

**PROJETO PROCURA E PLANEAMENTO 2018**

**GRUPO** 18

**Problema de Afectação em Empresas de Transporte**

Rafael Koener, ist176475

Kevin Batista Corrales, ist194131

**Conteúdo**

[1. Introdução 2](#_Toc531549890)

[2. Estrutura de dados 2](#_Toc531549891)

[3. MODELAÇÕES DO PROBLEMA 3](#_Toc531549892)

[4. HEURÍSTICAS DESENVOLVIDAS 3](#_Toc531549893)

[5. ESTRATÉGIDAS DE CORTE 3](#_Toc531549894)

[6. CONCLUSÃO 3](#_Toc531549895)

# 

# Introdução

Este projeto tem como objetivo desenvolver um programa que resolva um problema de afectação completa das tarefas de condução de veículos de transporte, a realizar pelos trabalhadores durante um dia de trabalho, a turnos de serviço, respeitando todas as restrições.

Neste relatório contém a avaliação de implementações alternativas tanto de ponto de vista quantitativo como qualitativo.

# Estrutura de dados

Para a estruturação dos dados, na implementação do projeto, foi decidido apenas estruturar o estado do problema (*state*):

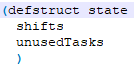


Figura 1 - G18.lisp - state

No estado decidimos implementar informação como: *shifts* (lista de turnos, cada lista de turno contém lista de tarefas) e *unusedTasks* (lista de tarefas não utilizadas).

A lista de tarefas não utilizadas (*unusedTasks*) será utilizada para o processamento de atribuição e organização de turnos, será verificado em que turno pertence cada tarefa da lista.

A lista de turnos (*shifts*) será retornada após o problema estiver completamente resolvido.

A razão da decisão de apenas utilizar estruturação para o estado do problema foi devido ao facto de que o tratamento da lista de tarefas, recebidas como argumento no método de resolução de problema *faz-afectação*, seria mais eficiente e mais rápida em relação ao tempo de processamento.

# MODELAÇÕES DO PROBLEMA

Na modelação do problema foi utilizado a função *cria-problema* (disponibilizada na biblioteca *procura.lisp*) em que recebe como argumento um estado-inicial, lista de operadores, o objetivo final, função de custo, heurística, o algoritmo de procura e profundidade máxima.

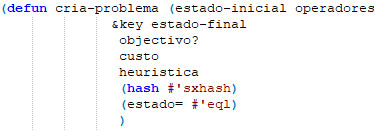


Figura – procura.lisp – cria-problema

* *estado-inicial* será uma lista de tarefas, em que cada tarefas contém uma lista com informação, ordenada, sobre: local de partida; local de chegada; instante de partida; instante de chegada.

Exemplo:

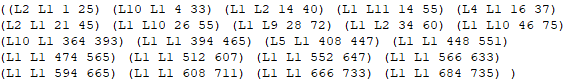


Figura Exemplo estado-inicial

* *operadores* será uma lista de operadores possíveis para a alteração do estado do problema. Consideramos como operadores possíveis: adicionar turno (*addShift*) e adicionar tarefa (*addTask*).
  + *addShift* nesta operação é criado um novo turno na lista de turnos do estado (*shifts*)

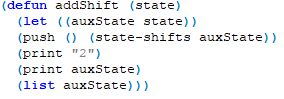


Figura - G18.lisp - addShift

* + *addTask* nesta operação é adicionada uma tarefa ao um turno que pertence, que se encontra na lista de turnos do estado (*shifts*).

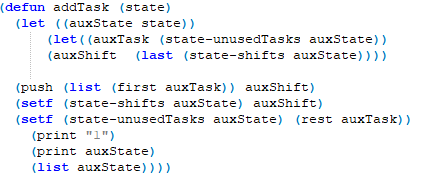


Figura - G18.lisp - addTask

* *objetivo?* será uma função que representa o objetivo final do problema, como objetivo final será considerado o esvaziamento total de *unusedTasks* do estado (lista de tarefas não utilizadas). Este facto significa que todas as tarefas do estado inicial foram atríbuidas em turnos e o problema foi resolvido.

https://i.gyazo.com/522b994dc5808a1c9e051c10066da886.png

Figura - g18.lisp - objetivo?

* *custo* será uma função de cálculo de custo de cada operação realizada.
* *heurística* é um argumento opcional dependendo do algoritmo de procura escolhido.
* *algoritmo/estratégia de procura*, tal como o nome indica, este argumento recebe o nome do algoritmo de procura a utilizar.
* *profundidade máxima*, este argumento é opcional, indica o valor da profundidade máxima. Este argumento depende do algoritmo de procura utilizado.

# HEURÍSTICAS DESENVOLVIDAS

# ESTRATÉGIDAS DE CORTE

# CONCLUSÃO