Aplicação da otimização por colônia de formigas ao problema de múltiplos caixeiros viajantes no atendimento de ordens de serviço nas empresas de distribuição de energia elétrica

Alternative Title: Applying a variation of the ant colony optimization algorithm to solve the multiple traveling salesmen problem to route the teams of the electric power distribution companies

Denilson F. Barbosa
Programa de Pós-Graduação em Informática
Universidade Tecnológica Federal do Paraná
denilsonbarbosa@utfpr.edu.br

Carlos N. Silla Jr.
Programa de Pós-Graduação em Informática
Universidade Tecnológica Federal do Paraná
carlosjunior@utfpr.edu.br

André Y. Kashiwabara
Programa de Pós-Graduação em Informática
Universidade Tecnológica Federal do Paraná
kashiwabara@utfpr.edu.br

RESUMO

Este artigo apresenta uma metodologia para otimização do atendimento comercial nas empresas de distribuição de energia elétrica, atividade que representa uma parcela significativa dos custos operacionais dessas empresas. Como distribuir as ordens de serviço entre as equipes de atendimento e como definir rotas eficientes são problemas complexos devido à grande quantidade de combinações possíveis. Através da criação de instâncias de Problemas de Múltiplos Caixeiros Viajantes a partir das posições de execução das ordens e da aplicação de um algoritmo baseado na Otimização por Colônia de Formigas nessas instâncias, foram obtidas soluções otimizadas para distribuição das ordens e roteamento das equipes. O método desenvolvido foi aplicado a 17 instâncias construídas a partir de dados reais de uma agência de atendimento e mostrou-se eficiente ao diminuir em média 42,23% das distâncias percorridas pelas equipes em comparação com as rotas reais. Para possibilitar a replicação dos experimentos realizados, o código fonte do protótipo desenvolvido e os dados reais utilizados encontram-se disponíveis em https://github.com/denilsonfag/STACS.

Palavras-Chave

Distribuição de energia elétrica, problema de múltiplos caixeiros viajantes, otimização por colônia de formigas.

ABSTRACT

This paper presents a methodology to optimize the execution of the commercial services of electric power distribution companies. This activity represents a significant portion of the operational costs of these companies. How to distribute the services to the teams and define efficient routes is a complex problem due to the great number of possible combinations. The set of geographical positions of the services can be seen as an instance of the multiple traveling salesmen problem. We created a variation of the ant colony optimization algorithm to obtain optimized solutions for the distribution of the services and the route of each team. Our methodology was applied to 17 real word instances obtained from an electric power distribution company from the city of Cornélio Procópio, Brazil. Our method obtained an improvement of 42,23% on average when compared to the solutions that were performed in the real world. To allow the reproduction of our results, the source-code of our system and the dataset are freely available at https://github.com/denilsonfag/STACS.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*heuristic methods*; G.1.6 [**Numerical Analysis**]: Optimization—*constrained optimization, integer programming, stochastic programming*

General Terms

Algorithms, Experimentation

Keywords

Distribution of electric power, multiple traveling salesmen problem, ant colony optimization.

1. INTRODUÇÃO

As principais atividades das empresas de distribuição de energia elétrica são a manutenção do sistema elétrico de distribuição e a gestão comercial de seus consumidores. Para a realização destas atividades, equipes técnicas percorrem diariamente a área sob concessão dessas empresas, o que representa uma parcela significativa de seus custos operacionais. Estes custos podem ser reduzidos com a aplicação de uma metodologia que utilize de forma eficiente as informações disponíveis para gerenciamento das equipes, como a apresentada neste artigo.

Em geral, a solução adotada nas empresas para gerenciamento do atendimento comercial é a execução dessa atividade por técnicos que realizam manualmente a análise das informações disponíveis ao distribuir as ordens entre as equipes no início dos dias de trabalho. Após distribuídas as ordens, cabe a cada equipe definir também manualmente a sequência em que realizará seus atendimentos. Tanto a distribuição das ordens quanto as sequências de atendimento admitem um grande número de combinações, o que as tornam tarefas difíceis de serem otimizadas sem o auxílio de uma ferramenta computacional.

A partir de informações obtidas da agência de atendimento de Cornélio Procópio, estado do Paraná, este trabalho apresenta uma metodologia para realizar a distribuição das ordens e criar rotas otimizadas para as equipes simultaneamente, através da aplicação de um algoritmo de otimização por colônia de formigas em instâncias de problemas de múltiplos caixeiros viajantes representando os dias de trabalho. Os experimentos realizados comprovam a eficácia da metodologia ao reduzir em 42,23%, em média, o deslocamento total das equipes em um dos experimentos realizados.

O restante deste trabalho está organizando da seguinte forma: na seção 2 são apresentados os trabalhos relacionados ao problema real e à abordagem deste artigo. Na seção 3 é detalhada a metodologia desenvolvida e na seção 4 são apresentados os experimentos realizados e os resultados obtidos. A seção 5 analisa a eficácia deste trabalho e aponta as questões a serem consideradas na sua continuação.

2. REFERENCIAL TEÓRICO

Na literatura, o trabalho de Costa et al. [3] aborda o problema da distribuição de ordens de serviço entre as equipes através da análise do histórico de atendimento de uma agência, buscando definir a melhor forma de particionar a área de atendimento para equilibrar a carga de trabalho das equipes. O problema é modelado como um como um Problema de p-Medianas Capacitado e é utilizada a Programação Linear Inteira e um Algoritmo Genético específico para tratá-lo.

A proposta nos trabalhos de Garcia et al. [7] e Mascia et al. [11] para distribuir as ordens entre as equipes também é através do Problema de Agrupamento Capacitado (outra denominação do Problema de p-Medianas Capacitado), só que os agrupamentos são realizados diretamente sobre as ordens a executar, definidas por suas posições. Nesta abordagem, após distribuídas as ordens entre as equipes, ainda cabe a cada uma delas definir manualmente a sequência em que realizará seus atendimentos. Uma maneira de distribuir as ordens entre as equipes e definir as sequências de atendimento simultaneamente é abordar o problema como um Problema de Múltiplos Caixeiros Viajantes, que é explicado na seção 2.1.

2.1 O Problema de Múltiplos Caixeiros Viajantes

O TSP (*Travelling Salesman Problem*, Problema do Caixeiro Viajante) é um dos problemas de otimização combinatória mais pesquisados, que se destaca por ser de fácil declaração mas de difícil resolução, pertencendo à classe dos problemas NP-completos [1]. Em sua definição clássica, em uma instância do problema existem n cidades a serem visitadas por um caixeiro, que objetiva realizar a menor rota possível que passe por todas as cidades exatamente uma vez e retorne à cidade de partida. Várias aplicações do mundo real podem ser modeladas como instâncias de um TSP, como rotas de transporte escolar, sequenciamento de DNA, manufatura de circuitos de microchips e inspeções em plataformas de petróleo [9].

Uma instância de TSP pode ser representada em um grafo completamente conectado e não direcionado $G = (V, A)$, no qual V é o conjunto de n vértices representando as cidades e A é o conjunto de arestas que conectam todas as cidades entre si. Cada elemento de A possui um custo associado, que geralmente é uma medida de distância ou tempo, os quais são organizados numa matriz C na qual cada elemento c_{ij} é o custo do deslocamento iniciando no vértice $i \in V$ até o vértice $j \in V$. A matriz de custos é dita simétrica quando $c_{ij} = c_{ji}, \forall\ i, j \in V$, ou assimétrica caso contrário.

O MTSP (*Multiple Traveling Salesman Problem*, Problema de Múltiplos Caixeiros Viajantes) é uma generalização do TSP na qual mais de um caixeiro participa da solução. Como explica Bektas [2], o MTSP é mais apropriado para aplicações do mundo real que o TSP e pode ser estendido para uma grande variedade de VRPs (*Vehicle Routing Problem*, Problema de Roteamento de Veículos) com a adição de restrições às soluções. VRP é o nome dado a toda uma classe de problemas envolvendo a visita de clientes por veículos.

Na definição geral do MTSP, existem $m > 1$ caixeiros inicialmente posicionados em um vértice comum, normalmente indexado como o primeiro vértice da instância, $v_0 \in V$, e denominado depósito. Os demais vértices da instância são chamados de intermediários. O objetivo é minimizar o custo total da solução, que corresponde aos custos das rotas de todos os caixeiros somados, sendo que todas as rotas devem começar e terminar no depósito e todos os nós intermediários devem ser visitados exatamente uma vez. Neste artigo, o termo "rota" é empregado ao conjunto de cidades visitadas por um caixeiro ordenadas pelo instante da visita, e o termo "solução" é empregado para uma solução completa do problema.

Entre as formas possíveis de modelar o MTSP, a mais comum é a formulação baseada em atribuição utilizando a programação linear inteira, como a apresentada a seguir, extraída de [2]. Nessa formulação, n corresponde ao número de elementos de V, que é o conjunto de vértices da instância incluindo o depósito; m é o número de caixeiros; i e j são vértices da instância; x_{ij} é a variável binária que informa se a aresta (i, j) faz parte da solução do problema; c_{ij} é o custo associado à aresta (i, j); e S é o número de elementos dos subconjuntos de V, que constituem os vértices atribuídos a cada caixeiro. Na criação do modelo, inicialmente é definida a variável binária x_{ij}:

$$x_{ij} = \begin{cases} 1 & \text{se a aresta } (i,j) \text{ é usada na solução,} \\ 0 & \text{caso contrário.} \end{cases} \quad (1)$$

A função objetivo é:

$$Min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \quad (2)$$

Sujeita às seguintes restrições:

$$\sum_{j=2}^{n} x_{1j} = m \quad (3)$$

$$\sum_{j=2}^{n} x_{j1} = m \quad (4)$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 2, ..., n \quad (5)$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 2, ..., n \quad (6)$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq 1, \quad \forall\, S \subseteq V \backslash \{1\}, \quad S \neq 0 \quad (7)$$

$$x_{ij} \in \{0,1\}, \forall\, i, j \in V \quad (8)$$

As restrições 3 e 4 garantem que exatamente m caixeiros farão parte da solução. As restrições 5, 6 e 8 representam as regras de atribuição e a restrição 7 previne a criação de subconjuntos de cardinalidade S que não contenham o depósito.

Uma variação importante do MTSP está relacionada ao objetivo do problema. Okonjo-Adigwe [12] aponta que o MTSP sem restrições não considera importantes parâmetros de problemas reais, entre eles o equilíbrio da carga de trabalho (*workload balance*) entre os caixeiros. Para buscar o equilíbrio entre os custos das rotas em um MTSP, ao invés de minimizar o custo total da solução, o custo a ser minimizado passa a ser o da maior rota da solução, que é composta por m rotas, uma de cada caixeiro.

2.2 A Otimização por Colônia de Formigas

A Otimização por Colônia de Formigas (*Ant Colony Optmization*, ACO) está entre as principais meta-heurísticas usadas para o TSP [9]. Ela tem alcançado resultados consistentes para problemas de otimização combinatória, recebendo esta denominação por ser baseada na maneira como colônias de formigas reais encontram rotas mais curtas até suas fontes de alimento. A partir do primeiro algoritmo ACO, denominado *Ant System* [5], que obteve resultados encorajadores para o TSP, a ACO tem sido aplicada com sucesso para vários problemas NP-difíceis de otimização combinatória [6].

Na ACO, as soluções são geradas num processo iterativo no qual agentes inteligentes, as formigas, percorrem todas as cidades de uma instância de TSP, por exemplo, escolhendo a cada iteração a próxima cidade de sua rota de maneira estocástica através da Regra de Transição de Estado (RTE). A RTE do *Ant System* é mostrada na equação 9:

$$p_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}} & \text{se } j \text{ é uma aresta permitida à } k \\ 0 & \text{caso contrário} \end{cases} \quad (9)$$

Na equação 9, que determina a probabilidade de um vértice ser escolhido por uma formiga, i representa o vértice no qual a formiga k se encontra, e j um dos possíveis vértices para o qual ela pode se deslocar. Os vértices permitidos à formiga são aqueles ainda não visitados até a fase atual da construção da solução. Cada formiga armazena os nós já visitados em sua memória, que é denominada como lista tabu. $\tau_{ij}(t)$ corresponde à quantidade de feromônio presente na aresta entre os vértices i e j no instante t. η_{ij} corresponde à visibilidade da aresta, que é definida pelo valor $1/c_{ij}$, onde c_{ij} é o custo de deslocamento da aresta. α e β são parâmetros que definem o peso da trilha de feromônio e da visibilidade, respectivamente, na escolha do próximo vértice pela formiga.

2.2.1 O Ant Colony System

O ACS (*Ant Colony System*, Sistema de Colônia de Formigas) é um dos vários algoritmos que surgiram a partir de aperfeiçoamentos no *Ant System* (AS), entre eles o *Elitist AS*, o *Ant-Q*, o *MAX-MIN AS* e o *Rank-based AS* [6]. O ACS difere do AS em três pontos principais: utiliza uma RTE mais agressiva; a RAF (Regra de Atualização de Feromônio) determina que a evaporação e o depósito de feromônio, ao final de cada ciclo, acontecem apenas nas arestas pertencentes à *best-so-far solution*, ou seja, a melhor solução encontrada até o momento em uma execução do algoritmo; e, a cada vez que uma formiga utiliza uma aresta numa solução, uma quantidade pré-definida de feromônio é removida da aresta, para aumentar a exploração de caminhos alternativos. Outra diferença importante é em relação ao número N de formigas que percorrem o grafo e constroem suas próprias soluções simultaneamente: enquanto no *Ant System* o melhor valor encontrado experimentalmente é $N = n$, para o ACS o melhor valor encontrado é $N = 10$.

$$j = \begin{cases} \text{argmax}_{l \in S_i^k}\{\tau_{il}[\eta_{il}]^{\beta}\}, & \text{se } q \leq q_0; \\ J, & \text{caso contrário.} \end{cases} \quad (10)$$

A RTE do ACS está mostrada na equação 10, na qual j representa a cidade escolhida por uma formiga k que se encontra no vértice i ao se movimentar. Esta equação é chamada de regra proporcional pseudo-aleatória, pois o parâmetro q_0 define a porcentagem das escolhas que serão feitas de forma determinística pelas formigas, onde $0 \leq q_0 \leq 1$. Sendo q uma variável aleatória uniformemente distribuída em $[0, 1]$ e atualizada a cada movimento, se $q_0 = 1$, todas as escolhas das formigas serão realizadas de forma determinística, pelo valor máximo de $\tau_{il}[\eta_{il}]^{\beta}$, onde $l \in S_i^k$ corresponde ao conjunto de cidades permitidas à formiga no movimento. Se, ao contrário, $q_0 = 0$, todas as escolhas serão realizadas de forma estocástica, pois J representa uma cidade escolhida através das probabilidades calculadas pela equação 9, com a única diferença que o parâmetro α foi excluído, ou seja, seu valor foi fixado em 1. O melhor valor definido experimentalmente é $q_0 = 0.9$, o que significa que 90% das escolhas serão determinísticas.

A RAF do ACS define que a atualização de feromônio ocorre em dois momentos: uma atualização global, ao final de cada ciclo do algoritmo; e uma atualização local, logo depois que uma formiga se move de uma cidade a outra, ou seja, inclui mais uma aresta na sua rota. A regra para Atualização Global de Feromônio (AGF) está apresentada nas equações 11 e 12, onde ρ é o coeficiente de persistência

do feromônio, cujo melhor valor encontrado para o ACS é $\rho = 0.1$. T^{bs} corresponde ao conjunto de arestas que fazem parte da melhor solução encontrada até o momento atual da execução do algoritmo e C^{bs} corresponde ao custo de T^{bs}.

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \rho\Delta\tau_{ij}^{bs}, \quad \forall(i,j) \in T^{bs}, \tag{11}$$

$$\Delta\tau_{ij}^{bs} = 1/C^{bs} \tag{12}$$

A regra de Atualização Local de Feromônio (ALF) está apresentada na equação 13, na qual ξ é um parâmetro cujo melhor valor experimentalmente calculado é $\xi = \rho = 0.1$. τ_0 também é um parâmetro que, além de ser utilizado na atualização local de feromônio, também corresponde à quantidade de feromônio depositada em todas as arestas no início da execução do algoritmo. O melhor valor encontrado para τ_0 é calculado a partir do custo de uma solução criada aplicando um algoritmo do vizinho mais próximo à instância em análise. A fórmula para cálculo de τ_0 é mostrada na equação 14, na qual n é igual ao número de cidades da instância e C^{nn} é igual ao custo da solução obtido utilizando um algoritmo do vizinho mais próximo sobre a instância.

$$\tau_{ij} = (1 - \xi)\tau_{ij} + \xi\tau_0, \tag{13}$$

$$\tau_0 = 1/nC^{nn} \tag{14}$$

2.2.2 O Team Ant Colony Optmization

O TACO (*Team Ant Colony Optmization*, Otimização por Colônia com Equipes de Formigas) é o algoritmo desenvolvido por Vallivaara [13] que adapta o ACS para instâncias de MTSP. Isto é feito substituindo cada formiga do ACS, que gera soluções para o TSP, por um time de formigas formado por m elementos, onde cada formiga de um time corresponde a um caixeiro na construção de uma solução para MTSP, de forma que cada time possui sua própria lista tabu.

No início da construção da solução, todas as formigas de todos os times são posicionadas no depósito. Para realizar a distribuição da carga de trabalho, a formiga que tem a menor rota parcial, em qualquer momento da construção do solução, escolhe sua próxima cidade de acordo com a regra definida na equação 10. Segundo Vallivaara [13], este método é eficiente para a distribuição da carga de trabalho, mas em muitos casos as escolhas das formigas tendem para soluções não ótimas. Para tratar este problema, é verificado a cada escolha se, caso a cidade escolhida fosse cedida a qualquer outra formiga do time, não haveria um tamanho total menor da solução, considerando também o retorno ao depósito. Se houver, é a formiga com menor rota parcial que se movimenta, podendo realizar uma nova escolha do seu próximo vértice a partir da regra 10.

Matematicamente, estando uma formiga k posicionada em um vértice v_k; sendo v_0 o depósito da instância; $c(v_i, v_j)$ o custo de deslocamento entre dois vértices v_i e v_j quaisquer; $CP(k)$ o custo parcial total da rota já percorrida pela formiga k; e tendo sido escolhido o vértice v_j como seu próximo vértice de acordo com a regra 10; o método consiste em verificar, antes da formiga de mover, se existe qualquer outra formiga l da instância em que

$$c(v_l, v_j) + c(v_j, v_0) + CP(l) < c(v_k, v_j) + c(v_j, v_0) + CP(k) \tag{15}$$

Se houver, é permitido que a formiga l escolha seu próximo vértice novamente de acordo com a equação 10, e se movimente para o novo vértice escolhido.

Outra técnica utilizada por Vallivaara [13] é a aplicação de algoritmos de buscas locais para melhoramento das soluções geradas. A busca local 2-opt, que tenta trocar duas arestas quaisquer da solução, é aplicada a todas as soluções geradas. A busca local 3-opt, que tenta comutar quaisquer três arestas de uma solução ao mesmo tempo, é aplicada apenas à melhor solução dentre as N geradas simultaneamente pelos times. Para a otimização das instâncias de MSPT geradas a partir do problema real em estudo, foi desenvolvido um protótipo que implementa uma variação do TACO de Vallivaara [13], que é explicado na seção 3.

3. METODOLOGIA

A metodologia desenvolvida consiste na criação de instâncias de MTSP a partir de cenários estáticos do ambiente em estudo e na aplicação de um algoritmo ACO a estas instâncias para criação de soluções otimizadas para distribuição das ordens e roteamento das equipes. O ambiente analisado é o atendimento no município de Cornélio Procópio, estado do Paraná, que possui 637,95 km^2 de extensão territorial e 19.276 consumidores de energia elétrica, dos quais 3,76% estão localizados na área rural [8]. As equipes de atendimento ficam concentradas na mesma garagem e estão aptas a executar qualquer tipo de ordem de serviço, não havendo, portanto, restrições quanto à habilidade das equipes na distribuição das ordens.

Na figura 1 estão destacadas em vermelho as 31 ordens de serviço executadas no município em 3 de fevereiro de 2014, um dia de trabalho típico. O ponto em azul representa a garagem da agência de atendimento, local onde as equipes normalmente iniciam e concluem suas rotas. Nos eixos da figura 1 estão representadas as coordenadas no sistema de projeção Universal Transverso de Mercator (UTM), que definem as posições da garagem e das ordens executadas. O UTM é o sistema de coordenadas cartesianas bidimensional oficial brasileiro para mapeamentos [10]. Essas coordenadas foram obtidas do histórico da agência de atendimento, a partir do qual também foi possível obter a sequência em que as equipes executaram os serviços, o que tornou possível o cálculo das rotas reais realizadas.

Para criar e otimizar soluções para instâncias de MTSP como a da figura 1, foi desenvolvido na linguagem de programação C++ um protótipo que implementa um algoritmo ACO denominado STACS (*Single Team Ant Colony System*, Sistema de Colônia com Equipe Única de Formigas). O STACS é baseado no *Team Ant Colony Optmization* (TACO) [13], que por sua vez adapta o *Ant Colony System* (ACS) [4] para o MTSP. Esta implementação é apresentada no algoritmo 1. A principal diferença entre os dois algoritmos é que no TACO N soluções são geradas simultaneamente por N times, enquanto que no protótipo desenvolvido as soluções são geradas uma a uma e, quando N soluções são geradas, o que corresponde ao final de um ciclo, ocorre a atualização global de feromônio. Outra diferença é em relação à busca local 3-opt, que não é aplicada às soluções geradas pelo STACS devido ao tempo computacional exigido.

O protótipo desenvolvido aceita como entrada instâncias de MTSP representadas por coordenadas planas, como a da figura 1, para as quais são calculadas as respectivas matrizes de custos com as distâncias euclidianas entre os pontos;

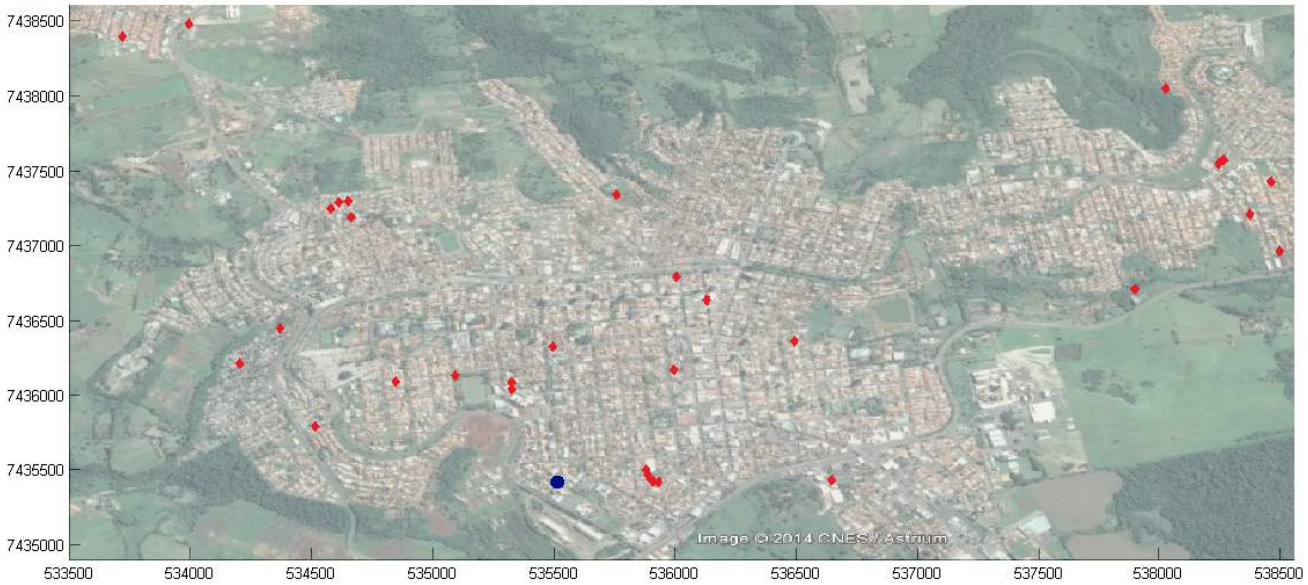

Figura 1: Ordens de serviço executadas no município de Cornélio Procópio em um dia de trabalho típico (pontos em vermelho) e a garagem da agência de atendimento (ponto em azul). Fontes: imagem de satélite de *Google Maps* e coordenadas dos pontos de COPEL Distribuição.

Algoritmo 1: STACS (*Single Team Ant Colony System*)

Entrada: Instância MTSP e parâmetros ACS
Saída: Melhor solução MTSP criada
Início
  $m \leftarrow$ número de caixeiros da instância;
  $n \leftarrow$ número de cidades da instância;
  $v_0 \leftarrow$ índice do vértice depósito da instância;
  $N \leftarrow$ número de soluções geradas em cada ciclo;
  Inicialize a matriz de feromônio; `//equação 14`
  Enquanto *o critério de parada não for atingido* **faça**
    Para *contador_de_soluções* $\leftarrow 1$ *até* N **faça**
      Esvazie a lista tabu da colônia;
      Posicione m formigas em v_0;
      Insira v_0 na lista tabu;
      Enquanto *houver vértices não visitados* **faça**
        $k \leftarrow$ formiga com menor rota parcial;
        $v_j \leftarrow$ próximo vértice de k; `//equação 10`
        Verifique o movimento de k; `//inequação 15`
        se *houver uma formiga* l *melhor que* k **então**
          $k \leftarrow l$;
          $v_j \leftarrow$ próximo vértice de k; `//equação 10`
        fim
        $v_i \leftarrow$ vértice atual de k;
        Movimente a formiga k para o vértice v_j;
        Insira v_j na lista tabu;
        Aplique ALF à aresta (v_i, v_j); `//equação 13`
      Fim
      Movimente todas as m formigas para v_0;
      Monte a solução MTSP criada em S_{atual};
      Aplique a busca local 2-opt à S_{atual};
      Atualize a melhor solução do ciclo S_{m_ciclo};
    Fim
    Atualize a melhor solução da execução S_{m_exec};
    Aplique a AGF à S_{m_exec}; `//equações 11 e 12`
  Fim
  Retorne S_{m_exec};
Fim

e instâncias representadas por matrizes de custo assimétricas, o que permite a utilização de custos que representem melhor o ambiente real da instância em análise, como tempos estimados de deslocamento, por exemplo. A distância euclidiana também é chamada de distância métrica e, para dois pontos i e j representados por coordenadas planas x e y, seu valor é igual a $[(x_i - x_j)^2 + (y_i - y_j)^2]^{1/2}$ [5]. Nos experimentos apresentadas na seção 4 são utilizadas matrizes com as distâncias euclidianas nas otimizações.

4. EXPERIMENTOS COMPUTACIONAIS

Os experimentos apresentados nesta seção compravam a eficácia da metodologia desenvolvida aplicada ao problema real. Utilizando dados de uma agência de atendimento, são realizados dois experimentos. No primeiro, o protótipo é configurado para minimizar o custo total das soluções, sem considerar a distribuição da carga de trabalho entre as equipes, como na descrição geral do MTSP. No segundo experimento, o protótipo minimiza o custo da maior rota individual das soluções, objetivando a construção de soluções formadas por rotas com custos equilibrados entre os caixeiros, como no MTSP com equilíbrio da carga de trabalho. Para possibilitar a replicação destes experimentos, em https://github.com/denilsonfag/STACS encontram-se disponíveis o código fonte do protótipo desenvolvido e os dados reais utilizados.

4.1 Detalhes experimentais

As ordens de serviço da base de dados reais foram separadas quanto ao seu dia de execução e, a partir das coordenadas UTM que indicam suas posições de execução, foi montada uma instância de MTSP para cada dia de trabalho. Além das coordenadas das ordens, as instâncias contém as coordenadas da agência de atendimento (obtidas através de uma consulta no *Google Maps* a partir de seu endereço)

e o número de equipes em serviço no dia, que corresponde ao número m de caixeiros.

Para a confecção das matrizes de custo que representam as instâncias, foram utilizadas as distâncias euclidianas, as quais foram calculadas a partir das coordenadas UTM dos pontos divididas por 1000, o que gera valores em quilômetros, já que cada unidade no sistema de projeção UTM corresponde a um metro na superfície da Terra [10].

4.1.1 Base de dados reais

A base de dados reais utilizada nos experimentos consiste no histórico de atendimento do dia 1ºde fevereiro a 23 de fevereiro de 2014 do município de Cornélio Procópio. Os dados foram obtidos a partir do sistema de gerenciamento das ordens de serviço da COPEL Distribuição que, dentre outras funções, armazena as informações sobre as ordens desde a sua geração até a sua conclusão. Os atendimentos realizados pelas equipes nos 23 dias está resumido na tabela 1, que traz a divisão das ordens de acordo com seu dia de execução e o número de equipes em serviço em cada dia. Nos dias 1º, 2, 8, 9 e 23, havia apenas uma equipe em serviço, o que gera instâncias para TSP, pois $m = 1$. Como o foco deste artigo é o MTSP, estes dias de trabalho não foram utilizados nos experimentos, apesar de isto ser possível no protótipo desenvolvido. No dia 16 não houve nenhum atendimento, sendo também excluído da base de dados. Dessa forma, os experimentos foram realizados sobre os 17 dias de trabalho restantes.

Tabela 1: Ordens de serviço executadas no município de Cornélio Procópio de 1ºde fevereiro a 23 de fevereiro de 2014. Fonte: COPEL Distribuição.

Dia de trabalho	Equipes	Ordens de serviço
1	1	7
2	1	1
3	3	31
4	4	38
5	4	40
6	4	33
7	4	40
8	1	4
9	1	1
10	3	37
11	4	27
12	4	53
13	4	35
14	4	33
15	2	12
16	0	0
17	2	26
18	4	64
19	4	35
20	4	30
21	4	36
22	2	9
23	1	5

A partir do histórico de atendimento foram obtidas as coordenadas UTM de todas as ordens executadas, que possibilitam a construção de instâncias de MTSP representando os 17 dias de trabalho. Analisando o horário de execução das ordens, também disponível no histórico, foi possível definir a sequência real dos atendimentos nos dias de trabalho, que corresponde a uma solução para MTSP, cujos custos calculados pelo protótipo com a utilização das distâncias euclidianas servem de referência para a avaliação dos resultados obtidos e estão mostrados nas tabelas 2 e 4.

4.1.2 Protocolo experimental

Em cada um dos experimentos foram realizadas 100 execuções independentes do algoritmo STACS para cada uma das 17 instâncias. O algoritmo foi implementado em C++ e compilado com o pacote de desenvolvimento *Cygwin 1.7.32*. Foi utilizado um equipamento de 64 bits executando o sistema operacional *Windows 8.1* em um processador com dois núcleos de 2,40 GHz e 8 GB de memória RAM. As sementes dos números pseudo-randômicos de todas as execuções foram armazenadas no arquivo de *log* do experimento para possibilitar a replicação dos resultados.

O critério de parada do algoritmo STACS foi de 1000 iterações em cada execução. O conjunto de parâmetros utilizado foi : $q_0 = 0.9$, $\alpha = 1$, $\beta = 2$ e $\xi = \rho = 0.1$. Foram geradas $N = 10$ soluções a cada ciclo. Esses valores foram os melhores obtidos experimentalmente por [6] ao aplicar o ACS a um grande conjunto de instâncias euclidianas de TSP e também foram os mesmos utilizados por [13] em seus experimentos.

4.2 Minimização do custo total das soluções

O primeiro experimento executado com os dados e o protocolo descritos foi realizado minimizando-se o custo total das soluções, ou seja, a soma de todos os custos das rotas individuais das equipes. Isto significa que, nas comparações entre duas soluções durante a execução do algoritmo, a que é considerada como sendo melhor é a que tiver menor custo total. Esta configuração, que pode ser útil no ambiente real caso se deseje exclusivamente reduzir o deslocamento total das equipes, sem se preocupar com a distribuição equilibrada das ordens entre elas. Os resultados obtidos estão sintetizados na tabela 2.

4.2.1 Tempos de execução do algoritmo

A tabela 3 apresenta o tempo médio de execução do algoritmo para cada instância. Este valor foi obtido a partir da média do tempo gasto para execução dos 1000 ciclos do STACS nas 100 execuções do algoritmo, para cada dia de trabalho.

Tabela 3: Tempo médio de execução de 1.000 ciclos do STACS para cada dia de trabalho.

Dia de trabalho	Número de equipes	Número de ordens de serviço	Tempo médio de execução (ms)
3	3	31	2794
4	4	38	3805
5	4	40	4426
6	4	33	3327
7	4	40	4368
10	3	37	3534
11	4	27	2414
12	4	53	7197
13	4	35	4010
14	4	33	3667
15	2	12	677
17	2	26	1846
18	4	64	9993
19	4	35	3448
20	4	30	2658
21	4	36	3629
22	2	9	658

Tabela 2: Resultados obtidos minimizando-se o custo total das soluções MTSP para os 17 dias de trabalho.

	Solução Real		Médias de 100 execuções do STACS				Melhoramento	
Dia	**Custo Total (CTr)**	**Maior Rota (MRr)**	**Custo Total (CTp)**	**D. P.**	**Maior Rota (MRp)**	**D. P.**	**Custo Total (CTr - CTp)**	**Maior Rota (MRr - MRp)**
3	60,83	22,37	19,53	0,29	13,86	2,67	41,30	8,51
4	84,41	41,39	52,03	0,38	39,45	5,01	32,38	1,94
5	131,98	84,36	94,00	0,44	75,17	4,56	37,98	9,19
6	116,16	54,24	75,24	1,14	52,71	8,05	40,92	1,53
7	109,53	51,15	68,94	0,46	43,83	6,80	40,59	7,32
10	62,22	23,47	21,36	0,27	15,36	3,38	40,86	8,11
11	48,95	18,61	18,87	0,35	15,44	2,58	30,08	3,17
12	188,19	74,20	131,72	0,74	115,40	9,41	56,47	-41,20
13	52,25	23,05	21,01	0,32	13,43	1,51	31,24	9,62
14	60,53	23,78	23,07	0,44	14,42	3,73	37,46	9,36
15	140,44	133,77	111,08	0,01	104,84	0,00	29,36	28,93
17	37,75	20,22	16,75	0,46	13,26	3,44	21,00	6,96
18	71,73	24,05	27,13	0,26	15,32	3,89	44,60	8,73
19	56,26	21,65	26,09	0,60	19,61	1,91	30,17	2,04
20	55,31	23,83	22,23	0,43	12,05	3,40	33,08	11,78
21	70,62	28,70	24,22	0,43	15,55	3,32	46,40	13,15
22	138,59	108,06	105,03	0,00	103,05	0,00	33,56	5,01
Totais	**1485,75**	**776,90**	**858,29**		**682,75**		**627,46**	**94,15**

4.3 Minimização da maior rota individual das soluções

Um segundo experimento foi realizado minimizando-se a maior rota individual das soluções e mantendo-se os demais parâmetros como no primeiro experimento. Esta configuração é adequada ao ambiente real caso se deseje que as equipes tenham deslocamentos equilibrados entre si ao final da execução das ordens, o que também resulta que mais ordens sejam executados no mesmo intervalo de tempo (uma equipe não fica parada devido a ter uma rota significativamente menor que as outras, por exemplo). A tabela 4 apresenta os resultados obtidos para esta configuração.

4.4 Análise dos Resultados

O STACS apresentou resultados satisfatórios para as duas variações de MTSP experimentadas: minimizando o custo total da solução (tabela 2) e minimizando o custo da maior rota individual da solução (tabela 4). Os pequenos valores dos desvios padrão nas duas tabelas confirmam a robustez do algoritmo ao gerar soluções com custos próximos à média das 100 execuções.

No primeiro experimento, houve uma diminuição de 627,46 km no deslocamento total das equipes nos 17 dias de trabalho, comparando os custos reais com a média dos custos das soluções obtidas nas 100 execuções do STACS, o que corresponde a uma melhora de 42,23% em relação às rotas reais das equipes. Além disso, para todas as 17 instâncias foi encontrada uma solução com custo total menor e apenas para o dia de trabalho 12 houve um aumento da maior rota individual da solução.

No segundo experimento, configurado para minimizar a menor rota individual, a redução do deslocamento total das equipes nos 17 dias foi de 294,82 km, significativamente inferior ao ganho obtido no experimento anterior. Porém, quando verificando a maior rota individual das equipes, tem-se uma melhora de 94,15 km no primeiro experimento e 269,90 km no segundo. Apesar do ganho total nos deslocamentos ser menor, o ganho nas rotas individuais informa que as equipes trabalharam de maneira mais uniforme, concluindo suas rotas em instantes próximos, o que leva à execução de mais ordens em menos tempo, aumentando a qualidade do atendimento prestado.

Quanto aos tempos de execução do algoritmo, mostrados na tabela 3, verifica-se que para a maior instância experimentada, a do dia 18, com 64 ordens de serviço, foram gastos em média aproximadamente 10 segundos para execução dos 1000 ciclos do algoritmo, que representam um tempo aceitável para o problema em estudo.

5. CONCLUSÕES

A metodologia apresentada mostrou-se eficiente na distribuição das ordens de serviço entre as equipes e na criação de rotas otimizadas para realização dos atendimentos. Entretanto, algumas questões precisam ser analisadas, como a utilização de distâncias euclidianas para representação dos custos de deslocamento reais das equipes. A continuação deste trabalho prevê a utilização de custos que representem de forma mais fiel o ambiente real, como previsões de deslocamento entre dois pontos fornecidas por Sistemas de Informações Geográficas.

Quanto à confirmação da eficácia do método, é necessária a comparação dos resultados apresentados neste artigo com outros obtidos a partir da aplicação de diferentes metodologias ao problema. Para a criação de soluções otimizadas para MTSP neste trabalho, foi implementado um algoritmo baseado na meta-heurística ACO. Outra meta-heurística que tem obtido resultados consistentes para problemas de otimização combinatória são os Algoritmos Genéticos (AG) [3], que serão utilizados para comparação com os resultados obtidos pelo STACS em novos experimentos.

Quanto ao conjunto de parâmetros do algoritmo ACO utilizado nos experimentos, que foram obtidos a partir de trabalhos relacionados, uma experimentação utilizando o STACS e instâncias construídas a partir de dados reais podem resultar em um conjunto de parâmetros mais eficiente para o problema em estudo. Porém, devido à grande quantidade de combinações possíveis de parâmetros, esta experimentação assume elevada complexidade e será realizada na continua-

Tabela 4: Resultados obtidos minimizando-se o custo da maior rota individual das soluções MTSP para os 17 dias de trabalho.

	Solução Real		Médias de 100 execuções do STACS				Melhoramento	
Dia	**Custo Total (CTr)**	**Maior Rota (MRr)**	**Custo Total (CTp)**	**D. P.**	**Maior Rota (MRp)**	**D. P.**	**Custo Total (CTr - CTp)**	**Maior Rota (MRr - MRp)**
3	60,83	22,37	24,16	0,54	8,32	0,03	36,67	14,05
4	84,41	41,39	61,80	2,38	33,89	0,00	22,61	7,50
5	131,98	84,36	103,94	1,60	73,29	0,00	28,04	11,07
6	116,16	54,24	127,12	12,25	39,58	0,08	-10,96	14,66
7	109,53	51,15	80,79	1,91	39,81	0,00	28,74	11,34
10	62,22	23,47	24,96	0,26	8,39	0,11	37,26	15,08
11	48,95	18,61	28,15	1,54	7,73	0,00	20,80	10,88
12	188,19	74,20	185,48	18,73	60,80	0,00	2,71	13,40
13	52,25	23,05	32,16	2,26	9,26	0,02	20,09	13,79
14	60,53	23,78	30,55	0,47	7,90	0,02	29,98	15,88
15	140,44	133,77	179,35	1,54	90,21	0,67	-38,91	43,56
17	37,75	20,22	17,49	0,10	8,77	0,06	20,26	11,45
18	71,73	24,05	32,00	0,41	8,14	0,09	39,73	15,91
19	56,26	21,65	38,77	1,58	11,64	0,02	17,49	10,01
20	55,31	23,83	28,90	0,21	7,41	0,02	26,41	16,42
21	70,62	28,70	32,44	0,14	8,15	0,02	38,18	20,55
22	138,59	108,06	162,86	3,74	83,70	0,00	-24,27	24,36
Totais	**1485,75**	**776,90**	**1190,93**		**507,00**		**294,82**	**269,90**

ção deste trabalho.

Outro ponto diz respeito ao caráter dinâmico do ambiente em estudo. Enquanto a metodologia proposta mostra-se viável para cenários estáticos, no qual todas as ordens e as posições das equipes são conhecidas antes da otimização, no ambiente real serviços surgem durante o dia de trabalho das equipes e podem ter maior prioridade que outros, como as ordens emergenciais. Para tratamento desta questão, a metodologia desenvolvida será adaptada para gerar novas soluções otimizadas a cada mudança no ambiente real, como o surgimento de novas ordens.

6. AGRADECIMENTOS

À Alexandre da Silva, técnico de atendimento comercial e emergencial da COPEL Distribuição na agência de Cornélio Procópio, pelo fornecimento dos dados reais utilizados nos experimentos.

7. REFERÊNCIAS

[1] J. Augustine. *Offline and online variants of the traveling salesman problem*. PhD thesis, Louisiana State University, Department of Electrical and Computer Engineering, 2002.

[2] T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, June 2006.

[3] C. E. d. S. Costa, D. M. B. Costa, and A. R. T. Goes. Determinação de setores de atendimento em uma concessionária de energia. In *Anais do Simpósio Brasileiro de Pesquisa Operacional - SBPO*, pages 951–962, Goiânia, 2006.

[4] M. Dorigo and L. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, Apr. 1997.

[5] M. Dorigo, V. Maniezzo, and a. Colorni. The Ant System: optimization by a colony of cooperating agents. *IEEE transactions on systems, man, and cybernetics.*, 26(1):29–41, Jan. 1996.

[6] M. Dorigo and T. Stutzle. *Ant Colony Optimization*. The MIT Press, Cambridge, Massachusetts, 2004.

[7] V. Garcia, D. Bernardon, M. Sperandio, C. do Vale, and J. Fernandes. Service order dispatching in electric utilities. In *45th International Universities Power Engineering Conference (UPEC)*, pages 1 – 7, Cardiff, Wales, 2010. IEEE.

[8] IPARDES. Caderno estatístico do município de Cornélio Procópio. *Instituto Paranaense de Desenvolvimento Econômico e Social*, 2014.

[9] J. Y. Kanda. Sistema de meta-aprendizado para a seleção de meta-heurísticas para o problema do caixeiro viajante. In *X SBSI - Simpósio Brasileiro de Sistemas de Informação*, pages 651–662, Londrina, 2014.

[10] D. M. Machado Jr and M. A. Dal Santo. Considerações acerca de Trabalhos em Áreas de Divisa de Fusos UTM. In *Anais do Congresso Brasileiro de Cadastro Técnico Multifinalitário*, pages 1–14, 2004.

[11] A. R. Mascia, W. d. M. Reck, V. J. Garcia, D. P. Bernardon, M. Sperandio, and C. do Vale. Algoritmo heurístico para agrupamento de ordens de serviço em concessionárias de distribuição de energia elétrica considerando priorização. In *Anais do Congreso Internacional de Distribuicion Electrica*, pages 1–5, Buenos Aires, 2010.

[12] C. Okonjo-Adigwe. An effective method of balancing the workload amongst salesmen. *Omega*, 16(2):159–163, Jan. 1988.

[13] I. Vallivaara. A team ant colony optimization algorithm for the multiple travelling salesmen problem with minmax objective. In *Proceedings of the 27th IASTED International Conference on Modelling, Identification and Control*, pages 387–392, Anaheim, CA, USA, 2008.