METAHEURÍSTICA COLÔNIA DE FORMIGA APLICADA AO PROBLEMA DE ROTEAMENTO DE VEÍCULOS COM COLETA E ENTREGA E JANELA DE TEMPO

Eduardo G. Carabetti, **Sérgio R. de Souza**, **Marcelo C. P. Fraga**
Centro Federal de Educação Tecnológica de Minas Gerais
Av Amazonas 7675 - Nova Gameleira
30260-250 - Belo Horizonte - MG - Brasil
carabetti@gmail.com, sergio@dppg.cefetmg.br, caramuru@hotmail.com

RESUMO

Nesse artigo, propomos uma metodologia para resolver o Problema de Roteamento de Veículos com Coleta e Entrega e Janelas de Tempo(PRVCEJT). A resolução deste problema consiste em determinar as rotas de custo mínimo, que devem ser executadas por uma frota de veículos de mesma capacidade, para atender à demanda de um conjunto de clientes por um determinado produto. Para cada consumidor, o atendimento somente é possível dentro de um intervalo de tempo, chamado janela de tempo. Nossa abordagem utiliza o algoritmo Colônia de Formiga, *threads* para aumentar o número de tarefas executadas por unidade de tempo (*throughput*) associado a cada formiga, e técnicas de refinamento realizadas em tempo de execução para melhorar a qualidade das soluções. Para testar a eficiência do algoritmo, experimentos computacionais foram feitos gerando resultados competitivos com os melhores encontrados na literatura.

Palavras-Chave: Problema de Roteamento de Veículo com Coleta e Entrega e Janela de Tempo, Colônia de Formigas, Metaheurística.

ABSTRACT

In this paper, we proposed a metaheuristic to solve the Pickup and Delivery Problem with Time Windows. The solution of this problem consists in determining the routes of minimum cost for a fleet of vehicles, of same capacity, in charge to satisfy the demand of a set of customers which the attendance is only possible inside a specific time interval. Our approach uses the Ant Colony algorithm, threads to increase the numbers of tasks performed per unit time (throughput) associated at each ant, and refinement techniques performed at runtime to improve the quality of the solutions. To test the efficiency of the algorithm, computational experiment was benchmarked generating some competitive results with the best found in literature.

Keywords: Vehicle Routing Problem with Pickup and Delivery and Time Window, Ant Colony, Metaheuristic.

1. INTRODUÇÃO

O Problema de Roteamento de Veículos com Coleta e Entrega e Janela de Tempo (PRVCEJT), variação do PRV clássico, pode ser descrito como um problema no qual uma frota de veículos, inicialmente situada em um depósito, deverá atender a um conjunto de consumidores que possuem diferentes demandas por coleta ou entrega de produtos. O atendimento a cada consumidor deve ser iniciado em um intervalo de tempo denominado janela de tempo. A solução do problema consiste em encontrar o número mínimo de veículos, capaz de atender a todos os consumidores, e um conjunto de rotas, que minimize a distância total percorrida pelos veículos.

A metaheurística *Ant Colony Optimization* (ACO), proposta em [8], foi inspirada no comportamento real de formigas. De acordo com [3], os algoritmos baseados em ACO são os exemplos mais eficientes de sistema de inteligência coletiva. Os algoritmos de formigas foram utilizados com sucesso em vários problemas combinatórios, incluindo o Problema do Caixeiro Viajante (PCV), o Problema de Roteamento de Veículos (PRV), o Problema de Atribuição Quadrático (PQA) e *job-shop Scheduling*.

Uma metodologia chamada ACS-LS que utiliza o algoritmo Ant Colony System (ACS) para a construção da solução inicial e buscal local, (Local Search (LS)), para realizar o refinamento das soluções é proposta para a resolução desse problema. O algoritmo é dividido em duas fases: construção e refinamento. Na fase de construção, a geração de uma solução inicial é feita através da metaheurística *Ant Colony Optimization* (ACO) utilizando o conceito de elitismo. Somente a melhor formiga é que poderá depositar feromônio, tentando assim melhorar as soluções geradas pelas formigas a cada iteração. Na sequência, a solução gerada passa por uma fase de refinamento, através do uso de uma heurística de busca local, qual seja, o Método da Descida com Primeira de Melhora, associado a três estruturas de vizinhança. Depois do refinamento da solução, o feromônio é lançado sobre os arcos cobertos, ajudando a guiar as próximas formigas. Esse processo é autocatalítico, por exemplo, as próximas iterações utilizam as informações a partir dos resultados gerados anteriores, ajudando na convergência e precisão.

Este trabalho combina dois movimentos para exploração da estrutura de vizinhança, quais sejam: a realocação e a troca de pares de consumidores de coleta e entrega. O movimento de realocação consiste na remoção de um par de consumidores de coleta e entrega da sua posição original e sua inserção em uma posição diferente dentro da rota original ou em uma outra rota diferente da original. O movimento de troca consiste na remoção de dois pares distintos de consumidores de coleta e entrega de suas posições originais e a inserção dos pares com as posições invertidas, seja dentro da mesma rota ou em rotas distintas. A metodologia desenvolvida foi testada em um conjunto de instâncias criadas por [10] para testar a eficiência do algoritmo de solução proposto para o PRVCEJT, e produziu resultados tão bons quanto os encontrados na literatura.

O restante desse artigo está estruturado como se segue. Na Seção 2 é feita uma apresentação de trabalhos relacionados ao problema. A seção 3 fornece uma visão global do PRVCEJT. A metaheurística ACO é detalhada na seção 4. A seção 5 contém a metodologia utilizada nesse artigo. A Seção 6 traz os resultados computacionais obtidos a partir da aplicação do algoritmo proposto nesse trabalho em um conjunto

de instâncias disponibilizadas em [10]. Finalmente, na seção 7 tem-se as conclusões.

2. TRABALHOS RELACIONADOS

De acordo com [5], o PRVCEJT pertence à classe de problemas NP-difíceis. Logo, métodos heurísticos têm sido amplamente utilizados para a solução de problemas-teste de diversas ordens, sofrendo, no entanto, das dificuldades inerentes à indefinição da qualidade da solução encontrada a partir do uso dessa classe de métodos, no tocante à otimalidade. Quando o cenário do problema é pequeno, soluções exatas podem ser obtidas a médio prazo.

Em [4], é resolvido o PRVCE para um único veículo considerando o metaheurística *Variable Neighborhood Search* (VNS). O problema abordado inclui uma restrição adicional sobre a carga, ou seja, os ítens só podem ser entregues respeitando-se uma estrutura de fila em que o último produto a entrar é o primeiro a sair (*Last In First Out* (LIFO)). Oito diferentes técnicas de construção são utilizadas para gerar a solução inicial. Os vizinhos são definidos através da troca dos pares de coleta e entrega, troca dos pares em blocos, realocação de blocos, 2-*opt*-L e operadores de multi-realocação.

Em [10], é apresentada uma metodologia baseada na metaheurísitca Busca Tabu incorporada à metaheurística *Simulated Annealing* (SA) para resolver o PRVCEJT com múltiplos veículos. Uma troca dos pares de coleta e entrega é realizada baseado-se nos movimentos chamados *shift*, *exchange* e *rearrange*. Os dois primeiros definem a estrutura de vizinhança realizada pela metaheurística e o terceiro é usado para tentar refinar os resultados gerados pelos dois movimentos anteriores, ou seja, é considerado como um procedimento de pós-otimização.

Em [11], é proposto, para a solução do PRVCEJT, um algoritmo denominado *Grouping Genetic Algoritm* (GGA). Esse algoritmo difere-se do algoritmo genético tradicional na forma em que um grupo orientado para a codificação genética é utilizado. A codificação utilizada por [11] corresponde ao conjunto de solicitações de criação de uma frota. O aspecto do roteamento, não compreendido na codificação, é acrescentado à codificação do cromossomo. Já [6] utiliza a heurística de inserção associada ao GGA para gerar soluções factíveis, implementa novas estruturas de dados e três estratégias de ajuste de rota para o *Multi-Strategy Grouping Genetic Algorithm* (MSGGA).

No trabalho de [14], o algoritmo *Adaptative Large Neighborhood Search*(ALNS) é apresentado para a solução do PRVCEJT. Os autores incluem a consideração de múltiplos depósitos e a variável tempo de serviço para a análise do problema. Em [14], o método proposto é utilizado para resolver o PRV e instâncias do PRV com *Backhauls*, sendo transformadas no PRVCEJT.

Em [1], é apresentando um algoritmo híbrido para o PRVCEJT dividido em duas fases. Em sua fase inicial, utiliza-se à metaheurística *Simulated Annealing* (SA) para diminuir o número de veículos; na fase seguinte, o algoritmo *Large Neighborhood Search* (LNS) é usado, para reduzir a distância total percorrida.

3. O PROBLEMA DE ROTEAMENTO DE VEÍCULOS

A situação em que uma frota de veículos deve satisfazer a todos os pedidos dos consumidores, informando seu respectivo local e horário de coleta e entrega das mercadorias, caracteriza o PRVCEJT. O pedido consiste na retirada de mercadorias de um local e, logo em seguida, realizar a entrega em um outro local distinto. Duas janelas de tempo são disponibilizadas para cada solicitação: a janela de tempo de coleta, que especifica quando as mercadorias serão coletadas, e a janela de tempo de entrega, que define quando as mercadorias devem ser entregues. Além disso, a variável tempo de serviço é inserida e está associada a cada coleta e a cada entrega. O tempo de serviço indica quanto tempo levará para realizar o processo de coleta e de entrega. O veículo pode chegar a um consumidor antes do início da janela de tempo, mas o mesmo deverá esperar até que a janela de tempo se abra, e logo dar-se início à operação. A capacidade do veículo deverá ser respeitada a todo o tempo, assim como a janela de tempo que foi definida para cada local. Cada consumidor poderá ser visitado uma única vez. A solução para o PRVCEJT consiste em designar um conjunto de rotas que atenda a essas restrições, minimize o número total de veículos utilizado e o custo total da rota (distância total percorrida).

O PRVCEJT pode ser utilizado para modelar vários problemas no campo da logística e tráfego urbano. Encontrar boas soluções para estes problema é de suma importância, pois permite ao operador ou o coordenador logístico de tráfego urbano utilizar a frota existente, com o melhor custo-benefício e atender às demandas dos consumidores. Uma completa análise do PRVDEJT é apresentada em [13] e [12].

4. ALGORITMOS DE COLÔNIA DE FORMIGAS

O algoritmo *Ant Colony System* (ACS), introduzido por [7], foi o escolhido para encontrar soluções iniciais para o problema. Para cada arco de uma rota, um valor de feromônio é introduzido, sendo que cada formiga tem a tarefa de construir uma solução factível da seguinte maneira: um dos vértices do grafo PRVCEJT é escolhido de forma aleatória como o vértice principal. A formiga, então, constrói uma rota para o grafo PRVCEJT, movendo-se, a cada etapa de construção, do seu vértice corrente para o próximo ainda não visitado. Essa escolha do caminho é feita baseando-se na quantidade de feromônio associado ao arco e a sua visibilidade (inverso da distância), e o equilíbrio entre ambos os parâmetros é definido como a atratividade do arco. A cada passo, o arco percorrido é adicionado à solução ainda em construção. Quando não restarem mais arcos, as formigas terminam a rota, movendo-se do vértice corrente para o vértice em que a construção se iniciou. Este tipo de construção significa que uma formiga terá uma memória ou uma lista tabu para armazenar os vértices que já foram visitados.

Uma melhoria realizada no algoritmo foi a utilização de um critério de decisão, chamado de regra pseudo-randômico-proporcional, apresentado em 1, em que o parâmetro q_0 controla a formação das soluções, se realizada de forma probabilística ou de forma gulosa. Quando a formiga escolhe o próximo vértice para se mover, um número randômico q é gerado e comparado ao parâmetro q_0. Se q é menor que q_0, a geração da solução valoriza o aprendizado gerado pela deposição do feromônio, trabalhando de forma gulosa. Se q é maior do que q_0, a solução será obtida de forma

probabilística, segundo expressão (2).

$$s = \begin{cases} \arg\max_{u \notin M_k} \{[\tau_{(r,u)}]^{\alpha}[\eta_{(r,u)}]^{\beta}\} \ , & \text{se } \ q \leq q_0 \\ \\ \text{Equação (2)}, & \text{caso contrário.} \end{cases} \tag{1}$$

$$P_k(r,s) = \begin{cases} \dfrac{[\tau_{(r,s)}]^{\alpha}[\eta_{(r,s)}]^{\beta}}{\sum\limits_{u \notin M_k} [\tau_{(r,u)}]^{\alpha}[\eta_{(r,u)}]^{\beta}} \ , & \text{se } \ \notin M_k \\ \\ 0 \qquad\qquad , & \text{caso contrário.} \end{cases} \tag{2}$$

sendo:

- $P_k(r,s)$: probabilidade da formiga k mover-se do vértice r para s;
- $\tau_{(r,s)}$: quantidade de feromônio associoado com o arco (r,s);
- $\eta_{(r,s)}$: atratividade do arco(r,s);
- U: conjunto de arcos ainda não visitados;
- M_k: memória ou lista tabu da formiga k;
- α, β: parâmetro de controle de influência de $\tau_{(r,s)}$ e $\eta_{(r,s)}$. Localizados na faixa [0,1].

Após todas as formigas construírem sua solução, a evaporação de feromônio é atualizada de forma local e global. A atualização global é aplicada apenas àquelas que pertencem a melhor solução global. De acordo com [7], a atualização global do ACS impede a lenta convergência dos resultados, concentrando-se nas buscas sobre os vizinhos com as melhores soluções. A atualização global de feromônio é calculada com a equação (3):

$$\tau_{(r,s)} = (1-\rho)\tau_{(r,s)} + \rho/\triangle_{(r,s)} \tag{3}$$

sendo $\tau_{(r,s)}$ a quantidade de feromônio associada ao arco (r,s); ρ a taxa de evaporação do feromônio, para $\rho \in [0,1]$; e $\triangle_{(r,s)}$ definido como a distância associada à melhor solução encontrada desde o início do processo computacional, conforme posto em [7]. Um processo chamado *Daemon Actions* pode ser usado, opcionalmente, para auxiliar no processo centralizado de tomada de decisões pelo ACS, em situações como ativar um procedimento de busca local, foco do trabalho, ao final de cada iteração ou depositar uma quantidade extra de feromônio sobre a melhor solução. Esses são exemplos de decisões que não podem ser tomadas pelas formigas individualmente. Uma revisão completa desta metaheurística inspirada no comportamento das formigas pode ser encontrada em [2], [7] e [9].

5. METODOLOGIA ACS-LS

A metodologia proposta para a solução do PRVCEJT é denominada *Ant Colony System* com Busca Local (ACS-LS). Foram utilizadas *threads* para permitir computação em paralelo e aumentar o número de tarefas por unidade de tempo (*throughput*). Cada *thread* está associada a uma formiga, sendo responsável por construir uma solução completa e factível para o PRVCEJT, e é executada por um determinado período de tempo. A formiga com a melhor solução é escolhida através de uma função objetivo lexicográfica, que prioriza, inicialmente, aquelas soluções com o menor número de veículos, para, em seguida, selecionar as que possuem menores distâncias percorridas.

```
1 Loop
2  Aleatoriamente posicione m formigas artificiais em n consumidores/vértices
3  Para consumidor = 1 até n faça
4      Para formiga = 1 até m faça
5          Escolha o próximo vértice para se mover
6          Aplique a regra de atualização na trilha de feromônio local
7      fim para
7      Calcule o comprimento da rota feito pela formiga m
8      Atualize a trilha de feromônio da formiga m
9  fim para
10 Execute daemon-actions
11 Execute Busca Local, Descida com Primeira de Melhora
12 Aplique a regra de atualização global na trilha de feromônio
13 Até Fim-condição
```

Figura 1. Pseudocódigo da Metodologia Ant Colony System com Busca Local (ACS-LS).

O algoritmo é dividido em duas fases: construção e refinamento. Na fase de construção, a geração de uma solução inicial é feita através da metaheurística *Ant Colony Optimization* (ACO) com o conceito de elitismo. Somente a melhor formiga é que poderá depositar feromônio, tentando, assim, melhorar as soluções geradas pelas formigas a cada iteração. Na seqüencia, a solução gerada passa por uma fase de refinamento através da heurística de busca local denominada Método da Descida com Primeira de Melhora. Neste método, dois movimentos para exploração da estrutura de vizinhança são usados: a realocação e a troca de pares de consumidores de coleta e entrega. Depois do refinamento da solução, o feromônio é lançado sobre os arcos cobertos, ajudando a guiar as próximas formigas.

O movimento de realocação consiste na remoção de um par de consumidores de coleta e entrega da sua posição original e sua inserção em uma posição diferente, dentro da rota original ou em uma rota diferente da original. O movimento de troca consiste na remoção de dois pares distintos de consumidores de coleta e entrega de suas posições originais e a inserção dos pares com as posições invertidas, seja dentro da mesma rota ou em rotas distintas.

A Figura 1 mostra o pseudo-código da metodologia ACS-LS proposta para a resolução do PRVCEJT.

5.1. MÉTODO DA DESCIDA COM PRIMEIRA DE MELHORA

Variante do método da Descida, o método da Descida com Primeira de Melhora evita a pesquisa exaustiva pelo melhor vizinho, pois consiste em interromper a exploração da vizinhança quando um vizinho melhor é encontrado. Se o vizinho escolhido não for de melhoria do resultado da solução, a solução corrente permanece inalterada e outro vizinho é gerado. Desta forma, apenas no pior caso toda a vizinhança é explorada. O procedimento é interrompido após um certo número fixo de iterações sem melhora no valor da melhor solução obtida até então. A solução final é um ótimo local com respeito à vizinhança considerada.

Seja $\mathcal{S}$ o espaço de busca de um problema de otimização, f a função objetivo a

Procedimento DescidaPrimeiraMelhora $(f(\cdot), N(\cdot), s)$;
1 $V = \{s' \in N(s)\}$;
2 enquanto (|V| > 0) faça
3 Selecione $s' \in V$;
4 Se $f(s') < f(s)$;
5 então $s \leftarrow s'$;
6 $V = \{s' \in N(s)\}$;
7 senão $V \leftarrow V \setminus \{s'\}$;
8 fim Se;
9 fim enquanto;
10 Retorne s;
Fim Procedimento;

Figura 2. Pseudo-Código do Método da Descida com Primeira de Melhora.

otimizar (minimizar) e s uma solução qualquer do problema, isto é, $s \in S$. Seja N uma função que associa, a cada solução $s \in S$, sua vizinhança $N(S) \subseteq S$. Cada solução $s' \in N(s)$ é chamada vizinho de s. Denomina-se movimento a uma modificação m que transforma uma solução s em outra s', que esteja em sua vizinhança, na forma $s' \leftarrow s \oplus m$. Dada uma função f, uma solução s e uma vizinhança $N(\cdot)$, a figura 2 mostra o procedimento de descida com primeira de melhora.

5.2. ESTRATÉGIAS DE REFINAMENTO

Os movimentos de realocação são aplicados dentro de três estratégias de refinamento chamadas de Elimina Rota ou PD-SRK (*Pickup and Delivery Small Route Killer*), PD-DHFI (*Pickup an Delivery Down Hill First Improvement*) e PD-Ex (*Pickup and Delivery Exchange*).

A Elimina Rota ou PD-SRK é definida pelo uso do movimento de realocação associado ao refinamento por Descida com Primeira Melhora. Inicia-se sempre a remoção dos pares de consumidores a partir da menor rota do conjunto de rotas e a inserção a partir da maior, privilegiando a eliminação de consumidores das menores rotas e a manutenção das maiores rotas. Neste refinamento, não existe a preocupação em melhorar a função objetivo (FO), mas apenas em manter a factibilidade da solução. A idéia é impedir que um movimento que poderia eliminar rotas futuramente não seja executado por não melhorar a FO imediatamente. Caso ocorra a eliminação de toda a rota, mesmo que a FO aumente, o processo se reinicia. Se algum consumidor da rota não possa ser realocado, a rota deverá permanecer inalterada e o processo continuará para as demais rotas até que os consumidores pertencentes à maior rota sejam analisados. Essa estratégia consegue reduzir o número de veículos principalmente nas iterações iniciais em que o aprendizado dado pelo feromônio ainda não influencia as formigas na tomada de decisões.

A segunda estratégia PD-DHFI é muito parecida com a PD-SRK, pois também utiliza a Descida com Primeira Melhora associada ao movimento de realocação e parte da menor para a maior rota. Porém, os movimentos de refinamento somente são aceitos caso melhorem a FO. Se o par de consumidores realocado diminuir a FO, a rota é definitivamente alterada e processo se reinicia. Diferentemente da PD-SRK, não é necessário eliminar toda a rota para que o algoritmo aceite o novo conjunto

de rotas, basta apenas que a FO tenha um valor melhor que o atual. Seu objetivo é melhorar a FO e eliminar rotas baseando-se no imediatismo produzido pela FO.

Por fim, a terceira estratégia PD-Ex é semelhante a PD-DHFI, porém utiliza a Descida com Primeira Melhora associada ao movimento de Troca. A única diferença efetiva é que, ao invés de fazer a realocação dos pares de consumidores, procede-se a sua troca. Esta estratégia também é norteada pela melhoria da FO, ou seja, o movimento só é aceito caso gere uma FO melhor que a corrente.

Um exemplo para melhor entendimento é dado pelo conjunto de 3 rotas a seguir:

Rota 1: 0 - 1 - 2 - 3 - 4 - 0
Rota 2: 0 - 7 - 8 - 9 - 10 - 11 - 12 - 0
Rota 3: 0 - 13 - 14 - 15 - 16 - 17 - 18 - 0

Supondo que em todas as rotas cada consumidor ímpar faça entrega no próximo consumidor par consecutivo. Sabendo-se que a Rota (1) é a menor, é possível remover o 1º par de Coleta e Entrega (1 e 2) da mesma, obtendo assim uma nova Rota 1:

Rota 1: 0 - 3 - 4 - 0

Inserindo o par (1-2) na maior rota (3) no início:

Rota 3: 0 - **1** - **2** - 13 - 14 - 15 - 16 - 17 - 18 - 0

Supondo que a realocação não tenha violado restrições e tenha piorado a FO, o movimento não teria sido aceito pela estratégia PD-DHFI e o par deverá ser realocado em outra posição. Para a estratégia PD-DHFI, somente quando melhorar a FO o novo par é aceito na rota.

Inserindo o par (1-2) na maior rota (3) no fim:

Rota 3: 0 - 13 - 14 - 15 - 16 - 17 - 18 - **1** - **2** - 0

Se a nova rota não violar restrições e o custo da FO melhorar, esta rota é aceita e o processo continua tentando inserir os consumidores restantes (3 e 4) da menor rota (1) na maior rota (3).

Caso ocorra uma violação de qualquer uma das restrições, não será possível inserir o par (3 e 4) na rota (3). Repete-se o processo para a rota (2) e se ocorrerem violações ao inserir na rota (2) o processo continuará até que seja selecionado o último par de consumidores (17 e 18) da maior rota (3).

O refinamento PD-SRK poderia ter obtido sucesso neste caso se ao inserir o 1º par (1 e 2) na rota (3) não houvesse a preocupação em melhorar a FO, apenas a manter a solução factível. Dessa forma, o próximo par da rota (1) poderia "entrar" ainda na rota (3), apesar do aumento na FO, mas com o benefício da eliminação da rota (1). Produzindo:

Rota 1: 0 - 0
Rota 2: 0 - 7 - 8 - 9 - 10 - 11 - 12 - 0
Rota 3: 0 - 1 - 2 - 13 - 14 - 15 - 16 - 17 - 18 - 3 - 4 -0

Como é possível perceber, a rota 1 pode ser eliminada, pois não possui mais nenhum consumidor.

Sempre que as estratégias de refinamento são utilizadas e produzem resultados que violem restrições, chama-se um método de refinamento por força-bruta para tentar tornar factível a solução. O método de força-bruta consiste em utilizar os movimentos de realocação ou de troca em todas as posições possíveis para o par de coleta e entrega. Um exemplo pode ser visto abaixo:

Rota 1: 0 - 1 - 2 - 3 - 4 - 0
Rota 2: 0 - 7 - 8 - 9 - 10 - 11 - 12 - 0

Supondo que deseja-se realocar o par de consumidores (1 e 2) da rota (1) para a rota (2). A estratégia de refinamento irá remover o par da posição inicial e inserir consecutivamente na rota destino.

Rota 2: 0 - 1 - 2 - 7 - 8 - 9 - 10 - 11 - 12 - 0
Rota 2: 0 - 7 - 8 - 1 - 2 - 9 - 10 - 11 - 12 - 0
Rota 2: 0 - 7 - 8 - 9 - 10 - 1 - 2 - 11 - 12 - 0
Rota 2: 0 - 7 - 8 - 9 - 10 - 11 - 12 - 1 - 2 - 0

Porém, às vezes é necessário que os consumidores de coleta e entrega não fiquem posicionados consecutivamente. O método de força-bruta irá gerar as variações possíveis para cada uma das posições de inserção ou de troca, sempre mantendo o consumidor de coleta antes do de entrega.

Rota 2: 0 - 1 - 2 - 7 - 8 - 9 - 10 - 11 - 12 - 0

Se a rota acima violar restrições, gera-se as seguintes variações:

Rota 2: 0 - 1 - 7 - 2 - 8 - 9 - 10 - 11 - 12 - 0
Rota 2: 0 - 1 - 7 - 8 - 2 - 9 - 10 - 11 - 12 - 0
Rota 2: 0 - 1 - 7 - 8 - 9 - 2 - 10 - 11 - 12 - 0
Rota 2: 0 - 1 - 7 - 8 - 9 - 10 - 2 - 11 - 12 - 0
Rota 2: 0 - 1 - 7 - 8 - 9 - 10 - 11 - 2 - 12 - 0
Rota 2: 0 - 1 - 7 - 8 - 9 - 10 - 11 - 12 - 2 - 0

Por fim, é possível perceber que, no método de força-bruta, enquanto um consumidor fica fixo na posição original atribuída pela estratégia de refinamento, o outro fica pivoteando, à procura de uma solução factível ou de melhor FO.

6. RESULTADOS COMPUTACIONAIS

O algoritmo ACS-LS foi implementado em C++ utilizando a plataforma Eclipse e executado em um computador Intel Core 2 Duo E8400, com 2GB de memória RAM, sob plataforma Windows XP Pro. Foram utilizadas, para teste, as 56 instâncias propostas em [10], usadas amplamente como referência de desempenho de algoritmos para o PRVCEJT, sendo derivadas do mesmo número de instâncias originais de [15], usadas para análise do PRVJT.

Inicialmente, os parâmetros são configurados. O primeiro parâmetro é o número de iterações. O algoritmo foi executado 10 vezes para cada instância, cada uma delas partindo de uma semente diferente de números aleatórios, com intervalos de 60s,

Tabela 1. Resultado para as instâncias de 100 consumidores de [10]

	Melhor Literatura		ACS-LS	
Instância	**NV**	**TD**	**NV**	**TD**
lc101	10	828,94	10	828,94
lc102	10	828,94	10	831,68
lc103	9	1035,35	10	920,38
lc104	9	860,01	9	1093,46
lc105	10	828,94	10	830,53
lc106	10	828,94	10	851,56
lc107	10	828,94	10	844,72
lc108	10	826,44	10	882,37
lc109	9	827,82	10	916,58
lc201	3	591,56	3	591,55
lc202	3	591,56	3	777,66
lc203	3	585,56	3	970,52
lc204	3	590,6	4	807,34
lc205	3	588,88	4	767,56
lc206	3	588,49	4	834,80
lc207	3	588,29	4	764,89
lc208	3	588,32	4	807,85
lr101	19	1650,8	19	1650,79
lr102	17	1487,57	17	1503,13
lr103	13	1292,68	13	1379,60
lr104	9	1013,39	10	1084,43
lr105	14	1377,11	14	1409,69
lr106	12	1252,62	12	1319,49
lr107	10	1111,31	10	1159,74
lr108	9	968,97	10	1128,06
lr109	11	1208,96	12	1321,55
lr110	10	1159,35	12	1249,03
lr111	10	1108,9	11	1204,41
lr112	9	1003,77	10	1188,51
lr201	4	1253,23	4	1436,41
lr202	3	1197,67	4	1419,20
lr203	3	949,4	3	1288,58
lr204	2	849,05	3	1152,94
lr205	3	1054,02	4	1321,86
lr206	3	931,63	3	1393.71
lr207	2	903,06	3	1219,57
lr208	2	734,85	3	988,76
lr209	3	930,59	3	1205,48
lr210	3	964,22	3	1354,54
lr211	2	911,52	3	1085,01
lrc101	14	1708,8	14	1760,80
lrc102	12	1558,07	13	1650,00
lrc103	11	1258,74	11	1371,68
lrc104	10	1128,74	11	1313,88
lrc105	13	1637,62	13	1662,57
lrc106	11	1424,73	11	1456,38
lrc107	11	1230,15	11	1421,29
lrc108	10	1147,43	11	1284,41
lrc201	4	1406,94	5	1704,10
lrc202	3	1374,27	4	1714,26
lrc203	3	1089,07	4	1222,28
lrc204	3	818,66	3	1226,93
lrc205	4	1302,2	4	1543,29
lrc206	3	1159,03	4	1571,67
lrc207	3	1062,05	4	1410,71
lrc208	3	852,76	3	1561,70

600s, 3600s e 43200s, a fim de determinar seu melhor comportamento. Estes valores afetam a qualidade da solução e o tempo computacional. Números elevados de iterações aumentam a probabilidade de alcançar melhores soluções em detrimento de um maior tempo computacional. O segundo parâmetro ρ é configurado com 0,1, devido a

baixa taxa de evaporação (para valores de 0.0001 e 0.001) não garantir a diversificação da solução para um novo ponto; ρ é o parâmetro que controla a taxa de evaporação do feromônio. O terceiro parâmetro controla a influência do rastro de feromônio na fórmula de probabilidade, e é definido como $\beta = 1$. O quarto parâmetro controla a influência da trilha de feromônio na fórmula de probabilidade, e então $\alpha = 1$. O quinto parâmetro controla a forma de atuação das formigas, sendo ela gulosa ou probabilística. Finalmente, o último parâmetro utiliza 5 formigas por iteração, sendo que somente a melhor solução global de todas é que deposita feromônio.

Tabela 2. Comparação final

	Melhor Lit.	**ACS-LS**	**%**
Total NV	402	412	2,48
Total TD	57881,51	60274,10	3,94

Um estudo comparativo entre a abordagem proposta e os melhores resultados conhecidos da literatura com base nos problemas testes de [10] é apresentado na Tabela 1. Os melhores resultados publicados foram obtidos em `http://www.sintef.no/Projectweb/TOP/Problems/PDPTW/Li--Lim-benchmark/`. Nessa tabela, NV significa o número de veículos utilizados e TD significa a distância total. Para o grupo LC1 de instâncias, todos os ótimo globais são conhecidos na literatura. Em 6 das 9 instâncias desse grupo, o algoritmo ACS-LS foi capaz de encontrá-lo. Para o grupo LC2 e LRC1, em 2 das 8 instâncias o algoritmo ACS-LS também foi capaz de encontrá-los. Para os subgrupos LRC2 e LR2, não foram alcançados resultados de maior relevância.

Os resultados computacionais comprovam a viabilidade da metodologia proposta. O resultado final da metaheurística ACS-LS está apenas 2,48% atrás dos melhores resultados conhecidos no caso do número total de veículos. Por outro lado, o resultado final em relação à distância total foi de 3,94% acima dos resultados mais conhecidos. Neste caso, é necessário um esforço para melhorar a metodologia, a fim de obter melhores resultados.

7. CONCLUSÃO

Nesse trabalho, é apresentado uma arquitetura para a solução do PRVCEJT que utiliza a metaheurística ACS associada com um método de Busca Local, Descida com Primeira de Melhora, e três diferentes tipos de estrutura de vizinhança. A metodologia foi testada nas 56 instâncias testes proposta por [10], gerando soluções iguais às melhores encontradas na literatura, mostrando que o método desenvolvido é eficiente e robusto para resolver o PRVCEJT. Apesar das soluções alcançadas serem relativamente boas, a metodologia pode ser melhor explorada criando mais estruturas de buscas locais com diferentes tipos de movimentos para produzir diferentes vizinhos e, portanto, diferentes soluções. Aumentando a quantidade de soluções, possivelmente melhores resultados podem ser encontrados. A associação entre a metaheurística *Ant Colony* e Busca Local foi capaz de fornecer resultados interessantes e devem ser melhor estudados, através de novas implementações e testes.

AGRADECIMENTOS

Os autores gostariam de agradecer ao CEFET/MG, CAPES, CNPq e FAPEMIG por apoiar o desenvolvimento desta pesquisa.

Referências

[1] Bent, R., and Hentenryck, P. V. A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. *Computers & Operations Research 33* (2006), 875–893.

[2] Blum, C. Ant colony optimization: Introduction and recent trends. *Physics of Life Reviews 2 2* (2005), 353–373.

[3] Bonabeau, E., Dorigo, M., and Thraulaz, G. From natural to artificial swarm inteligence. *Oxford University Press* (1999).

[4] Carrabs, F., Cordeau, J.-F., and Laporte, G. Variable neighborhood search for the pickup and delivery traveling salesman problem with lifo loading. *INFORMS J. on Computing 19*, 4 (2007), 618–632.

[5] Cordeau, J. F., Laporte, G., Salvelbergh, M. W. P., and Vigo, D. Vehicle routing. *Handbooks in Operational Research and Management Science* (2005).

[6] Ding, G., Li, L., and Ju, Y. Multi-strategy grouping genetic algorithm for the pickup and delivery problem with time windows. 97–104.

[7] Dorigo, M., and Gambardella, L. M. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation 1* (1996), 53–66.

[8] Dorigo, M., Maniezzo, V., and Colorni, A. The ant system: An autocatalytic optimization process. Tech. Rep. n. 91-016 Revised, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1991.

[9] Gendreau, M., and Tarantilis, C. D. *Solving Large-Scale Vehicle Routing Problems with Time Windows: The State-of-the-Art.* Operations Research/Computer Science Interfaces Series. CIRRELT, Montreal, Canada, 2010.

[10] Li, H., and Lim, A. A metaheuristic for the pickup and delivery problem with time windows. *IEEE International Conference on Tools with Artificial Intelligence (ICTAI'01) 13* (2001), 160.

[11] Panktraz, G. A grouping genetic algorithm for the pickup and delivery problem with time windows. Tech. rep., Department of Business Administration and Economics, 2006.

[12] Parragh, S., Doerner, K., and Hartl, R. A survey on pickup and delivery problems part i: Transportation between customers and depot. *Journal für Betriebswirtschaft 58*, 2 (June 2008), 21–51.

[13] Parragh, S., Doerner, K., and Hartl, R. A survey on pickup and delivery problems part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft 58*, 2 (June 2008), 81–117.

[14] Ropke, S., and Pisinger, D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science 40* (2004), 455–472.

[15] Solomon, M. M. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operational Research 35*, 2 (1987), 254–265.