META-HEURÍSTICA CS PARA O PROBLEMA DE LOCALIZAÇÃO DE CONTADORES DE TRÁFEGO EM REDES DE TRANSPORTE

Geraldo Regis Mauri
UFES - Universidade Federal do Espírito Santo
Centro de Ciências Exatas, Naturais e da Saúde, Alegre - ES
geraldo.mauri@ufes.br

Glaubos Climaco, Bruno Salezze Vieira, Glaydston Mattos Ribeiro
PESC/COPPE e PET/COPPE - Universidade Federal do Rio de Janeiro
Centro de Tecnologia Bloco H - Sala 106, Cidade Universitária - RJ
glaubos@cos.ufrj.br, {bruno.vieira,glaydston}@pet.coppe.ufrj.br

Pedro Henrique González
CEFET/RJ - Centro Federal de Educação Tecnológica Celso Suckow da Fonseca
Av. Maracanã, 229 - Maracanã, Rio de Janeiro - RJ
pegonzalez@eic.cefet-rj.br

Nilo Flávio Rosa Campos Júnior, Leonel Antônio da Rocha Teixeira Júnior, André de Oliveira Nunes
DNIT - Departamento Nacional de Infraestrutura de Transportes - Coordenação de Planejamento
SAN Quadra 03 lote "A", Brasília - DF
{nilo.junior,leonel.teixeira,andre.nunes}@dnit.gov.br

RESUMO

Este trabalho propõe a aplicação da meta-heurística *Clustering Search* (CS) para resolução do Problema de Localização de Contadores de Tráfego (PLCT). O PLCT consiste em determinar o número e a localização de postos de contagem cobrindo uma rede de transporte. Os postos de contagem são geralmente utilizados para monitorar e descrever as características do tráfego, que são utilizadas para auxiliar nas estimativas de matrizes de origem e destino (O-D). A partir dessas matrizes, é possível verificar a distribuição espacial das viagens entre as zonas de tráfego em uma rede de transporte. O CS é utilizado para definir o número mínimo e a localização de postos de contagem necessários para cobrir redes de transportes definidas a partir da malha rodoviária brasileira. O desempenho do método proposto foi comparado com outro método apresentado na literatura, e os resultados foram superiores para todos os Estados Brasileiros.

PALAVRAS CHAVE. Clustering Search, Localização de Contadores de Tráfego, Meta-heurística.
Tópicos: L&T - Logística e Transportes; MH - Meta-heurísticas.

ABSTRACT

This paper applies a Clustering Search (CS) metaheuristic to solve the Traffic Counting Location Problem (TCLP). The TCLP consists of determining the number and location of counting stations to cover a road network. Counting stations are used to check and depict traffic characteristics which will be used to assist in estimating origin-destination (O-D) matrices. Using these matrices we can analize the spatial distribution of trips among traffic zones in the network. Our CS is applied to define the minimum number and location of counting stations covering road networks based on the brazilian roads. The performance of the proposed CS was compared with another method reported in the literature, and improved results were obtained for all Brazilian States.

KEYWORDS. Clustering Search. Traffic Counting Location. Metaheuristic.
Paper topics: L&T - Logistics and Transport; MH - Metaheuristics.

1. Introdução

Características do tráfego em uma rede de transporte podem ser observadas e descritas por meio de contadores de tráfego. Esses equipamentos são capazes de medir, por exemplo, o número de veículos que passa por um determinado trecho de rodovia durante um período de tempo específico. Dessa forma, é possível determinar algumas características como o tráfego médio diário anual, o volume de veículos nos hórários de pico, etc. [Garber e Hoel, 1999].

O uso dos contadores ainda pode auxiliar na determinação da distribuição das viagens entre as diferentes zonas de tráfego em uma rede de transporte, por meio da definição de matrizes de origem e destino (O-D). Uma matriz O-D é uma fonte de informação importante para muitos estudos de transporte, como previsão de demanda futura de viagens, gestão de transporte e controle.

De uma forma geral, uma matriz O-D é estimada a partir de um estudo em grande escala que normalmente é custoso, demorado e trabalhoso. Porém, as estimativas de matrizes O-Ds a partir de contagens de tráfego podem ser atualizadas com frequência, e são relativamente baratas em comparação com os métodos de pesquisa convencionais. Portanto, as matrizes O-D estimadas a partir de contagens de tráfego são consideradas como uma forma conveniente e prática para obter informações atualizadas com frequência sobre os padrões de demanda de viagens de uma região.

Na literatura, é possível encontrar diversos trabalhos com diferentes abordagens para a definição de matrizes O-D. Van Zuylen e Willumsen [1980], Maher [1983], Bell [1984], Cascetta [1984], Spiess [1987], Fisk [1988], Yang et al. [1992], Ashok e Ben-Akiva [1993], Sherali et al. [1994], Bell e Shield [1995], Yang [1995], Bell et al. [1997], Hazelton [2000], Maher et al. [2001], Chen et al. [2005], Chootinan et al. [2005] e Yang et al. [2006] apresentam diferentes alternativas para o uso de contadores de tráfego de forma a melhorar as estimativas das matrizes O-D.

Tratando diretamente o Problema de Localização de Contadores de Tráfego (PLCT), tem-se uma abordagem tradicional que considera uma estimativa da matriz O-D. Nesse caso, não se encontra tantos trabalhos na literatura, apesar de ser um problema importante na prática [Yang e Zhou, 1998].

Yang et al. [1991] formulam o PLCT como um problema de programação inteira (PI) no qual a cobertura de cada par da matriz O-D é incorporada como uma restrição, e o tráfego total observado é tomado como a função objetivo a ser otimizada. Os autores também apresentaram um modelo de PI para determinar o número mínimo de pontos de contagem necessários para observar parte do fluxo total de tráfego em uma rede.

Yang et al. [2001] e Yang et al. [2006] propuseram modelos matemáticos para o PLCT visando selecionar, de forma otimizada, estações de contagem de tráfego em redes rodoviárias, considerando a separação de todos os pares O-D. Dessa forma, o problema foi "simplificado" a somente um grafo com peso binário nas aresta, que tem a função de indicar se existe um contador naquele trecho ou não. Viagens entre um determinado par O-D são consideradas observadas (ou separadas) se e somente se nenhum caminho pode ignorar os locais de contagem selecionados.

O PLCT é um problema combinatório NP-Difícil [Chen et al., 2007], e uma abordagem para sua resolução consiste em enumerar todos os caminhos possíveis para cada par O-D. Contudo, esse procedimento necessita de um tempo computacional inviável para redes de grandes dimensões, uma vez que o número de caminhos entre cada par O-D cresce exponencialmente de acordo com o tamanho da rede.

Chen et al. [2007] demonstram a complexidade do PLCT e propõem um Algoritmo Genético [Goldberg e Holland, 1988] para resolução do problema. Bianco et al. [2014] realizaram um estudo detalhado sobre o PLCT e desenvolveram uma abordagem exata baseada em um algoritmo *branch-and-price*, além de um Algoritmo Genético para resolver o problema.

Por fim, González et al. [2016] propuseram um conjunto de instâncias baseadas nos Estados Brasileiros, além de três heurísticas específicas para resolvê-las.

Com o intuito de resolver o PLCT e procurar por melhores soluções para as rodovias brasileiras, este trabalho apresenta uma aplicação da meta-heurística CS para resolução do PLCT.

O CS é utilizado para definir o número mínimo e a localização de postos de contagem necessários para cobrir a malha viária de cada Estado Brasileiro. O desempenho do método proposto foi comparado com outro método apresentado na literatura, e os resultados foram superiores para todas as instâncias consideradas.

O restante deste trabalho apresenta a definição matemática do problema na Seção 2. Já na Seção 3 é apresentado o *Clustering Search* (CS) implementado. Os experimentos computacionais são detalhados na Seção 4 e, por fim, as conclusões são apresentadas na Seção 5.

2. Definição Matemática do Problema

Uma rede de transporte pode ser definida por meio de um grafo não orientado $G = (N, A)$, em que N é o conjunto de nós e A é o conjunto de trechos na rede (arestas). O conjunto de pares O-D é representado por W, sendo cada par O-D $w \in W$ formado por nós $i \in N$ e $j \in N$ (origem e destino). Uma rede é chamada de conexa se existe pelo menos um caminho simples (um caminho que não contém trechos repetidos e nenhum nó repetido) entre cada par O-D $w \in W$.

Assim, considere então $x_a \in \{0, 1\}$, $a \in A$ como sendo uma variável binária tal que se $x_a = 1$, uma estação de contagem de tráfego deverá ser instalada no trecho a, e 0 caso contrário. Além disso, considere ainda que **x** denota o vetor de variáveis binárias correspondentes aos elementos x_a. O comprimento do caminho mais curto entre o par $w \in W$ é dado por $u_w(\mathbf{x})$, e é determinado como uma função do vetor binário **x**. Se $u_w(\mathbf{x}) > 0$, o caminho mais curto entre o par $w \in W$ obtido, por exemplo, pelo método de Dijkstra [Ahuja et al., 1993; Bertsekas, 1998] inclui, pelo menos, uma estação de contagem. Caso contrário, existe pelo menos um caminho de tempo de viagem igual a zero entre a origem i e o destino j do par O-D $w \in W$ que não passa por qualquer estação de contagem.

O modelo matemático utilizado no PLCT tratado neste trabalho é um dos modelos apresentados em Yang et al. [2006] para cobertura completa da rede, e consiste em determinar o número mínimo e a localização dos contadores de tráfego que separam todos os pares O-D na rede G:

$$Minimizar\ z = \sum_{a \in A} x_a \tag{1}$$

Sujeito a:

$$\sum_{a \in A} \delta_{ra}^{w} x_a \geq 1 \qquad r \in R_w, w \in W \tag{2}$$

$$x_a \in \{0, 1\} \qquad a \in A \tag{3}$$

sendo que R_w é o conjunto de caminhos entre o w e δ_{ra}^{w} é um parâmetro que é igual a 1 caso o trecho a esteja no caminho r entre o par w, e 0 caso contrário. A Função Objetivo (1) calcula o número mínimo de contadores de tráfego necessários para separar todos os pares O-D na rede G, enquanto as Restrições (2) garantem que todo par O-D é separado por no mínimo um contador (trecho selecionado). Por fim, as Restrições (3) definem o domínio das variáveis de decisão.

3. Clustering Search

O CS foi proposto por Oliveira e Lorena [2007] como uma maneira genérica de se combinar meta-heurísticas com agrupamentos (*clustering*) com o objetivo de identificar as regiões promissoras do espaço de busca antes de aplicar métodos de busca local. A possibilidade do acoplamento com qualquer meta-heurística e de aplicação a qualquer problema de otimização contínua ou combinatória faz do CS um modelo para construção de meta-heurísticas híbridas.

O CS pode ser considerado como um método híbrido que combina meta-heurísticas e buscas locais, no qual a busca é intensificada somente em áreas do espaço de busca que se mostrem

promissoras. A ideia principal é particionar o espaço de busca e localizar "regiões" supostamente promissoras [Oliveira et al., 2013].

Um *cluster* é definido por uma conjunto de três elementos $C = \{c, v, r\}$, sendo o seu centro c (melhor solução da região), o seu volume v (quantidade de soluções que foram associadas ao *cluster*) e o índice de ineficácia r, que indica quantas vezes o centro do *cluster* não foi melhorado por uma busca local [Ribeiro et al., 2012].

Se o volume v alcançar um valor λ, ele se torna promissor, e o seu índice de ineficácia r é analisado. Se r atingir o valor r_{max}, uma perturbação é realizada no centro do *cluster*, caso contrário, uma busca local é aplicada no centro do *cluster* c.

O pseudo-código do CS implementado é apresentado no Algoritmo 1. Como tal método exige que alguma técnica seja utilizada como geradora de soluções iniciais, foi utilizado o *Simulated Annealing* (SA) [Kirkpatrick et al., 1983]. O mesmo é executado até que um tempo limite ($Tempo_{max}$) seja atingido e, a cada redução de temperatura (linha 21) a solução corrente do SA é enviada para o CS.

Inicialmente, o número de *clusters* criados e seu volume e ineficácia são inicializados (linha 2). A seguir, uma solução inicial é gerada e armazenada como a melhor solução (s^* - linha 3). A partir de então, a cada iteração do SA uma solução vizinha é gerada a partir da solução corrente (Linha 10), sendo essa aceita caso melhore a solução corrente ou, caso contrário, com uma certa probabilidade.

A temperatura é reduzida (linha 21) e, caso o número máximo de *clusters* não tenha sido atingido, a solução corrente do SA é colocada no centro de um novo *cluster*. Após a inicialização dos γ *clusters*, as novas soluções geradas pelo SA passam a ser incluídas nos *clusters* existentes (a partir da linha 25). A assimilação é definida com base na semelhança entre a solução corrente do SA e o centro de todos os *clusters*. Assim, a solução é assimilada ao *cluster* com centro mais adequado (linha 26). Por fim, verifica-se se a região se tornou promissora e se deve ser aplicada a busca local ou uma perturbação, ou ainda se uma nova melhor solução foi encontrada (linhas 27 a 43).

Os parâmetros do CS (e do SA) utilizados no algoritmo proposto são:

- γ: Número máximo de *clusters*;
- λ: Limite para o volume dos *clusters*;
- r_{max}: Limite para o índice de ineficácia dos *clusters*;
- T_0: Temperatura inicial;
- T_f: Temperatura de congelamento;
- SA_{max}: Número máximo de iterações por temperatura; e
- α: Taxa de resfriamento.

Algoritmo 1: Clustering Search para o PLCT

Início
  $clt \leftarrow 0$; $r_i \leftarrow 0$, $v_i \leftarrow 0\ \forall i = 1...\gamma$;
  $s \leftarrow$ solução inicial; $s^* \leftarrow s$;
  Enquanto $Tempo < Tempo_{max}$ **Faça**
    $T \leftarrow T_0$;
    Enquanto $T > T_f$ **Faça**
      $iter \leftarrow 0$;
      Enquanto $iter < SA_{max}$ **Faça**
        $iter \leftarrow iter + 1$;
        $s' \leftarrow N(s)$;
        Se $f(s') < f(s)$ **Então**
          $s \leftarrow s'$;
          Se $f(s') < f(s^*)$ **Então**
            $s^* \leftarrow s'$;
          Fim
        Fim
        Senão
          Com probabilidade $e^{\frac{-(f(s')-f(s))}{T}}$ $s \leftarrow s'$;
        Fim
      Fim
      $T \leftarrow \alpha T$;
      Se $clt < \gamma$ **Então**
        $clt \leftarrow clt + 1$, $v_{clt} \leftarrow v_{clt} + 1$, $c_{clt} \leftarrow s$;
      Fim
      Senão
        $i \leftarrow argmin_{i \in \{1...\gamma\}}\{H_i\}$, $v_i \leftarrow v_i + 1$, $c_i \leftarrow melhor(c_i, s)$;
        Se $v_i = \lambda$ **Então**
          $v_i \leftarrow 1$;
          Se $r_i = r_{max}$ **Então**
            $r_i \leftarrow 0$, $c_i \leftarrow N(c_i)$;
          Fim
          Senão
            $BuscaLocal(c_i)$;
            Se c_i *melhorou* **Então**
              $r_i \leftarrow 0$;
            Fim
            Senão
              $r_i \leftarrow r_i + 1$;
            Fim
          Fim
          $s^* \leftarrow melhor(s^*, c_i)$;
        Fim
      Fim
    Fim
  Fim
  Retorna s
Fim

3.1. Modelagem Computacional do Problema

Em relação à modelagem do problema e estrutura de dados utilizada, o modelo matemático apresentado na Seção 2 foi tomado como base, e a estrutura utilizada para representar uma solução foi a mesma proposta em Chen et al. [2007], na qual uma solução é representada por um vetor de números binários com tamanho igual ao número de arestas da rede, indicando se tal aresta (trecho da rede) possui um contador instalado ou não. Como método verificador de cobertura, foi implementada uma versão do algoritmo de Dijkstra, que espalha caminhos mínimos a partir de cada nó de origem dos pares O-D para todos os nós de destino, e uma cobertura completa ocorre quando o menor caminho entre O-Ds é maior ou igual a 1, ou seja, $u_w(\mathbf{x}) \geq 1\ \forall\ w \in W$. Também foram realizados testes aceitando soluções inviáveis com penalidades e probabilidades de viabilidade, mas os resultados não se mostraram promissores. Então, a modelagem utilizada considera apenas soluções viáveis durante toda a execução do CS.

3.2. Geração da Solução Inicial

Uma heurística denominada C3 proposta por González et al. [2016] foi utilizada para geração de uma solução inicial para o CS. Essa heurística foi utilizada pois a mesma é capaz de gerar soluções de boa qualidade com baixos tempos computacionais.

3.3. Vizinhança $N(s)$

Esse método é responsável por receber uma solução s e retornar uma outra solução s' dentro da vizinhança de s, ou seja, $s' \in N(s)$. A implementação proposta apresentou grande impacto positivo no desempenho do CS. A ideia geral desse método consiste em selecionar uma aresta aleatoriamente, trocar seu valor binário e tentar encontrar uma solução viável sem alterar o número de arestas selecionadas. O pseudo-código do método é apresentado no Algoritmo 2.

Algoritmo 2: $N(s)$

```
1  Início
2      a ←aleatório |A|;
3      Se x_a = 0 Então
4          x_a ← 1;
5          Enquanto i < |A| Faça
6              b ← aleatório |A| sendo b ≠ a e x_b = 1;
7              x_b = 0;
8              Se s é inviável Então
9                  x_b ← 1;
10             Fim
11             i ← i + 1;
12         Fim
13     Fim
14     Senão
15         x_a ← 0;
16         Enquanto s é inviável Faça
17             b ← aleatório |A| sendo x_b = 0;
18             x_b ← 1;
19             Se o número de pares O-D cobertos não aumentou Então
20                 x_b ← 0;
21             Fim
22         Fim
23     Fim
24 Fim
```

Uma aresta a é selecionada aleatoriamente (linha 2) e, caso ela não possua um contador instalado, um contador é instalado (linha 4) e, por $|A|$ vezes, outra aresta b é selecionada aleatoriamente tal que b tenha um contador e seja diferente de a (linha 6). A aresta b tem o seu contador removido (linha 7) e caso algum par O-D fique descoberto (solução inviável), o contador é adicionado novamente (linha 9). É importante que a seleção da segunda aresta b seja aleatória (linha 6) ao invés de fazer uma busca sequencial, pois gera uma maior diversidade de soluções.

Já se a tiver um contador instalado (linha 14), esse posto é retirado (linha 15), e arestas aleatórias b tem contadores instalados até que seja encontrada uma solução viável. Caso o contador seja instalado e não "aumente a cobertura" (linha 19), o posto é removido, pois é desnecessário. A condição de que a aresta seja diferente da primeira não foi utilizada pois, caso contrário, o método pode entrar em um ciclo.

3.4. Similaridade

A similaridade entre duas soluções (linha 26 do Algoritmo 1) foi definida pela distância de Hamming (H_i) [Hamming, 1950], que é calculada pelo número de "bits" diferentes (arestas com diferentes valores binários) entre as soluções. Por exemplo se $s_1 = 10010$ e $s_2 = 10001$, a distância é 2. Assim, a solução corrente do SA é assimilada ao *cluster* cujo centro possua a menor distância em relação a ela.

3.5. Busca Local

O pseudo-código é apresentado no Algoritmo 3. Enquanto a solução estiver melhorando, duas arestas são selecionadas aleatoriamente (a e b), com valores diferentes na solução (uma possui contador e outra não) e que trocando o valor das duas a solução continue viável (linhas 3 a 12). Uma vez trocadas duas arestas quaisquer, por $|A|$ vezes, outra aresta c é selecionada (também aleatoriamente), sendo esta diferente das duas primeiras e com um contador instalado. O contador de c é removido e, se a solução for inviável, o contador é adicionado novamente.

Algoritmo 3: $BuscaLocal(s)$

```
1  Início
2      Repita
3          Repita
4              a ← aleatório |A|;
5              b ← aleatório |A| sendo x_b ≠ x_a;
6              x_a ← 1 − x_a;
7              x_b ← 1 − x_b;
8              Se Solução Inviável Então
9                  x_a ← 1 − x_a;
10                 x_b ← 1 − x_b;
11             Fim
12         até Encontrar Solução Viável;
13         Enquanto i < |A| Faça
14             c ← aleatório |A| sendo c ≠ b e c ≠ a e x_c = 1;
15             x_c ← 0;
16             Se Solução Inviável Então
17                 x_c ← 1;
18             Fim
19             i ← i + 1;
20         Fim
21     até Solução s Não Melhore;
22 Fim
```

3.6. Perturbação

A fim de "perturbar" o centro de um *cluster* (linha 30 do Algoritmo 1), foi utilizada a geração de uma solução vizinha por meio da vizinhança definida anteriormente (Seção 3.3).

4. Experimentos Computacionais

Considerando separadamente cada Estado Brasileiro, foi gerada uma instância a partir da base rodoviária georeferenciada brasileira, do ano de 2015, disponibilizada pelo Departamento Nacional de Infraestrutura de Transportes (DNIT), que atualmente está responsável por implantar o Plano Nacional de Contagem de Tráfego (PNCT). Assim, foram definidas 26 instâncias, nas quais cada nó representa um município ou um cruzamentos ao longo das rodovias e a cada aresta representa um segmento de rodovia estadual ou federal presente no estado de cada instância. Cada município foi utilizado para definição dos pares O-D, ou seja, foram consideradas "viagens" (O-D) de todos os municípios para todos os municípios de um dado estado.

A Tabela 1 apresenta as características das instâncias utilizadas sendo OD o número de municípios. Com isso, $|W| = \frac{OD \times (OD-1)}{2}$, ou seja, cada município é considerado uma origem e destino para todos os outros municípios da rede. É importante destacar que o número OD (municípios) é diferente de $|N|$ (nós), pois os cruzamentos existentes nas rodovias são nós no grafo G, porém não são considerados na definição dos pares O-D. A maior instância é a de Minas Gerais (MG), que possui 1917 arestas, 1474 nós, 803 municípios e pouco mais de 322 mil pares O-D.

Para efeito de comparação, foi implementado e calibrado um Algoritmo Genético (AG) de acordo com o proposto no trabalho de Chen et al. [2007]. Foi utilizada a linguagem C (compilador GCC versão 6.3), e os testes foram executados em um computador com processador Intel i7 4.60GHz com 16GB de RAM, e sistema operacional Windows 10. A Tabela 2 apresenta o resultado

da calibração do AG, cujos valores foram utilizados para execução do AG por 10 vezes para cada uma das 26 instâncias.

Para calibração do CS, foram realizados testes com três instâncias de médio porte (AL, MS e RS), pois se mostraram representativas o suficiente, e as de grande porte levariam muito tempo para serem resolvidas. A Tabela 3 apresenta o resultado final do processo de calibração. Com os parâmetros definidos, cada instância foi resolvida 10 vezes com sementes aleatórias em um computador com processador Intel i7 3.4GHz com 16GB de RAM e Windows 7. O código foi desenvolvido em C++ (Dev-C++ 5.11 e GCC 4.9.2).

Tabela 1: Características das instâncias utilizadas

Instância	$\|N\|$	$\|A\|$	OD	$\|W\|$
AC	61	84	20	190
AL	169	219	97	4656
AM	74	77	37	666
AP	52	77	13	78
BA	812	1113	395	77815
CE	401	612	177	15576
ES	283	394	75	2775
GOeDF	799	1165	241	28920
MA	257	355	163	13203
MG	1474	1917	803	322003
MS	343	497	76	2850
MT	711	1069	140	9730
PA	289	370	122	7381
PB	384	480	213	22578
PE	362	472	172	14706
PI	405	550	212	22366
PR	780	1083	381	72390
RJ	502	721	86	3655
RN	333	433	160	12720
RO	185	258	50	1225
RR	75	97	13	78
RS	675	859	391	76245
SC	481	604	266	35245
SE	183	248	74	2701
SP	1280	1683	606	183315
TO	372	524	134	8911

Tabela 2: Parâmetros do Algoritmo Genético

Parâmetro	Valor Final
Tamanho da população	150
Novos indivíduos por geração	50
Taxa de elitismo	8%
Taxa de mutação	2%
Número de gerações	250

Tabela 3: Parâmetros do *Clustering Search*

Parâmetro	Descrição	Valor Final	Valores Testados
γ	Número Máximo de *Clusters*	3	3, 5 e 7
λ	Volume Máximo dos *Clusters*	2	2, 3 e 4
r_{max}	Índice Máximo de Ineficácia dos *Clusters*	3	2, 3 e 4
T_0	Temperatura Inicial	$f(s_0)$	10^2, 10^3 e 10^5
T_f	Temperatura de Congelamento	0,01	-
SA_{max}	Número Máximo de Iterações por Temperatura	$2\|A\|$	-
α	Taxa de Resfriamento	0,975	0,975 e 0,995
$Tempo_{max}$	Tempo Limite (segundos)	7200,00	1800, 3600 e 7200

O valor de T_0 foi definido como sendo o valor da função objetivo da solução inicial. Foram realizados testes com valores empíricos como 10^2, 10^3 e 10^5, porém, os resultados encontrados

foram de baixa qualidade. Quanto ao tempo limite, o valor final foi de 7200 segundos, de modo a possibilitar a convergência para todas as instâncias.

4.1. Resultados

A partir do CS e AG calibrados, foram realizadas 10 execuções de ambos para cada instância. A Tabela 4 apresenta os resultados. A primeira coluna indica o nome (estado de referência) de cada instância e a segunda coluna (Melhor $f(s)$) apresenta o valor da função objetivo da melhor solução encontrada. As terceira ($f(s)$ Média) e quinta (Tempo Médio) colunas apresentam, respectivamente, os valores e tempos médios de solução, e a quarta coluna indica os desvios obtidos $\left(\left(\frac{f(s)\text{Mdia}-\text{Melhor}f(s)}{\text{Melhor}f(s)}\right) \times 100\right)$. Todos os tempos computacionais são apresentados em segundos. A última linha apresenta as médias associadas as 10 execuções do CS em todas as instâncias. Os resultados mostram um CS robusto, com seu alto índice de convergência e desvios muito baixos, sendo que a grande maioria é igual a zero, e o desvio máximo de $0,96\%$.

Tabela 4: Resultados do CS

Instância	**Melhor** $f(s)$	$f(s)$ **Média**	**Desvio**(%)	**Tempo Médio (s)**
AC	30	30,00	0,00	0,03
AL	137	137,00	0,00	0,19
AM	39	39,00	0,00	0,01
AP	22	22,00	0,00	0,03
BA	630	630,00	0,00	527,64
CE	329	329,00	0,00	149,32
ES	144	144,00	0,00	19,92
GOeDF	464	466,20	0,47	948,09
MA	250	250,00	0,00	0,13
MG	1122	1124,80	0,25	6101,23
MS	150	150,00	0,00	603,35
MT	310	311,60	0,52	3705,36
PA	174	174,00	0,00	127,28
PB	296	296,00	0,00	4,83
PE	252	252,00	0,00	45,49
PI	316	316,00	0,00	46,23
PR	599	599,60	0,10	817,11
RJ	166	167,60	0,96	2866,44
RN	233	233,00	0,00	3,00
RO	88	88,00	0,00	4,71
RR	19	19,00	0,00	10,18
RS	544	544,00	0,00	264,98
SC	371	371,00	0,00	137,91
SE	112	112,00	0,00	9,83
SP	877	878,00	0,11	1301,82
TO	231	231,00	0,00	66,13
Média	304,04	304,42	0,09	683,12

4.2. Comparação do CS com o AG

A Tabela 5 apresenta a comparação dos resultados do CS com o melhor resultado de cada uma das 10 execuções do AG de Chen et al. [2007]. A primeira coluna indica o nome de cada instância, e as segunda e terceira colunas apresentam os resultados médios do CS. A quarta e quinta colunas representam os resultados do AG, e a sexta coluna apresenta a diferença percentual entre as soluções do CS e do AG. Todos os tempos de processamento estão em segundos. A última linha apresenta as médias, e os valores de solução em negrito representam o valor da melhor solução apresentada para cada instância.

Diante dos resultados da Tabela 5, nota-se que o CS foi capaz de encontrar soluções médias iguais ou melhores que as do AG em todas as instâncias testadas, com tempos computacionais inferiores. Destaca-se que nas instâncias SP, RJ, MT, MS, e GOeDF, que representam estados com alta densidade de rodovias, o CS apresentou melhores resultados com uma melhora de 4,33 até 8,11% em relação ao AG. Além disso, o AG apresentou resultados iguais em apenas 9 instâncias, possivelmente pelo fato de usar soluções iniciais aleatórias. Com isso, acredita-se que o uso da variação aleatória do C3, ou alguma outra melhor heurística como geradora de soluções iniciais para o AG, possa melhorar consideravelmente o seu desempenho.

Tabela 5: Comparação dos resultados do CS com o AG

Instância	CS		AG		Diferença
	$f(s)$ Média	Tempo Médio	Melhor $f(s)$	Tempo	(%)
AC	**30**	0,03	**30**	2,56	0,00
AL	**137**	0,19	**137**	18,90	0,00
AM	**39**	0,01	**39**	2,01	0,00
AP	**22**	0,03	**22**	2,11	0,00
BA	**630**	527,64	645	2591,47	2,38
CE	**329**	149,32	337	283,97	2,43
ES	**144**	19,92	148	64,99	2,78
GOeDF	**466,2**	948,09	496	1292,24	8,11
MA	**250**	0,13	**250**	73,28	0,00
MG	**1124,8**	6101,23	1136	9731,56	1,53
MS	**150**	603,35	158	113,01	6,67
MT	**311,6**	3705,36	329	741,22	5,58
PA	**174**	127,28	178	68,43	2,30
PB	**296**	4,83	**296**	141,24	0,00
PE	**252**	45,49	255	140,76	1,19
PI	**316**	46,23	322	209,51	1,90
PR	**599,6**	817,11	616	2443,91	2,74
RJ	**167,6**	2866,44	176	304,34	6,21
RN	**233**	3,00	238	105,63	2,15
RO	**88**	4,71	**88**	28,56	2,27
RR	**19**	10,18	**19**	5,93	0,00
RS	**544**	264,98	549	1322,20	0,92
SC	**371**	137,91	373	323,29	0,54
SE	**112**	9,83	**112**	26,08	0,89
SP	**878**	1301,82	909	5213,28	4,33
TO	**231**	66,13	240	144,68	4,33
Média	304,42	683,12	311,50	976,74	2,68

5. Conclusões

Este trabalho apresentou uma nova alternativa para resolução do Problema de Localização de Contadores de Tráfego (PLCT) em redes de transporte rodoviário. O PLCT é um problema de otimização combinatória complexo que envolve redes de grandes dimensões e possui características e abordagens muito variadas.

Para resolução do PLCT, foi proposta uma aplicação da meta-heurística *Clustering Search* (CS), que vem apresentando bons resultados na resolução de outros problemas combinatoriais e que ainda não foi utilizada para resolução do PLCT.

Para avaliar o desempenho do método proposto, foram utilizados dados reais da malha rodoviária brasileira, e os resultados obtidos demonstram a eficiência do método, uma vez que soluções com baixo desvio foram obtidas em baixo tempo computacional, superando um AG proposto na literatura. Logo, o CS pode ser considerado como uma nova e boa alternativa para resolução do PLCT.

Agradecimentos: Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e

Tecnológico - CNPq (processos 313408/2014-9, 454569/2014-9 e 301725/2016-0), à Fundação de Amparo à Pesquisa e Inovação do Espírito Santo - FAPES (processos 67627153/2014 e 73290475/2016) pelo apoio financeiro e ao Departamento Nacional de Infraestrutura de Transportes - DNIT pela disponibilização dos dados.

Referências

Ahuja, R. K., Magnanti, T. L., e Orlin, J. B. (1993). *Network flows: theory, algorithms, and applications*. Prentice hall.

Ashok, K. e Ben-Akiva, M. E. (1993). Dynamic origin-destination matrix estimation and prediction for real-time traffic management systems. In *XII International Symposium on the Theory of Traffic Flow and Transportation*, Berkeley.

Bell, M. G. (1984). The estimation of junction turning volumes from traffic counts: the role of prior information. *Traffic Engineering & Control*, 25(5):279–283.

Bell, M. G. e Shield, C. M. (1995). A log-linear model for path flow estimation. In *Applications of Advanced Technologies in Transportation Engineering*, p. 695–699. ASCE - American Society of Civil Engineers.

Bell, M. G., Shield, C. M., Busch, F., e Kruse, G. (1997). A stochastic user equilibrium path flow estimator. *Transportation Research Part C: Emerging Technologies*, 5(3–4):197–210.

Bertsekas, D. P. (1998). *Network optimization: continuous and discrete models*. Athena Scientific.

Bianco, L., Cerrone, C., Cerulli, R., e Gentili, M. (2014). Locating sensors to observe network arc flows: exact and heuristic approaches. *Computers & Operations Research*, 46:12–22.

Cascetta, E. (1984). Estimation of trip matrices from traffic counts and survey data: a generalized least squares estimator. *Transportation Research Part B: Methodological*, 18(4–5):289–299.

Chen, A., Chootinan, P., e Recker, W. (2005). Examining the quality of synthetic origin–destination trip table estimated by path flow estimator. *Journal of Transportation Engineering*, 131(7):506–513.

Chen, A., Pravinvongvuth, S., Chootinan, P., Lee, M., e Recker, W. (2007). Strategies for selecting additional traffic counts for improving od trip table estimation. *Transportmetrica*, 3(3):191–211.

Chootinan, P., Chen, A., e Recker, W. (2005). Improved path flow estimator for origin-destination trip tables. *Transportation Research Record: Journal of the Transportation Research Board*, p. 9–17.

Fisk, C. S. (1988). On combining maximum entropy trip matrix estimation with user optimal assignment. *Transportation Research Part B: Methodological*, 22(1):69–73.

Garber, N. J. e Hoel, L. A. (1999). *Traffic and highway engineering*. CL Engineering.

Goldberg, D. E. e Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning*, 3:95–99.

González, P. H., Clímaco, G., Simonetti, L., Gomes, H. A., e Ribeiro, G. M. (2016). Heurísticas para o problema de localização de contadores de tráfego em redes de transporte. In *Anais do XXX ANPET - Congresso de Pesquisa e Ensino em Transportes*.

Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell Labs Technical Journal*, 29(2):147–160.

Hazelton, M. L. (2000). Estimation of origin–destination matrices from link flows on uncongested networks. *Transportation Research Part B: Methodological*, 34(7):549–566.

Kirkpatrick, S., Gelatt, C. D., e Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Maher, M. J. (1983). Inferences on trip matrices from observations on link volumes: a bayesian statistical approach. *Transportation Research Part B: Methodological*, 17(6):435–447.

Maher, M. J., Zhang, X., e Van Vliet, D. (2001). A bi-level programming approach for trip matrix estimation and traffic control problems with stochastic user equilibrium link flows. *Transportation Research Part B: Methodological*, 35(1):23–40.

Oliveira, A. C. M., Chaves, A. A., e Lorena, L. A. N. (2013). Clustering search. *Pesquisa Operacional*, 33(1):105–121.

Oliveira, A. C. M. e Lorena, L. A. N. (2007). Hybrid evolutionary algorithms and clustering search. In *Hybrid Evolutionary Algorithms*, p. 77–99. Springer.

Ribeiro, G. M., Laporte, G., e Mauri, G. R. (2012). A comparison of three metaheuristics for the workover rig routing problem. *European Journal of Operational Research*, 220:28–36.

Sherali, H. D., Sivanandan, R., e Hobeika, A. G. (1994). A linear programming approach for synthesizing origin-destination trip tables from link traffic volumes. *Transportation Research Part B: Methodological*, 28(3):213–233.

Spiess, H. (1987). A maximum likelihood model for estimating origin-destination matrices. *Transportation Research Part B: Methodological*, 21(5):395–412.

Van Zuylen, H. J. e Willumsen, L. G. (1980). The most likely trip matrix estimated from traffic counts. *Transportation Research Part B: Methodological*, 14(3):281–293.

Yang, H., Gan, L., e Tang, W. H. (2001). Determing cordons and screen lines for origin-destination trip studies. *Proceedings of the Eastern Asia Society for Transportation Studies*.

Yang, H. (1995). Heuristic algorithms for the bilevel origin-destination matrix estimation problem. *Transportation Research Part B: Methodological*, 29(4):231–242.

Yang, H., Iida, Y., e Sasaki, T. (1991). An analysis of the reliability of an origin-destination trip matrix estimated from traffic counts. *Transportation Research Part B: Methodological*, 25(5): 351–363.

Yang, H., Sasaki, T., Iida, Y., e Asakura, Y. (1992). Estimation of origin-destination matrices from link traffic counts on congested networks. *Transportation Research Part B: Methodological*, 26 (6):417–434.

Yang, H., Yang, C., e Gan, L. (2006). Models and algorithms for the screen line-based traffic-counting location problems. *Computers & Operations Research*, 33(3):836–858.

Yang, H. e Zhou, J. (1998). Optimal traffic counting locations for origin–destination matrix estimation. *Transportation Research Part B: Methodological*, 32(2):109–126.