MOTOR DE PROGRAMAÇÃO DE ESCALA

1. Introdução

O presente documento tem como objetivo apresentar um motor de programação de escala, bem como um pseudocódigo para sua heurística construtiva e uma explicação detalhada sobre a implementação da metaheurística de busca tabu para otimização da escala gerada.

2. Motor de Programação de Escala

Pseudocódigo da heurística construtiva

```
Inicialize capMotorista como 7.34 (horas)
Inicialize linhas_de_onibus com informações sobre as linhas de ônibus
Inicialize motoristas_disponiveis com uma lista de  motoristas
Inicialize EscalaGeral como um dicionário vazio

Para cada linha de ônibus em linhas_de_onibus:
   Inicialize ContHora como 0
   Escolha um motorista aleatório de motoristas_disponiveis
   Remova o motorista escolhido de motoristas_disponiveis
   Obtenha o primeiro horário de partida da linha de ônibus
   Inicialize escala como um dicionário vazio
   Inicialize dadosViagem como uma lista vazia contendo três listas vazias
   Inicialize trechoMot como uma lista vazia
   Inicialize hrPartHrCheg como uma lista vazia

   Para cada horário de partida na linha de ônibus:
      Calcule a diferença de tempo entre o horário atual e o próximo horário
      Se a diferença de tempo for maior ou igual a 4 horas:
         Adicione "Intervalo" ao trechoMot do motorista
         Adicione [horário de partida, horário de chegada] a hrPartHrCheg
         Adicione trechoMot, hrPartHrCheg e um intervalo de 30 minutos ao escala[motorista]

      Se a diferença de tempo for menor ou igual a capMotorista:
         Adicione o trecho da cidade ao trechoMot do motorista
         Adicione [horário atual, próximo horário] a hrPartHrCheg
         Adicione trechoMot e hrPartHrCheg ao escala[motorista]

      Se a diferença de tempo exceder capMotorista:
         Adicione o trecho da cidade ao trechoMot do motorista
         Adicione [horário atual, próximo horário] a hrPartHrCheg
         Adicione um intervalo entre jornadas de 11 horas ao dadosViagem
         Escolha um novo motorista aleatório de motoristas_disponiveis
         Remova o motorista escolhido de motoristas_disponiveis
         Atualize o horário inicial do motorista para o próximo horário
         Reinicie as listas trechoMot e hrPartHrCheg
         Adicione trechoMot, hrPartHrCheg e IntervaloEntre ao escala[motorista]

   Adicione escala à EscalaGeral para a linha de ônibus atual

Retorne EscalaGeral
```

Para utilizar a metaheurística de busca tabu para otimizar a EscalaGeral e minimizar o custo total, as seguintes etapas serão seguidas:

Definição da Função de Avaliação

Primeiramente, é necessário definir uma função de avaliação que calcule o custo total da escala considerando o custo fixo de cada motorista, o custo das horas extras e o custo adicional do DSR, se aplicável. Essa função deve levar em consideração as restrições do problema, como a capacidade de carga horária dos motoristas e os intervalos entre jornadas.

Implementação da Busca Tabu

Em seguida, é possível implementar o algoritmo de busca tabu. A busca tabu é uma metaheurística que permite explorar o espaço de soluções de forma mais eficiente, evitando ciclos e incentivando a diversificação das soluções encontradas.

A busca tabu mantém uma lista de soluções recentes (lista tabu) que não podem ser revisadas imediatamente. Isso ajuda a evitar que o algoritmo fique preso em mínimos locais e explore soluções mais variadas.

Definição de Movimentos e Critérios de Parada

É importante definir movimentos que alterem a escala dos motoristas de maneira a melhorar o custo total. Por exemplo, um movimento poderia ser trocar o motorista de uma viagem por outro motorista disponível que reduza o custo total.

Além disso, é importante definir critérios de parada para a busca tabu, como um número máximo de iterações sem melhoria significativa no custo total ou um tempo limite de execução.

Execução da Busca Tabu

Com a função de avaliação, os movimentos e os critérios de parada definidos, é possível executar a busca tabu na EscalaGeral. Durante a busca tabu, os movimentos são aplicados na escala atual, o custo total de cada nova solução é avaliado e as melhores soluções são escolhidas, considerando a lista tabu para evitar movimentos repetidos.

Análise e Seleção da Melhor Solução

Ao final da busca tabu, várias soluções candidatas otimizadas serão obtidas. É importante analisar essas soluções para encontrar aquela que possui o menor custo total, considerando todas as restrições e objetivos do problema.

A busca tabu é uma técnica poderosa para resolver problemas de otimização combinatória, como a otimização de escalas de motoristas. Ajustar os parâmetros da busca tabu, como tamanho da lista tabu, frequência de atualização e critérios de aspiração, é essencial para obter os melhores resultados para o problema específico.

Pseudocódigo Busca de Vizinhança inteligente:

```
Função AvaliarCusto(Escala):
   Calcular o custo total da Escala considerando o custo fixo de cada motorista, custo das horas extras e custo adicional do DSR, se aplicável
   Retornar o custo total

Função GerarMovimento(Escala):
   Implementar um movimento que altere a Escala de motoristas (por exemplo, trocar um motorista de uma viagem por outro motorista disponível)
   Retornar a nova Escala após o movimento

Função BuscaTabu(EscalaInicial, TamanhoListaTabu, NúmeroMáximoIterações):
   MelhorEscala <- EscalaInicial
   MelhorCusto <- AvaliarCusto(EscalaInicial)
   ListaTabu <- Lista vazia
   ContadorIterações <- 0

   Enquanto ContadorIterações < NúmeroMáximoIterações:
      NovaEscala <- GerarMovimento(MelhorEscala)
      NovaCusto <- AvaliarCusto(NovaEscala)

      Se NovaEscala não está na ListaTabu e NovaCusto < MelhorCusto:
         MelhorEscala <- NovaEscala
         MelhorCusto <- NovaCusto
         Adicionar NovaEscala à ListaTabu

      Atualizar ListaTabu removendo soluções antigas
      Incrementar ContadorIterações

   Retornar MelhorEscala e MelhorCusto
```

A função AvaliarCusto calcula o custo total da Escala considerando todos os fatores relevantes. A função GerarMovimento implementa um movimento na Escala para explorar novas soluções.

A função BuscaTabu é a implementação principal da metaheurística de busca tabu. Ela recebe uma EscalaInicial, um TamanhoListaTabu (tamanho máximo da lista tabu) e um NúmeroMáximoIterações (número máximo de iterações sem melhoria significativa).

A busca tabu é realizada através de um loop que gera movimentos na Escala e avalia o custo das novas soluções. Movimentos que não estão na lista tabu e melhoram o custo são aceitos. A lista tabu é atualizada e o processo continua até atingir o número máximo de iterações. No final, a função retorna a MelhorEscala encontrada e o MelhorCusto associado a essa escala.