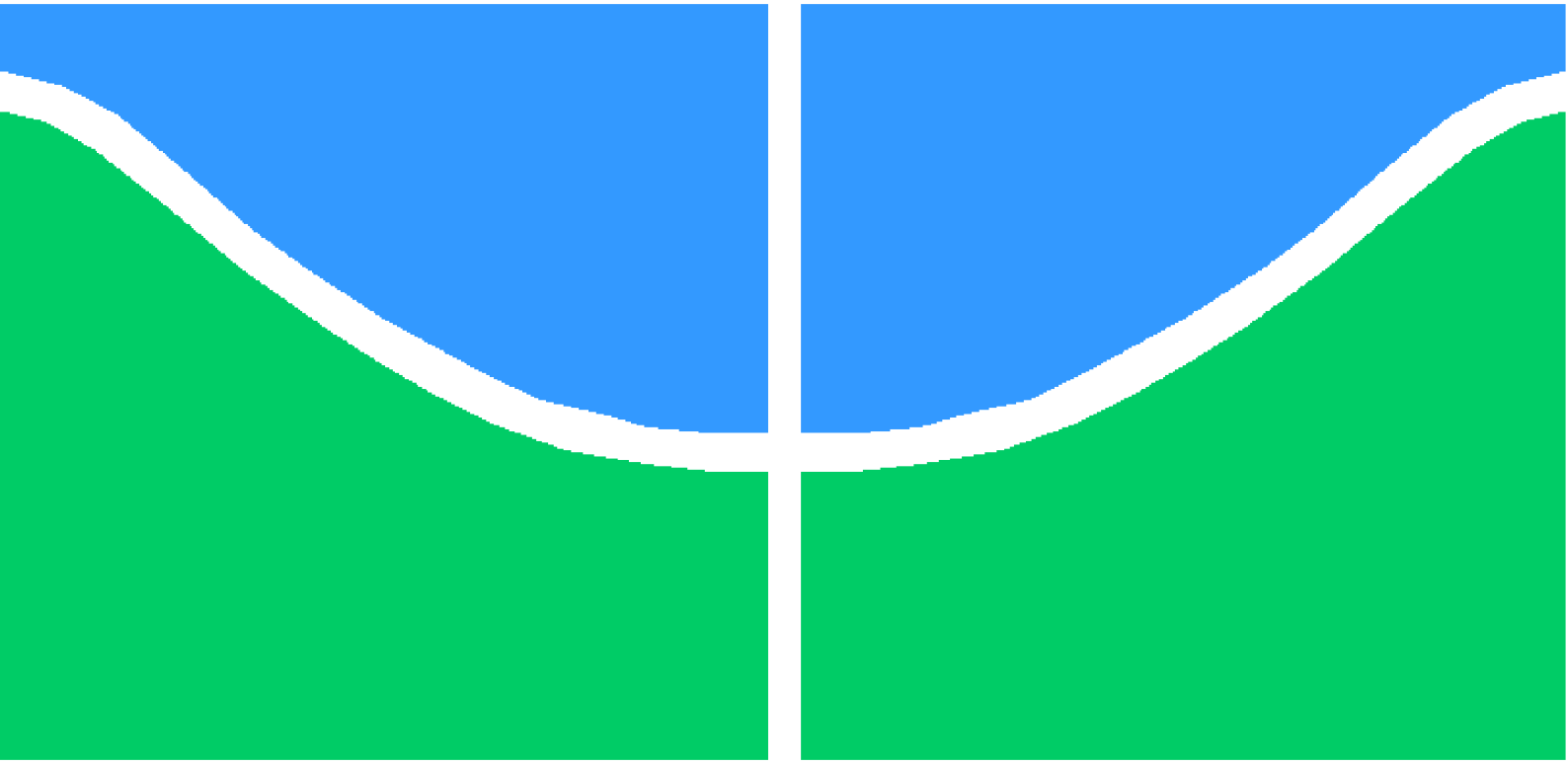


Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Fundamentos Lógicos de Inteligência Artificial

Distribuição Hoteleira PDDL

Autor: Thiago Oliveira Cunha
Gabriele Ribeiro Gomes
Arthur Taylor de Jesus Popov

Brasília, DF
2023

Sumário

Introdução

O presente trabalho visa a utilização de um solver PDDL para a resolução de um problema apresentado ao grupo como o "problema da distribuição hoteleira". O problema consiste em alocar o maior número de pessoas em quartos de hoteis respeitando diversas regras, como por exemplo minimizar o custo total dos quartos, casais devem ser alocados juntos entre outras regras de negócio que devem ser respeitadas.

Levando isso em consideração e também os estudos realizados pelo grupo com o PDDL Wiki,(GREEN BENJAMIN JACOB REJI, 2020) e com o trabalho de gerenciamento de pacotes da própria disciplina,(KADESH, 2023) modelamos o problema de forma a tentar respeitar a maior parte destas regras, utilizando a linguagem PDDL e utilizando um solver de planejamento o "Fast Downward"para a obtenção de soluções ótimas ou sub ótimas para o problema que nos foi apresentado.

Ao decorrer deste documento temos duas partes, sendo elas

- Formulação Proposta: Onde será mostrada a formulação proposta pelo grupo para resolver o problema da dsitribuição hoteleira, contendo o código de dominio em PDDL, com explicações de seus elementos e como ele funciona logicamente, visto que é ele que rege a boa execução de um código de problema, e os códigos de problema foram gerados por um código separado em python que gera o problema de acordo com as entradas especificadas.
- Experimentos: Onde é mostrado uma tabela para cada um dos 10 experimentos realizados contendo os quartos om preços, os homens, mulheres, casais e métricas obtidas a partir da execução com o planner "Fast Downward", bem como uma imagem do resultado do melhor planejamento obtido.

E por fim a conclusão, onde analisamos os dados obtidos a partir dos experimentos e vemos os problemas e capacidades do codigo realizado, verificando gargalos e dando uma analise geral do que foi feito.

Formulação Proposta

Para resolver o problema da distribuição hoteleira, foi utilizado uma modelagem do problema com PDDL. O domínio foi definido da seguinte maneira:

```
(define (domain hotel)
    (:requirements :typing :negative-preconditions
       :disjunctive-preconditions :equality)

    (:types person room couple)

    (:functions (total-cost)
                (price ?room - room)
    )

    (:predicates (double-room ?doubleRoom - room)
                 (triple-room ?tripleRoom - room)
                 (quadruple-room ?quadrupleRoom - room)
                 (couple-room ?coupleRoom - room)
                 (allocated ?person - person)
                 (allocated-couple ?couple - couple)
                 (is-available ?room - room)
                 (male ?person - person)
                 (female ?person - person)
    )

    (:action allocate-one-person
        :parameters (?person - person ?room - room)
        :precondition (and (not (allocated ?person))
                           (or (double-room ?room)
                               (triple-room ?room)
                               (quadruple-room ?room)
                               (couple-room ?room)
                           )
                           (is-available ?room)
        )
        :effect (and (allocated ?person)
                     (not (is-available ?room))
```

```
33                      (increase (total-cost) (price ?room)))
34      )
35
36      (:action allocate-two-people
37          :parameters (?person1 - person
38                       ?person2 - person
39                       ?room - room
40          )
41          :precondition (and (not (allocated ?person1))
42                             (not (allocated ?person2))
43                             (not (= ?person1 ?person2))
44                             (or (and (male ?person1)
45                                      (male ?person2))
46                                 (and (female ?person1)
47                                      (female ?person2))
48                             )
49                             (or (double-room ?room)
50                                 (triple-room ?room)
51                                 (quadruple-room ?room)
52                             )
53                             (is-available ?room)
54
55          )
56          :effect (and (allocated ?person1)
57                       (allocated ?person2)
58                       (not (is-available ?room))
59                       (increase (total-cost) (price ?room))
60          )
61      )
62
63      (:action allocate-three-people
64          :parameters (?person1 - person
65                       ?person2 - person
66                       ?person3 - person
67                       ?room - room
68          )
69          :precondition (and (not (allocated ?person1))
70                             (not (allocated ?person2))
71                             (not (allocated ?person3))
```

```
                            (not (= ?person1 ?person2))
                            (not (= ?person1 ?person3))
                            (not (= ?person2 ?person3))
                            (or (and (male ?person1)
                                     (male ?person2)
                                     (male ?person3))
                                (and (female ?person1)
                                     (female ?person2)
                                     (female ?person3))
                            )
                            (or (triple-room ?room)
                                (quadruple-room ?room)
                            )
                            (is-available ?room)

        )
        :effect (and (allocated ?person1)
                     (allocated ?person2)
                     (allocated ?person3)
                     (not (is-available ?room))
                     (increase (total-cost) (price ?room))
        )
    )

    (:action allocate-four-people
        :parameters (?person1 - person
                     ?person2 - person
                     ?person3 - person
                     ?person4 - person
                     ?room - room
        )
        :precondition (and (not (allocated ?person1))
                           (not (allocated ?person2))
                           (not (allocated ?person3))
                           (not (allocated ?person4))
                           (not (= ?person1 ?person2))
                           (not (= ?person1 ?person3))
                           (not (= ?person1 ?person4))
                           (not (= ?person2 ?person3))
```

```
                        (not (= ?person2 ?person4))
                        (not (= ?person3 ?person4))
                        (or (and (male ?person1)
                                 (male ?person2)
                                 (male ?person3)
                                 (male ?person4))
                            (and (female ?person1)
                                 (female ?person2)
                                 (female ?person3)
                                 (female ?person4))
                        )
                        (quadruple-room ?room)
                        (is-available ?room)
        )
        :effect (and (allocated ?person1)
                     (allocated ?person2)
                     (allocated ?person3)
                     (allocated ?person4)
                     (not (is-available ?room))
                     (increase (total-cost) (price ?room))
        )
    )

    (:action allocate-couple
        :parameters (?couple - couple
                     ?room - room
        )
        :precondition (and (not (allocated-couple ?couple))
                        (or (double-room ?room)
                            (triple-room ?room)
                            (quadruple-room ?room)
                            (couple-room ?room)
                        )
                        (is-available ?room)

        )
        :effect (and (allocated-couple ?couple)
                     (not (is-available ?room))
                     (increase (total-cost) (price ?room))
```

```
150             )
151         )
152 )
```

Existem três tipos do problema: *person* que define uma pessoa, *room* que define um quarto e *couple* que define um casal.

Foram criados nove predicados, sendo eles:

- *double-room* que representa um quarto duplo (com duas camas de solteiro).
- *triple-room* que representa um quarto triplo (com três camas de solteiro).
- *quadruple-room* que representa um quarto quadruplo (com quatro camas de solteiro).
- *couple-room* que representa um quarto de casal (com uma cama de casal).
- *allocated* que indica que uma pessoa está alocada em algum quarto.
- *allocated-couple* que indica que um casal está alocado em algum quarto.
- *is-available* que indica que um quarto está disponível.
- *male* que indica que uma pessoa é do sexo masculino.
- *female* que indica que uma pessoa é do sexo feminino.

Também foi definida uma função *total-cost* que determina o valor total dos quartos usados na alocação das pessoas e uma função *price* que indica o valor individual de cada quarto. No problema, é definido como métrica a minimização desse valor.

No domínio foram definidas cinco ações, sendo elas:

- *allocate-one-person*: que aloca uma pessoa em algum quarto. A pré-condição é de que exista uma pessoa não alocada, que o quarto seja duplo, triplo, quadruplo ou de casal e que esse quarto esteja disponível. O efeito dessa ação é a alocação da pessoa, a indisponibilidade do quarto e um aumento no custo baseado no valor do quarto escolhido.
- *allocate-two-people*: que aloca duas pessoas em algum quarto. A pré-condição é de que existam duas pessoas desalocadas, que essas pessoas não sejam as mesmas, que as duas pessoas sejam do sexo masculino ou feminino, que o quarto seja do tipo duplo, triplo ou quadruplo, e que o quarto esteja disponível. O efeito dessa ação é a alocação das pessoas, a indisponibilidade do quarto e um aumento no custo baseado no valor do quarto escolhido.

- *allocate-three-people*: que aloca três pessoas em algum quarto. A pré-condição é de que existam três pessoas desalocadas, que essas pessoas não sejam as mesmas, que as três pessoas sejam do sexo masculino ou feminino, que o quarto seja do tipo triplo ou quadruplo, e que o quarto esteja disponível. tipo duplo, triplo ou quadruplo, e que o quarto esteja disponível. O efeito dessa ação é a alocação das pessoas, a indisponibilidade do quarto e um aumento no custo baseado no valor do quarto escolhido.

- *allocate-four-people*: que aloca quatro pessoas em algum quarto. A pré-condição é de que existam quatro pessoas desalocadas, que essas pessoas não sejam as mesmas, que as quatro pessoas sejam do sexo masculino ou feminino, que o quarto seja do quadruplo e esteja disponível. tipo duplo, triplo ou quadruplo, e que o quarto esteja disponível. O efeito dessa ação é a alocação das pessoas, a indisponibilidade do quarto e um aumento no custo baseado no valor do quarto escolhido.

- *allocate-couple*: que aloca um casal em algum quarto. A pré-condição é de que exista um casal desalocado, um quarto duplo, triplo, quadruplo ou de casal e que esteja disponível. O efeito é a alocação do casal, a indisponibilidade do quarto e um aumento no custo baseado no valor do quarto escolhido.

Para gerar os problemas usados nos experimentos, foi criado um script em Python que gera aleatoriamente os valores de entrada em um arquivo. Definindo quantidade de quartos, tipo e valor de cada quarto. Bem como a quantidade de pessoas do sexo masculino e feminino, e quais dessas pessoas são um casal.

Também foi criado um outro script em Python que recebe essas entradas e gera automaticamente um arquivo *problem.pddl*.Ambos os códigos se encontram no GitHub.

O *solver* usado para resolver os problemas gerados foi o **fast-downward**.

Um exemplo de código de problema pode ser visto a seguir:

```
(define (problem aloca-pessoas-no-hotel)
    (:domain hotel)
    (:requirements :universal-preconditions)
    (:objects Casimiro Igor DavyJones Iasmine Chris - person
              Bruno&Rose Otaviano&Vanessa - couple
              room0 room1 room2 room3 room4 room5 room6
                 room7 room8 room9 - room
    )
    (:init
           (not (allocated Casimiro))
           (male Casimiro)
```

```
            (not (allocated Igor))
            (male Igor)
            (not (allocated DavyJones))
            (male DavyJones)
            (not (allocated Iasmine))
            (female Iasmine)
            (not (allocated Chris))
            (female Chris)
            (not (allocated-couple Bruno&Rose))
            (not (allocated-couple Otaviano&Vanessa))
            (is-available room0)
            (couple-room room0)
            (= (price room0) 100)
            (is-available room1)
            (couple-room room1)
            (= (price room1) 150)
            (is-available room2)
            (couple-room room2)
            (= (price room2) 1000)
            (is-available room3)
            (double-room room3)
            (= (price room3) 150)
            (is-available room4)
            (double-room room4)
            (= (price room4) 200)
            (is-available room5)
            (triple-room room5)
            (= (price room5) 250)
            (is-available room6)
            (triple-room room6)
            (= (price room6) 600)
            (is-available room7)
            (triple-room room7)
            (= (price room7) 250)
            (is-available room8)
            (quadruple-room room8)
            (= (price room8) 360)
            (is-available room9)
            (quadruple-room room9)
```

```
50              (= (price room9) 990)
51              (= (total-cost) 0)
52      )
53      (:goal
54        (and (forall (?person - person) (and(allocated
              ?person)))
55              (forall (?couple - couple) (and(allocated-couple
                 ?couple)))
56        )
57      )
58      (:metric minimize (total-cost))
59  )
```

Em cada arquivo de problema são listadas todas as pessoas, quartos, casais. Bem como o sexo das pessoas e o preço dos quartos. O *goal* estabelece que todas as pessoas e todos os casais devem ser alocados. E a métrica é a minimização do custo total dos quartos.

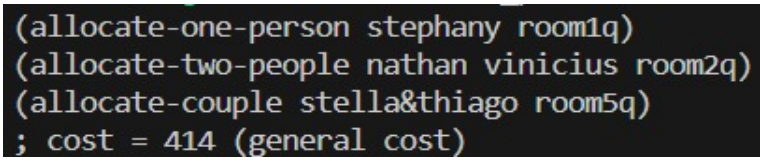
Experimentos

Esta seção está destinada para a demonstração das 10 entradas escolhidas e do resultado das mesmas, com tempo de execução total, total de memória utilizado e tempo de busca total.

- Ex 01:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
C 655 Q 102 Q 187 D 942 C 647 Q 125 T 782 T 411	Vinicius Nathan Thiago	Stephany Stella	Stella Thiago	- Tempo Execução Total : 0.0194361s - Memória Utilizada: 10820 KB - Tempo Busca Total: 0.0153588s

Tabela 1 – Dados Entrada/Saída 01



- Ex 02:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
D 648 C 739 C 880 T 406 T 396 T 120 Q 365 C 925 D 738 C 758 T 752 D 967 C 422 C 476 Q 628 Q 331 C 306 C 656 T 450 C 491 T 203 T 617 Q 120 D 589 D 669 C 837 D 118 T 175	Diogo Matheus Andre Bryan Gabriel	Carolina Ana Mariana Fernanda Milena Maria	Carolina Milena	- Tempo Execução Total : 13.8982s - Memória Utilizada: 50836KB - Tempo Busca Total: 13.8383s

Tabela 2 – Dados Entrada/Saida 02

```
(allocate-two-people ana fernanda room26d)
(allocate-couple carolina&milena room5t)
(allocate-three-people andre bryan diogo room18t)
(allocate-two-people maria mariana room27t)
(allocate-two-people gabriel matheus room22q)
; cost = 983 (general cost)
```

- Ex 03:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
D 921 T 560 C 310 D 543 Q 969 D 119 C 725	Lorenzo Andre	Eloah Esther Maysa	Esther Eloah	- Tempo Execução Total : 0.0199746s - Memória Utilizada: 10820KB - Tempo Busca Total: 0.235664s

Tabela 3 – Dados Entrada/Saida 03

```
(allocate-two-people andre lorenzo room5d)
(allocate-one-person maysa room2c)
(allocate-couple esther&eloah room3d)
; cost = 972 (general cost)
```

- Ex 04:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
D 266 Q 356 Q 71 Q 738 T 720 Q 419 T 138 T 653 C 582 C 256 T 358 Q 644 C 634	Gabriel Leonardo Joao	Sabrina Helena Ana	Helena Joao Sabrina Leonardo	- Tempo Execução Total : 0.0173286s - Memória Utilizada: 10812KB - Tempo Busca Total: 0.0207788s

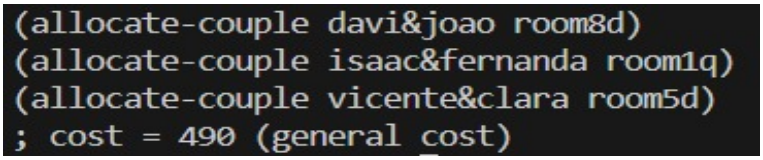
Tabela 4 – Dados Entrada/Saida 04

```
(allocate-one-person ana room6t)
(allocate-one-person gabriel room9c)
(allocate-couple sabrina&leonardo room2q)
(allocate-couple helena&joao room0d)
; cost = 731 (general cost)
```

- Ex 05:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
T 517 Q 79 T 672 D 500 D 826 D 350 Q 115 D 946 D 61 D 849 T 818 T 907 Q 871 Q 514 T 629 C 770	Davi Isaac Joao Vicente	Clara Fernanda	Davi Joao Vicente Clara Isaac Fernanda	- Tempo Execução Total : 0.019691s - Memória Utilizada: 10816KB - Tempo Busca Total: 0.023259s

Tabela 5 – Dados Entrada/Saida 05



- Ex 06:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
Q 238 Q 540 D 204 D 546 T 208 D 63 T 910 T 975 Q 637 Q 966 Q 406 T 506 T 342 Q 375 Q 516 D 553 Q 104 C 616 T 967	Davi	Rebeca Laura Ana Alexia Gabrielly	Alexia Rebeca	- Tempo Execução Total : 0.710449s - Memória Utilizada: 11472KB - Tempo Busca Total: 0,0189551s

Tabela 6 – Dados Entrada/Saida 06

```
(allocate-three-people ana gabrielly laura room16q)
(allocate-one-person davi room5d)
(allocate-couple alexia&rebeca room2d)
; cost = 371 (general cost)
```

- Ex 07:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
Q 283 D 770 C 855 C 460 T 267 C 767 Q 820 Q 263 C 929 T 819 C 692 C 285 C 484 C 86	Pietro Thales Diogo Bryan	Nicole Alice Maria Nina	Alice Bryan Maria Nicole	- Tempo Execução Total :0,203412s - Memória Utilizada: 12628KB - Tempo Busca Total: 0,19847s

Tabela 7 – Dados Entrada/Saida 07

```
(allocate-three-people diogo pietro thales room7q)
(allocate-couple alice&bryan room13c)
(allocate-one-person nina room0q)
(allocate-couple maria&nicole room4t)
; cost = 899 (general cost)
```

- Ex 08:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
T 234 Q 343 C 586 D 632 D 68 D 822 C 470 Q 343 Q 948 C 932 T 769 Q 285 C 524 C 648 Q 653	Thales Otavio Joao Thiago Andre Vitor Erick Luigi	Lavinia Livia Sophie Catarina	Erick Sophie Andre Lavinia Livia Otavio Thales Thiago Vitor Catarina	- Tempo Execução Total :1,61197s - Memória Utilizada: 21644KB - Tempo Busca Total: 1,60772s

Tabela 8 – Dados Entrada/Saida 08

```
(allocate-two-people joao luigi room0t)
(allocate-couple erick&sophie room4d)
(allocate-couple andre&lavinia room11q)
(allocate-couple livia&otavio room1q)
(allocate-couple thales&thiago room7q)
(allocate-couple vitor&catarina room6c)
; cost = 1743 (general cost)
```

- Ex 09:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
T 344 T 626 D 317 C 878 D 498 T 573 C 998 Q 61 C 615 Q 324 T 162 T 317 Q 720 T 469	Joao Lorenzo Igor Vicente Gabriel	Lavinia Ana Stella Giovanna Nicole	Gabriel Igor Stella Vicente Lavinia Joao	- Tempo Execução Total :3,03824s - Memória Utilizada: 31312KB - Tempo Busca Total: 3,03167s

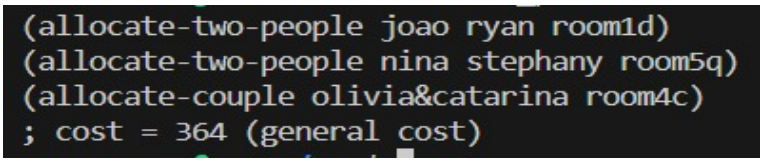
Tabela 9 – Dados Entrada/Saida 09

```
(allocate-three-people ana giovanna nicole room10t)
(allocate-couple gabriel&igor room7q)
(allocate-couple lavinia&joao room2d)
(allocate-one-person lorenzo room11t)
(allocate-couple stella&vicente room9q)
; cost = 1181 (general cost)
```

- Ex 10:

Quartos/Preços	Homens	Mulheres	Casais	Métricas
Q 688 D 142 D 635 T 661 C 60 Q 162 T 452	Ryan Joao	Catarina Stephany Olivia Nina	Olivia Catarina	- Tempo Execução Total :0,0211308s - Memória Utilizada: 10816KB - Tempo Busca Total: 0,0126803s

Tabela 10 – Dados Entrada/Saida 10



Conclusão

Após realizarmos a apresentação da forma como foi modelado o problema proposto e apresentarmos alguns resultados podemos assim analisar de forma um pouco mais profunda os resultados obtidos e o processo de realização do trabalho em si.

A modelagem do problema em si foi bastante desafiadora, uma vez que seguir as regras de negócio propostas utilizando PDDL para a modelagem do problema pela primeira vez foi um tanto quanto desafiador, devido a nova forma de se pensar necessária para modelar problemas, mas isso abriu também espaço para encarar problemas de uma forma diferente. Tendo sido atendidas pelo trabalho as regras mais básicas como minimização de custo, alocação de casais e também a separação por sexos dos quartos. Isso nos proporcionou uma visão geral e uma grande aprendizagem sobre modelagens de problemas em PDDL.

Então como citado o trabalho atende as regras básicas, com isso não foi implementado uma lógica para a alocação considerando as afinidades das pessoas para a divisão de quartos que também nos foi proposta e nenhuma outra regra adicional, outro ponto que vale ressaltar é a possibilidade de refatoração e melhoramentos possíveis na modelagem, uma vez que problemas com muitas pessoas acaba estourando a memória da maquina onde estamos rodando o planejador e não conseguimos um plano ótimo para aquele caso.

Diante do apresentado o trabalho possui sim margem para melhora, como implementação de novas regras de negócio para uma melhor acomodação dos hóspedes e uma otimização da modelagem para que possamos trabalhar com um número maior de pessoas de forma correta. Mas para uma primeira versão da modelagem e também primeiro contato dos membros com a modelagem PDDL conseguimos alcançar os objetivos básicos.

Referências

GREEN BENJAMIN JACOB REJI, C. C. M. E. S. F. M. F. M. R. H. S. J. D. M. M. J. M. A. *Planning.Wiki The AI Planning PDDL wiki*. 2020. Disponível em: <https://planning.wiki/>. Citado na página 3.

KADESH, E. *Gerenciador de Dependencias*. 2023. Disponível em: <https://gitlab.com/sergiosacj/dependencies-manager/-/tree/main/>. Citado na página 3.