} + K_g), (P - 10^{-6})(e^{-0.4} + K_g)\}.\end{aligned}$$

Escolhendo $P = 1$ como exemplo, nesse caso π seria a política ótima sob o critério GUBS, ou seja, $V^{\pi}_{GUBS}(s_0) > V^{\pi'}_{GUBS}(s_0)$, quando:

$$K_g > 6.70321 \times 10^5. \tag{5.4}$$

Da mesma maneira, para π' ser a política ótima, $V^{\pi}_{GUBS}(s_0)$ precisa ser menor que $V^{\pi'}_{GUBS}(s_0)$. Nesse caso, e considerando a terceira condição do Teorema 5, a seguinte condição precisa ser verdadeira:

$$1 < K_g < 6.70321 \times 10^5. \tag{5.5}$$

Enquanto a política ótima para o GUBS pode ser π ou π' dependendo do valor da constante K_g, a política ótima para os critérios MAXPROB, S^{3}P, e MCMP sempre será π. Como demonstrado no Teorema 1, isso acontece porque $P_G^{\pi} > P_G^{\pi'}$ mesmo que a diferença entre esses dois valores seja bem pequena, decorrendo em uma grande diferença em custo.

Considere novamente o SSP-MDP da Figura 4.1, e π'' como a política que toma a ação c em s_0. Note que π e π' tem valores de probabilidade à meta positivos e π'' não tem chance de alcançar a meta. O critério GUBS, priorizando metas sobre históricos com $K_g > 0$ e $U_{max} - U_{min} > 0$, nunca escolhe π'' como a política ótima (Corolário 2). Por outro lado, para os critérios fSSPUDE e de custo descontado (que possuem priorização de probabilidade à meta apenas-0-forte), o mesmo não pode ser garantido, como demonstrado nas provas dos Teoremas 3 e 4. Para o fSSPUDE, considerando qualquer valor de D tal que $L \geq D$, ambas políticas π e π'' teriam a mesma preferência, por exemplo. Para o critério de custo descontado, para γ satisfazendo a inequação $L \geq \frac{1}{1-\gamma}$ (o que é equivalente a $\gamma < \frac{L-1}{L}$), π'' seria preferível sobre π. Por exemplo, considerando $L = 1000$ e um valor de γ tal que $\gamma < 999/1000 = 0.999$, a preferência do agente é $\pi'' \succ \pi$.

Pode-se também ilustrar a relação entre esse exemplo e a condição da Definição 3. Como mencionado anteriormente, nesse problema existem três políticas possíveis a partir do estado inicial: π, π' e π'', com os respectivos valores de probabilidade à meta $P_G^{\pi}(s_0) = P$, $P_G^{\pi'}(s_0) = P - 10^{-6}$ e $P_G^{\pi''}(s_0) = 0$. Ao considerar o valor da razão na Definição 3 para todos pares de políticas e $P = 1$, pode-se obter o seguinte ao combinar π, π' e π'':

$$\begin{aligned}
\frac{P_G^{\pi}(s_0)}{P_G^{\pi'}(s_0)} &= \frac{1}{1-10^{-6}} = 1.000001,\\
\frac{P_G^{\pi}(s_0)}{P_G^{\pi}(s_0)} &= 1,\\
\frac{P_G^{\pi'}(s_0)}{P_G^{\pi}(s_0)} &= \frac{1-10^{-6}}{1} = 0.999999,\\
\frac{P_G^{\pi'}(s_0)}{P_G^{\pi'}(s_0)} &= 1,\\
\frac{P_G^{\pi''}(s_0)}{P_G^{\pi}(s_0)} &= \frac{0}{1} = 0,\\
\frac{P_G^{\pi''}(s_0)}{P_G^{\pi'}(s_0)} &= \frac{0}{1-10^{-6}} = 0.
\end{aligned}$$

A partir disso, se pode concluir que as condições $\pi \succeq \pi''' \implies \frac{P_G^{\pi}(s_0)}{P_G^{\pi'''}(s_0)} \geq 1$, $\pi' \succeq \pi''' \implies \frac{P_G^{\pi'}(s_0)}{P_G^{\pi'''}(s_0)} \geq 0.999999$, e $\pi'' \succeq \pi''' \implies \frac{P_G^{\pi''}(s_0)}{P_G^{\pi'''}(s_0)} = 0$ são verdadeiras, para toda política π''' aplicável a partir de s_0 tal que $P_G^{\pi'''}(s_0) > 0$. Pelo fato de já ter sido demonstrado que, ao considerar $P = 1$: (i) para $\pi \succ \pi'''$ ser verdade, $K_g > 6.70321 \times 10^5$ precisa ser verdadeiro; (ii) para $\pi' \succ \pi'''$ ser verdade, $1 < K_g < 6.70321 \times 10^5$ precisa ser verdadeiro; e (iii) para $\pi'' \succ \pi'''$ também ser verdade, $K_g \leq 1$ precisa ser verdadeiro. Então, se pode concluir que

as seguintes expressões são verdadeiras:

$$K_g > 6.70321 \times 10^5 \implies \pi \succeq \pi''' \implies \frac{P_G^{\pi}(s_0)}{P_G^{\pi'''}(s_0)} \geq 1 = \alpha$$

$$1 < K_g < 6.70321 \times 10^5 \implies \pi' \succeq \pi''' \implies \frac{P_G^{\pi'}(s_0)}{P_G^{\pi'''}(s_0)} \geq 0.999999 = \alpha$$

$$K_g \leq 1 \implies \pi'' \succeq \pi''' \implies \frac{P_G^{\pi''}(s_0)}{P_G^{\pi'''}(s_0)} \geq 0 = \alpha$$

$\forall \pi'''$aplicável de s_0 tal que $P_G^{\pi'''}(s_0) > 0$.

Já que se está considerando apenas um único SSP-MDP (o problema da Figura 4.1), isso não significa que o critério GUBS é 1-forte no primeiro caso, nem 0.999999-forte no outro.

5.4 Algoritmo Aproximado para Resolver o Critério GUBS

Para propor novos algoritmos para resolver o critério GUBS, uma reformulação de custo discreto do problema original foi proposta em Freire e Delgado, 2017. Essa reformulação considera um conjunto de estados aumentados, que são estados estendidos com o custo acumulado C gasto até o momento. O custo imediato é redefinido ao considerar a utilidade do custo do problema original e a utilidade do custo acumulado C. Finalmente, K_g é associada com a recompensa terminal para estados meta.

Um algoritmo aproximado baseado em iteração de valor para encontrar soluções para o critério GUBS utilizando essa reformulação de custos discretos é também proposto em Freire e Delgado, 2017. O algoritmo itera a partir de um valor de custo acumulado máximo dado como parâmetro (C_{max}) até 0, como no algoritmo de iteração de valor para horizonte finito, computando os valores de qualidade de cada estado aumentado em cada iteração.

Apesar do GUBS permitir um compromisso adequado entre custo e probabilidade à meta, encontrar uma política ótima para esse critério possui três problemas principais. Primeiro, uma política ótima é não markoviana porque ela é definida sobre um espaço de estados aumentado com o custo acumulado. Segundo, assim como em MDPs de horizonte finito, a política ótima precisa ser obtida de trás para frente. Terceiro, muita memória pode ser necessária para guardar a política ótima considerando um espaço de estados aumentados.

Capítulo 6

Critério eGUBS

O critério GUBS permite uma escolha de qualquer função de utilidade sobre custos u e constante K_g tal que um agente utilizando esses parâmetros prioriza metas, se eles satisfazem as condições no Teorema 5. No presente capítulo, o *GUBS Exponencial* (*Exponential GUBS* - eGUBS) (Freire, Delgado e Reis, 2019), um caso particular do GUBS em que a função de utilidade $u(\cdot)$ é exponencial e satisfaz a primeira e segunda condições do Teorema 5, é apresentado. A escolha dessa função é baseada em SSP-MDPs sensíveis ao risco (Patek, 2001).

6.1 SSP-MDPs Sensíveis ao Risco e Função de Utilidade Exponencial Utilizada pelo eGUBS

Um SSP-MDP sensível ao risco (Patek, 2001) é definido pela tupla $\mathcal{RSSSP} = \langle \mathcal{M}, \lambda \rangle$ em que $\mathcal{M}$ é um SSP-MDP e λ é o fator de risco que modela a atitude de risco do agente. Se $\lambda < 0$, o agente considera uma atitude propensa ao risco; se $\lambda > 0$, o agente considera uma atitude aversa ao risco; e no limite, se $\lambda \to 0$, o agente considera uma atitude neutra ao risco. A função de utilidade utilizada em SSP-MDPs sensíveis ao risco é definida como o seguinte:

$$u(x) = -\operatorname{sgn}(\lambda) e^{\lambda x},$$

em que $\operatorname{sgn}(\lambda)$ é a função que retorna o sinal de λ, $x = \sum_{t=0}^{M-1} c(s_t, \pi(s_t))$, tal que $M \in \mathbb{N}$ é o passo de tempo em que a meta é alcançada, e $M = \infty$ se um estado meta não é alcançado. Para o critério eGUBS, o valor $\lambda < 0$ é considerado. Quando é esse o caso, a função valor da política π em SSP-MDPs sensíveis ao risco é bem-definida e dada pela seguinte equação:

$$V_\lambda^\pi(s) = \lim_{T \to \infty} \mathbb{E}\left[e^{\lambda C_T^\pi} \mid s_0 = s \right], \tag{6.1}$$

em que $C_T^\pi = \sum_{t=0}^{T-1} c(s_t, \pi(s_t))$ e o valor da política π pode ser computado ao resolver o seguinte sistema de equações:

$$V_\lambda^\pi(s) = \begin{cases} 1, & \text{se } s \in \mathcal{G} \\ e^{\lambda c(s,\pi(s))} \sum_{s' \in S} P(s, \pi(s), s') V_\lambda^\pi(s'), & \text{caso contrário} \end{cases} \tag{6.2}$$

A seguir um critério lexicográfico para SSP-MDPs sensíveis ao risco é introduzido. Esse critério é utilizado para descrever algumas propriedades do critério eGUBS.

Definição 6 (Critério lexicográfico sensível ao risco[1]). *O critério lexicográfico sensível ao risco considera o seguinte para todo $s \in S$:*

(i) *se $P_G^\pi(s) > P_G^{\pi'}(s)$, então $\pi \succ \pi'$; ou*

(ii) *se $P_G^\pi(s) = P_G^{\pi'}(s)$ e $V_\lambda^\pi > V_\lambda^{\pi'}$, então $\pi \succ \pi'$.*

Políticas π_L^* estacionárias ótimas sob o critério lexicográfico sensível ao risco satisfazem o seguinte:

- π_L^* maximiza probabilidade à meta (logo, $\pi_L^* \in \Pi^{MP}$). Em outras palavras, $P_G^{\pi_L^*}(s) \geq P_G^\pi(s), \forall s \in S$ e $\forall \pi \in \Pi$, em que Π é o conjunto de todas as políticas estacionárias;
- π_L^* maximiza a utilidade exponencial V_λ^π. Isso significa que $V_\lambda^{\pi_L^*}(s) \geq V_\lambda^\pi(s), \forall s \in S$ e $\forall \pi \in \Pi^{MP}$.

6.2 Definições e Propriedades do Critério eGUBS

Como já mencionado, para um agente seguindo o critério GUBS priorizar metas (Teorema 5), a função de utilidade u precisa ter duas propriedades: (i) ela precisa ser uma função na forma $u : \mathbb{R} \cup \{\infty\} \to [U_{min}, U_{max}]$; e (ii) a função $u(C_T)$ precisa ser estritamente decrescente em C_T. Por essas razões, a seguinte função exponencial modificada é definida como a função de utilidade para o critério eGUBS.

Definição 7 (Critério eGUBS [2]). *O critério eGUBS considera o critério GUBS em que a função de utilidade u é definida como a seguinte:*

$$u(C_T) = \begin{cases} 0, \textit{se } C_T = \infty \\ e^{\lambda C_T}, \textit{ senão,} \end{cases}$$

sobre algum custo acumulado C_T e um fator de risco $\lambda < 0$.

Como $\lambda < 0$ na função $u(C_T)$, as condições 1 e 2 do Teorema 5 são verdadeiras:

- Quando $C_T = 0$, $u(0) = e^0 = 1$. Além disso, quando $C_T = \infty$, $u(C_T) = 0$. Logo, $u(C_T) : \mathbb{R} \cup \{\infty\} \to [0, 1]$ (condição 1);

[1] Essa definição é uma pequena modificação da Definição 5 introduzida em Freire, Delgado e Reis, 2019.

[2] Essa definição é uma modificação da Definição 2 introduzida em Freire, Delgado e Reis, 2019.

- $u(C_T)$ é estritamente decrescente em C_T (condição 2).

Como já explicado, λ modela a atitude de risco do agente em um cenário de utilidade exponencial. Já que o eGUBS considera $\lambda < 0$, a partir dessa função utilidade, agentes seguindo esse critério serão propensos ao risco considerando a métrica de custo acumulado. Ademais, quando $\lambda \to 0$, o critério aproxima neutralidade ao risco considerando essa métrica. No entanto, o termo K_g da definição do GUBS (Definição 5) atribui um peso para probabilidade à meta. Logo, ambos parâmetros do eGUBS (λ e K_g) podem modificar a atitude do agente para possuir tanto uma atitude mais propensa ao risco relacionada ao custo acumulado ou uma maior preferência por alcançar metas.

Propriedades teóricas anteriores podem ter resultados mais específicos se utilizada a função de utilidade exponencial descrita na Definição 7.

Corolário 4 (Priorização de probabilidade à meta α-forte para o critério eGUBS). *Para um valor arbitrário $0 \leq \alpha < 1$, o critério eGUBS tem priorização de probabilidade à meta α-forte se a seguinte condição é verdadeira:*

$$K_g \geq \frac{\alpha}{1-\alpha}$$

Demonstração. A imagem da função de utilidade exponencial utilizada pelo eGUBS é o intervalo $[0, 1]$. Em outras palavras, $U_{min} = 0$ e $U_{max} = 1$. Então, seguindo do Corolário 3:

$$K_g \geq \frac{(U_{max} - U_{min})\alpha}{1-\alpha},$$
$$K_g \geq \frac{\alpha}{1-\alpha}.$$

□

Como demonstrado no Capítulo 5, para resolver o critério GUBS, o custo acumulado precisa ser levado em conta para computar valores de utilidade. Por essa razão, é necessário definir uma função valor para o eGUBS que depende não apenas em um estado s, mas em um custo acumulado C. Essa função será chamada de função valor estado-custo:

Definição 8 (Função valor estado-custo (Freire, Delgado e Reis, 2019)). *Considere que o agente já pagou um custo acumulado C, está no estado s, e então segue a política π a partir desse ponto. A função valor estado-custo é definida por:*

$$V^{\pi}_{GUBS}(s, C) = \lim_{T \to \infty} \mathbb{E}[U(C + C_T, \beta_T) | \pi, s_0 = s], \tag{6.3}$$

descrevendo a utilidade esperada obtida pelo agente nesse cenário.

O próximo teorema mostra como a função valor de uma política estacionária avaliada sob o critério GUBS pode ser expressada em termos de $V^{\pi}_{\lambda}(s)$ (a função valor da política π para SSP-MDPs sensíveis ao risco) e $P^{\pi}_{G}(s)$ (a função de probabilidade à meta da política π).

Teorema 7 (Valor estado-custo de políticas estacionárias sob o critério eGUBS (Freire, Delgado e Reis, 2019)). *Considere uma política estacionária π e o critério*

eGUBS. A função valor estado-custo de π é:

$$V^{\pi}_{GUBS}(s, C) = e^{\lambda C} V^{\pi}_{\lambda}(s) + K_g P^{\pi}_{G}(s). \tag{6.4}$$

em que $V^{\pi}_{\lambda}(s)$ e $P^{\pi}_{G}(s)$ são definidos como nas equações 6.2 e 3.1, respectivamente.

Demonstração.

$$\begin{aligned} V^{\pi}_{GUBS}(s, C) &= \lim_{T\to\infty} \mathbb{E}[U(C + C_T, \beta_T)|\pi, s_0 = s] \\ &= \lim_{T\to\infty} \mathbb{E}[u(C + C^{\pi}_T) + K_g \beta^{\pi}_T | s_0 = s] \\ &= \lim_{T\to\infty} \mathbb{E}[e^{\lambda(C+C^{\pi}_T)} + K_g \beta^{\pi}_T | s_0 = s] \\ &= e^{\lambda C} V^{\pi}_{\lambda}(s) + K_g P^{\pi}_{G}(s) \end{aligned}$$

□

Note novamente que soluções para SSP-MDPs sob o critério GUBS são políticas não markovianas. O Teorema 8 mostra, então, como uma política estacionária pode ser melhorada ao aumentar o espaço de estados com o custo acumulado, o que gera, por fim, uma política não markoviana.

Teorema 8 (Melhoria de política sob o critério eGUBS[3]). *Seja π uma política estacionária, π pode ser melhorada em uma política não markoviana π', $\forall s \in \mathcal{S}$ ao utilizar:*

$$\pi'(s, C) = \arg\max_{a\in\mathcal{A}} \left\{ \sum_{s'\in\mathcal{S}} P(s, a, s') \left(e^{\lambda(C+c(s,a))} V^{\pi}_{\lambda}(s') + K_g P^{\pi}_{G}(s') \right) \right\}.$$

Demonstração. É necessário provar que, para todo $s \in \mathcal{S}$, $V^{\pi}_{GUBS}(s, C) \leq V^{\pi'}_{GUBS}(s, C)$. Para

[3] Esse teorema é baseado na Definição 4 introduzida em Freire, Delgado e Reis, 2019

$s \in \mathcal{G}$, isso é trivial. Para todos estados $s \in \mathcal{S} \setminus \mathcal{G}$:

$$
\begin{aligned}
V^{\pi}_{GUBS}(s, C) &= \sum_{s' \in \mathcal{S}} P(s, \pi(s), s') V^{\pi}_{GUBS}(s', C + c(s, \pi(s))) \\
&\leq \max_{a \in \mathcal{A}} \left\{ \sum_{s' \in \mathcal{S}} P(s, a, s') V^{\pi}_{GUBS}(s', C + c(s, a)) \right\} \\
&= \max_{a \in \mathcal{A}} \left\{ \sum_{s' \in \mathcal{S}} P(s, a, s') \left(e^{\lambda(C + c(s,a))} V^{\pi}_{\lambda}(s') + K_g P^{\pi}_G(s') \right) \right\} \\
&= \max_{a \in \mathcal{A}} \left\{ \sum_{s' \in \mathcal{S} \setminus \mathcal{G}} P(s, a, s') \sum_{s'' \in \mathcal{S}} \left(P(s', \pi(s'), s'') \right.\right. \\
&\qquad \left. \left(e^{\lambda(C + c(s,a) + c(s', \pi(s')))} V^{\pi}_{\lambda}(s'') + K_g P^{\pi}_G(s'') \right) \right) \\
&\qquad \left. + \sum_{s' \in \mathcal{G}} P(s, a, s') \left(e^{\lambda(C + c(s,a))} + K_g \right) \right\} \\
&= \left\{ \sum_{s' \in \mathcal{S} \setminus \mathcal{G}} P(s, \pi'(s, C), s') \sum_{s'' \in \mathcal{S}} \left(P(s', \pi(s'), s'') \right.\right. \\
&\qquad \left. \left(e^{\lambda(C + c(s, \pi'(s,C)) + c(s', \pi(s')))} V^{\pi}_{\lambda}(s'') + K_g P^{\pi}_G(s'') \right) \right) \\
&\qquad \left. + \sum_{s' \in \mathcal{G}} P(s, \pi'(s, C), s') \left(e^{\lambda(C + c(s, \pi'(s,C)))} + K_g \right) \right\} \\
&\leq \max_{a' \in \mathcal{A}} \left\{ \sum_{s' \in \mathcal{S} \setminus \mathcal{G}} P(s, \pi'(s, C), s') \sum_{s'' \in \mathcal{S}} \left(P(s', a', s'') \right.\right. \\
&\qquad \left. \left(e^{\lambda(C + c(s, \pi'(s,C)) + c(s', a'))} V^{\pi}_{\lambda}(s'') + K_g P^{\pi}_G(s'') \right) \right) \\
&\qquad \left. + \sum_{s' \in \mathcal{G}} P(s, \pi'(s, C), s') \left(e^{\lambda(C + c(s, \pi'(s,C)))} + K_g \right) \right\} \\
&= \Pr(s_1 \notin \mathcal{G} \mid s_0 = s, \pi') \\
&\qquad \mathbb{E}\left[e^{\lambda(C + c(s_0, a_0) + c(s_1, a_1))} V^{\pi}_{\lambda}(s_2) + K_g P^{\pi}_G(s_2) \middle| s_0 = s, s_1 \notin \mathcal{G}, \pi' \right] \\
&\quad + \Pr(s_1 \in \mathcal{G} \mid s_0 = s, \pi') \mathbb{E}\left[\left(e^{\lambda(C + c(s_0, a_0))} + K_g \right) \middle| s_0 = s, s_1 \in \mathcal{G}, \pi' \right] \\
&\leq \lim_{T \to \infty} \Pr(s_{T-1} \notin \mathcal{G} \mid s_0 = s, \pi') \\
&\qquad \mathbb{E}\left[e^{\lambda(C + \sum_{t=0}^{T-1} c(s_t, a_t))} V^{\pi}_{\lambda}(s_T) + K_g P^{\pi}_G(s_T) \middle| s_0 = s, s_{T-1} \notin \mathcal{G}, \pi' \right] \\
&\quad + \sum_{t=1}^{T-1} \Pr(s_t \in \mathcal{G} \mid s_0 = s, \pi') \mathbb{E}\left[\left(e^{\lambda(C + \sum_{t=0}^{T-2} c(s_t, a_t))} + K_g \right) \middle| s_0 = s, s_{T-1} \in \mathcal{G}, \pi' \right] \\
&= V^{\pi'}_{GUBS}(s, C).
\end{aligned}
$$

□

6.3 Como Encontrar Políticas Ótimas para o Critério eGUBS

A principal propriedade do critério eGUBS é que um custo máximo pode ser obtido tal que quando o agente alcança esse custo acumulado, a política ótima a partir desse ponto é estacionária. As próximas duas definições descrevem essa propriedade formalmente. Antes disso, seguem definições adicionais de notações relacionadas a SSP-MDPs, que serão utilizadas a partir da presente seção.

Seja $\mathcal{S}_{succ}(s) = \{s' \mid \exists a \in \mathcal{A} \text{ tal que } P(s,a,s') > 0, \forall s' \in \mathcal{S}\}$ o conjunto de sucessores imediatos de s; $\mathcal{S}_{reach}(s) = \{s' \mid \exists \pi \in \Pi \text{ tal que } s' \text{ é alcançável de } s \text{ quando } \pi \text{ é seguida }\}$ o conjunto de todos estados s' alcançáveis de s; e $\mathcal{A}_s(s') = \{a \in \mathcal{A} \mid P(s,a,s') > 0\}, \forall (s,s') \in \mathcal{S} \times \mathcal{S}$ o conjunto de ações que levam o agente de s a s'.

Definição 9 ($C_{stat}(s)$ **e políticas estacionárias ótimas sob o critério eGUBS**). *No critério eGUBS, existe um valor mínimo $C_{stat}(s)$ tal que se o custo acumulado alcança $C_{stat}(s)$, a política ótima a partir desse ponto é estacionária. Ou seja, dado que o processo está no estado s e o custo acumulado alcança o valor $C_{stat}(s)$, uma política estacionária π^*_L pode ser seguida a partir desse ponto. Essa política estacionária maximiza o critério lexicográfico sensível ao risco. Em outras palavras, para qualquer valor $C' \geq C_{stat}(s)$, o valor da política ótima π^* sob o critério eGUBS é $V^{\pi^*}_{GUBS}(s', C') = V^{\pi^*_L}_{\lambda}(s')$, $\forall s' \in \{s\} \cup \mathcal{S}_{reach}(s)$, o que não é verdade para qualquer custo acumulado C'' tal que $C'' < C_{stat}(s)$.*

Definição 10 (Custo-admissibilidade de s). *Para um estado inicial $s \in \mathcal{S}$, um custo acumulado C é Custo-admissível de s se $C \geq C_{stat}(s)$.*

O próximo teorema mostra como um valor Custo-admissível pode ser computado considerando todos estados do SSP-MDP. Para isso, seja a função $\mathcal{W}(s,a)$ e o conjunto $\mathcal{X}$ definidos como:

$$\mathcal{W}(s,a) = -\frac{1}{\lambda} \log \frac{V^{\pi^*_L}_{\lambda}(s) - e^{\lambda c(s,a)} \sum_{s' \in \mathcal{S}} P(s,a,s') V^{\pi^*_L}_{\lambda}(s')}{K_g \left(\sum_{s' \in \mathcal{S}} P(s,a,s') P^{\pi^*_L}_{G}(s') - P^{\pi^*_L}_{G}(s) \right)}, \tag{6.5}$$

e

$$\mathcal{X} = \left\{ (s \in \mathcal{S}, a \in \mathcal{A}) \middle| \left(V^{\pi^*_L}_{\lambda}(s) - e^{\lambda c(s,a)} \sum_{s' \in \mathcal{S}} P(s,a,s') V^{\pi^*_L}_{\lambda}(s') \right) < 0 \right\}, \tag{6.6}$$

em que $\mathcal{X}$ é o conjunto de pares estado-ação (s,a) tal que a melhora a utilidade exponencial esperada em s, comparada com a utilidade da política ótima lexicográfica sensível ao risco.

Teorema 9 (C_{max} **é Custo-admissível de qualquer** $s \in \mathcal{S}$ [4]). *No critério eGUBS, o valor*

[4] Esse teorema é uma pequena modificação do Teorema 4 definido em Freire, Delgado e Reis, 2019.

de C_{max} é Custo-admissível de qualquer $s \in \mathcal{S}$, e é definido pela seguinte equação:

$$C_{max} = \max_{(s,a)\in\mathcal{X}} [\mathcal{W}(s,a)]\,. \tag{6.7}$$

Demonstração. A política π_L^* é ótima se não pode ser melhorada por *look-ahead*, ou seja, não existe qualquer estado s cujo valor pode ser melhorado ao tomar uma ação $a \neq \pi_L^*(s)$. Já que π_L^* otimiza a probabilidade à meta, apenas pares $(s,a) \in \mathcal{X}$ que melhoram a utilidade exponencial esperada são candidatos para melhorar a política π_L^*. Por isso, para a política π_L^* ser ótima, a seguinte restrição precisa ser levada em conta para cada par $(s,a) \in \mathcal{X}$:

$$e^{\lambda C}V_\lambda^\pi(s) + K_g P_G^\pi(s) \geq \sum_{s'\in\mathcal{S}} P(s,a,s')\left(e^{\lambda(C+c(s,a))}V_\lambda^\pi(s') + K_g P_G^\pi(s')\right),$$

$$e^{\lambda C}\left(V_\lambda^\pi(s) - \sum_{s'\in\mathcal{S}} P(s,a,s')e^{\lambda c(s,a)}V_\lambda^\pi(s')\right) \geq \sum_{s'\in\mathcal{S}} P(s,a,s')K_g P_G^\pi(s') - K_g P_G^\pi(s),$$

$$\frac{V_\lambda^\pi(s) - \sum_{s'\in\mathcal{S}} P(s,a,s')e^{\lambda c(s,a)}V_\lambda^\pi(s')}{\sum_{s'\in\mathcal{S}} P(s,a,s')K_g P_G^\pi(s') - K_g P_G^\pi(s)} \leq e^{-\lambda C}.$$

Enquanto a primeira passagem é feita de manipulações matemáticas simples, na segunda a inequação $\sum_{s'\in\mathcal{S}} P(s,a,s')K_g P_G^\pi(s') - K_g P_G^\pi(s) < 0$ é levada em conta. Aplicando log em ambos os lados da inequação:

$$\log\frac{V_\lambda^\pi(s) - \sum_{s'\in\mathcal{S}} P(s,a,s')e^{\lambda c(s,a)}V_\lambda^\pi(s')}{\sum_{s'\in\mathcal{S}} P(s,a,s')K_g P_G^\pi(s') - K_g P_G^\pi(s)} \leq -\lambda C,$$

$$C \geq -\frac{1}{\lambda}\log\frac{V_\lambda^\pi(s) - \sum_{s'\in S} P(s,a,s')e^{\lambda c(s,a)}V_\lambda^\pi(s')}{K_g\left(\sum_{s'\in S} P(s,a,s')P_G^\pi(s') - P_G^{\pi^*}(s)\right)}.$$

A partir dessa inequação, pode ser concluído que para cada custo C que segue a sua restrição, a política ótima no estado s, considerando a ação a e esse custo, é estacionária. Se um valor C_{max} for computado ao maximizar esse valor para todo par $(s,a) \in \mathcal{X}$, esse valor é então Custo-admissível a partir de qualquer $s \in \mathcal{S}$. □

Além disso, note que o valor resultante da Equação 6.5 pode ser negativo, o que no caso de custos positivos significa que a partir do custo acumulado 0 a política ótima para o estado específico já é estacionária. Isso também significa que o valor de $C_{stat}(s)$ pode ser negativo.

O resultado do Teorema 9 pode ser melhorado ao considerar apenas um estado s. O Teorema 10, então, mostra como encontrar um valor Custo-admissível a partir de um estado s sem a necessidade de realizar uma maximização sobre todos os estados. Para esse teorema, considere a função $W(s)$ definida como o seguinte:

$$W(s) = \begin{cases} \max_{a \in \mathcal{A}_C(s)}[\mathcal{W}(s,a)], \text{ se } \mathcal{A}_C(s) \neq \emptyset \\ -\infty, \text{ caso contrário,} \end{cases} \tag{6.8}$$

em que $\mathcal{A}_C(s)$ é o conjunto de ações que com s formam um par estado-ação que pertence a $\mathcal{X}$ (como definido pela Equação 6.6):

$$\mathcal{A}_C(s) = \{a \in \mathcal{A} \mid (s,a) \in \mathcal{X}\}. \tag{6.9}$$

Em outras palavras, ações $a \in \mathcal{A}_C(s)$ são ações que melhoram a utilidade exponencial a partir de s quando comparada com a utilidade de uma política estacionária ótima sob o critério lexicográfico sensível ao risco.

$W(s)$ é o custo máximo em (somente) s a partir do qual uma política ótima sob o critério eGUBS é garantidamente estacionária e lexicográfica, se $\mathcal{A}_C(s) \neq \emptyset$. Se $\mathcal{A}_C(s) = \emptyset$, então $W(s) = -\infty$, o que significa que a política ótima a partir de s é a política ótima estacionária, a partir de qualquer custo acumulado.

Ademais, o custo para o qual uma política estacionária ótima sob o critério lexicográfico sensível ao risco é ótima sob o critério eGUBS para estados s' alcançáveis a partir de s ($s' \in \mathcal{S}_{reach}(s)$) precisa ser computado, porque pode existir um estado s' alcançável de s tal que $W(s') > W(s)$. Isso significa que não se pode garantir que uma política ótima em s sob o critério eGUBS é estacionária para um custo $W(s)$, porque um agente pode ainda chegar a s' a partir de s com custo acumulado $W(s) \leq C < W(s')$. Logo, se $C_{min}(s,s')$ é o menor custo possível de um histórico que pode ser gerado de s para s', então $W(s') - C_{min}(s,s')$ pode ser levado em conta, ao contrário de apenas considerar $W(s')$. $C_{min}(s,s')$ pode então ser definido como o seguinte:

$$C_{min}(s,s') = \min_{h_T \in \mathcal{H}_{s,s'}} \left[\sum_{t=0}^{T-1} c_t \right], \tag{6.10}$$

em que $\mathcal{H}_{s,s'}$ é o conjunto de históricos $h_T = \{\langle s_0, a_0, c_0\rangle, \langle s_1, a_1, c_1\rangle, \ldots, s_T\}$, tal que $s_0 = s$ e $s_T = s'$.

Baseado nessas ideias, o Teorema 10 mostra como computar um valor Custo-admissível de um estado s, e o Corolário 5 mostra como esse valor é um limite superior mais restrito sobre $C_{stat}(s)$ que C_{max}.

Teorema 10 ($\bar{C}_{max}(s)$ **é Custo-admissível a partir de** s)**.** *No critério eGUBS, o valor de* $\bar{C}_{max}(s)$*, computado pela seguinte equação, é Custo-admissível a partir de* s*.*

$$\bar{C}_{max}(s) = \max\{W(s), \max_{s' \in \mathcal{S}_{reach}(s)} [W(s') - C_{min}(s,s')]\}. \tag{6.11}$$

Demonstração. $C_{min}(s,s')$ é o menor custo possível que precisa ser pago ao seguir um histórico de s até s'. Se o agente está no estado s e precisa pagar pelo menos $C_{min}(s,s')$ para chegar em s', então quando o custo acumulado $W(s') - C_{min}(s,s')$ é alcançado, a política certamente será estacionária a partir desse ponto se essa diferença é maior ou igual a $W(s)$.

Logo, se $W(s') - C_{min}(s, s') \geq W(s)$, $W(s') - C_{min}(s, s')$ é Custo-admissível de s.

Caso contrário, se $W(s) \geq W(s') - C_{min}(s, s')$, $\bar{C}_{max}(s) = W(s)$. Já que $W(s)$ é Custo-admissível de s por definição, então $\bar{C}_{max}(s)$ é Custo-admissível de s. □

Corolário 5 ($\bar{C}_{max}(s)$ **é um limite superior mais restrito sobre** $C_{stat}(s)$ **que** C_{max})**.** *Seja $\mathcal{M}$ um SSP-MDP com um dado valor C_{max} como definido no Teorema 9 e $\bar{C}_{max}(\cdot)$ como no Teorema 10, então:*

$$\bar{C}_{max}(s) \leq C_{max}, \forall s \in \mathcal{S}.$$

Demonstração. Seja $S_w = \{s_w \mid s_w = \arg\max_{s' \in \mathcal{S}} W(s')\}$. Primeiro, será demonstrado que $\bar{C}_{max}(s_w) = C_{max}, \forall s_w \in \mathcal{S}_w$, e então demonstrado que $\bar{C}_{max}(s_w) \leq C_{max}, \forall s_w \notin \mathcal{S}_w$.

A partir da Equação 6.8, $W(s) = \max_{a \in \mathcal{A}_C(s)} \mathcal{W}(s, a)$ quando $\mathcal{A}_C(s) \neq \emptyset$. Para qualquer $s_w \in S_w$, pela Equação 6.11, $\bar{C}_{max}(s_w) = W(s_w)$, já que $\mathcal{A}_C(s_w) \neq \emptyset$ e $C_{min}(s_w, s') \geq 0, \forall s' \in \mathcal{S}_{reach}(s_w)$. Uma vez que $C_{max} = \max_{(s,a) \in \mathcal{X}} \mathcal{W}(s, a) = \max_{s \in \mathcal{S}}[\max_{a \in \mathcal{A}_C(s)} \mathcal{W}(s, a)] = W(s_w)$ a partir da Equação 6.7, $\bar{C}_{max}(s_w) = C_{max} = W(s_w)$.

Para $s \notin \mathcal{S}_w$, $\bar{C}_{max}(s) \leq C_{max}$, porque nesse caso $\bar{C}_{max}(s) = \max_{s' \in \mathcal{S}_{reach}(s)}[W(s') - C_{min}(s, s')]$ pela Equação 6.11 e $W(s') \leq \max_{(s',a) \in \mathcal{X}} \mathcal{W}(s', a) = C_{max}$ pela Equação 6.8 já que $C_{min}(s, s') \geq 0$.

Então, $\bar{C}_{max}(s) \leq C_{max}, \forall s \in \mathcal{S}$. □

O seguinte teorema mostra como $\bar{C}_{max}(s)$ pode ser computado recursivamente:

Teorema 11 (Definição recursiva de $\bar{C}_{max}(s)$**).**

$$\bar{C}_{max}(s) = \max\{W(s), \max_{s' \in \mathcal{S}_{succ}(s), a \in \mathcal{A}_s(s')}[\bar{C}_{max}(s') - c(s, a)]\}. \tag{6.12}$$

Demonstração. Pela Equação 6.11, o valor de $\bar{C}_{max}(s)$ é dado por:

$$\bar{C}_{max}(s) = \max\{W(s), \max_{s' \in \mathcal{S}_{reach}(s)}[W(s') - C_{min}(s, s')]\}.$$

A diferença máxima $W(s') - C_{min}(s, s')$ para os estados s' alcançáveis de s pode ser definida como o valor máximo entre o máximo dessa mesma diferença para os sucessores de s e o máximo dela para estados alcançáveis de s exceto os sucessores de s.

$$\begin{aligned}\bar{C}_{max}(s) = \max\{&W(s),\\ &\max\{\\ &\quad\max_{s' \in \mathcal{S}_{succ}(s)}[W(s') - C_{min}(s, s')],\\ &\quad\max_{s' \in (\mathcal{S}_{reach}(s) \setminus \mathcal{S}_{succ}(s))}[W(s') - C_{min}(s, s')]\}\}.\end{aligned}$$

Ao considerar apenas os estados sucessores s' de s, $C_{min}(s, s')$ pode ser reescrito como o custo mínimo $c(s, a)$ para qualquer ação a que pode levar o agente de s a s':

$$\begin{aligned}\bar{C}_{max}(s) = \max\{W(s), & \\ \max\{ & \\ & \max_{s' \in S_{succ}(s)} [W(s') - \min_{a \in \mathcal{A}_s(s')} c(s,a)], \\ & \max_{s' \in (\boldsymbol{S_{reach}(s) \setminus S_{succ}(s)})} [\boldsymbol{W(s') - C_{min}(s,s')}]\} \\ \}.\end{aligned} \tag{6.13}$$

O segundo termo da maximização interior pode ser rearranjado como o máximo, para todo estado s' sucessor de s, da diferença máxima entre $W(s'')$ e $C_{min}(s,s'')$

$$\begin{aligned}\bar{C}_{max}(s) = \max\{W(s), & \\ \max\{ & \\ & \max_{s' \in S_{succ}(s)} [W(s') - \min_{a \in \mathcal{A}_s(s')} c(s,a)], \\ & \max_{s' \in \boldsymbol{S_{succ}(s)}} \{ \max_{s'' \in \boldsymbol{S_{reach}(s')}} [\boldsymbol{W(s'') - C_{min}(s,s'')}]\}\} \\ \}.\end{aligned} \tag{6.14}$$

Note que, mesmo o termo rearranjado (o termo em negrito na Equação 6.14) não sendo exatamente o mesmo que o original (o termo em negrito na Equação 6.13), a igualdade geral é válida.

Seja $\xi(s)$ definido como (o mesmo que o termo em negrito na Equação 6.14):

$$\xi(s) = \max_{s' \in S_{succ}(s)} \{ \max_{s'' \in S_{reach}(s')} [W(s'') - C_{min}(s,s'')]\}.$$

O custo mínimo entre s e s'' pode ser dividido entre a soma do mínimo custo imediato entre s e o seu sucessor s' e o custo mínimo entre s' e s'':

$$\begin{aligned}\xi(s) &= \max_{s' \in S_{succ}(s)} \{ \max_{s'' \in S_{reach}(s')} \{W(s'') - [\min_{a \in \mathcal{A}_s(s')} c(s,a) + C_{min}(s',s'')]\}\} \\ &= \max_{s' \in S_{succ}(s)} \{ \max_{s'' \in S_{reach}(s')} [W(s'') - C_{min}(s',s'')] - \min_{a \in \mathcal{A}_s(s')} c(s,a)\}.\end{aligned}$$

Então, substituindo $\xi(s)$ de volta na Equação 6.14, os itens sendo maximizados quando considerando o mesmo conjunto $S_{succ}(s)$ podem ser agrupados em um único termo de

maximização:

$$\begin{aligned}
\bar{C}_{max}(s) = \max\{W(s), & \\
& \max_{s' \in \mathcal{S}_{succ}(s)} [W(s') - \min_{a \in \mathcal{A}_s(s')} c(s,a)], \\
& \max_{s' \in \mathcal{S}_{succ}(s)} \{ \max_{s'' \in \mathcal{S}_{reach}(s')} [W(s'') - C_{min}(s', s'')] - \min_{a \in \mathcal{A}_s(s')} c(s,a) \} \\
\} & \\
\bar{C}_{max}(s) = \max\{W(s), & \\
& \max_{s' \in \mathcal{S}_{succ}(s)} \{ \\
& \quad W(s') - \min_{a \in \mathcal{A}_s(s')} c(s,a), \\
& \quad \max_{s'' \in \mathcal{S}_{reach}(s')} [W(s'') - C_{min}(s', s'')] - \min_{a \in \mathcal{A}_s(s')} c(s,a) \\
& \} \\
\}. &
\end{aligned}$$

Como $\min_{a \in \mathcal{A}_s(s')} c(s,a)$ é um termo comum em ambos os dois termos sendo maximizados, ele pode ser retirado para fora da maximização mais interna:

$$\begin{aligned}
\bar{C}_{max}(s) = \max\{W(s), & \\
& \max_{s' \in \mathcal{S}_{succ}(s)} \{ \\
& \quad \max\{W(s'), \max_{s'' \in \mathcal{S}_{reach}(s')} [W(s'') - C_{min}(s', s'')]\} \\
& \quad - \min_{a \in \mathcal{A}_s(s')} c(s,a) \\
& \} \\
\}. &
\end{aligned}$$

A maximização mais interna pode então ser substituída por $\bar{C}_{max}(s')$:

$$\bar{C}_{max}(s) = \max\{W(s), \max_{s' \in \mathcal{S}_{succ}(s)} \{\bar{C}_{max}(s') - \min_{a \in \mathcal{A}_s(s')} [c(s,a)]\}\}.$$

Finalmente, os termos podem ser rearranjados tal que ambos $s' \in \mathcal{S}_{succ}(s)$ e $a \in \mathcal{A}_s(s')$ são os termos que maximizam a diferença $\bar{C}_{max}(s') - c(s,a)$:

$$\bar{C}_{max}(s) = \max\{W(s), \max_{s' \in \mathcal{S}_{succ}(s), a \in \mathcal{A}_s(s')} [\bar{C}_{max}(s') - c(s,a)]\}.$$

□

O seguinte teorema mostra como o valor de $C_{stat}(s)$ pode ser definido com uma equação. Para computar $\bar{C}_{max}(s)$ como no Teorema 10, juntamente com os valores de $W(s)$ e $W(s'), \forall s' \in \mathcal{S}_{reach}(s)$, é necessário computar os valores de $C_{min}(s, s')$, os quais são os custos mínimos de históricos que partem de s até s' ao seguir *qualquer* política. Para definir $C_{stat}(s)$, no entanto, apenas as ações ótimas precisam ser consideradas.

Para isso, primeiro considere $\mathcal{H}^{\pi^*_{GUBS}}_{s,s'}$ como o conjunto de históricos h_T que podem ser gerados ao seguir a política π^*_{GUBS} de $s_0 = s$ para $s_T = s'$:

$$\begin{aligned} h_T = \{&\langle s_0, \pi^*_{GUBS}(s_0, 0), c_0\rangle, \\ &\langle s_1, \pi^*_{GUBS}(s_1, c_0), c_1\rangle, \\ &\cdots, \\ &\langle s_{T-1}, \pi^*_{GUBS}(s_{T-1}, c_0 + c_1 + \cdots + c_{T-2}), c_{T-1}\rangle, s_T\}, \end{aligned}$$

Em seguida, seja $C^{\pi^*_{GUBS}}_{min}(s, s') = \min_{h_T \in \mathcal{H}^{\pi^*_{GUBS}}_{s,s'}} \left[\sum_{t=0}^{T-1} c_t\right]$ o custo mínimo de um histórico que vai de s até s' a partir de uma política ótima sobre o eGUBS.

Ademais, ao invés de considerar os valores de $W(s)$, os quais levam em conta todas as ações em $\mathcal{A}_C(s)$, pode-se limitar o espaço de interesse ao apenas considerar ações ótimas. Seja $W^*(s)$ definido como o seguinte:

$$W^*(s) = \begin{cases} \max\limits_{C \in C_W(s)} [\mathcal{W}(s, \pi^*_{GUBS}(s, C))], \text{ se } C_W(s) \neq \emptyset, \\ -\infty, \text{ caso contrário,} \end{cases}$$

em que $C_W(s) = \{C \mid C \geq 0 \text{ e } \pi^*_{GUBS}(s, C) \in \mathcal{A}_C(s)\}$ é o conjunto de custos C que formam um estado aumentado com s, para o qual $\pi^*_{GUBS}(s, C)$ pertence a $\mathcal{A}_C(s)$.

Teorema 12 (Valor de $C_{stat}(s)$). *O valor de $C_{stat}(s)$ pode ser definido pela seguinte equação:*

$$C_{stat}(s) = \max\{W^*(s), \max_{s' \in \mathcal{S}_{reach}(s)} [W^*(s') - C^{\pi^*_{GUBS}}_{min}(s, s')]\}. \tag{6.15}$$

Demonstração. Para definir $C_{stat}(s)$, que é o mínimo custo possível a partir do qual a política ótima sob o critério eGUBS é garantidamente estacionária, podem-se considerar históricos que são gerados apenas ao seguir a política ótima não markoviana sob o critério eGUBS (π^*_{GUBS}) a partir de s até s', ao invés de considerar qualquer histórico possível.

Esses históricos são então representados pelo conjunto $\mathcal{H}^{\pi^*_{GUBS}}_{s,s'}$, e o menor custo possível obtido ao seguir qualquer um desses históricos, por $C^{\pi^*_{GUBS}}_{min}(s, s')$. Para então definir $C_{stat}(s)$ pode-se utilizar a mesma equação definida no Teorema 10 para $\bar{C}_{max}(s)$ (Equação 6.11), mas substituindo $C_{min}(s, s')$ por $C^{\pi^*_{GUBS}}_{min}(s, s')$, e $W(s)$ por $W^*(s)$, já que ações ótimas precisam ser consideradas e $W^*(s)$ maximiza sobre essas ações quando se consideram estados aumentados (s, C), tal que $C \in C_W(s)$, porque a política π^*_{GUBS} é não markoviana. Isso leva à Equação 6.15, e completa a prova. □

6.4 Exemplo Ilustrativo

Considere o SSP-MDP da Figura 6.1. Esse exemplo é similar ao da Figura 4.1, mas com dois estados adicionais: um novo estado inicial s_0 e um *dead end* s_{d_2}. O estado s_0 da Figura 4.1 é renomeado para s_1 na Figura 6.1. Tomar a ação a em s_0 leva a s_1 com custo associado 2ℓ. Ações b e c em s_0 tem custo $L/2$ e levam a s_g com probabilidade $P/2$ e para s_{d_2} com probabilidade $1 - (P/2)$.

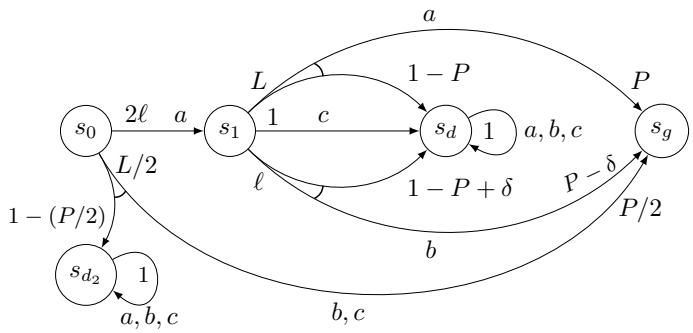


Figura 6.1: *Exemplo de SSP-MDP modificado a partir do exemplo da Figura 4.1*

Além disso, considere valores de $P = 0.8, \delta = 0.1, L = 20, \ell = 1, \lambda = -0.1$, e $K_g = 0.1$.

6.4.1 Calculando C_{max}

Primeiramente, os valores de P_G^{π} para os *dead ends* s_d e s_{d_2} são 0 por definição, para toda política $\pi \in \Pi$. Os valores de $V_{\lambda}^{\pi_L^*}$ para esses estados são:

$$V_{\lambda}^{\pi_L^*}(s_d) = V_{\lambda}^{\pi_L^*}(s_{d_2}) = e^{-0.1\infty} = 0,$$

já que ao alcançar um *dead end* o agente obterá um custo acumulado infinito. Para o estado meta, $V_{\lambda}^{\pi_L^*}(s_g) = e^0 = 1$ e $P_G(s_g) = 1$.

Sejam políticas π_a, π_b e π_c as políticas que tomam as ações a, b, e c respectivamente em s_1 e tomam ações ótimas nos outros estados. Os valores de $V_{\lambda}^{\pi}(s_1)$ para essas 3 políticas são dados como:

$$\begin{aligned}
V_{\lambda}^{\pi_a}(s_1) &= e^{-0.1\times 20}(0.8V_{\lambda}^{\pi_L^*}(s_g) + 0.2V_{\lambda}^{\pi_L^*}(s_d)) = 0.8e^{-0.1\times 20} = 0.108 \\
V_{\lambda}^{\pi_b}(s_1) &= e^{-0.1}\{(0.8-0.1)V_{\lambda}^{\pi_L^*}(s_d) + [1-(0.8-0.1)]V_{\lambda}^{\pi_L^*}(s_d)\} \\
&= (0.8-0.1)e^{-0.1} = 0.633 \\
V_{\lambda}^{\pi_c}(s_1) &= e^{-0.1}V_{\lambda}^{\pi_L^*}(s_d) = 0.
\end{aligned}$$

De maneira similar, os valores de $P_G^{\pi}(s_1)$ para essas mesmas políticas são:

$$\begin{aligned}
P_G^{\pi_a}(s_1) &= P \times P_G^{\pi_a}(s_g) = P = 0.8 \\
P_G^{\pi_b}(s_1) &= (P-\delta)P_G^{\pi_b}(s_g) + [1-(P-\delta)]P_G^{\pi_b}(s_d) \\
&= 0.8 - 0.1 = 0.7 \\
P_G^{\pi_c}(s_1) &= P_G^{\pi_c}(s_d) = 0.
\end{aligned}$$

Então, como π_a é a única política que maximiza probabilidade à meta em s_1, $V_{\lambda}^{\pi_L^*}(s_1) = V_{\lambda}^{\pi_a}(s_1) = 0.108$.

Sejam π'_a, π'_b e π'_c as políticas que tomam as ações a, b, e c respectivamente em s_0 e tomam ações ótimas nos outros estados. Os valores de $V_{\lambda}^{\pi}(s_0)$ para essas 3 políticas são

dados como:

$$\begin{aligned} V_\lambda^{\pi_a'}(s_0) &= e^{-0.2} V_\lambda^{\pi_L^*}(s_1) = 0.088 \\ V_\lambda^{\pi_b'}(s_0) = V_\lambda^{\pi_c'}(s_0) &= e^{-0.1\times 10}(0.4 V_\lambda^{\pi_L^*}(s_g) + 0.6 V_\lambda^{\pi_L^*}(s_{d_2})) \\ &= 0.4 e^{-0.1\times 10} = 0.147. \end{aligned}$$

Os valores de $P_G^{\pi'}(s_0)$ para essas mesmas políticas são:

$$\begin{aligned} P_G^{\pi_a'}(s_0) &= P_G^{\pi_a'}(s_0) = 0.8 \\ P_G^{\pi_b'}(s_0) = P_G^{\pi_c'}(s_0) &= (P/2) P_G(s_g) + (1 - P/2) P_G(s_{d_2}) = P/2 = 0.4. \end{aligned}$$

π_a' é a política que maximiza probabilidade à meta em s_0, então $V_\lambda^{\pi_L^*}(s_0) = V_\lambda^{\pi_a'}(s_0) = 0.088$.

Para calcular os valores de $\mathcal{W}$, os pares estado-ação que pertencem ao conjunto $\mathcal{X}$ (Equação 6.6) precisam ser encontrados. Para um par estado-ação $(s, a) \in \mathcal{S} \times \mathcal{A}$ ser parte de $\mathcal{X}$, $(V_\lambda^{\pi_L^*}(s) - \sum_{s' \in \mathcal{S}} P(s, a, s') e^{\lambda c(s,a)} V_\lambda^{\pi_L^*}(s')) < 0$ precisa ser verdade. Seja $V_{diff}(s, a) = V_\lambda^{\pi_L^*}(s) - \sum_{s' \in \mathcal{S}} P(s, a, s') e^{\lambda c(s,a)} V_\lambda^{\pi_L^*}(s')$. Os valores de V_{diff} para cada par estado-ação no presente exemplo são dados abaixo:

$$\begin{aligned} V_{diff}(s_0, a) &= V_\lambda^{\pi_a}(s_0) - e^{-0.2} V_\lambda^{\pi_L^*}(s_1) = V_\lambda^{\pi_a}(s_0) - V_\lambda^{\pi_a}(s_0) = 0 \\ \boldsymbol{V_{diff}(s_0, b)} &= V_\lambda^{\pi_a}(s_0) - 0.4 e^{-1} V_\lambda^{\pi_L^*}(s_g) = 0.088 - 0.147 = -0.059 < 0 \\ \boldsymbol{V_{diff}(s_0, c)} &= V_{diff}(s_0, b) = -0.059 < 0 \\ V_{diff}(s_1, a) &= V_\lambda^{\pi_a}(s_1) - V_\lambda^{\pi_a}(s_1) = 0 \\ \boldsymbol{V_{diff}(s_1, b)} &= V_\lambda^{\pi_a}(s_1) - 0.7 e^{-0.1} V_\lambda^{\pi_L^*}(s_g) = 0.108 - 0.633 = -0.525 < 0 \\ V_{diff}(s_1, c) &= V_\lambda^{\pi_a}(s_1) - e^{-0.1} V_\lambda^{\pi_L^*}(s_d) = V_\lambda^{\pi_a}(s_1) > 0 \\ V_{diff}(s_d, \cdot) &= V_{diff}(s_{d_2}, \cdot) = 0 \\ V_{diff}(s_g, \cdot) &= 0. \end{aligned}$$

Como destacado nos valores em negrito, os únicos pares estado-ação que pertencem a $\mathcal{X}$ são (s_0, b), (s_0, c), e (s_1, b). Então, os valores de $\mathcal{W}(s_0, b)$, $\mathcal{W}(s_0, c)$ e $\mathcal{W}(s_1, b)$ precisam ser calculados a seguir:

$$\begin{aligned} \mathcal{W}(s_1, b) &= -\frac{1}{-0.1} \log \frac{0.108 - 0.633}{1(-0.1)} \\ &= \frac{1}{0.1} \log \frac{-0.525}{-0.1} \\ \mathcal{W}(s_1, b) &= 16.582, \end{aligned}$$

e:

$$\begin{aligned}\mathcal{W}(s_0, b) = \mathcal{W}(s_0, c) &= -\frac{1}{-0.1}\log\frac{0.088 - 0.147}{1(-0.4)} \\ &= \frac{1}{0.1}\log\frac{-0.059}{-0.4} \\ \mathcal{W}(s_0, b) = \mathcal{W}(s_0, c) &= -19.140.\end{aligned}$$

Finalmente, o valor de C_{max} é dado por:

$$\begin{aligned}C_{max} &= \max_{(s,a)\in\mathcal{X}} \mathcal{W}(s, a) \\ &= \max\{\mathcal{W}(s_0, b), \mathcal{W}(s_1, b), \mathcal{W}(s_0, c)\} \\ &= \max\{-19.140, 16.582\} \\ &= 16.582.\end{aligned}$$

6.4.2 Calculando $\bar{C}_{max}(\cdot)$

Primeiramente, já que b e c, as únicas ações que pertencem a $\mathcal{A}_C(s_0)$, tem os mesmos valores de $\mathcal{W}(s_0, \cdot)$, então $W(s_0) = \mathcal{W}(s_0, b) = \mathcal{W}(s_0, c)$. Para s_1, $\mathcal{A}_C(s_1) = \{b\}$, o que significa que $W(s_1) = \mathcal{W}(s_1, b)$.

Além disso, note que $\bar{C}_{max}(s_d) = \bar{C}_{max}(s_{d_2}) = \bar{C}_{max}(s_g) = -\infty$, porque esses estados são ou *dead ends*, ou a meta. Então, os valores de $\bar{C}_{max}(s_0)$ e $\bar{C}_{max}(s_1)$ podem ser calculados como segue:

$$\begin{aligned}\bar{C}_{max}(s_1) &= \max\{16.582, \max[\bar{C}_{max}(s_d) - 1, \bar{C}_{max}(s_g) - 20, \bar{C}_{max}(s_g) - 1]\} \\ &= \max\{16.582, -\infty\} \\ &= 16.582. \\ \bar{C}_{max}(s_0) &= \max\{-19.140, \\ &\qquad \max[\bar{C}_{max}(s_1) - 2, \bar{C}_{max}(s_{d_2}) - 10, \bar{C}_{max}(s_g) - 10]\} \\ &= \max\{-19.140, \max[14.582, -\infty]\} \\ &= 14.582.\end{aligned}$$

6.4.3 Calculando $C_{stat}(\cdot)$

Para calcular os valores de C_{stat}, é necessário primeiro encontrar a política π^*_{GUBS}. Para computar $\pi^*_{GUBS}(s_0, 0)$, é necessário avaliar as ações a e b (c tem o mesmo valor que b em s_0) a partir do estado aumentado $(s_0, 0)$. Seja $\pi^{aug}_a(s_0, \cdot) = \pi'_a(s_0)$ e $\pi^{aug}_b(s_0, \cdot) = \pi'_b(s_0)$.

$$V_{GUBS}^{\pi_a^{aug}}(s_0, 0) = V_{GUBS}^*(s_1, 2\ell) = V_{GUBS}^*(s_1, 2),$$
$$V_{GUBS}^{\pi_b^{aug}}(s_0, 0) = \frac{P}{2}\left(u\left(\frac{L}{2}\right) + K_g\right) + \left(1 - \frac{P}{2}\right)u(\infty) = 0.4(e^{-1} + 0.1),$$
$$V_{GUBS}^{\pi_b^{aug}}(s_0, 0) = 0.187.$$

Note que $V_{GUBS}^{\pi_a^{aug}}(s_0, 0) = V_{GUBS}^*(s_1, 2\ell)$ porque ao seguir a política π_a^{aug} a partir do estado $(s_0, 0)$, o conjunto de possíveis históricos gerados é o mesmo que quando seguindo a mesma política a partir de $(s_1, 2\ell) = (s_1, 2)$. Isso significa que seguir π_a^{aug} a partir de tanto $(s_0, 0)$ quanto $(s_1, 2)$ leva ao mesmo valor de utilidade esperada.

$\pi_{GUBS}^*(s_0, 0)$ é então dada pelo seguinte:

$$\pi_{GUBS}^*(s_0, 0) = \arg\max_{a,b}\{V_{GUBS}^{\pi_a^{aug}}(s_0, 0), V_{GUBS}^{\pi_b^{aug}}(s_0, 0)\}.$$

Como $V_{GUBS}^{\pi_a^{aug}}(s_0, 0) = V_{GUBS}^*(s_1, 2)$, é necessário computar $V_{GUBS}^*(s_1, 2)$ para então computar $V_{GUBS}^{\pi_a}(s_0, 0)$. Para isso, seja $\pi_a^{aug'}(s_1, \cdot) = \pi_a(s_1)$, $\pi_b^{aug'}(s_1, \cdot) = \pi_b(s_1)$, e $\pi_c^{aug'}(s_1, \cdot) = \pi_c(s_1)$. $V_{GUBS}^{\pi_a^{aug'}}(s_1, 2)$, $V_{GUBS}^{\pi_b^{aug'}}(s_1, 2)$ e $V_{GUBS}^{\pi_c^{aug'}}(s_1, 2)$ são então dados por:

$$V_{GUBS}^{\pi_a^{aug'}}(s_1, 2) = 0.8(e^{-0.1(2+20)} + 0.1) + 0.2 \times 0 = 0.8(e^{-2.2} + 0.1) = 0.169,$$
$$V_{GUBS}^{\pi_b^{aug'}}(s_1, 2) = 0.7(e^{-0.1(2+1)} + 0.1) + 0.3 \times 0 = 0.7(e^{-0.3} + 0.1) = 0.589,$$
$$V_{GUBS}^{\pi_c^{aug'}}(s_1, 2) = u(\infty) = 0.$$

Então, $V_{GUBS}^*(s_1, 2) = \max\{V_{GUBS}^{\pi_a^{aug'}}(s_1, 2), V_{GUBS}^{\pi_b^{aug'}}(s_1, 2), V_{GUBS}^{\pi_c^{aug'}}(s_1, 2)\} = V_{GUBS}^{\pi_b^{aug'}}(s_1, 2) = 0.589$, e $\pi_{GUBS}^*(s_1, 2) = \pi_b^{aug'}(s_1, 2) = b$. Logo, $V_{GUBS}^{\pi_a^{aug}}(s_0, 0) = V_{GUBS}^*(s_1, 2) = 0.589$. Isso significa que $V_{GUBS}^*(s_0, 0) = V_{GUBS}^{\pi_b^{aug'}}(s_1, 2) = 0.589$, e $\pi_{GUBS}^*(s_0, 0) = \pi_a^{aug}(s_0, 0) = a$.

Da seção 6.4.2, é conhecido que $\mathcal{A}_C(s_0) = \{b, c\}$. $\pi_{GUBS}^*(s_0, C) = a, \forall C \geq 0$, já que $\mathcal{W}(s_0, b) = \mathcal{W}(s_0, c) = -19.140 < 0$. Em outras palavras, não existe custo C tal que $\pi_{GUBS}^*(s_0, C) \in \mathcal{A}_C(s_0)$, o que significa que $C_W(s_0) = \emptyset$. Logo, $W^*(s_0) = -\infty$.

Para s_1, $\mathcal{A}_C(s_1) = \{b\}$. $\pi_{GUBS}^*(s_1, C) = b, \forall C$ tal que $C < \mathcal{W}(s_1, b) = 16.582$, e $\pi_{GUBS}^*(s_1, C) = a, \forall C \geq \mathcal{W}(s_1, b) = 16.582$. Então, $C_W(s_1) = \{C \mid C < 16.582\}$, e $W^*(s_1) = \mathcal{W}(s_1, b) = 16.582$.

Para os estados s_d, s_{d_2} e s_g, os valores de W^* são iguais a $-\infty$.

Os valores de $C_{stat}(s_0)$ e $C_{stat}(s_1)$ podem então ser computados como o seguinte:

$$\begin{aligned}
C_{stat}(s_1) &= \max\{W^*(s_1), \max_{s' \in \mathcal{S}_{reach}(s_0)} [W^*(s') - C^{\pi^*_{GUBS}}_{min}(s_1, s')]\}, \\
&= \max\{W^*(s_1), \max[W^*(s_g) - C^{\pi^*_{GUBS}}_{min}(s_1, s_g), W^*(s_d) - C^{\pi^*_{GUBS}}_{min}(s_1, s_d)]\}, \\
&= \max\{16.582, \max[-\infty - \ell, -\infty - \ell]\}, \\
&= \max\{16.582, -\infty\}, \\
C_{stat}(s_1) &= 16.582.
\end{aligned}$$

$$\begin{aligned}
C_{stat}(s_0) &= \max\{W^*(s_0), \max_{s' \in \mathcal{S}_{reach}(s_0)} [W^*(s') - C^{\pi^*_{GUBS}}_{min}(s_0, s')]\}, \\
&= \max\{W^*(s_0), \max[\\
&\quad W^*(s_1) - C^{\pi^*_{GUBS}}_{min}(s_0, s_1), W^*(s_d) - C^{\pi^*_{GUBS}}_{min}(s_0, s_d), \\
&\quad W^*(s_{d_2}) - C^{\pi^*_{GUBS}}_{min}(s_0, s_{d_2}), W^*(s_g) - C^{\pi^*_{GUBS}}_{min}(s_0, s_g) \\
&\quad]\}, \\
&= \max\{-\infty, \max[16.582 - 2, -\infty - 3, -\infty - 3, -\infty - 10]\}, \\
&= \max\{-\infty, 14.582\}, \\
C_{stat}(s_0) &= 14.582.
\end{aligned}$$

Os valores de $C_{stat}(s_d)$, $C_{stat}(s_{d_2})$ e $C_{stat}(s_g)$ são iguais a $-\infty$.

Note que nesse exemplo, $\bar{C}_{max}(s) = C_{stat}(s)$ para todos estados s nesse problema, no entanto isso não é necessariamente verdadeiro para qualquer SSP-MDP.

Capítulo 7

Algoritmos Exatos para Resolver o Critério eGUBS

A Seção 6.2 mostra, entre outros resultados, como uma política estacionária pode ser melhorada em uma política aumentada finita e não markoviana utilizando os resultados do Teorema 8 no critério eGUBS, enquanto a Seção 6.3 mostra como uma política ótima finita pode ser encontrada a partir de custos que são Custo-admissíveis. O algoritmo eGUBS-VI (proposto em Freire, Delgado e Reis, 2019), baseado em iteração de valor, é explicado no presente capítulo. Além disso, o eGUBS-AO*, por sua vez baseado em busca heurística, é proposto neste capítulo. Ambos resolvem SSP-MDPs sob o critério eGUBS de maneira ótima, a partir dos resultados explicados no capítulo anterior. Antes disso, primeiramente um algoritmo para encontrar a política ótima para o critério lexicográfico sensível ao risco, chamado de LexicográficoSensívelAoRiscoIV (também proposto em Freire, Delgado e Reis, 2019), é exposto. Depois, são descritos algoritmos para computar os valores Custo-admissíveis C_{max} e $\bar{C}_{max}(s)$, que utilizam as saídas do algoritmo LexicográficoSensívelAoRiscoIV.

7.1 Encontrando Políticas Ótimas para o Critério Lexicográfico Sensível ao Risco

O critério lexicográfico sensível ao risco foi introduzido no presente texto na Seção 6.1. Na presente seção, será descrito um algoritmo para encontrar políticas ótimas para esse critério. Dado um SSP-MDP, um fator de risco λ e um limite de erro ϵ, o algoritmo LexicográficoSensívelAoRiscoIV (Algoritmo 5) computa uma política π que é ótima sob o critério lexicográfico sensível ao risco, sua função valor V_λ e a função de probabilidade à meta máxima P_G. O algoritmo primeiro inicializa V_λ e P_G (linhas 3 e 4). A cada iteração o algoritmo computa uma nova aproximação P_G para cada estado (exceto estados meta) utilizando os valores de probabilidade à meta dos sucessores da iteração anterior (linha 9). Na linha 10 o algoritmo calcula o conjunto de ações gulosas que maximizam probabilidade à meta para cada estado, $\mathcal{A}_s^{P_G}$. Então, na linha 11 o algoritmo calcula o valor de V_λ considerando apenas esse conjunto. O critério de parada para esse algoritmo depende nos valores

δ_1 e δ_2. δ_1 (calculado na linha 13) é a soma máxima da diferença absoluta dos valores de V_λ e da diferença absoluta dos valores de probabilidade à meta de cada iteração. δ_2 (vide linha 14) é a diferença mínima entre o valor de probabilidade à meta e da probabilidade de tomar outra ação que não é gulosa com relação à maximização da probabilidade à meta (ou seja, ações que não pertencem a $\mathcal{A}_s^{P_G}$). Então, o algoritmo é executado enquanto $\delta_1 \geq \epsilon$ ou $\delta_2 \leq 0$. A segunda condição garante que mesmo que o valor de ϵ não seja apropriado, o resultado do Algoritmo 5 ainda pode ser utilizado para calcular o valor de C_{max}. Finalmente, na linha 17 o algoritmo calcula a política ótima π para o critério lexicográfico sensível ao risco a partir da função valor V_λ.

Algoritmo 5: LexicográficoSensívelAoRiscoIV

Entrada: SSP-MDP $\mathcal{M} = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, \lambda, \epsilon$

Saída: Função valor do critério lexicográfico sensível ao risco, função de probabilidade à meta P_G e política do critério lexicográfico sensível ao risco π

1 **início**

▷ Seja $V_\lambda : \mathcal{S} \to \mathbb{R}$, $P_G : \mathcal{S} \to [0,1]$, e $\pi : \mathcal{S} \to \mathcal{A}$

▷ Inicialização

2 $\delta_1 \leftarrow \infty, \delta_2 \leftarrow 0$

3 $V_\lambda(s) \leftarrow 0 \ \forall s \in \mathcal{S} \setminus \mathcal{G}, V_\lambda(s) \leftarrow 1 \ \forall s \in \mathcal{G}$

4 $P_G(s) \leftarrow 0 \ \forall s \in \mathcal{S} \setminus \mathcal{G}, P_G(s) \leftarrow 1 \ \forall s \in \mathcal{G}$

5 **enquanto** $\delta_1 \geq \epsilon$ ou $\delta_2 \leq 0$ **faça**

6 $V' \leftarrow V_\lambda$

7 $P'_G \leftarrow P_G$

8 **para cada** $s \in \mathcal{S} \setminus \mathcal{G}$ **faça**

9 $P_G(s) \leftarrow \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} P(s,a,s') P'_G(s')$

10 $\mathcal{A}_s^{P_G} \leftarrow \left\{ a \mid P_G(s) = \sum_{s' \in \mathcal{S}} P(s,a,s') P'_G(s') \right\}$

11 $V_\lambda(s) \leftarrow \max_{a \in \mathcal{A}_s^{P_G}} \left\{ e^{\lambda c(s,a)} \sum_{s' \in \mathcal{S}} P(s,a,s') V'(s') \right\}$

12 **fim**

13 $\delta_1 \leftarrow \max_{s \in \mathcal{S}} \left\{ |V_\lambda(s) - V'(s)| + |P_G(s) - P'_G(s)| \right\}$

14 $\delta_2 \leftarrow \min_{s \in \mathcal{S}, a \in \mathcal{A} \setminus \mathcal{A}_s^{P_G}} \left\{ P_G(s) - \sum_{s' \in \mathcal{S}} P(s,a,s') P_G(s') \right\}$

15 **fim**

16 **para cada** $s \in \mathcal{S}$ **faça**

17 $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}_s^{P_G}} \left\{ e^{\lambda c(s,a)} \sum_{s' \in \mathcal{S}} P(s,a,s') V'(s') \right\}$

18 **fim**

19 **retorna** V_λ, P_G, π

20 **fim**

O consumo de tempo do algoritmo LexicográficoSensívelAoRiscoIV é $O(|\mathcal{S}^2||\mathcal{A}|n_{iter_{rs}})$, em que $n_{iter_{rs}}$ é o número de iterações executadas no laço da linha 5, que depende do valor de ϵ. O Teorema 13 mostra como o algoritmo converge para a política ótima.

Teorema 13 (Convergência do algoritmo LexicográficoSensívelAoRiscoIV[1]). *O*

[1] Esse teorema é uma modificação do Teorema 5 definido em Freire, Delgado e Reis, 2019.

algoritmo LexicográficoSensívelAoRiscoIV converge para a política ótima sob o critério lexicográfico sensível ao risco.

Demonstração. O algoritmo LexicográficoSensívelAoRiscoIV mantém duas funções valor: $V_\lambda(s)$ e $P_G(s)$. Ambas funções são inicializadas com o valor de 1 para estados meta, e 0 para todos os outros estados.

É conhecido que, se um algoritmo de iteração de valor é utilizado, os valores de $P_G(s)$ maximizarão a probabilidade à meta a partir de s após a sua execução (Kolobov, D. S. Weld *et al.*, 2011). Para $V_\lambda(s)$, uma política ótima e estacionária pode ser encontrada por iteração de valor (Patek, 2001), porque já que $\lambda < 0$ e $c(s,a) > 0$, $e^{\lambda c(s,a)}$ funciona como fator de desconto, o que garante convergência (Minami e Silva, 2012), já que todo histórico que não alcança metas tem custo infinito e valor associado 0. Além disso, pelo fato de que apenas ações que maximizam probabilidade à meta são consideradas quando o valor de $V_\lambda(s)$ é atualizado (ao computar o conjunto $\mathcal{A}_s^{P_G}$ na linha 10), o valor de probabilidade à meta será maximizado e, considerando apenas políticas que maximizam essa métrica, a utilidade exponencial esperada também é maximizada. □

7.2 Calculando C_{max} e $\bar{C}_{max}(s)$

Calcular C_{max} consiste em computar o valor da Equação 6.7. Especificamente, para calcular $\mathcal{W}(s,a)$ e $\mathcal{X}$, é preciso conhecer os valores de $V_\lambda^{\pi_L^*}$ e $P_G^{\pi_L^*}$ para pelo menos o estado inicial s_0 e qualquer estado alcançável de s_0 (o conjunto $\mathcal{S}_{reach}(s_0)$). O consumo de tempo de calcular C_{max} a partir da Equação 6.7 é $O(|\mathcal{S}|^2|\mathcal{A}|)$ no pior caso. Isso é verdade porque calcular $\mathcal{W}(s,a)$ tem um custo de $O(|\mathcal{S}|)$, e obter C_{max} consome $O(|\mathcal{S}||\mathcal{A}|)$, em função da maximização sobre o conjunto $\mathcal{X}$.

Para computar $\bar{C}_{max}(s)$ pode-se utilizar a definição recursiva da Equação 6.12. O algoritmo $\bar{C}_{max}$-Alcançáveis (Algoritmo 6) computa o valor de $\bar{C}_{max}(s)$ para cada $s \in \{s_0 \cup \mathcal{S}_{reach}(s_0)\}$. O algoritmo primeiro define os valores iniciais de $\bar{C}_{max}(s)$ (linhas 2–4) ao realizar uma busca em profundidade e atribuir a $\bar{C}_{max}(s)$ o valor de $W(s)$. Então, o algoritmo continua ao computar repetidamente a equação na linha 7 (quase o mesmo que a Equação 6.12, exceto que nela $W(s)$ é substituído pelo valor atual de $\bar{C}_{max}(s)$) para cada $s \in \{s_0 \cup \mathcal{S}_{reach}(s_0)\}$, até que o valor de $\bar{C}_{max}$ não mude para nenhum estado.

O consumo de tempo do algoritmo $\bar{C}_{max}$-Alcançáveis é $O(|\mathcal{S}|^2|\mathcal{A}|n_{iter_{cmax}})$. Isso se dá porque calcular $W(s)$ como na Equação 6.8 tem consumo de tempo $O(|\mathcal{S}||\mathcal{A}|)$, e a busca em profundidade da linha 2 tem custo $O(|\mathcal{S}|^2|\mathcal{A}|)$. Finalmente, a linha 7 tem consumo de tempo $O(|\mathcal{S}|^2|\mathcal{A}|n_{iter_{cmax}})$, em que $n_{iter_{cmax}}$ é o número de iterações executadas no laço da linha 5.

7.3 Algoritmo eGUBS-VI

Nessa seção o algoritmo eGUBS-VI (Freire, Delgado e Reis, 2019), o primeiro algoritmo a resolver SSP-MDPs sob o critério eGUBS, será descrito. Esse algoritmo pode ser dividido em três fases principais:

Algoritmo 6: $\bar{C}_{max}$-Alcançáveis

Entrada: $V^*_\lambda, P_G, \pi^*_\lambda, \lambda, K_g$, SSP-MDP $\mathcal{M} = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle$
Saída: $\bar{C}_{max}$

início
  Realize uma busca em profundidade em $\{s_0\} \cup \mathcal{S}_{reach}(s_0)$, e para cada $s \in (\{s_0\} \cup \mathcal{S}_{reach}(s_0))$:
    Compute $W(s)$ como na Equação 6.8
    $\bar{C}_{max}(s) \leftarrow W(s)$
  repita
    para cada $s \in (\{s_0\} \cup \mathcal{S}_{reach}(s_0))$ **faça**
      $\bar{C}_{max}(s) \leftarrow \max\{\bar{C}_{max}(s), \max_{s' \in \mathcal{S}_{succ}(s), a \in \mathcal{A}_s(s')}[\bar{C}_{max}(s') - c(s,a)]]\}$
    fim
  até $\bar{C}_{max}$ não mudar em nenhum estado;
  retorna $\bar{C}_{max}$
fim

1. Encontrar uma política ótima sob o critério lexicográfico sensível ao risco;
2. Definir *sequências de custo acumulado* $\mathcal{C}$ e $\mathcal{D}$, que são sequências não negativas estritamente monotônicas que determinam pontos de custo acumulado em que a política e o seu valor devem ser alterados. A *sequência de política* $\mathcal{C}$ contém um conjunto de valores para os quais o algoritmo realizará atualizações na **política** e a *sequência de valor* $\mathcal{D}$ contém os valores para os quais serão realizadas atualizações de **valor**; e
3. Melhorar uma política estacionária em uma política aumentada finita $\pi^* : \mathcal{S} \times \mathcal{C} \to \mathcal{A}$ baseada na sequência de custo acumulado $\mathcal{C}$ tal que:

$$a_t = \pi^*(s_t, C_{next}(C_t, \mathcal{C})),$$

em que $C_{next}(C_t, \mathcal{C}) = \min\{D \mid D \in \mathcal{C} \text{ e } D \geq C_t\}$. Para isso, o algoritmo precisa manipular três funções valor baseadas na sequência de custo acumulado $\mathcal{D}$:

- $V^*_\lambda(s, C)$ é a função valor do critério lexicográfico sensível ao risco começando do estado s e seguindo a política π^* a partir do custo acumulado C;
- $P^*_G(s, C)$ é a probabilidade à meta partindo do estado s seguindo a política π^* a partir do custo acumulado C; e
- $V^*_{GUBS}(s, C)$ é a função valor do critério eGUBS. Esse valor pode ser computado utilizando $V^*_\lambda(s, C)$ e $P^*_G(s, C)$, como demonstrado no Teorema 7.

7.3.1 Pseudocódigo do eGUBS-VI

O pseudocódigo do eGUBS-VI é demonstrado no Algoritmo 7. eGUBS-VI recebe como parâmetros um SSP-MDP $\mathcal{M}$, o fator de risco λ, um valor ϵ e a utilidade constante de metas K_g. O algoritmo primeiro computa a política ótima do critério lexicográfico sensível ao risco ao chamar o algoritmo LexicográficoSensívelAoRiscoIV (linha 2); então, são definidos

as sequências de valor e política, que dependem no tipo de função de custo utilizada (isso é explicado com mais detalhes na Seção 7.3.2). Finalmente, o eGUBS-VI chama o algoritmo MelhoraPolíticaEstacionária (Algoritmo 8) para encontrar uma política aumentada π^* e a função valor ótima sob o critério eGUBS V^*_{GUBS} (linha 4).

O algoritmo MelhoraPolíticaEstacionária (Algoritmo 8) recebe uma política π ótima sob o critério lexicográfico sensível ao risco, suas funções valor (V_λ e P_G), os parâmetros do modelo K_g e λ e as sequências de valor e política $\mathcal{C}$ e $\mathcal{D}$. O algoritmo passa por todos os valores de $\mathcal{D}$ do maior para o menor (laço da linha 8), atualizando os valores das funções V^*_λ, P^*_G e V^*_{GUBS} para cada estado $s \in \mathcal{S}$ (linhas 22 – 24). Para calcular esses três valores, o algoritmo primeiro calcula, para cada estado e ação, o próximo custo C' na sequência de custos $\mathcal{D}$ (linha 11). Esse valor é então utilizado para calcular os valores de Q e P'_G (linhas 12 e 13). Se $C \in \mathcal{C}$, então C é um ponto de melhoria de política e uma política aumentada finita é gerada ao utilizar o Teorema 8 (linhas 15–17). Caso contrário, a política aumentada finita atual é considerada, ou seja, a política para o custo $C' \geq C$ na sequência de valores (linhas 18–20).

Algoritmo 7: eGUBS-VI

Entrada: SSP-MDP $\mathcal{M} = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, \lambda, \epsilon, K_g$
Saída: V^*_{GUBS}, π^*
1 **início**
2 $V_\lambda, P_G, \pi \leftarrow$ LexicográficoSensívelAoRiscoIV$(\mathcal{M}, \lambda, \epsilon)$
3 $\mathcal{D}, \mathcal{C} \leftarrow$ DefineSequênciasDeValorEPolítica$(\cdot)$
4 $V^*_{GUBS}, \pi^* \leftarrow$ MelhoraPolíticaEstacionária$(\mathcal{M}, K_g, P_G, \pi, \lambda, V_\lambda, \mathcal{D}, \mathcal{C})$
5 **retorna** V^*_{GUBS}, π^*
6 **fim**

O consumo de tempo do MelhoraPolíticaEstacionária pode ser resumido pelo consumo de tempo da execução das linhas 11–13. A linha 11 tem um consumo de tempo de $O(|\mathcal{D}|^2|\mathcal{S}||\mathcal{A}|)$; a linha 12, de $O(|\mathcal{D}||\mathcal{S}|^2|\mathcal{A}|)$; e a linha 13, também de $O(|\mathcal{D}||\mathcal{S}|^2|\mathcal{A}|)$. Então, o algoritmo MelhorPolíticaEstacionária tem um custo de $O(|\mathcal{D}|^2|\mathcal{S}||\mathcal{A}| + |\mathcal{D}||\mathcal{S}|^2|\mathcal{A}|)$.

Logo, o consumo de tempo do eGUBS-VI pode ser definido como o seguinte:

- LexicográficoSensívelAoRiscoIV tem consumo de tempo de $O(|\mathcal{S}|^2|\mathcal{A}|n_{iter_{rs}})$;
- Seja o consumo de tempo do algoritmo DefineSequênciasDeValorEPolítica $O(\mathcal{T}_{seq})$;
- O algoritmo MelhoraPolíticaEstacionária tem consumo de tempo de $O(|\mathcal{D}|^2|\mathcal{S}||\mathcal{A}| + |\mathcal{D}||\mathcal{S}|^2|\mathcal{A}|)$.

Portanto, eGUBS-VI tem consumo de tempo $O(|\mathcal{S}|^2|\mathcal{A}|n_{iter_{rs}} + |\mathcal{D}|^2|\mathcal{S}||\mathcal{A}| + |\mathcal{D}||\mathcal{S}|^2|\mathcal{A}| + \mathcal{T}_{seq})$.

7.3.2 Definindo as Sequências de Valor e Política e Resultados Teóricos sobre a Otimalidade do Algoritmo eGUBS-VI

Na presente seção será demonstrado como definir sequências de valor e política quando o custo é racional ou inteiro e condições suficientes para o eGUBS-VI retornar uma política

Algoritmo 8: MelhoraPolíticaEstacionária

Entrada: SSP-MDP $\mathcal{M} = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, K_g, P_G, \lambda$, política do critério lexicográfico sensível ao risco π e $V_\lambda = V_\lambda^\pi$, sequência de valor $\mathcal{D} = \{D_1 = 0, D_2, \dots, D_N, \infty\}$, em que $D_i < D_j \Leftrightarrow i < j$, e sequência de política $\mathcal{C} = \{C_1, C_2, \dots, C_M, \infty\}$ em que $C_i < C_j \Leftrightarrow i < j$ e $\mathcal{C} \subseteq \mathcal{D}$

Saída: V^*_{GUBS}, π^*

início
- ▷Seja $V^*_\lambda : \mathcal{S} \times \mathcal{D} \to \mathbb{R}$, $P^*_G : \mathcal{S} \times \mathcal{D} \to [0,1]$, $\pi^* : \mathcal{S} \times \mathcal{C} \to \mathcal{A}$
- ▷Seja $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, $P'_G : \mathcal{S} \times \mathcal{A} \to [0,1]$
- **para cada** $s \in \mathcal{S}$ **faça**
  - $V^*_\lambda(s, \infty) = V_\lambda(s)$, $P^*_G(s, \infty) = P_G(s)$, $\pi^*(s, \infty) = \pi(s)$
  - **se** $s \in \mathcal{G}$ **então**
    - $V^*_\lambda(s, C) = V_\lambda(s), P^*_G(s, C) = P_G(s), \forall C \in \mathcal{D} \setminus \{\infty\}$
  - **fim**
- **fim**
- **para** $C \leftarrow D_N$ **até** D_1 **faça**
  - **para cada** $s \in \mathcal{S}$ **faça**
    - **para cada** $a \in \mathcal{A}$ **faça**
      - $C' \leftarrow \min\{D \mid D \in \mathcal{D} \text{ e } D \geq C + c(s,a)\}$
      - $Q(s,a) \leftarrow e^{\lambda c(s,a)} \sum_{s' \in \mathcal{S}} P(s,a,s') V^*_\lambda(s', C')$
      - $P'_G(s,a) \leftarrow \sum_{s' \in \mathcal{S}} P(s,a,s') P^*_G(s', C')$
    - **fim**
    - **se** $C \in \mathcal{C}$ **então**
      - $\pi^*(s,C) \leftarrow \arg\max_{a \in \mathcal{A}} e^{\lambda C} Q(s,a) + K_g P'_G(s,a)$,
      - $a^* \leftarrow \pi^*(s,C)$
    - **senão**
      - $C' \leftarrow \min\{D \mid D \in \mathcal{D} \text{ e } D \geq C\}$,
      - $a^* \leftarrow \pi^*(s,C')$
    - **fim**
    - $V^*_\lambda(s,C) = Q(s,a^*)$
    - $P^*_G(s,C) = P'_G(s,a^*)$
    - $V^*_{GUBS}(s,C) = e^{\lambda C} V^*_\lambda(s,C) + K_g P^*_G(s,C)$
  - **fim**
- **fim**
- **retorna** V^*_{GUBS}, π^*

fim

ótima. Primeiro, será considerado o caso em que o custo é racional.

Teorema 14 (Custo racional – política ótima [2]). *Considere que a função de custo tem sua imagem nos números racionais, ou seja, $c : \mathcal{S} \times \mathcal{A} \to \mathcal{I} \subset \mathcal{Q}$, e que todos valores $r_i \in \mathcal{I}$ tem uma representação $r_i = \frac{p_i}{q}$ com o mesmo denominador q (se esse não é o caso, esse valor pode ser obtido ao calcular o menor múltiplo comum). Finalmente, seja p o maior divisor comum do numerador de $\mathcal{I}$.*

Sejam

- C_{max} *definido como no Teorema 9; e*
- $\mathcal{C} = \mathcal{D} = \left\{0, \frac{p}{q}, \frac{2p}{q}, \frac{3p}{q}, \ldots, C_{max}, \infty\right\}$;

então, a entrada $(\mathcal{M}, \lambda, \epsilon, K_g)$ é suficiente para que o algoritmo eGUBS-VI retorne uma política ótima.

Demonstração. A parte principal do algoritmo eGUBS-VI é o algoritmo MelhoraPolíticaEstacionária, que recebe π (a política ótima sob o critério lexicográfico sensível ao risco no SSP-MDP $\mathcal{M}$), V_λ (a utilidade exponencial esperada de π) e P_G (probabilidade à meta de π).

Primeiro, no algoritmo MelhoraPolíticaEstacionária, pelo Teorema 9 e inicialização de $\pi^*(s, \infty)$, $V^*_\lambda(s, \infty)$, e $P^*_G(s, \infty)$ (linhas 3–6), é verdade que π^* é ótima para $C > C_{max}$. Essa propriedade é mantida porque o valor de C_{max} foi escolhido tal que π, que é ótima sob o critério lexicográfico sensível ao risco, tem a seguinte propriedade para qualquer $C \geq C_{max}$:

$$V^\pi_{GUBS}(s, C) = \max_{a \in \mathcal{A}} \left\{ \sum_{s' \in \mathcal{S}} P(s, a, s') V^\pi_{GUBS}(s', C + c(s, a)) \right\},$$

logo, π não pode ser melhorada para custos acumulados $C \geq C_{max}$.

Segundo, pelo Teorema 8 (Melhoria de política sob o eGUBS), é verdade que π será melhorada na política não markoviana π^* (linha 16). Além disso, pelo Teorema 7 (valor estado-custo de políticas estacionárias sob o eGUBS), também é verdade que as três funções V^*_λ, P^*_G, e V^*_{GUBS} são corretas para $C \leq C_{max}$, após executar as linhas 22–24.

Terceiro, as sequências de valor $\mathcal{D}$ e de política $\mathcal{C}$ tem todos os custos acumulados alcançáveis. Logo, porque custos são sempre positivos e porque o algoritmo MelhoraPolíticaEstacionária faz atualizações do maior custo acumulado para o menor (linha 8), assim como uma iteração de valor para MDPs de horizonte finito (Puterman, 1994; Mausam e Kolobov, 2012), todos os valores utilizados nas atualizações são corretos, e o resultado segue. □

A seguir, é considerado o caso em que o custo é inteiro.

Corolário 6 (Custo inteiro – política ótima [3]). *Considere que a função de custo tem a sua imagem nos números naturais. Ou seja, $c : \mathcal{S} \times \mathcal{A} \to \mathcal{I} \subset \mathbb{N}$. Seja $\mathcal{C} = \mathcal{D} =$*

[2] Esse Teorema é uma modificação do Teorema 6 definido em Freire, Delgado e Reis, 2019.

[3] Esse corolário é uma pequena modificação do Corolário 1 definido em Freire, Delgado e Reis, 2019.

$\{0, 1, 2, \dots, C_{max}, \infty\}$. *Então, a entrada* $(\mathcal{M}, \lambda, \epsilon, K_g)$ *é suficiente para que o algoritmo eGUBS-VI retorne uma política ótima.*

Demonstração. Considerando $p = 1$ e $q = 1$, pode-se afirmar que a função $c : \mathcal{S} \times \mathcal{A} \to \mathcal{I} \subset \mathbb{N}$ segue a restrição do Teorema 14, já que cada elemento da sua imagem é um número racional da forma $c(s, a) = \frac{c(s,a)p}{q}$. O mesmo pode ser denotado para as sequências $\mathcal{C}$ e $\mathcal{D}$, já que $\mathcal{C} = \mathcal{D} = \{0, 1, 2, \dots, C_{max}, \infty\} = \{\frac{0p}{q}, \frac{1p}{q}, \frac{2p}{q}, \dots, \frac{C_{max}p}{q}, \infty\}$. □

Corolário 7 (Função valor correta[4]). *Considere a função de custo* $c(\cdot, \cdot)$ *e as sequências* $\mathcal{C}$ *e* $\mathcal{D}$ *como no Teorema 14. Então, a entrada* $(\mathcal{M}, \lambda, \epsilon, K_g)$ *é suficiente para que o algoritmo eGUBS-VI retorne uma função valor* V^*_{GUBS} *correta considerando a política* π^*.

Demonstração. Pelo Teorema 14, segue o resultado de que a entrada $(\mathcal{M}, \lambda, \epsilon, K_g)$ é suficiente para que eGUBS-VI retorne uma política ótima sob o eGUBS. Na linha 24 do algoritmo MelhoraPolíticaEstacionária, o valor da política ótima π^* em (s, C) é calculada baseada no Teorema 8, o que faz esse valor correto considerando π^*. □

Resumindo, se a função de custo $c(\cdot)$ e sequências $\mathcal{C}$ e $\mathcal{D}$ respeitam o Teorema 14, o algoritmo eGUBS-VI retorna uma política ótima. Caso contrário, eGUBS-VI retorna uma política subótima. Logo, por exemplo, se a função de custo tem a sua imagem nos números inteiros, o algoritmo eGUBS-VI utilizando o Algoritmo 9 para definir as sequências de valor e política retorna uma política e função valor ótimas. Note que nesse caso, ambas sequências $\mathcal{C}$ e $\mathcal{D}$ são iguais e o valor de C_{max} utilizado é calculado como na Equação 6.7. Como elas são iguais, nesse caso, as linhas 19–20 do Algoritmo 8 não serão executadas quando os algoritmos 8 e 9 são chamados pelo Algoritmo 7.

Outro caso possível é o caso em que os custos são reais porém irracionais. Isso não é abordado pelo Teorema 14, e um procedimento de transformação seria possivelmente necessário para gerenciar as sequências $\mathcal{C}$ e $\mathcal{D}$. Por exemplo, isso pode ser alcançado ao transformar custos para os números racionais mais próximos ou ao normalizá-los por uma constante suficientemente grande e truncá-los para transformá-los em inteiros. No entanto, essas transformações introduzem erros e, nesse caso, o algoritmo eGUBS-VI poderia retornar uma política subótima.

Quando a função de custo tem uma imagem nos inteiros, o consumo de tempo do eGUBS-VI pode ser definido como $O(|\mathcal{S}|^2|\mathcal{A}|n_{iter_{rs}} + (C_{max})^2|\mathcal{S}||\mathcal{A}| + C_{max}|\mathcal{S}|^2|\mathcal{A}|)$, já que $|\mathcal{D}| = O(C_{max})$ e $\mathcal{T}_{seq} = O(|\mathcal{S}|^2|\mathcal{A}|)$.

7.4 Algoritmo GUBS-AO*

Apesar de ser ótimo sob certas condições, o algoritmo eGUBS-VI sempre precisa atualizar o espaço inteiro de estados [5] para encontrar a política ótima para eGUBS. Uma ideia comum nesses casos é utilizar um procedimento de busca heurística para potencialmente

[4] Esse corolário é uma pequena modificação do Corolário 2 definido em Freire, Delgado e Reis, 2019.

[5] Note que ao considerar um estado inicial s_0, o eGUBS-VI precisa processar apenas s_0 e o conjunto de estados alcançáveis a partir desse estado.

Algoritmo 9: DefineSequênciasDeValorEPolítica ▷ para custos inteiros

Entrada: SSP-MDP $\mathcal{M} = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, V_\lambda, P_G, K_g, \lambda$
Saída: $\mathcal{D}$, $\mathcal{C}$

1 **início**
  ▷ Compute C_{max} como na Equação 6.7
2 $C_{max} \leftarrow \text{ComputaCmax}(\mathcal{M}, V_\lambda, P_G, K_g, \lambda)$
3 $\mathcal{C} \leftarrow \{0, 1, 2, \ldots, C_{max}, \infty\}$
4 $\mathcal{D} \leftarrow \mathcal{C}$
5 **retorna** $\mathcal{D}$, $\mathcal{C}$
6 **fim**

encontrar a solução de maneira mais eficiente, como em algoritmos como o AO* (Martelli e Montanari, 1973) e o LAO* (Hansen e Zilberstein, 2001).

O espaço de estados aumentados que precisa ser mantido para representar uma política para o critério eGUBS pode ser representado por um hipergrafo, em que os nós são estados aumentados $(s, C) \in \mathcal{S} \times \mathbb{R}$ e as suas hiperarestas representam ações a que levam (s, C) para qualquer estados sucessores (s', C'), tal que $s' \in \mathcal{S}_{succ}(s)$ e $C' = C + c(s, a), \forall a \in \mathcal{A}_s(s')$.

Definição 11 (Grafo implícito do critério eGUBS para $\mathcal{M}$)**.** *O grafo implícito do critério eGUBS para um SSP-MDP $\mathcal{M}$ é um grafo tal que os nós são estados aumentados $(s, C) \in \mathcal{S} \times \mathbb{R}$ e as suas hiperarestas representam ações a que levam (s, C) para qualquer estados sucessores (s', C'), tal que $s' \in \mathcal{S}_{succ}(s)$ e $C' = C + c(s, a), \forall a \in \mathcal{A}_s(s')$.*

Note que o grafo da Definição 11 contempla apenas o espaço de estados aumentados ($(s, C) \in \mathcal{S} \times \mathbb{R}$). Dessa maneira, um algoritmo que precisa manter uma política que leva em conta o espaço de estados simples $\mathcal{S}$ precisa utilizar alguma estrutura adicional para armazenar esses estados.

Proposição 1. *O grafo implícito do eGUBS para um SSP-MDP $\mathcal{M}$ é acíclico.*

Demonstração. Um nó (s, C) estaria em um ciclo apenas se $\mathcal{M}$ tivesse ações com custos negativos ou nulos. Já que a função de custo de $\mathcal{M}$ tem sua imagem positiva para estados que não são metas (s, C) não pode estar em um ciclo. □

Proposição 2. *Dada a função $\bar{C}_{max}(s)$ como descrita no Teorema 10, o grafo implícito do eGUBS para um SSP-MDP $\mathcal{M}$ pode ser representado de maneira finita.*

Demonstração. O grafo terá um número finito de nós (s, C'), considerando $s \in \mathcal{S}_{reach}(s_0)$ e $C' \leq \bar{C}_{max}(s)$, pois nós com $C' > \bar{C}_{max}(s)$ não precisam ser representados já que a política ótima a partir desse ponto é estacionária. □

Já que o grafo implícito do eGUBS é acíclico e finito, utilizar uma variante do algoritmo AO* é suficiente [6] para resolver o critério eGUBS por busca heurística.

[6] Mesmo que o algoritmo eGUBS-AO* utilize uma política estacionária ótima para o critério lexicográfico sensível ao risco e o grafo de solução dessa política pode ter ciclos, não é necessário construí-lo, já

A presente seção então introduz o algoritmo eGUBS-AO*, um algoritmo de busca heurística baseado no AO* que encontra políticas ótimas para SSP-MDPs sob o critério eGUBS. Esse algoritmo tem três fases principais:

1. Computar o valor do critério lexicográfico sensível ao risco $V_\lambda(s)$ para cada $s \in \{s_0\} \cup \mathcal{S}_{reach}(s_0)$;
2. Computar $\bar{C}_{max}(s)$ também para cada $s \in \{s_0\} \cup \mathcal{S}_{reach}(s_0)$; e
3. Computar V^*_{GUBS} ao realizar busca heurística no SSP-MDP.

A primeira diferença entre os algoritmos eGUBS-VI e eGUBS-AO* é que o último computa a função $\bar{C}_{max}(s)$ para cada estado alcançável de s_0, enquanto que o eGUBS-VI computa um único valor escalar C_{max} e o utiliza para cada estado. Considerando isso, o passo 3 é a principal diferença entre os dois algoritmos. Para executar a busca, o eGUBS-AO* mantém dois grafos: G e G'. G é o grafo da melhor solução parcial, contendo a solução conhecida até um dado momento. G' é o grafo explícito, que guarda informações sobre todos estados aumentados visitados durante a busca. Já que é conhecido pela Proposição 1 que o grafo implícito do eGUBS de um SSP-MDP é acíclico e que G e G' são subconjuntos desse grafo, então ambos G e G' são acíclicos.

Todo nó (s, C) nesses grafos são estados aumentados do SSP-MDP sendo resolvido, e toda hiperaresta representa os resultados de tomar ações específicas do estado aumentado atual. Além disso, todo nó (s, C) no grafo contém os seguintes atributos:

- $v = V^*_\lambda(s, C)$ – a função cujo valor é a utilidade esperada começando do estado s e seguindo a melhor política atual a partir do custo acumulado C;
- $p = P^*_G(s, C)$ – o valor de probabilidade à meta a partir do estado s e seguindo a melhor política atual a partir do custo acumulado C;
- $a^* = \pi^*(s, C)$ – a melhor ação atual em (s, C) sob o critério eGUBS; e
- *resolvido* – uma propriedade que indica se (s, C) foi marcado como resolvido ou não.

Note que o valor de $V^*_{GUBS}(s, C)$ não é mantido pelo grafo mas pode ser facilmente calculado em termos das propriedades v e p como demonstrado no Teorema 7.

A busca pode ser resumida nos seguintes passos:

(i) um subconjunto de estados de $\mathcal{S}$ é escolhido para ser expandido, isto é, os estados vizinhos de cada um desses estados são adicionados ao grafo explícito G' e os seus valores para as propriedades v, p e a^* são respectivamente inicializados com uma função heurística admissível h_v, função de probabilidade à meta P_G e política lexicográfica sensível ao risco π^*_λ. Nesse caso, uma função heurística admissível $h_v : \mathcal{S} \to \mathbb{R}$ é uma função que superestima o valor de um estado $s \in \mathcal{S}$. Ou seja, $h_v(s) \geq V^*_\lambda(s)$. O subconjunto de interesse de $\mathcal{S}$ mencionado será referido como $\mathcal{S}^{all}_u$, e contém estados de G que não foram expandidos até esse ponto. Abordagens diferentes podem ser utilizadas para definir que estados $\mathcal{S}^{all}_u$ conterá em cada iteração da busca. Por exemplo, um único estado não expandido pode ser escolhido para ser

que nesse ponto do algoritmo o critério lexicográfico sensível ao risco já foi resolvido pelo algoritmo LexicográficoSensívelAoRiscoIV.

expandido, assim como todos estados não expandidos podem ser adicionados à S_u^{all} de uma vez só. A abordagem aqui utilizada será de expandir n_{exp} *níveis* de estados, em que n_{exp} é dado como parâmetro para o algoritmo. A razão para escolher esse método é a de que expandir vários níveis de uma vez só pode levar a valores mais informativos calculados durante a fase de iteração de valor, conforme a profundidade da busca fica maior, do que quando apenas um estado ou apenas um nível de estados é expandido;

(ii) os valores das propriedades v, p e a^* são atualizados via iteração de valor para estados em S_u^{all} e todos estados que podem alcançá-los ao seguir a melhor solução parcial. Durante essa fase, um estado pode ser marcado como resolvido se os seus sucessores ao seguir a melhor solução já tiverem sido resolvidos também;

(iii) finalmente, o grafo de melhor solução parcial G é atualizado considerando os novos valores calculados no passo anterior.

Esses passos continuam até que o estado inicial seja finalmente marcado como resolvido.

7.4.1 Pseudocódigo do eGUBS-AO*

O pseudocódigo do eGUBS-AO* é mostrado no Algoritmo 10. Ele recebe como entrada um SSP-MDP $\mathcal{M}$; uma função heurística h_v; o fator de risco λ; um limite de erro ϵ utilizado como parâmetro para o algoritmo LexicográficoSensívelAoRiscoIV; a utilidade constante da meta K_g, e o número de níveis a serem expandidos n_{exp}. O algoritmo começa ao computar a política ótima para o critério lexicográfico sensível ao risco (π_λ^*), a sua utilidade exponencial esperada (V_λ^*), e a probabilidade à meta (P_G). Então o valor da função $\bar{C}_{max}$ para o estado inicial s_0 e para todo estado alcançável de s_0 é computado ao chamar o algoritmo $\bar{C}_{max}$-Alcançáveis na linha 3. Se $\bar{C}_{max}(s_0) \leq 0$, isso significa que quando o processo inicia, a solução ótima sob o critério eGUBS para o estado inicial $(s_0, 0)$ é a própria política ótima do critério lexicográfico sensível ao risco. Então, nesse caso, essa política é retornada (linha 5). Caso contrário, ambos grafos de melhor solução parcial G e o grafo explícito G' são inicializados.

O eGUBS-AO* é executado enquanto o estado aumentado inicial $(s_0, 0)$ não é marcado como resolvido. Em cada iteração, a expansão dos estados é realizada pelo algoritmo eGUBS-AO*-Expande (linha 12). eGUBS-AO*-Expande retorna o grafo explícito G' atualizado com novos estados, e o conjunto Z, que contém estados para os quais um procedimento de iteração de valor (eGUBS-VI-mod) é executado na linha seguinte. Então, eGUBS-AO* atualiza o grafo de melhor solução parcial G baseado em G' a partir do procedimento AtualizaMelhoresAções (linha 14), que atualiza os arcos em G de acordo com os novos valores e ações ótimas atuais resultantes computadas pelo algoritmo eGUBS-VI-mod; e computa o conjunto atual de estados não expandidos (linha 15). Quando o laço finaliza (ou seja, quando o estado aumentado inicial $(s_0, 0)$ já foi resolvido), os valores da função valor ótima V_{GUBS}^* e a política ótima π^* são salvos de G' (linhas 17–20) e então retornados (linhas 21).

Algoritmo 10: eGUBS-AO*

Entrada: $h_v, \lambda, \epsilon, K_g, n_{exp}$, SSP-MDP $\mathcal{M} = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle$
Saída: V^*_{GUBS}, π^*

1 **início**
2 $V^*_\lambda, P_G, \pi^*_\lambda \leftarrow$ LexicográficoSensívelAoRiscoIV$(\mathcal{M}, \lambda, \epsilon)$
3 $\bar{C}_{max} \leftarrow \bar{C}_{max}$-Alcançáveis$(V^*_\lambda, P_G, \pi^*_\lambda, \lambda, K_g, \mathcal{M})$
4 **se** $\bar{C}_{max}(s_0) \leq 0$ **então**
5 **retorna** $V^*_\lambda, \pi^*_\lambda$
6 **fim**
7 $G \leftarrow \{(s_0, 0)\}$
8 $G' \leftarrow \{(s_0, 0)\}$
9 $G'[(s_0, 0)].v \leftarrow V^*_\lambda(s), G'[(s_0, 0)].p \leftarrow P_G(s)$
▷Seja $\mathcal{S}_u$ o conjunto de todos estados não expandidos em G
10 Compute $\mathcal{S}_u$
11 **enquanto** $G'[(s_0, 0)].resolvido = falso$ **faça**
12 $G', Z \leftarrow$ eGUBS-AO*-Expande$(G, G', V^*_\lambda, \pi^*_\lambda, P_G, \bar{C}_{max}, \mathcal{S}_u, n_{exp}, h_v, \mathcal{M})$
13 $G' \leftarrow$ eGUBS-VI-mod$(Z, \lambda, K_g, \mathcal{M}, G, G')$
14 $G \leftarrow$ AtualizaMelhoresAções(G')
15 Compute $\mathcal{S}_u$
16 **fim**
17 **para cada** $(s, C) \in G$ **faça**
18 $V^*_{GUBS}(s, C) \leftarrow e^{\lambda C} G'[(s, C)].v + K_g G'[(s, C)].p$
19 $\pi^*(s, C) \leftarrow G'[(s, C)].a^*$
20 **fim**
21 **retorna** V^*_{GUBS}, π^*
22 **fim**

7.4.2 eGUBS-AO*-Expande

eGUBS-AO*-Expande (Algoritmo 11) realiza a expansão de estados em um grafo explícito G'. O algoritmo funciona ao expandir n_{exp} (valor recebido como parâmetro) níveis para o grafo explícito G' recebido, e retornar o par (G', Z), em que Z é o conjunto de estados para o qual iteração de valor será depois aplicada. Mais especificamente, eGUBS-AO*-Expande recebe $\mathcal{S}_u$, o conjunto atual de estados não expandidos, como um de seus parâmetros. Se $\mathcal{S}_u$ é vazio, então Z será apenas o conjunto de estados no grafo de melhor solução parcial G. Caso contrário, as variáveis $\mathcal{S}^{all}_u$ e $\mathcal{S}^{new}_u$ são inicializadas (linha 2). Essas variáveis são respectivamente o conjunto de todos estados não expandidos processados pelo algoritmo, e o conjunto de novos estados não expandidos processados por ele em um nível específico. Cada um dos n_{exp} níveis é então expandido em uma iteração do laço externo da linha 3.

Nele, todo estado $(s, C) \in \mathcal{S}_u$ é expandido (linhas 5–22). O conjunto $\mathcal{S}_s$, que inclui os sucessores (s', C') de (s, C) que não foram processados é criado na linha 7 e adicionado a $\mathcal{S}^{new}_u$ na linha 8, tendo seus valores de v, p e a^* iniciados, respectivamente. Se (s', C') é um estado meta, v é inicializado com 1 (linha 13). Se (s, C) já foi resolvido ou $C' \geq \bar{C}_{max}(s')$, v é inicializado com $V^*_\lambda(s')$ e a^* com $\pi^*_\lambda(s')$ (linha 16). Se nenhuma dessas duas condições for

verdadeira, então v é inicializado com o valor heurístico $h_v(s)$ (linha 18). A propriedade p de (s', C') sempre é inicializada com $P_G(s')$ na linha 20. Finalmente, ao fim do laço mais externo , S_u recebe o valor de $\mathcal{S}_u^{new}$ (linha 23) e $\mathcal{S}_u^{new}$ é feito vazio (linha 24). Então, quando o laço é finalizado, Z recebe o valor de $\mathcal{S}_u^{all}$, que no momento tem todos estados que foram expandidos durante a execução do algoritmo e é retornado juntamente com G'.

Algoritmo 11: eGUBS-AO*-Expande

Entrada: $G, G', V_\lambda^*, \pi_\lambda^*, P_G, \bar{C}_{max}, \mathcal{S}_u, n_{exp}, h_v$, SSP-MDP $\mathcal{M} = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle$
Saída: G', Z

1 **se** $|\mathcal{S}_u| > 0$ **então**
2 $\mathcal{S}_u^{all} \leftarrow \{\}$, $\mathcal{S}_u^{new} \leftarrow \{\}$
3 **para cada** *nivel* $\leftarrow 0$ **to** $n_{exp} - 1$ **faça**
4 $\mathcal{S}_u^{all} \leftarrow \mathcal{S}_u^{all} \cup \mathcal{S}_u$
5 **enquanto** $|\mathcal{S}_u| > 0$ **faça**
6 Remova um estado de $\mathcal{S}_u$ e o chame de (s, C)
7 $\mathcal{S}_s \leftarrow \{(s', C') \mid s' \in \mathcal{S}_{succ}(s), s' \notin G' \text{ e } C' = C + c(s, a), \forall a \in \mathcal{A}\}$
8 $\mathcal{S}_u^{new} \leftarrow \mathcal{S}_u^{new} \cup \mathcal{S}_s$
9 **para cada** $(s', C') \in \mathcal{S}_s$ **faça**
10 $G' \leftarrow G' \cup \{(s', C')\}$
11 $G'[(s', C')].resolvido \leftarrow falso$
12 **se** $s' \in \mathcal{G}$ **então**
13 $G'[(s', C')].v \leftarrow 1$
14 **senão se** $(s', C') \in \mathcal{G}$ or $C' \geq \bar{C}_{max}(s')$ **então**
15 $G'[(s', C')].resolvido \leftarrow verdadeiro$
16 $G'[(s', C')].v \leftarrow V_\lambda^*(s'), G'[(s', C')].a^* \leftarrow \pi_\lambda^*(s')$
17 **senão**
18 $G'[(s', C')].v \leftarrow h_v(s')$
19 **fim**
20 $G'[(s', C')].p \leftarrow P_G(s')$
21 **fim**
22 **fim**
23 $\mathcal{S}_u \leftarrow \mathcal{S}_u^{new}$
24 $\mathcal{S}_u^{new} \leftarrow \{\}$
25 **fim**
26 $Z \leftarrow \mathcal{S}_u^{all}$
27 **senão**
28 $Z \leftarrow \{(s, C) \mid (s, C) \in G\}$
29 **fim**
30 **retorna** G', Z

7.4.3 eGUBS-VI-mod

eGUBS-VI-mod (Algoritmo 12) recebe Z e, no laço da linha 2, obtém um estado (s', C') desse conjunto em uma determinada ordem tal que não existe outro estado em Z que pode ser alcançado de (s', C'). Em outras palavras, (s', C') deve ser escolhido apenas se

Algoritmo 12: eGUBS-VI-mod

Entrada: Z, λ, K_g, SSP-MDP $\mathcal{M} = \langle \mathcal{S}, s_0, \mathcal{A}, P, c, \mathcal{G} \rangle, G, G'$
Saída: G'

1 **início**
  ▷$Q : \mathcal{S} \times \mathbb{R} \times \mathcal{A} \to \mathbb{R}$, $P'_G : \mathcal{S} \times \mathbb{R} \times \mathcal{A} \to [0, 1]$
2 **enquanto** Z não é vazio **faça**
3 Retire um estado (s', C') de Z tal que não existem outros estados em Z que podem ser alcançados ao seguir ações marcadas começando de (s', C')
4 **para cada** $a \in \mathcal{A}$ **faça**
5 $C'' \leftarrow C' + c(s', a)$
6 $Q(s', C', a) \leftarrow e^{\lambda c(s', a)} \sum_{s'' \in \mathcal{S}} P(s', a, s'') G'[(s'', C'')].v$
7 $P'_G(s', C', a) \leftarrow \sum_{s'' \in \mathcal{S}} P(s', a, s'') G'[(s'', C'')].p$
8 **fim**
9 $a^* \leftarrow \arg\max_{a \in \mathcal{A}} e^{\lambda C'} Q(s', C', a) + K_g P'_G(s', C', a)$
10 $v_{old} \leftarrow G'[(s', C')].v$
11 $G'[(s', C')].p \leftarrow P'_G(s', C', a^*)$
12 $G'[(s', C')].v \leftarrow Q(s', C', a^*)$
13 $G'[(s', C')].a^* \leftarrow a^*$
14 $\mathcal{S}_{a^*} \leftarrow \{(s'', C'') \mid s'' \in \mathcal{S}_{succ}(s') \text{ e } C'' = C' + c(s', a^*)\}$
15 **se** $G'[(s'', C'')].resolvido = verdadeiro, \forall (s'', C'') \in \mathcal{S}_{a^*}$ **então**
16 $G'[(s', C')].resolvido \leftarrow verdadeiro$
17 **fim**
18 **se** $G'[(s', C')].v < v_{old}$ ou $G'[(s', C')].resolvido = verdadeiro$ **então**
19 $Z \leftarrow Z \cup \{(s'', C'') \mid (s'', C'')$ é um estado predecessor de (s', C') em G' sob ações marcadas em $G\}$
20 **fim**
21 **fim**
22 **retorna** G'
23 **fim**

ele não depende de qualquer outro estado em Z para atualizar os seus valores. Então, nas linhas de 10 a 13 o algoritmo computa os valores dos estados e ações ótimas, na mesma maneira que no eGUBS-VI. Finalmente, se todos estados sucessores de (s', C') ao seguir a ação ótima $G'[(s', C')].a^*$ estiverem resolvidos, isso significa que (s', C') também já está resolvido. Se o novo valor de $G[(s', C')].v$ fica menor, ou se (s', C') está resolvido, todos os seus predecessores em G' sob ações marcadas em G são inseridos em Z (linhas 18–20), então seus valores são também atualizados.

7.4.4 Resultados Teóricos sobre eGUBS-AO*

O consumo de tempo do algoritmo eGUBS-AO* pode ser divido nos seguintes elementos:

1. O algoritmo LexicográficoSensívelAoRiscoIV tem consumo de tempo $O(|\mathcal{S}|^2 |\mathcal{A}| n_{iter_{rs}})$;
2. $\bar{C}_{max}$-Alcançáveis tem consumo de tempo $O(|\mathcal{S}|^2 |\mathcal{A}| n_{iter_{cmax}})$;

3. Apesar de não precisar construir sequências $\mathcal{C}$ e $\mathcal{D}$ como é feito no eGUBS-VI, no pior caso os custos acumulados em estados aumentados que serão processados serão da mesma ordem $|\mathcal{D}|$;

4. O custo para eGUBS-AO* realizar passos de melhoria de política será similar ao custo de fazer isso no algoritmo MelhoraPolíticaEstacionária no pior caso. A única diferença é que o eGUBS-AO* não precisa realizar a operação de minimização que acontece nas linhas 11 e 19 do algoritmo MelhoraPolíticaEstacionária (que é chamado pelo eGUBS-VI), então ao invés de $|\mathcal{D}|^2$, se tem $|\mathcal{D}|$ no consumo de tempo. Então, o consumo de tempo total de realizar passos de melhoria de política no algoritmo eGUBS-AO* é de $O(|\mathcal{D}||\mathcal{S}|^2|\mathcal{A}|)$.

Logo, o tempo total de consumo do algoritmo eGUBS-AO* no pior caso, é de $O(|\mathcal{S}|^2|\mathcal{A}|n_{iter_{rs}} + |\mathcal{S}|^2|\mathcal{A}|n_{iter_{cmax}} + |\mathcal{D}||\mathcal{S}|^2|\mathcal{A}|)$.

Para os seguintes lemas, note novamente que $a^* = G'[(s,C)].a^*$, ou seja, a melhor ação atual para (s,C) no grafo G', como na linha 9 do algoritmo eGUBS-VI-mod.

Lema 1 (**Admissibilidade do algoritmo eGUBS-VI-mod**). *Dado que o grafo G' recebido como entrada no eGUBS-VI-mod tem valores de $G'[(s,C)].v \geq V^*_\lambda(s,C)$ e $G'[(s,C)].p \geq P_G(s)$, $\forall (s,C) \in G'$ (isto é, eles são admissíveis), quando eGUBS-VI-mod é finalizado, os valores atualizados de $G'[(s,C)].v$ e $G'[(s,C)].p$ para cada estado $(s,C) \in Z$ continuam sendo admissíveis.*

Demonstração. $G'[(s,C)].v$ e $G'[(s,C)].p$ são calculados utilizando as respectivas equações de atualização:

(i) $G'[(s,C)].v \leftarrow e^{\lambda c(s,a^*)} \sum_{s' \in \mathcal{S}} P(s,a^*,s')G'[(s',C')].v$;

(ii) $G'[(s,C)].p \leftarrow \sum_{s' \in \mathcal{S}} P(s,a^*,s')G'[(s',C')].p$

Considerando o fato de que o valor de ambos $G'[(s',C')].v$ e $G'[(s',C')].p$ são admissíveis, isto é, $G'[(s',C')].v \geq V^*_\lambda(s',C')$ e $G'[(s',C')].p \geq P_G(s')$:

$$\begin{aligned} G'[(s,C)].v &\leftarrow e^{\lambda c(s,a^*)} \sum_{s' \in \mathcal{S}} P(s,a^*,s')G'[(s',C')].v, \\ &\geq e^{\lambda c(s,\pi^*_{GUBS}(s,C))} \sum_{s' \in \mathcal{S}} P(s,\pi^*_{GUBS}(s,C),s')V^*_\lambda(s',C + c(s,\pi^*_{GUBS}(s,C))), \\ &= V^*_\lambda(s,C). \end{aligned}$$

e:

$$\begin{aligned} G'[(s,C)].p &\leftarrow \sum_{s' \in \mathcal{S}} P(s,a^*,s')G'[(s',C')].p, \\ &\geq \sum_{s' \in \mathcal{S}} P(s,\pi^*_\lambda(s),s')P_G(s'), \\ &= P_G(s). \end{aligned}$$

□

Lema 2 (Laço principal do algoritmo eGUBS-AO*). *O laço das linhas de 11 a 16 no eGUBS-AO* executa em um número finito de iterações e, quando finalizado, os valores de $G'[(s,C)].v$ e $G'[(s,C)].p$ serão ótimos sob o critério eGUBS, $\forall (s,C) \in G'$ se uma função heurística admissível h_v é fornecida como entrada.*

Demonstração. A cada iteração do laço da linha 11, eGUBS-AO* irá expandir estados por n_{exp} níveis ao chamar o algoritmo eGUBS-AO*-Expande. Pelas proposições 1 e 2, eventualmente nenhum estado não expandido existirá em G' e $\mathcal{S}_u$ será vazio depois que a linha 15 é executada.

Pela admissibilidade do algoritmo eGUBS-VI-mod (Lema 1), eGUBS-AO* manterá valores admissíveis de $G'[(s,C)].v$, já que essa propriedade é primeiramente inicializada com a função heurística admissível $h_v(s)$ na linha 18 do algoritmo eGUBS-AO*-Expande. $G'[(s,C)].p$ também sempre será admissível já que é inicializado com o valor de $P_G(s)$, na linha 20 do algoritmo eGUBS-AO*-Expande, que é por si só admissível para a função $P_G^{\pi^*_\lambda}(s)$.

Adicionalmente, note que o algoritmo eGUBS-VI-mod tem um procedimento de marcação de que garante que um estado aumentado (s,C) é resolvido para $V^*_\lambda(s,C)$ se todos os seus vizinhos também já tiverem sido resolvidos.

Por essas propriedades, eventualmente o estado inicial $(s_0,0)$ será resolvido, e o laço principal do eGUBS-AO* finalizará após um número finito de iterações. □

O Teorema 15 é relacionado à otimalidade do eGUBS-AO*.

Teorema 15 (Otimalidade do algoritmo eGUBS-AO*). *O algoritmo eGUBS-AO* retorna uma política ótima sob o critério eGUBS quando utilizada uma heurística admissível h_v.*

Demonstração. A partir do Lema 2, é conhecido que o laço principal do algoritmo eGUBS-AO* finaliza após um número finito de iterações e, depois disso, os valores $G'[(s,C)].v$ e $G'[(s,C)].p$ são ótimos, se uma função heurística admissível h_v é fornecida como entrada.

Então, no laço da linha 17 a função valor $V^*_{GUBS}(s,C)$ e sua política $\pi^*_{GUBS}(s,C)$ serão construídas de maneira correta, e retornadas na linha 21.

□

Capítulo 8

Experimentos

Experimentos foram realizados em três domínios: *Navigation* (introduzido na IPPC'11 (Sanner e Yoon, 2011)), *River* (Freire e Delgado, 2017) e *Triangle Tireworld* (Little, Thiebaux *et al.*, 2007). Esses domínios foram descritos e utilizados como ambientes PDDLGym[1] (Silver e Chitnis, 2020). Uma busca de profundidade no MDP fatorado obtido da representação PDDL dos problemas foi realizada para gerar o conjunto de estados alcançáveis a partir do estado inicial ($S_{reach}(s_0)$). Esse conjunto, que poderia ser menor ou igual ao conjunto de todos os estados do problema, é utilizado como entrada do eGUBS-VI.

Com os experimentos, o presente trabalho tem como objetivo responder às seguintes questões:

1. Qual é a influência dos parâmetros λ e K_g na diferença entre C_{max} (utilizado no eGUBS-VI) e $\bar{C}_{max}(s_0)$ (utilizado no eGUBS-AO*)?
2. Qual é a influência dos parâmetros λ e K_g no número de estados armazenados em memória pelos algoritmos eGUBS-VI e eGUBS-AO*?
3. Qual é a influência dos parâmetros λ e K_g no número de atualizações realizadas pelos algoritmos eGUBS-VI e eGUBS-AO*?
4. Como eGUBS-VI e eGUBS-AO* escalam de acordo com o número de estados do problema em termos de tempo?
5. Existem direções sobre qual algoritmo utilizar entre o eGUBS-VI ou o eGUBS-AO*?
6. Como o critério eGUBS se compara a outros critérios para resolver SSP-MDPs com *dead ends*?

Os experimentos foram realizados em uma máquina com um processador de 4 núcleos a 2.60 GHz e 16 GB de memória.

[1] O código-fonte da implementação está disponibilizado em https://github.com/gcrispino/gubs-pddlgym.

8.1 Domínios, Instâncias e Heurísticas

Primeiramente, serão descritos os domínios e instâncias utilizados nos experimentos. Ademais, serão descritas as funções heurísticas utilizadas para estimar o valor de $V^*_\lambda(s, C)$ (a função valor não markoviana sensível ao risco) no eGUBS-AO*.

8.1.1 *Navigation*

O agente no domínio *Navigation* é um robô caminhando em uma grade de dimensões $N_x \times N_y$, em um estado $s \in \mathcal{S}$ com coordenadas (x, y). O objetivo é ir da posição (N_x, N_y) para $(N_x, 1)$. Na primeira e última linhas (em que $y \in \{1, N_y\}$) todas as ações (mover nas direções norte, sul, leste ou oeste) são determinísticas. Nas outras linhas, toda ação tem uma probabilidade de fazer o agente desaparecer, isto é, ir para um *dead end*. Essa probabilidade é mais próxima de 1 em colunas próximas à meta, e decaem para 0 conforme as colunas se distanciam da meta.

Na Figura 8.1 pode-se visualizar uma instância do domínio *Navigation* com uma grade de tamanho 10×5 (instância 7). A probabilidade do agente desaparecer é mostrada em cada célula. Note que os movimentos realizados em células mais escuras tem maior probabilidade de fazer o agente desaparecer.

0	0	0	0	0	0	0	0	0	g
0.02	0.15	0.23	0.32	0.44	0.52	0.64	0.74	0.84	0.93
0.02	0.15	0.23	0.32	0.44	0.52	0.64	0.74	0.84	0.93
0.02	0.15	0.23	0.32	0.44	0.52	0.64	0.74	0.84	0.93
0	0	0	0	0	0	0	0	0	s_0

Figura 8.1: *Instância 7 para o domínio Navigation com uma grade de tamanho* 10×5.

8.1.2 *River*

No domínio *River* (Freire e Delgado, 2017) o agente precisa atravessar um rio de um lado da sua margem até o outro. O ambiente é modelado como uma grade de dimensões $N_x \times N_y$, em que N_x é a sua largura, N_y, sua altura, e (x, y) são as coordenadas do agente em um estado $s \in \mathcal{S}$. No domínio também existe uma ponte, localizada no topo da grade, e a cachoeira, no fundo. Se o agente cair na cachoeira, não existe saída. O rio é localizado em qualquer estado restante (ou seja, entre as margens e embaixo da ponte). As possíveis ações são mover nas direções norte (↑), sul (↓), leste (→), ou oeste (←). As transições são determinísticas nas margens e na ponte. No rio existe uma probabilidade de 0.4 de que o agente caia um nível no eixo y, para qualquer ação executada nesses estados.

A Figura 8.2 mostra uma instância do domínio *River* com uma grade de tamanho 5×8 (instância 2). Uma política de exemplo é mostrada pelas setas em cada estado (exceto pelos estados da cachoeira e na meta).

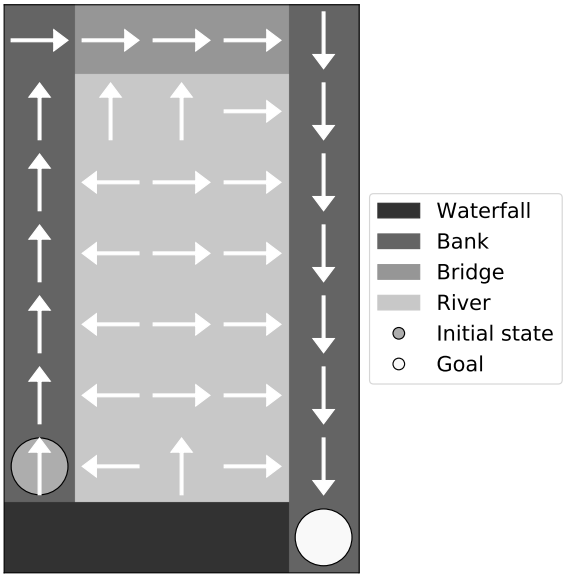


Figura 8.2: *Instância 2 do domínio River com uma grade de tamanho* 5×8.

8.1.3 *Triangle Tireworld*

No domínio *Triangle Tireworld*, o agente representa um carro que precisa ir de uma localização inicial a uma localização meta em um ambiente com formato de triângulo. O carro se move apenas se não tiver um pneu furado. Quando o carro se move a uma nova localização, existe uma probabilidade de 0.5 que o pneu fique furado após se locomover. Se isso acontecer e a nova localização tem um pneu disponível, a ação para trocar o pneu furado por esse pneu disponível pode ser tomada tal que o veículo possa se movimentar novamente. Caso contrário, o processo alcança um *dead end*, em que o carro não pode mais se mover.

8.1.4 Heurística para Estimar $V^*_\lambda(s, C)$

Para introduzir a heurística h_{sp} que estima o valor da função $V^*_\lambda(s, C)$ em estados aumentados (s, C) utilizados pelo GUBS nos experimentos, são feitas as seguintes definições. Seja G_d o grafo em que cada nó (x, y) representa uma localização em um problema e existe um arco de (x, y) para (x', y') se (x', y') é adjacente a (x, y) nesse problema; $(x(s), y(s))$ é a localização do agente no estado $s \in S$, e (x_g, y_g) é a localização da meta. A função heurística h_{sp} é definida como:

$$h_{sp}(s) = e^{\lambda d(s)},$$

em que $d(s)$ é a distância de $(x(s), y(s))$ para (x_g, y_g) no grafo G_d. h_{sp} utiliza $d(s)$ como uma estimativa para o custo esperado de alcançar a meta a partir de (s, C). $d(s)$ é pelo menos tão pequeno quanto o custo esperado para meta ótimo de s, o que também é verdade para qualquer estado aumentado $(s, C), \forall C \geq 0$. Já que $d(s)$ (que é utilizado como exponente em h_{sp}) é menor ou igual que o custo esperado ótimo para a meta de s e $\lambda < 0$, $h_{sp}(s) \geq V^*_\lambda(s, C)$, o que faz $h_{sp}(s)$ ser um valor admissível.

8.1.5 Instâncias

Primeiro, por conveniência, já que $\mathcal{W}(s, a)$ e consequentemente $W(s)$, C_{max} e $\bar{C}_{max}(s)$ podem ser negativos, C^+_{max} e $\bar{C}^+_{max}(s)$ são definidos da seguinte maneira:

$$C^+_{max} = \max\{0, C_{max}\}, \tag{8.1}$$

$$\bar{C}^+_{max}(s) = \max\{0, \bar{C}_{max}(s)\}, \forall s \in \mathcal{S}. \tag{8.2}$$

A Tabela 8.1 lista todas instâncias de todos domínios para os quais experimentos foram realizados. A terceira coluna mostra o número de estados alcançáveis a partir do estado inicial ($|\mathcal{S}_{reach}(s_0)|$). A quarta coluna mostra o valor $C^+_{maxinst}$ para cada instância, o qual é o valor máximo de C^+_{max} observado considerando todos valores dos parâmetros λ e K_g utilizados nos experimentos para essa instância específica. Por exemplo, o valor de $C^+_{maxinst}$ para a instância 2 do domínio *Navigation* é 0, o que significa que para cada experimento realizado nessa instância, o valor de C_{max} foi menor ou igual a 0. Isso indica que para cada configuração dessa instância o valor da política ótima sob o critério eGUBS é o valor da política ótima lexicográfica sensível ao risco a partir do estado inicial. A quinta coluna mostra $\hat{n}_{ext}(s_0) = |\mathcal{S}_{reach}(s_0)| \times C^+_{maxinst}$, isto é, o número máximo de estados aumentados processados para uma dada instância. Finalmente, a última coluna mostra $\bar{C}^+_{maxinst}(s_0)$, que é o valor máximo de $\bar{C}^+_{max}(s_0)$ observado considerando todos os valores dos parâmetros.

8.2 Desempenho dos Algoritmos eGUBS-VI e eGUBS-AO*

Os resultados reunidos e mostrados na presente seção são os seguintes: os valores de C^+_{max} (utilizado em eGUBS-VI) e $\bar{C}^+_{max}(s)$ (utilizada em eGUBS-AO*), número de estados armazenados em memória, número de atualizações e tempo de computação para ambos algoritmos eGUBS-VI e eGUBS-AO*.

Para melhor visualização dos resultados, cada gráfico apresentado nas seções 8.2.2, 8.2.3 e 8.2.4 foi limitado a conter dados de apenas 3 instâncias de cada domínio: instâncias 8, 9 e 10 do domínio *Navigation*; instâncias 3, 4 e 5 do domínio *River*; e instâncias 1, 2 e 3 do domínio *Tireworld*. Para todos esses experimentos os valores dos parâmetros K_g e λ são variados. Ademais, todos os valores mostrados nos gráficos foram normalizados da seguinte maneira: eles foram divididos por um valor base, considerado como o menor valor obtido pelo algoritmo eGUBS-VI para uma instância e parâmetro variado (K_g ou λ) específicos. Isso ajuda em visualizar os resultados para instâncias que podem ter valores

Domínio	Instância	$\lvert S_{reach}(s_0)\rvert$	$C^+_{maxinst}$	$\hat{n}_{ext}(s_0)$	$\bar{C}^+_{maxinst}(s_0)$
Navigation	1	13	0	13	0
	2	16	0	16	0
	3	21	237	4977	233
	4	31	623	19313	619
	5	31	413	12803	406
	6	41	786	32226	778
	7	51	1319	67269	1310
	8	61	817	49837	803
	9	81	2820	228420	2802
	10	101	2613	263913	2594
	11	201	2637	530037	2617
	12	451	6056	2731256	6032
	13	601	6095	3663095	6071
River	1	25	296	7400	293
	2	40	587	23480	580
	3	75	1245	93375	1231
	4	100	816	81600	807
	5	200	1698	339600	1674
Tri. Tireworld	1	42	59	2478	59
	2	946	59	55814	57
	3	19562	59	1154158	55

Tabela 8.1: *Informações para cada instância dos domínios utilizados nos experimentos.*

consideravelmente diferentes de números de estados, o que pode levar a resultados de diferentes magnitudes, dificultando na análise deles no mesmo gráfico. Como um exemplo do método de normalização utilizado, considere o seguinte caso: o menor valor de número de atualizações obtido para o eGUBS-VI na instância 8 do domínio *Navigation* ao variar o parâmetro K_g foi 9943, quando $K_g = 1$ ($\log_{10}(K_g) = 0$). Logo, todos os valores resultantes para essa instância quando K_g está sendo variado em ambos algoritmos serão divididos por 9943 quando mostrados no gráfico específico. Note que, diferentemente de outros métodos de normalização, valores maiores que 1 podem aparecer como um resultado desse método, no caso de um valor resultante específico (o dividendo) ser maior que o valor base (o divisor).

8.2.1 Configuração

Os valores dos parâmetros λ e K_g utilizados para os experimentos dessa seção em cada domínio são listados na Tabela 8.2. Esses valores foram escolhidos para fornecer um intervalo razoável dos valores de $\bar{C}^+_{max}(s_0)$ e C^+_{max}, os quais afetam o número de estados aumentados processados pelos algoritmos e, consequentemente, o desempenho deles. O número de níveis de expansão (n_{exp} nos algoritmos 10 e 11) é fixado em 5 para cada execução do algoritmo eGUBS-AO*. Finalmente, o valor de ϵ utilizado para ambos algoritmos foi de 10^{-8}.

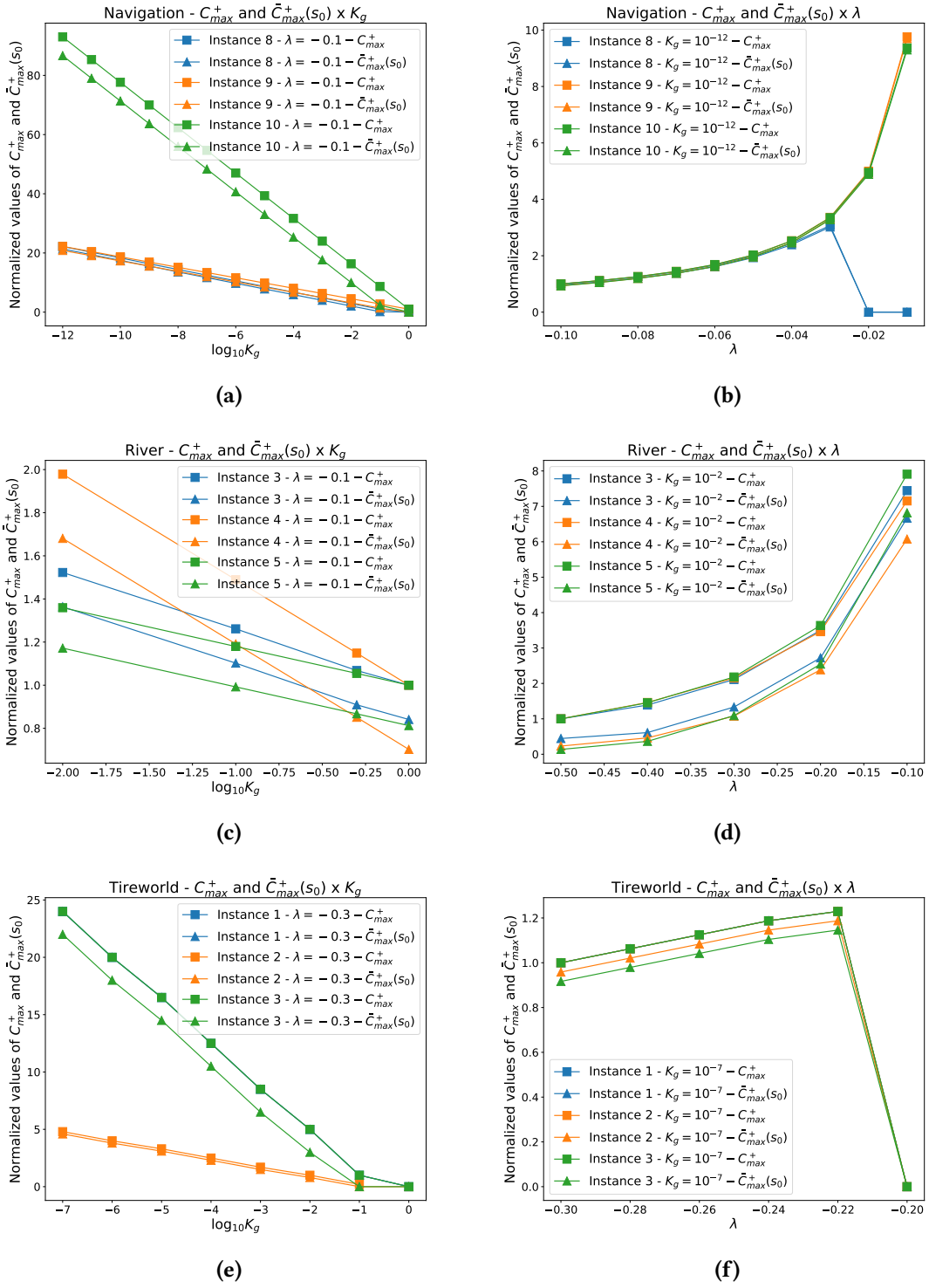


Figura 8.3: *Valores normalizados de C^+_{max} e $\bar{C}^+_{max}(s_0)$ por diferentes valores de λ e K_g.*

Domínio	Valores de λ	Valores de K_g
Navigation	$\{-0.1, -0.09, -0.08, -0.07, -0.06, -0.05, -0.04 -0.03, -0.02, -0.01\}$	$\{10^{-12}, 10^{-11}, 10^{10}, 10^{-9}, 10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$
River	$\{-0.5, -0.4, -0.3, -0.2, -0.1\}$	$\{0.01, 0.1, 0.5, 1\}$
Tri. Tireworld	$\{-0.3, -0.28, -0.26, -0.24, -0.22, -0.2\}$	$\{10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$

Tabela 8.2: *Valores de λ e K_g utilizados nos experimentos.*

8.2.2 $\bar{C}^{+}_{max}(s_0)$ e C^{+}_{max}

Os valores de $\bar{C}^{+}_{max}(s_0)$ e C^{+}_{max} obtidos como resultado dos experimentos podem ser visualizados na Figura 8.3. Na coluna da esquerda esses valores são mostrados para os domínios experimentados variando os valores do parâmetro K_g, e na coluna da direita, variando os valores de λ.

Para as instâncias dos domínios utilizados nos experimentos, os valores observados de $\bar{C}^{+}_{max}(s_0)$ são sempre menores ou iguais aos respectivos valores de C^{+}_{max}, o que segue o que é demonstrado pelo Corolário 5. Os valores de $\bar{C}^{+}_{max}(s_0)$ e C^{+}_{max} para cada instância dos domínios tende a aumentar tanto quando os valores de K_g diminuem quanto quando os valores de λ aumentam. Note que o valor de C^{+}_{max} vai para 0 a partir de $\lambda = -0.02$ (Figura 8.3b) na instância 8 do domínio *Navigation* e a partir de $\lambda = -0.2$ (Figura 8.3f) em todas instâncias do domínio *Triangle Tireworld*. Como mencionado na Seção 6.3, o valor resultante da Equação 6.5 pode ser menor ou igual a 0, o que para o cenário de custos positivos significa que a partir de um custo acumulado de 0 a política ótima para esse estado específico já é estacionária. Logo, ambos valores de C_{max} e $\bar{C}_{max}(s_0)$ podem ser menores ou iguais a 0, e nesse caso C^{+}_{max} e $\bar{C}^{+}_{max}(s_0)$ serão iguais a 0, respectivamente. As diferenças entre C^{+}_{max} e $\bar{C}^{+}_{max}(s_0)$ geralmente não mudam muito quando os parâmetros são variados e a diferença é pequena. A maior dessas diferenças que foi observada nos experimentos foi de 24 na instância 5 do domínio *River*. Para esse domínio, isso pode indicar que essa diferença depende no domínio em si, e não nesses parâmetros.

Para ilustrar melhor a distribuição dos valores de $\bar{C}^{+}_{max}$ para estados diferentes de s_0, a Figura 8.4 mostra um mapa de calor dos valores dessa função para todos estados da instância 7 do domínio *Navigation*, para parâmetros $\lambda = -0.02$ e $K_g = 10^{-12}$. Para essa instância, o único estado que tem o valor de $W(s) > 0$ é o estado cuja localização $(3, 3)$, para o qual $W(s) = 1319$. Pelo mapa de calor, se pode observar que esse valor é propagado nos valores de $\bar{C}^{+}_{max}$ nos estados vizinhos, e daí vão sendo subtraídos por 1, já que todas ações nesse domínio tem custos unitários. Então, em s_0 (na localização $(5, 10)$), $\bar{C}^{+}_{max}(s_0) = 1310$. Também note que para s_g (na localização $(1, 10)$), $\bar{C}^{+}_{max}(s_g) = 0$ por definição.

8.2.3 Número de Estados Armazenados

A presente seção contém uma análise no número de estados armazenados em memória pelos algoritmos. Esse número corresponde à soma do número de estados de cada instância do domínio em específico (que são sempre armazenados quando LexicográficoSensívelA-

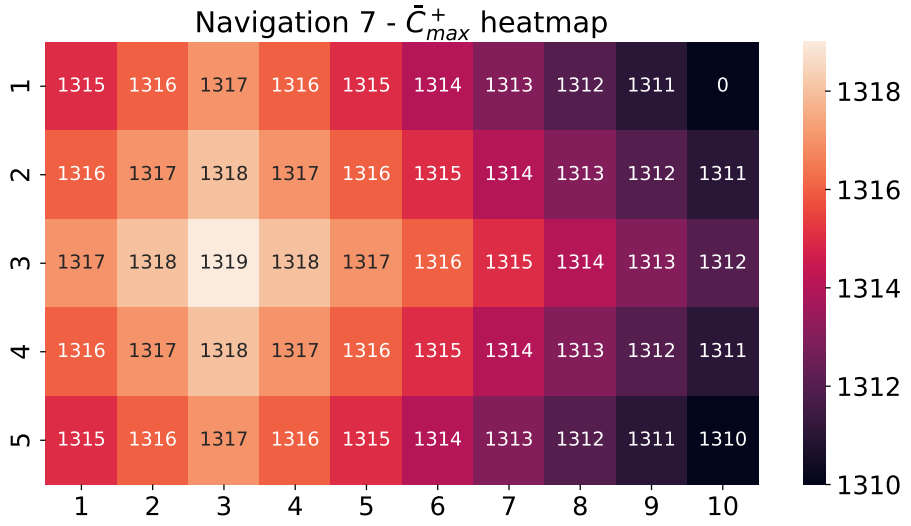


Figura 8.4: *Mapa de calor dos valores de $\bar{C}^{+}_{max}$ para estados da instância 7 do domínio Navigation.*

oRiscoIV é executado), e o número de estados aumentados que são salvos em memória durante a execução dos algoritmos eGUBS-VI e eGUBS-AO*.

A Figura 8.5 mostra esse número para cada domínio em cada valor de K_g e λ utilizados. Os valores de número de estados armazenados para cada instância dos domínios tende a crescer tanto quando os valores de K_g diminuem quanto quando os valores de λ aumentam. Isso acontece em geral em função do aumento dos valores de C^{+}_{max} e $\bar{C}^{+}_{max}(s_0)$ nesses casos, o que leva ao processamento de mais estados aumentados. Note que para algumas instâncias o número normalizado de estados nos gráficos diminui para valores muito pequenos a partir de certos pontos de valores de λ. Isso acontece nesses casos porque o valor de C^{+}_{max} vai para 0 como mostrado nas Figuras 8.3b e 8.3f. Ademais, note que os números normalizados de estados armazenados em memória observados ao executar o eGUBS-VI chegam a ser em torno de 10 vezes maiores que os mesmos números para o eGUBS-AO* na Figura 8.5b para as instâncias 9 e 10. [2]

Em geral, o algoritmo eGUBS-AO* precisa armazenar menos estados para encontrar a política ótima se comparado ao eGUBS-VI. eGUBS-AO* é capaz disso principalmente pela busca heurística que realiza, e secundariamente pelos valores de $\bar{C}_{max}$ calculados no algoritmo $\bar{C}_{max}$-Alcançáveis para cada estado alcançável de s_0 serem menores que os valores de C_{max}, valores esses que são utilizados no eGUBS-VI. Por outro lado, o eGUBS-VI precisa armazenar o conjunto de estados alcançáveis a partir do estado inicial na implementação utilizada nos experimentos (vide Figura 8.5). Por exemplo, quando o valor de C_{max} é alto, o eGUBS-VI precisa processar C_{max} vezes um conjunto no mínimo tão grande quanto o número de estados alcançáveis, enquanto que o eGUBS-AO* pode não precisar visitar estados aumentados com custos acumulados próximos a esse valor se caminhos para esses estados não se mostraram promissores pela busca heurística, o que faz o algoritmo não precisar explorá-los nesse caso.

[2] Os números de estados das instâncias 9 e 10 parecem muito próximos de 0 em função das escalas diferentes de cada linha do gráfico.

A maior diferença desses valores foi observada na instância 9 do domínio *Navigation*, para os valores de $K_g = 1$ e $\lambda = -0.1$, para os quais o número de estados é em torno de 1700 vezes menor no eGUBS-AO* (82 estados armazenados) que no eGUBS-VI (145740 estados armazenados) [3].

No pior caso o número de estados armazenados é o mesmo para ambos algoritmos eGUBS-AO* e eGUBS-VI.

8.2.4 Número de Atualizações

O número de atualizações obtido de cada instância é mostrado na Figura 8.6. É considerada como uma atualização cada vez que o valor de um estado é atualizado. No eGUBS-VI, isso é o caso quando tanto o LexicográficoSensívelAoRiscoIV quanto MelhoraPolíticaEstacionária executam a linha 8 no caso do primeiro, e a linha 9, no caso do segundo. No eGUBS-AO*, uma atualização é contada quando tanto a linha 8 é executada no algoritmo LexicográficoSensívelAoRiscoIV, quanto quando a linha 2 é executada no algoritmo eGUBS-VI-mod.

Para as instâncias do domínio *Navigation*, o eGUBS-AO* chega em um ponto estável de desempenho conforme o número de estados aumentados cresce quando valores de K_g diminuem, enquanto que o número de atualizações no eGUBS-VI continuam aumentando (Figura 8.6a). Ao variar os valores de λ (Figura 8.6b), os resultados não são exatamente os mesmos. Na maioria dos casos, eGUBS-AO* realiza um número menor de atualizações que o eGUBS-VI, e para ambos algoritmos o número de atualizações cresce conforme os valores de λ também crescem.

Por outro lado, eGUBS-AO* em geral oferece pior desempenho que o eGUBS-VI em instâncias do domínio *River* (figuras 8.6c e 8.6d). Isso pode acontecer em função das possibilidades de caminhos para o agente quando o rio é atravessado nesse domínio, caso em que não existe uma probabilidade de ir diretamente para um *dead end* como no *Navigation*, por exemplo. Ao invés disso, o agente pode cair um nível no rio quando realiza a tentativa de nadar por ele. Isso pode dificultar com que estados não promissores sejam descartados logo no início da busca, o que pode por sua vez requerer que uma função heurística mais informativa fosse utilizada para não ter que executar um maior número de atualizações que o realizado no eGUBS-VI, por exemplo.

Em todas instâncias do *Triangle Tireworld*, um menor número de atualizações (figuras 8.6e e 8.6f) é geralmente observado em resultados do eGUBS-AO* quando variando valores de ambos parâmetros K_g e λ, em comparação com o eGUBS-VI.

8.2.5 Tempo

A Figura 8.7 mostra o tempo médio de CPU ao executar os algoritmos eGUBS-VI e eGUBS-AO* por 10 rodadas para os domínios *Navigation*, *River*, e *Triangle Tireworld*. O eixo x representa as instâncias testadas, e o eixo y representa o tempo médio de execução em

[3] Note novamente que a Figura 8.5 mostra o número de estados armazenados em memória dividido pelo menor valor encontrado para o algoritmo eGUBS-VI para cada instância. Para a instância 9 do domínio *Navigation* e valores de $\lambda = -0.1$ e $K_g = 1$ o seu valor normalizado é de $82/145740 = 0.0005$.

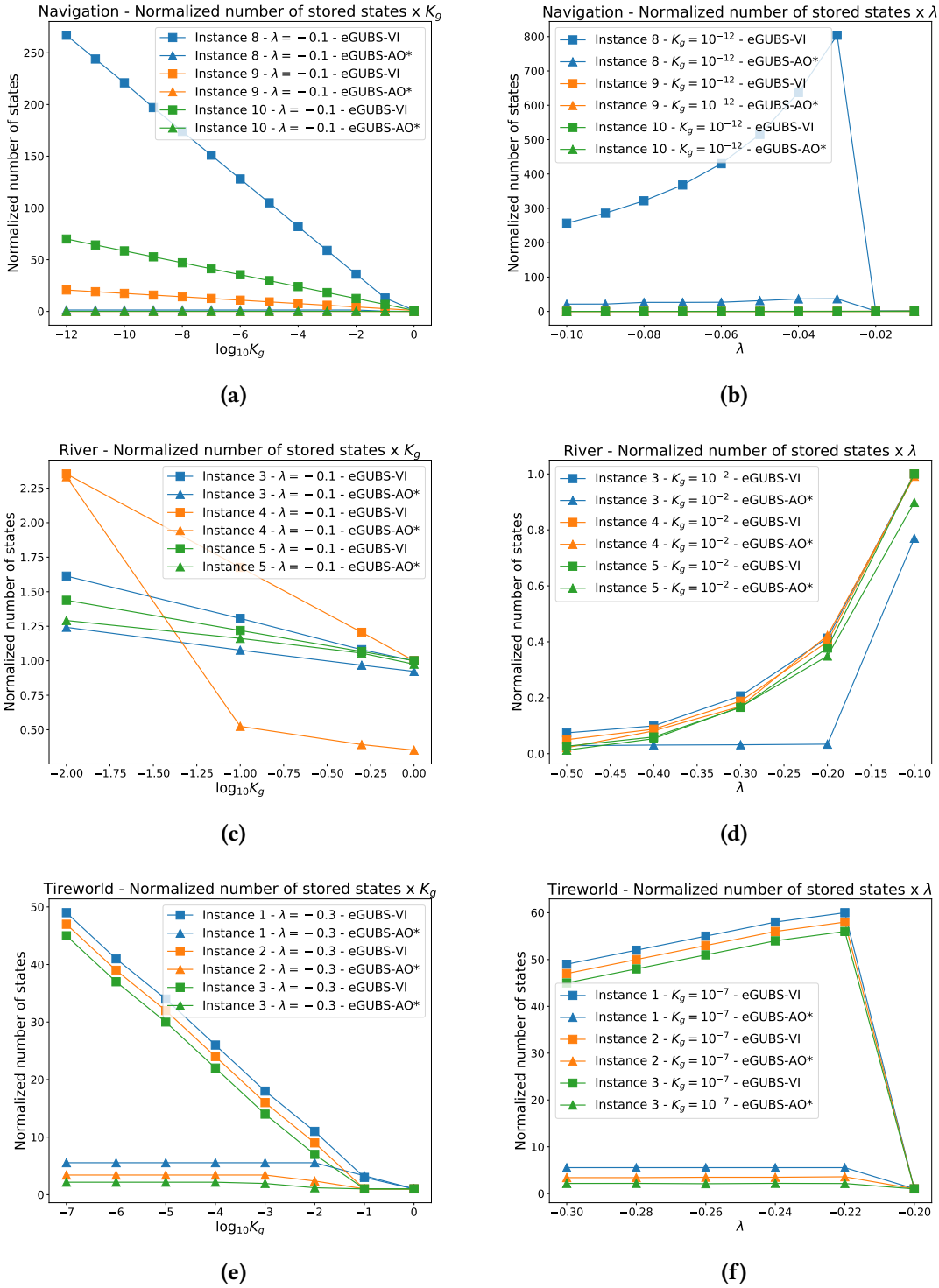


Figura 8.5: *Número normalizado de estados armazenados em memória por diferentes valores de λ e K_g.*

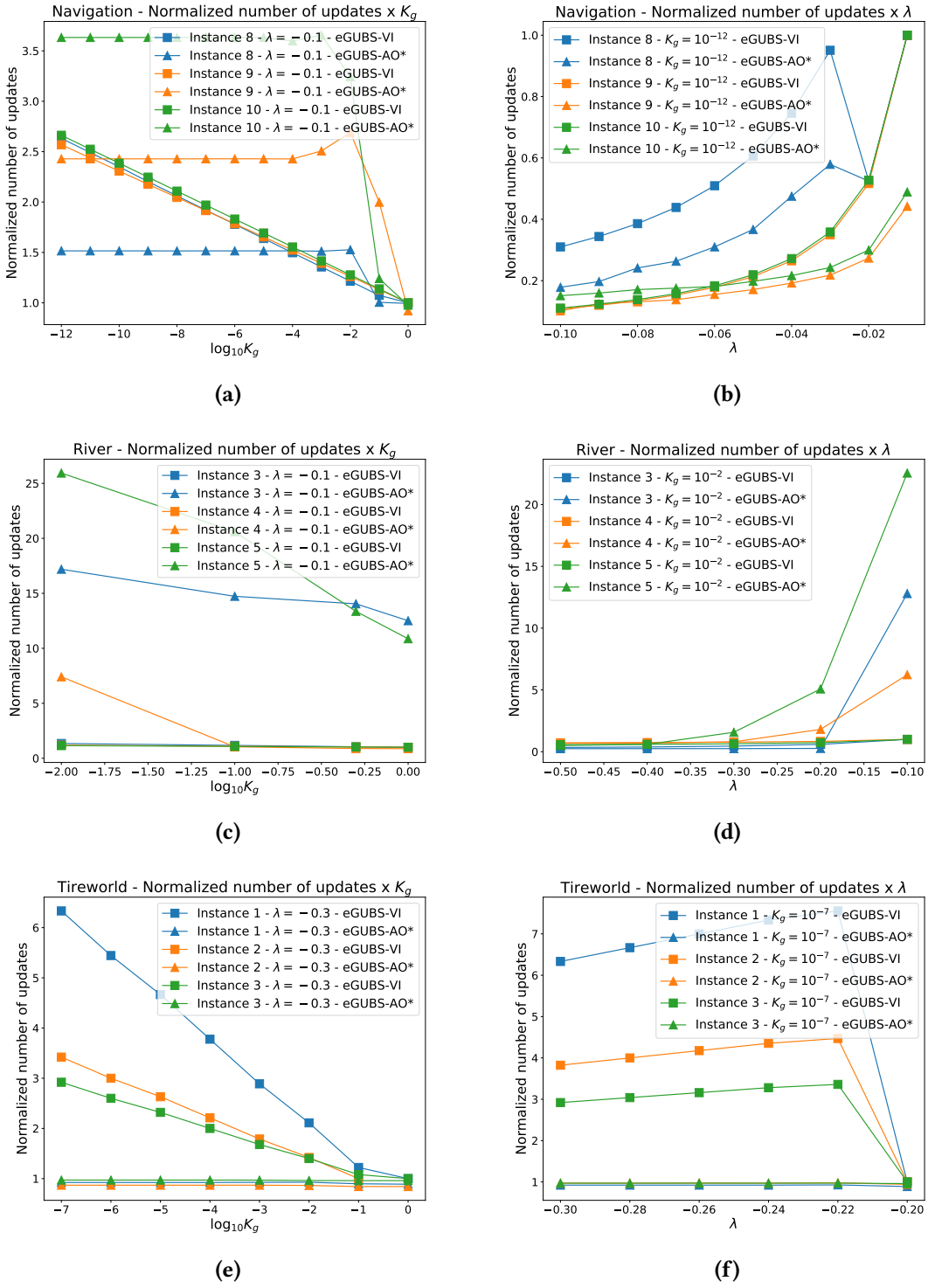


Figura 8.6: *Número normalizado de atualizações realizadas por diferentes valores de λ e K_g.*

segundos em escala logarítmica na base 2 ($\log_2$). Um limite de 4500 segundos foi utilizado para cada rodada. Note que, já que existem instâncias para as quais os experimentos são executados em menos de um segundo (por exemplo, as primeiras 4 instâncias do domínio *Navigation*), calcular o valor do $\log_2$ desses resultados resultaria em valores negativos. Por causa disso, um valor de deslocamento foi somado a cada resultado de cada instância para evitar a visualização de barras invertidas em função da existência de valores negativos. O valor utilizado para isso foi de 4 unidades para todos domínios. Além disso, os valores de desvio padrão são representados por uma marca no topo de cada barra dos gráficos.

Para essas execuções, os parâmetros λ e K_g foram fixados para cada domínio. Os valores dos pares (λ, K_g) utilizados para os domínios *Navigation*, *River*, e *Triangle Tireworld* são, respectivamente, $(-0.01, 10^{-12})$, $(-0.1, 0.01)$ e $(-0.3, 10^{-7})$.

Os resultados são em geral correlacionados com os resultados de número de atualizações, para as instâncias referenciadas na Seção 8.2.4, tendo o tempo médio de execução para o eGUBS-AO* menor que o do eGUBS-VI, exceto para o domínio *River*, para o qual o eGUBS-VI teve um melhor desempenho. Outra exceção é a 13ª instância do domínio *Navigation*, para a qual o tempo médio de execução do eGUBS-VI foi menor que o do eGUBS-AO*. Além disso, para instâncias menores, os resultados entre os dois algoritmos podem não ter diferença significativa. Isso é observável em sua maioria nas instâncias menores do *Navigation*.

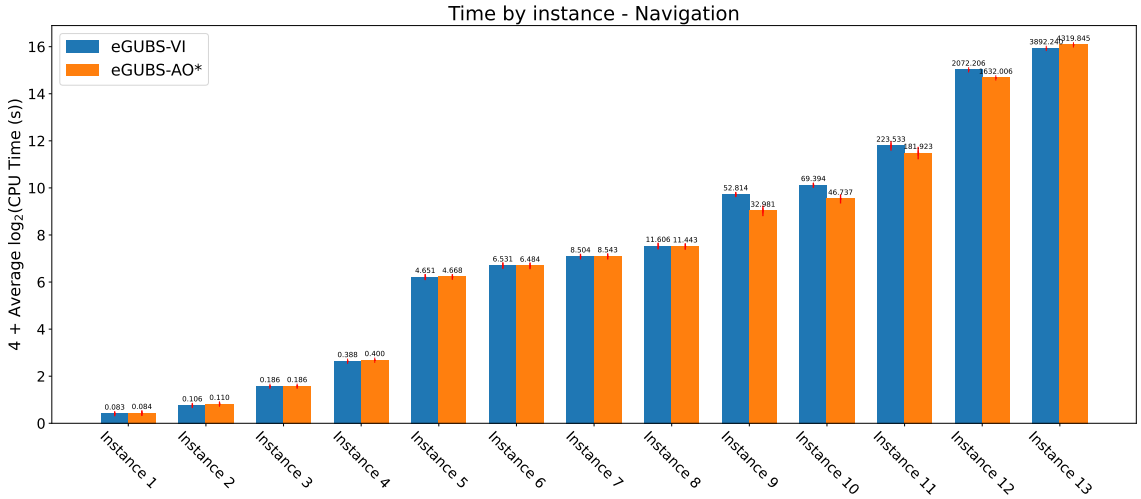
A seguir são mencionados mais detalhes sobre o desempenho de ambos algoritmos, para os valores fixados de λ e K_g como definidos anteriormente e considerando a média de cada instância sobre as 10 execuções realizadas. Para o domínio *Navigation*, o eGUBS-AO* foi até 1.6 vez mais rápido que o eGUBS-VI, o que aconteceu na instância 9. Na instância 13, o eGUBS-AO* foi até aproximadamente 1.11 vez mais lento que o eGUBS-VI no seu pior caso para esse domínio. No domínio *River*, no seu melhor caso, o eGUBS-AO* foi até 1.15 vez mais rápido que o eGUBS-VI na instância 1. No seu pior caso, ele foi até 6.52 vezes mais lento que o eGUBS-VI na instância 3. No domínio *Triangle Tireworld*, no seu melhor caso, o eGUBS-AO* foi até 6.77 vezes mais rápido que o eGUBS-VI na instância 1. No seu pior caso, o eGUBS-AO* foi até 4.4 vezes mais lento que o eGUBS-VI na instância 3.

8.2.6 Considerações sobre os Resultados

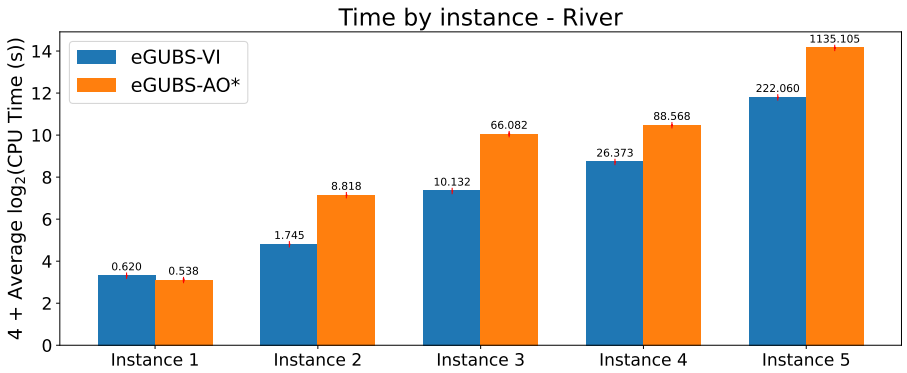
Resumindo, os resultados obtidos indicam que o eGUBS-AO* em geral terá um melhor desempenho que o eGUBS-VI se uma heurística com informação suficiente está disponível, sendo uma escolha mais adequada nesses casos. Caso contrário, a busca heurística pode não ser eficiente o bastante e produzir um *overhead* de modo que o seu desempenho não será pelo menos tão bom quanto o eGUBS-VI, fazendo com que o último seja uma melhor escolha.

Além disso, baseado nos resultados, o algoritmo eGUBS-AO* em geral teve melhor desempenho que o eGUBS-VI conforme os valores dos parâmetros K_g e λ ficavam respectivamente menores e maiores. Isso aconteceu porque em geral nessas situações os valores de $\bar{C}^+_{max}(s_0)$ e C^+_{max} cresciam, o que faz com que o número de estados aumentados também aumente, especialmente no caso do eGUBS-VI.

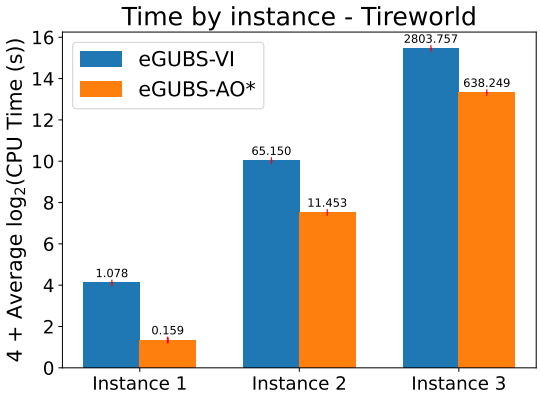
Ademais, foi observado que o eGUBS-AO* em geral armazena menos estados em



(a) *Navigation*



(b) *River*



(c) *Triangle Tireworld*

Figura 8.7: *Tempo médio de CPU obtido para resolver o critério eGUBS para ambos algoritmos eGUBS-VI e eGUBS-AO* nas 10 rodadas para os domínios Navigation, River, e Triangle Tireworld.*

memória que o eGUBS-VI, tendo no pior caso ambos algoritmos armazenando o mesmo número de estados. Tendo isso em vista, em aplicações para domínios muito grandes, o eGUBS-AO* pode ser uma melhor escolha, dependendo do seu desempenho no número de atualizações.

8.3 Comparação entre eGUBS e Outros Critérios

Na presente seção, serão mostrados resultados de experimentos feitos em instâncias selecionadas dos domínios para comparar o critério eGUBS com outros critérios utilizados para resolver SSP-MDPs com *dead ends*.

Mais especificamente, os critérios utilizados para comparação são fSSPUDE, critério de custo descontado e MCMP. Para eles, os seus parâmetros (D, γ, e p_{max}, respectivamente) [4] foram variados do menor valor para o maior no conjunto de valores escolhidos, exceto para os valores de p_{max} para o critério MCMP, que foram variados do maior valor para o menor. Os intervalos de valores passados como parâmetro para cada critério foram os seguintes: para a instância 10 do domínio *Navigation*, D recebeu 11 valores de 40 a 50, γ recebeu 10 valores de 0.5 a 0.999, e p_{max} recebeu 20 valores de 0.01 a 0.851. Para a instância 5 do domínio *River*, D recebeu 9 valores de 1 a 1500, γ recebeu 9 valores de 0.1 a 0.999, e p_{max} recebeu 20 valores de 0.01 a 1. Por fim, essas políticas foram avaliadas sob o critério eGUBS e os valores resultantes disso foram então comparados com o valor da política ótima sob o eGUBS.

Para os critérios fSSPUDE e de custo descontado, foram utilizados algoritmos de iteração de valor com as respectivas modificações necessárias (uso da penalidade e fator de desconto). Para o MCMP, foi utilizado o *solver* CBC (Forrest *et al.*, 2022) para resolver a sua formulação de programação linear enumerando suas variáveis para todos os estados. No caso do eGUBS, foi utilizado o algoritmo eGUBS-VI.[5]

Por essa comparação, pode-se avaliar se diferentes critérios podem encontrar políticas ótimas para o eGUBS apenas ao escolher parâmetros apropriados e, se não, o quão próximo do valor ótimo eles conseguem chegar. As figuras 8.8a e 8.8b mostram esses resultados para a instância 10 do domínio *Navigation* e instância 5 do domínio *River*, respectivamente. Os valores de K_g e λ para a instância 10 do domínio *Navigation* foram fixados em 10^{-12} e -0.1, e para a instância 5 do domínio *River*, 0.01 e -0.1, respectivamente.

Para ambas configurações de domínios, o único critério que foi capaz de encontrar uma política que o valor chegou próximo do ótimo sob o eGUBS foi o critério de custo descontado para o valor de $\gamma = 0.9$ em ambas instâncias. Como não existe um procedimento claro para escolher um certo valor para os parâmetros desses critérios com alguma garantia dos valores das políticas resultantes sob o eGUBS, um conjunto indefinido de valores precisa

[4] Apesar de originalmente considerar apenas políticas que maximizam probabilidade à meta, o critério MCMP pode receber outro valor p_{max} na sua formulação de programação linear para encontrar uma política com probabilidade à meta igual a p_{max}. Esse é o parâmetro variado para o MCMP nos experimentos descritos nessa seção.

[5] O código-fonte da implementação utilizada para os experimentos dessa seção pode ser encontrado em https://github.com/GCrispino/ssp-deadends.

ser variado para esses parâmetros com esse propósito. Em algumas situações, isso pode ser impraticável se é desejado ter garantias no valor das políticas resultantes sob o critério eGUBS.

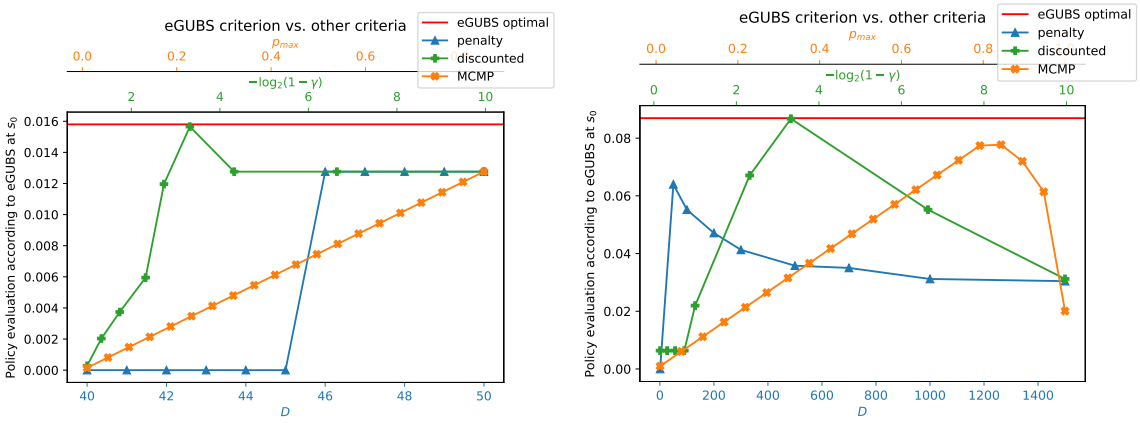
Por outro lado, se apenas uma aproximação é suficiente, pode ser o caso de que poucos valores de cada parâmetro precisem ser testados para cada critério para que se aproxime do valor ótimo do critério eGUBS. Além disso, note novamente que para os critérios fSSPUDE e de custo descontado, existem garantias de que no limite o critério MAXPROB é atingido (para o critério MCMP, isso acontece por padrão). Na Figura 8.8a ambos critérios fSSPUDE e de custo descontado obtém a política que maximiza a probabilidade à meta no último ponto testado ($D = 50, \gamma = 0.999$), enquanto que na Figura 8.8b apesar de valores próximos desse máximo serem alcançados, o valor exato não é obtido ($D = 1500, \gamma = 0.999$).

É também importante ressaltar que um tomador de decisão pode não ter recursos suficientes para resolver diferentes critérios para variados valores de cada um de seus parâmetros, a fim de encontrar uma aproximação razoável ao valor ótimo do critério eGUBS, especialmente sem garantias. No caso dos experimentos descritos na presente seção, para a instância 10 do domínio *Navigation* o tempo gasto para computar o valor ótimo do critério eGUBS foi suficiente para resolver os critérios de custo descontado, fSSPUDE, e MCMP para respectivamente 5, 8 e 6 valores diferentes de seus parâmetros (γ de 0.5 a 0.9, D de 40 a 48, e p_{max} de 0.851 a 0.585, em que os valores finais não são incluídos). Para a instância 5 do domínio *River*, o tempo foi suficiente para resolver os critérios base para 6, 2, e 3 valores de parâmetros diferentes (γ de 0.1 a 0.9, D de 1 a 100, e p_{max} de 1 a 0.844, em que os valores finais não são incluídos).

Outro aspecto importante ao comparar esses critérios é a natureza das políticas geradas por cada um deles. Já foi mencionado que a solução ótima para o critério GUBS é uma política não markoviana, enquanto que todos os outros critérios analisados no presente trabalho tem políticas estacionárias como soluções. Por causa dessa característica, uma pergunta importante a se fazer é de se políticas estacionárias ótimas sob critérios diferentes do GUBS são capazes de aproximar políticas não markovianas que são ótimas sob esse critério.

Essa análise foi realizada no contexto dos experimentos aqui apresentados. Para o caso da instância 10 do domínio *Navigation*, a política ótima sob o critério eGUBS pode ser obtida por critérios que geram apenas políticas estacionárias. Isso acontece porque nesse domínio a política ótima não contém ciclos, já que o agente sempre vai para frente e, quando cruza os estados com uma probabilidade positiva de desaparecer, ele irá ter sucesso em atravessá-los ou desaparecer, o que finaliza o processo. A política ótima para o eGUBS, nesse caso, é então na prática estacionária porque todos estados serão alcançados apenas uma vez (isto é, existirá um único custo acumulado C para cada s nos estados aumentados (s, C) alcançáveis do estado inicial) quando ela é seguida. Nos experimentos realizados, apenas quando $\gamma = 0.9$ para o critério de custo descontado, a política resultante teve um valor próximo ao da política ótima sob o eGUBS. Para o domínio *River*, esse não é exatamente o caso. Algo que pode acontecer nesse domínio é que o agente pode tentar atravessar o rio mas cair alguns níveis, e a política ótima pode fazer o agente tentar voltar para a margem e tentar novamente em um nível mais alto. Uma das consequências desse fato é a de que uma política estacionária não pode representar uma política ótima

nesse contexto exatamente, já que um mesmo estado s pode ser alcançado com diferentes custos acumulados. No entanto, seu valor pode ser muito próximo ao da política ótima não markoviana e então não ter uma diferença notável em casos específicos. Isso acontece nos experimentos para o critério de custo descontado nesse mesmo domínio, quando o valor de γ é igual a 0.9. O valor da política ótima nessa configuração é muito próximo ao valor ótimo do eGUBS, mesmo a política sendo estacionária e esses valores não sendo exatamente iguais.



(a) *Instância 10 do domínio Navigation*

(b) *Instância 5 do domínio River*

Figura 8.8: *Comparação do valor da política ótima para o eGUBS com o valor das políticas ótimas para os critérios fSSPUDE, de custo descontado e MCMP. Para todos critérios exceto o eGUBS, cada parâmetro (D, γ, e p_{max}, respectivamente) foi variado, e o valor da política ótima para cada um desses critérios sob o critério eGUBS foi calculado no estado inicial.*

Capítulo 9

Considerações Finais

O presente trabalho fornece uma análise do critério GUBS, um critério que combina priorização de metas sobre históricos com Teoria da Utilidade Esperada, comparando-o com outros critérios da literatura que resolvem SSP-MDPs com *dead ends*, e também apresenta resultados teóricos referentes às suas propriedades e relações. Entre outros resultados, é demonstrado que o GUBS é o único critério que garante a propriedade de priorização de probabilidade à meta α-forte para $0 \leq \alpha \leq 1$, e que permite a realização de compromissos infinito-infinitesimais.

O critério eGUBS, uma instância do critério GUBS que utiliza uma função exponencial com um fator de risco negativo, também é analisado. A partir desse caso particular, é possível derivar uma equação para calcular a função valor de estados aumentados a partir da função valor de SSP-MDPs sensíveis ao risco. O presente trabalho introduz a propriedade de custos Custo-admissíveis para tratar do caso em que é garantido que a partir de um determinado custo acumulado, a política ótima sob o critério eGUBS é também ótima sob o critério lexicográfico sensível ao risco. Ao longo do trabalho são expostos resultados teóricos relacionados a esse conceito.

O algoritmo eGUBS-VI é um algoritmo existente que realiza o cálculo do C_{max} (um valor Custo-admissível de qualquer estado do processo) e realiza o cálculo da política ótima de trás para frente para estados aumentados a partir de uma política ótima do critério lexicográfico sensível ao risco obtido previamente. O presente trabalho propõe o eGUBS-AO*, um algoritmo de busca heurística que realiza uma busca do estado inicial guiada por uma função heurística, também tendo como entrada uma política ótima do critério lexicográfico sensível ao risco. A busca pode parar quando valores do custo acumulado de estados aumentados (s, C) alcançam $\bar{C}_{max}(s)$ (uma versão melhorada do C_{max}, um valor Custo-admissível de s introduzido no presente trabalho), mas também pode parar antes dependendo dos valores calculados durante a busca.

Experimentos foram realizados em 3 domínios: *Navigation*, *River* e *Triangle Tireworld*. Os seus resultados indicam que, para esses domínios, o algoritmo eGUBS-AO* pode em geral ter um melhor desempenho quando a função heurística é boa o suficiente. Ademais, quando o espaço de estados é muito grande para um algoritmo enumerativo como o eGUBS-VI, o eGUBS-AO* pode realizar a busca sem precisar enumerar todos os estados

aumentados entre o custo 0 e C_{max}. Em outros casos, o eGUBS-VI pode ser uma melhor escolha, já que ele geralmente incorre em menos *overhead* por não precisar manter grafos auxiliares para busca, como é feito no eGUBS-AO*.

9.1 Contribuições

Como o presente trabalho tem um forte aspecto de revisão da literatura, vários conceitos já presentes em trabalhos correlatos são aqui apresentados. Como parte disso, além de novas contribuições relacionadas às propriedades teóricas do GUBS e dos critérios relacionados, alguns resultados no formato de definições, corolários e teoremas são modificados com adições referentes aos conceitos de priorização de probabilidade à meta α-forte e da realização de compromissos infinito-infinitesimais são realizados ao longo do trabalho. Abaixo, segue a lista de todos esses pontos que foram modificados de trabalhos anteriores ($*$) ou introduzidos exclusivamente neste trabalho ($\diamond$):

Definições:

- Definição 3$^\diamond$ – **Priorização de probabilidade à meta α-forte**;
- Definição 4* – **Priorização de metas sobre históricos**;
- Definição 6* – **Critério lexicográfico sensível ao risco**.

Corolários:

- Corolário 1* – **Compromisso mínimo no critério GUBS quando existe uma política própria**;
- Corolário 2$^\diamond$ – **Preferência do critério GUBS considerando políticas com probabilidade à meta igual a 0**;
- Corolário 3$^\diamond$ – **Priorização de probabilidade à meta α-forte para o critério GUBS**;
- Corolário 4$^\diamond$ – **Priorização de probabilidade à meta α-forte para o critério eGUBS**;
- Corolário 6* – **Custo inteiro - política ótima**;
- Corolário 7* – **Função valor correta**.

Teoremas:

- Teorema 1$^\diamond$ – **Falta de compromissos infinito-infinitesimais para critérios que maximizam probabilidade à meta**;
- Teorema 2$^\diamond$ – **Garantia de priorização de probabilidade à meta 1-forte para critérios lexicográficos e MAXPROB**;
- Teorema 3$^\diamond$ – **Critério fSSPUDE não tem priorização de probabilidade à meta α-forte para $\alpha > 0$**;
- Teorema 4$^\diamond$ – **Critério de custo descontado não tem priorização de probabilidade à meta α-forte para $\alpha > 0$**;

- Teorema 5* – **Condições do critério GUBS**;
- Teorema 6* – **Compromisso mínimo no critério GUBS**;
- Teorema 8* – **Melhoria de política sob o critério eGUBS**;
- Teorema 13* – **Convergência do algoritmo LexicográficoSensívelAoRiscoIV**;
- Teorema 14* – **Custo racional – política ótima**;

Além disso, este trabalho propõe o algoritmo eGUBS-AO*. Além da definição e realização de experimentos, neste algoritmo são utilizados conceitos relacionados com os custos acumulados e suas relações com políticas ótimas sob o critério eGUBS. Esses conceitos são descritos nos seguintes pontos do trabalho:

- Definição 9° – $C_{stat}(s)$ **e políticas estacionárias ótimas sob o critério eGUBS**;
- Definição 10° – **Custo-admissibilidade de** s;
- Teorema 9* – C_{max} **é Custo-admissível de qualquer** $s \in \mathcal{S}$;
- Teorema 10° – $\bar{C}_{max}(s)$ **é Custo-admissível a partir de** s;
- Corolário 5° – $\bar{C}_{max}(s)$ **é um limite superior mais restrito sobre** $C_{stat}(s)$ **que** C_{max};
- Teorema 11° – **Definição recursiva de** $\bar{C}_{max}(s)$;
- Teorema 12° – **Valor de** $C_{stat}(s)$;
- Teorema 15° – **Otimalidade do algoritmo eGUBS-AO***.

9.2 Publicações

9.2.1 Algoritmo de *Monte Carlo Tree Search* para SSPs sob o critério GUBS

O trabalho *Algoritmo de Monte Carlo Tree Search para SSPs sob o critério GUBS* (G. Crispino *et al.*, 2020) propõe o algoritmo UCT-GUBS, uma adaptação do algoritmo de busca em árvore Monte Carlo UCT (Kocsis e Szepesvári, 2006; Keller e Eyerich, 2012) para o critério GUBS. O UCT-GUBS escolhe uma ação para um estado s ao realizar repetidas simulações (ou *rollouts*) a partir desse estado e, quando alguma condição de parada (como tempo ou horizonte limite) for atingida, a ação escolhida é a ação que maximiza a estimativa do valor do critério GUBS calculada a partir do seu valor computado para cada histórico obtido como resultado das simulações. A diferença principal do UCT-GUBS para uma implementação padrão do UCT está justamente nessa avaliação dos históricos. Nesse caso, ao invés de computar uma média incremental dos custos resultantes das simulações realizadas, é calculada a média dos valores dos históricos obtidos sob o critério GUBS.

Esse trabalho foi publicado no XVII Encontro Nacional de Inteligência Artificial e, além da definição do UCT-GUBS, contém os resultados de experimentos preliminares realizados utilizando esse algoritmo para o ambiente *Navigation* com uma implementação baseada no PROST (Keller e Eyerich, 2012; Keller e Helmert, 2013). Os resultados indicam que

UCT-GUBS é uma alternativa válida como um método *anytime* para resolver o critério GUBS. No entanto, experimentos adicionais seriam necessários para avaliar o UCT-GUBS em outros domínios, além de uma comparação desse algoritmo com os algoritmos eGUBS-VI e eGUBS-AO*. Além disso, pelo fato desse algoritmo tratar o problema como um MDP de horizonte finito, uma exploração sobre que tipos de resultados teóricos se obtém para o GUBS nessa configuração pode ser relevante.

9.2.2 *GUBS Criterion: Arbitrary Trade-offs between Cost and Probability-to-goal in Stochastic Planning based on Expected Utility Theory*

O trabalho *GUBS Criterion: Arbitrary Trade-offs between Cost and Probability-to-goal in Stochastic Planning based on Expected Utility Theory* (G. N. Crispino *et al.*, 2023) é um trabalho que reúne as contribuições dos trabalhos publicados em conferências disponíveis em Freire e Delgado, 2017 e Freire, Delgado e Reis, 2019, adicionando mais resultados teóricos, a introdução do algoritmo eGUBS-AO* e novos experimentos com os algoritmos eGUBS-VI e eGUBS-AO*.

9.3 Trabalhos Futuros

Trabalhos futuros podem incluir a adaptação de funções heurísticas baseadas em métodos de estado da arte, como a que é apresentada em F. Trevizan, Thiébaux e Haslum, 2018, que pode ser utilizada para melhorar o desempenho do eGUBS-AO*. Além disso, um estudo para a adaptação de métodos de busca heurística também pode ser feito para encontrar tanto a política ótima do critério lexicográfico sensível ao risco, quanto o valor de $\bar{C}_{max}(s)$. No primeiro caso, é necessário utilizar algum método como o FRET (Kolobov, D. S. Weld *et al.*, 2011) ou o i-dual (F. Trevizan, Thiébaux, Santana *et al.*, 2016) para encontrar os valores de $P_G(s)$ (juntamente com os valores de $V^*_\lambda(s)$) com busca heurística, já que um algoritmo clássico desse tipo como o LAO* ou LRTDP (Bonet e Geffner, 2003) não garante a solução ótima para essa função. No segundo caso, é necessário se investigar como seria possível computar valores de $\bar{C}_{max}(s)$ a partir de estimativas admissíveis. Trabalhos futuros podem também incluir a análise de procedimentos de transformação da função de custo quando a sua imagem está localizada nos números reais e a análise da magnitude dos erros introduzidos por tal procedimentos, já que nesse caso o algoritmo eGUBS-VI não garante encontrar uma solução ótima. Outro possível caso de trabalho futuro é o estudo do relacionamento entre o critério GUBS e de *(chance) constraint* SSPs (C-SSPs) (Altman, 1999; F. W. Trevizan *et al.*, 2017; Ono e B. C. Williams, 2008; Blackmore *et al.*, 2011; Ono, Kuwata *et al.*, 2012; Thiébaux, B. Williams *et al.*, 2016). Pode-se utilizar uma restrição no valor de α e construir um C-SSP baseado nisso. Nesse caso, um novo critério que garante priorização de probabilidade à meta α-forte é obtido. No entanto, esse novo critério não é equivalente ao critério GUBS. A análise desse novo critério e dos seus prós e contras necessita de um estudo mais detalhado.

Referências

[Altman 1999] Eitan Altman. *Constrained Markov decision processes: stochastic modeling*. Routledge, 1999 (citado na pg. 90).

[Bellman *et al.* 1957] R. Bellman, Rand Corporation e Karreman Mathematics Research Collection. *Dynamic Programming*. Rand Corporation research study. Princeton University Press, 1957 (citado nas pgs. 8, 10).

[Bertsekas 1995] Dimitri Bertsekas. *Dynamic Programming and Optimal Control*. Belmont, Mass: Athena Scientific, 1995 (citado nas pgs. 1, 6).

[Blackmore *et al.* 2011] Lars Blackmore, Masahiro Ono e Brian C Williams. "Chance-constrained optimal path planning with obstacles". Em: *IEEE Transactions on Robotics* 27.6 (2011), pgs. 1080–1094 (citado na pg. 90).

[Bonet e Geffner 2003] Blai Bonet e Hector Geffner. "Labeled RTDP: improving the convergence of real-time dynamic programming." Em: *ICAPS*. Vol. 3. 2003, pgs. 12–21 (citado na pg. 90).

[G. Crispino *et al.* 2020] Gabriel Crispino, Valdinei Freire e Karina Valdivia Delgado. "Algoritmo de Monte Carlo Tree Search para SSPs sob o critério GUBS". Em: *Anais do XVII Encontro Nacional de Inteligência Artificial e Computacional*. Evento Online: SBC, 2020, pgs. 402–413 (citado na pg. 89).

[G. N. Crispino *et al.* 2023] Gabriel Nunes Crispino, Valdinei Freire e Karina Valdivia Delgado. "GUBS criterion: arbitrary trade-offs between cost and probability-to-goal in stochastic planning based on expected utility theory". Em: *Artificial Intelligence* 316 (2023), pg. 103848. issn: 0004-3702 (citado na pg. 90).

[Forrest *et al.* 2022] John Forrest *et al. Coin-or/cbc: release releases/2.10.8*. Versão releases/2.10.8. Mai. de 2022. doi: 10.5281/zenodo.6522795. url: https://doi.org/10.5281/zenodo.6522795 (citado na pg. 84).

[Freire e Delgado 2017] Valdinei Freire e Karina Valdivia Delgado. "GUBS: a utility-based semantic for goal-directed Markov decision processes". Em: *Proceedings of the 16th Conference on Autonomous Agents and Multiagent Systems*. 2017, pgs. 741–749 (citado nas pgs. 2, 20, 26–30, 32, 35, 71, 72, 90).

[Freire, Delgado e Reis 2019] Valdinei Freire, Karina Valdivia Delgado e Willy Arthur Silva Reis. "An exact algorithm to make a trade-off between cost and probability in SSPs". Em: *Proceedings of the International Conference on Automated Planning and Scheduling*. Vol. 29. 2019, pgs. 146–154 (citado nas pgs. 2, 37–40, 42, 55–57, 61, 62, 90).

[Hansen e Zilberstein 2001] Eric A Hansen e Shlomo Zilberstein. "LAO*: a heuristic search algorithm that finds solutions with loops". Em: *Artificial Intelligence* 129.1-2 (2001), pgs. 35–62 (citado nas pgs. 12, 13, 15, 63).

[Howard 1960] R.A. Howard. *Dynamic Programming and Markov Processes*. Published jointly by the Technology Press of the Massachusetts Institute of Technology and, 1960 (citado na pg. 10).

[Keller e Eyerich 2012] Thomas Keller e Patrick Eyerich. "PROST: probabilistic planning based on UCT". Em: *Proceedings of the Twenty-Second International Conference on International Conference on Automated Planning and Scheduling*. 2012, pgs. 119–127 (citado na pg. 89).

[Keller e Helmert 2013] Thomas Keller e Malte Helmert. "Trial-based heuristic tree search for finite horizon MDPs". Em: *Proceedings of the Twenty-Third International Conference on International Conference on Automated Planning and Scheduling*. 2013, pgs. 135–143 (citado na pg. 89).

[Kocsis e Szepesvári 2006] Levente Kocsis e Csaba Szepesvári. "Bandit based Monte-Carlo planning". Em: *European conference on machine learning*. Springer. 2006, pgs. 282–293 (citado na pg. 89).

[Kolobov, D. Weld *et al.* 2012] Andrey Kolobov, Daniel Weld *et al.* "A theory of goal-oriented MDPs with dead ends". Em: *Uncertainty in artificial intelligence : proceedings of the Twenty-eighth Conference [on uncertainty in artificial intelligence] (2012)*. 2012, pgs. 438–447 (citado nas pgs. 1, 2, 7, 18, 20, 32).

[Kolobov, D. S. Weld *et al.* 2011] Andrey Kolobov, Daniel S Weld e Hector Geffner. "Heuristic search for generalized stochastic shortest path MDPs". Em: *Proceedings of the Twenty-First International Conference on International Conference on Automated Planning and Scheduling*. 2011, pgs. 130–137 (citado nas pgs. 1, 17, 32, 57, 90).

[Kuo e Freire 2021] Isabella Kuo e Valdinei Freire. "Probability-to-goal and expected cost trade-off in stochastic shortest path". Em: *Computational Science and Its Applications - ICCSA 2021 - 21st International Conference, Cagliari, Italy, September 13-16, 2021, Proceedings, Part III*. Vol. 12951. Lecture Notes in Computer Science. Springer, 2021, pgs. 111–125 (citado na pg. 2).

[Little, Thiebaux *et al.* 2007] Iain Little, Sylvie Thiebaux *et al.* "Probabilistic planning vs. replanning". Em: *ICAPS Workshop on IPC: Past, Present and Future*. 2007, pgs. 1–10 (citado na pg. 71).

[Martelli e Montanari 1973] Alberto Martelli e Ugo Montanari. "Additive and/or graphs." Em: *IJCAI*. Vol. 73. 1973, pgs. 1–11 (citado nas pgs. 2, 12, 13, 63).

[Martelli e Montanari 1978] Alberto Martelli e Ugo Montanari. "Optimizing decision trees through heuristically guided search". Em: *Communications of the ACM* 21.12 (1978), pgs. 1025–1039 (citado nas pgs. 2, 12, 13).

[Mausam e Kolobov 2012] Mausam e Andrey Kolobov. *Planning with Markov decision processes : an AI perspective*. San Rafael, Calif. (1537 Fourth Street, San Rafael, CA 94901 USA: Morgan & Claypool, 2012 (citado nas pgs. 1, 10, 19, 61).

[Minami e Silva 2012] Renato Minami e Valdinei Freire da Silva. "Shortest stochastic path with risk sensitive evaluation". Em: *11th Mexican International Conference on Artificial Intelligence (MICAI 2012)*. Vol. 7629. Lecture Notes in Artificial Intelligence. 2012, pgs. 370–381 (citado na pg. 57).

[Nilsson 1982] Nils J Nilsson. *Principles of artificial intelligence*. Springer Science & Business Media, 1982 (citado nas pgs. 2, 12, 13).

[Ono, Kuwata *et al.* 2012] Masahiro Ono, Yoshiaki Kuwata e J Balaram. "Joint chance-constrained dynamic programming". Em: *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE. 2012, pgs. 1915–1922 (citado na pg. 90).

[Ono e B. C. Williams 2008] Masahiro Ono e Brian C Williams. "Iterative risk allocation: a new approach to robust model predictive control with a joint chance constraint". Em: *2008 47th IEEE Conference on Decision and Control*. IEEE. 2008, pgs. 3427–3432 (citado na pg. 90).

[Patek 2001] Stephen D Patek. "On terminating Markov decision processes with a risk-averse objective function". Em: *Automatica* 37.9 (2001), pgs. 1379–1386 (citado nas pgs. 37, 57).

[Puterman 1994] Martin Puterman. *Markov decision processes: discrete stochastic dynamic programming*. New York: Wiley, 1994 (citado nas pgs. 1, 5, 7, 10, 61).

[Sanner e Yoon 2011] Scott Sanner e Sungwook Yoon. "IPPC results presentation". Em: *International Conference on Automated Planning and Scheduling*. Accesses. 2011. url: http://users.cecs.anu.edu.au/~ssanner/IPPC_2011/IPPC_2011_Presentation.pdf (citado na pg. 71).

[Silver e Chitnis 2020] Tom Silver e Rohan Chitnis. "PDDLGym: gym environments from PDDL problems". Em: *International Conference on Automated Planning and Scheduling (ICAPS) PRL Workshop*. 2020, pgs. 1–6. url: https://github.com/tomsilver/pddlgym (citado na pg. 71).

[Teichteil-Königsbuch 2012] Florent Teichteil-Königsbuch. "Stochastic safest and shortest path problems". Em: *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*. 2012, pgs. 1825–1831 (citado nas pgs. 1, 2, 18, 32).

[Teichteil-Königsbuch *et al.* 2011] Florent Teichteil-Königsbuch, Vincent Vidal e Guillaume Infantes. "Extending classical planning heuristics to probabilistic planning with dead-ends". Em: *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*. 2011, pgs. 1017–1022 (citado nas pgs. 1, 20, 32).

[Thiébaux, B. Williams *et al.* 2016] Sylvie Thiébaux, Brian Williams *et al.* "RAO*: an algorithm for chance-constrained POMDP's". Em: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 30. 1. 2016 (citado na pg. 90).

[F. Trevizan, Thiébaux e Haslum 2018] Felipe Trevizan, Sylvie Thiébaux e Patrik Haslum. "Operator counting heuristics for probabilistic planning". Em: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, jul. de 2018, pgs. 5384–5388 (citado na pg. 90).

[F. Trevizan, Thiébaux, Santana *et al.* 2016] Felipe Trevizan, Sylvie Thiébaux, Pedro Santana e Brian Williams. "Heuristic search in dual space for constrained stochastic shortest path problems". Em: *Proceedings of the International Conference on Automated Planning and Scheduling* 26.1 (mar. de 2016), pgs. 326–334. doi: 10.1609/icaps.v26i1.13768. url: https://ojs.aaai.org/index.php/ICAPS/article/view/13768 (citado na pg. 90).

[F. W. Trevizan *et al.* 2017] Felipe W Trevizan, Florent Teichteil-Königsbuch e Sylvie Thiébaux. "Efficient solutions for stochastic shortest path problems with dead ends." Em: *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence (UAI) (2017)*. 2017, pgs. 1–10 (citado nas pgs. 1, 2, 18, 32, 90).