ALGORITMO GENÉTICO COM LISTA TABU APLICADO AO PROBLEMA DE ENTREGA DE REFEIÇÕES

Michel Ehrlich
Universidade Federal do Paraná
michelehrlich@gmail.com

Neida Maria Patias Volpi
Universidade Federal do Paraná
neida@ufpr.br

Resumo

O problema do roteamento de veículos com janelas de tempo consiste em determinar um roteiro de distância mínima para entregas ou coletas de produtos à clientes com localizações definidas através de caminhões partindo de um depósito. Cada caminhão possui uma capacidade definida e cada cliente uma demanda específica e uma janela de tempo na qual deve ser atendido. É de suma importância para a redução de custos e garantia de qualidade no serviço. Como se trata de um problema da classe NP-hard, a resolução exata através de um modelo de programação inteira binária é aplicável somente para problemas de pequenas dimensões, sugerindo a utilização de heurísticas ou meta-heurísticas como os algoritmos genéticos. Um problema recorrente na utilização destes é a convergência prematura para ótimos locais. Este trabalho apresentou como alternativa a esse problema a utilização de lista-tabu, tendo como objetivo analisar o impacto da mesma na aplicação do algoritmo genético. O método foi testado em dados reais de uma empresa que realiza entregas de refeições no município de Colombo. Foram realizados experimentos com o problema reduzido (possibilitando comparar com a solução exata) e com o problema completo de 53 clientes. Em todos os casos, os experimentos com a lista-tabu apresentaram resultados médios melhores do que sem a utilização da mesma, com um incremento pequeno no tempo computacional
Palavras Chave: Roteamento de veículos com janelas de tempo, algoritmo genético, lista tabu, meta-heurísticas.

Abstract

The vehicle routing problem with time windows consists in to determine a minimum distance route for deliveries or collections of products to customers with defined locations through trucks leaving a deposit. Each truck has a defined capacity and each customer a specific demand and a time window which should be attended. This is very important for cost reduction and quality assurance in service. Since it is a NP-hard class problem, an exact resolution through an integer-binary programming model is only applicable for small dimensions problems, suggesting the use of heuristics or meta-heuristics such as genetic algorithms. A recurring problem in their execution is the premature convergence to local optimums. This work presented as alternative to this problem the use of tabu-list, aiming to analyze the impact of it on the application of genetic algorithm. The method was tested on real data from a company that carries out deliveries of meals in the city of Colombo. Experiments were carried out with the reduced problem (enabling comparison with the exact solution) and with the complete problem of 53 customers. In all cases, the experiments with tabu-list had better average results than without the use of it, with a small increment in the computational time.
Key-words: vehicle routing problem with time windows; genetic algorithm, tabu-list,meta-heuristics.

1. INTRODUÇÃO

Problemas de roteamento de veículos genericamente envolvem o projeto de rotas de entrega ou coleta de itens partindo de determinado número de depósitos para um número de clientes, visando a minimização da distância percorrida ou do custo relacionado, tendo papel fundamental na área de gerenciamento da distribuição e logística [1]. A esse caso geral podem ser adicionados outras restrições, como o peso de carregamento, a quantidade e os tipos de veículos disponíveis e as janelas de tempo de atendimento de cada cliente, como é o caso do presente trabalho. Neste estudo, considera-se um único depósito ou centro de distribuição.

O problema de roteamento de veículos com janelas de tempo é uma extensão dos problemas de roteamento, no qual o atendimento a cada cliente somente é possível dentro de um intervalo de tempo pré-determinado, chamado janela de tempo. Janelas de tempo *soft* podem ser violadas, enquanto que janelas de tempo rígidas (*hard*) só permitem que o cliente seja atendido durante sua janela [12]. O presente trabalho foca nas janelas de tempo rígidas.

O problema de roteamento de veículos com janelas de tempo (PRVJT) é de fundamental importância na prática, uma vez que, de acordo com [13], pesquisas sugerem que entre 10% e 15% do valor final das mercadorias comercializadas se devem ao custo de seu transporte. Em várias destas aplicações, tais como entregas bancárias, correios e rotas de ônibus escolares, um intervalo de tempo no qual o cliente deve ser atendido é adicionado [17]. Neste trabalho, a metodologia foi testada em dados de um problema de entrega de refeições, que devido aos intervalos de tempo para refeições e da necessidade dos alimentos serem entregues quentes precisam de atendimento dentro de janelas de tempo específicas.

Metodologias exatas podem ser utilizadas para obtenção da solução ótima para o PRVJT. Porém, devido à sua alta complexidade (trata-se de um problema classificado como NP-Hard), mesmo softwares poderosos não são capazes de chegar a uma solução em tempo hábil quando o problema é de grandes dimensões (muitos pontos no grafo). Sendo assim, para alcançar uma boa solução ao menos próxima da solução ótima recomenda-se para estes casos métodos heurísticos [9], os quais são procedimentos que provavelmente irão encontrar uma excelente solução viável, mesmo que não necessariamente ótima, para o problema [8].

Dentre os métodos heurísticos se destacam as meta-heurísticas. Estas são definidas como métodos de resolução geral que fornecem tanto estrutura como diretrizes estratégicas para desenvolver uma heurística específica que se ajuste ao problema em questão [8].

Algoritmos genéticos são métodos de busca e otimização baseados nos princípios darwinistas de seleção natural dos seres vivos. Foram inicialmente propostos por John Holland na década de 70. Cada cromossomo representa uma possível solução do problema e são avaliados a cada geração por uma função de adaptação. Analogamente aos processos biológicos, os indivíduos mais aptos possuem maior probabilidade de reproduzirem-se através de cruzamentos com outros indivíduos, produzindo descendentes com características de ambas as partes [2]. Mutações também modificam as características da população. Ao longo das gerações espera-se que os indivíduos apontem uma solução viável e próxima da ótima.

Um aspecto crucial para o sucesso do algoritmo é evitar a convergência rápida para ótimos locais. Uma dificuldade observada na aplicação do AG na resolução do PRVJT é que pequenas alterações na população geralmente direcionam o algoritmo para uma parte do espaço de busca onde ele não consegue evoluir, causando, em poucas iterações, uma perda de diversidade muito prejudicial à obtenção de boas soluções, conforme é demonstrado em [20].

Com o intuito de contornar esse problema, utilizou-se no algoritmo genético uma lista-tabu, ideia extraída de outra meta-heurística: a Busca-Tabu. Este método consiste na repetição de uma busca local por uma melhoria dentro de uma vizinhança da solução atual, permitindo, no entanto, a piora da solução caso não seja possível melhoria na vizinhança. Para evitar que, após afastar-se de um ótimo local, o processo retorne imediatamente para o mesmo ponto, a busca-tabu impede temporariamente movimentações na solução que possam retornar para uma solução visitada recentemente. Uma lista tabu registra esses movimentos proibidos [8].

Este trabalho visa a elaboração deste algoritmo e testes do mesmo com dados de um caso real, comparando os resultados obtidos com e sem a presença da lista-tabu, para avaliar seu impacto. Na seção 2 apresenta-se a revisão de literatura com os principais trabalhos e conceitos utilizados como referência. Na seção 3 é detalhado o método proposto e um estudo de caso. Na seção seguinte são apresentados os resultados seguidos das conclusões obtidas.

2. REVISÃO DE LITERATURA

Algoritmos genéticos procuram resolver problemas gerais baseados no princípio da seleção natural darwinista. John Holland é considerado aquele que deu inicio à ideia, através de seu artigo de 1975 "*Adaptation in Natural and Artificial Systems"*. Seu aluno David Goldberg foi o primeiro a implementar com sucesso o algoritmo genético em aplicação industrial, na década de 80 [5]. O algoritmo é constituído dos seguintes passos básicos:

- Escolha da população inicial, geralmente formada por indivíduos aleatórios (o tamanho da população deve ser grande suficiente para garantir a diversidade e pequena o suficiente para não requerer recursos computacionais indisponíveis);
- Avaliação de toda a população de indivíduos segundo algum critério determinado por uma função chamada de função de aptidão ou "fitness";
- Aplicação do operador de "seleção", escolhendo-se assim os indivíduos "pais" que servirão de base para a criação de um novo conjunto de soluções (nova geração), sendo que o operador seleção tende a escolher os indivíduos mais aptos;
- Obtenção da nova geração aplicando-se sobre os indivíduos selecionados operações que misturem suas características (genes) através do cruzamento ou que as alterem aleatoriamente através da mutação;
- Esses passos (com exceção do primeiro) são repetidos até alcançar-se um critério de parada, que pode ser a obtenção de uma solução aceitável, um número máximo de iterações ou a não-melhoria da solução por diversas iterações seguidas [5].

A Busca Tabu é outra meta-heurística, proposta por Fred Glover em 1986 para problemas de otimização combinatória. Essencialmente, surge para evitar que a busca por soluções convirja para um ótimo local [10]. É genericamente representada pelos seguintes passos:

- A partir de uma solução inicial realiza-se uma série de movimentações, que representam uma busca local por soluções vizinhas
- Verificação se o melhor movimento não pertence a lista tabu. Caso não pertença, realizá-lo e adicionar à lista tabu. Caso pertença, desconsiderá-lo e procurar o seguinte melhor e repetir a verificação (admite-se piora na solução atual).
- Esses passos são repetidos até alcançar-se um critério de parada [8].

[10] e [5] apresentam um algoritmo genético restrito por listas tabu baseados no trabalho de Kurahashi e Terano, que em 2000 propuseram o uso de múltiplas listas tabu de modo a auxiliar o algoritmo genético a alcançar a solução de problemas multimodais ou multiobjetivos, aplicados à mineração de dados. À diferença do presente trabalho, no entanto, os estudos citados utilizavam as listas tabu na seleção (portanto antes do cruzamento).

[1] e [3] apresentam modelos matemáticos de programação inteira binária para o PRVJT. Os modelos exatos, contudo, não são práticos para problemas de grandes dimensões.

Diversos autores propuseram metodologias para a resolução do PRVJT, dos quais se pode citar [15], que propõem utilizar adaptações dos algoritmos busca-tabu e colônia de formigas e [9], os quais propõem uma metodologia híbrida de duas fases para resolução, com algoritmo genético agrupando e a busca-tabu roteirizando, obtendo excelentes resultados, de forma similar à [11], que combinam busca local com algoritmos evolucionários em duas fases.

Já [18] comparam o desempenho de diversas heurísticas e meta-heurísticas, incluindo algoritmos genéticos com melhorias locais, na resolução do PRVJT e [4] propõem um algoritmo genético híbrido paralelo, com duas populações evoluindo para objetivos diferentes, obtendo resultados muito bons. [19] propõe usar um algoritmo genético padrão, avaliando-o

sob duas métricas distintas: quantidade de veículos utilizados e distância percorrida, obtendo resultados satisfatórios, ainda que inferiores a outros já obtidos para as mesmas instâncias.

Alguns autores desenvolveram trabalhos para contornar a convergência para ótimos locais, entre os quais [13], que aumenta a aleatoriedade na determinação da ordem dos indivíduos a serem roteados em um algoritmo evolucionário e [20], que altera as taxas de cruzamento e mutação em um algoritmo genético, obtendo excelentes resultados.

3. METODOLOGIA PROPOSTA

A título de comparação, além da resolução pela heurística proposta, o PRVJT foi resolvido também através de um modelo matemático exato de programação inteira binária. No entanto, por tratar-se de um problema NP-Hard, a utilização deste modelo somente é viável para casos pequenos (poucos clientes). O modelo exato, baseado em [1], é apresentado a seguir. Neste modelo C={1,...,n} é o conjunto de nós que representam os clientes. O conjunto N=C ∪ {0,n+1} inclui todos os pontos, uma vez que 0 e n+1 representam o depósito (0 para início e n+1 para o final da rota). O conjunto E={(i,j): i,j ∈ N, i ≠j, i≠n+1, j≠0} corresponde aos arcos ligando os nós. A cada arco está associado um custo *Cij* e um tempo de viagem *Tij*. Cada cliente i tem um uma demanda *Di* a ser atendida por um conjunto *K* de veículos de capacidades iguais a *Q*. Cada cliente *i* deve ser atendido dentro de uma janela de tempo de limite inferior *Ai* e superior *Bi*. As variáveis *Xijk* representam a escolha ou não da rota *(i,j)* pelo veículo *k*. As variáveis *Sik* são o instante de tempo no qual o veículo *k* atende o cliente *i*. M representa um número muito grande, adequado de acordo com as dimensões das distâncias.

$$min \sum_{k \epsilon K} \sum_{(i,j) \epsilon E} Cij\ Xijk \qquad (1)$$

$$\sum_{k \epsilon K} \sum_{j \epsilon N} Xijk = 1 \qquad \forall \in C \qquad (2)$$

$$\sum_{i \epsilon C} Di \sum_{j \epsilon N} Xijk \leq Q \qquad \forall k \in K \qquad (3)$$

$$\sum_{j \epsilon N} Xojk = 1 \qquad \forall k \in K \qquad (4)$$

$$\sum_{i \in N} Xihk - \sum_{j \in N} Xhjk = 0 \qquad \forall h \in C\,, \qquad \forall k \in K \qquad (5)$$

$$\sum_{i \in N} Xi, n+1, k = 1 \qquad \forall k \in K \qquad (6)$$

$$Ai \leq Sik \leq Bi \qquad \forall i \in N - \{0\},\ \forall k \in K \qquad (7)$$

$$Sik + Tij \leq Sjk + (1 - Xijk)M, \qquad \forall (i,j) \in E,\ \forall k \in K \qquad (8)$$

$$Xiik = 0 \qquad \forall k \in K, \quad \forall i \in N \qquad (9)$$

$$Xijk = 0\ ou\ 1 \quad Sik \geq 0 \quad \in R \quad \forall (i,j) \in E, \qquad \forall k \in K \qquad (10)$$

A função objetivo (1) minimiza o custo total das rotas. O grupo de restrições (2) assegura que cada cliente é designado a um único veículo. As restrições (3) garantem que a demanda total de cada rota do veículo k não ultrapassa a capacidade Q. Os conjuntos de restrições (4), (5), (6) representam restrições de fluxo, exigindo que cada veículo k parta do depósito (nó 0) somente uma vez, deixe um nó h somente se tiver entrado neste nó e retorne obrigatoriamente e somente uma vez ao depósito (nó n+1). As restrições (7) garantem o

atendimento às janelas de tempo. O conjunto de restrições (8) impõe que, se o veículo deixa o nó i e viaja para o nó j, somente pode chegar em j depois de transcorrido o tempo de viagem Tij. As restrições (9) não permitem a ocorrência de rotas ligando um nó a si mesmo e as restrições (10) indicam o tipo de variável (no caso binária) de Xijk. As variáveis Sik são reais não negativas.

O modelo (1)-(10) será aplicado a um problema real de entrega de refeições e resolvido pela heurística proposta a seguir.

3.1 REPRESENTAÇÃO E DECODIFICAÇÃO DO CROMOSSOMO

A representação é o genótipo, ou seja, como a solução será trabalhada pelo algoritmo genético. Já a decodificação é necessária para fazer, a partir do genótipo, a leitura do fenótipo, ou seja, o que aquele genótipo de fato representa no problema em questão.

A representação do cromossomo (individuo da população representando um candidato a solução) proposta pelos primeiros algoritmos genéticos e até hoje de larga utilização é a binária [18]. Para o PRVJT e problemas de roteamento em geral, no entanto, essa representação não é a mais recomendada, pois aumenta o tamanho dos cromossomos (podendo gerar sobrecarga computacional) e dificulta o cruzamento por tratar-se de um problema em que todos os pontos estão em cada cromossomo. No caso específico do PRVJT ainda há a quantidade nos caminhões e restrições de janela de tempo e capacidade.

Assim, [19] propõe uma forma de representação e decodificação específica para o PRVJT. O cromossomo é representado por uma sequência dos clientes (que são representados por números), cuja ordem determina como eles serão inseridos nas rotas decodificadas. Desta forma, uma solução para um PRVJT de um grafo contendo n pontos é representada por um cromossomo de n -1 elementos, uma vez que o depósito não é representado na solução.

Quanto à decodificação, esta precisa ser feita de tal modo que crie soluções viáveis, ou seja, que atendam às restrições impostas pelo PRVJT. É feita através de um algoritmo descrito da seguinte maneira:

1. A cada caminhão é designada uma rota com inicio e término no depósito.
2. À rota sendo construída no momento são adicionados os clientes ainda não alocados em nenhuma rota de acordo com a ordem especificada no cromossomo. No entanto, o cliente somente é adicionado a essa rota caso respeite todas as restrições (demanda, capacidade, janelas de tempo)
3. Ao chegar ao final do cromossomo verifica-se se ainda há clientes no cromossomo ainda não alocados em nenhuma rota. Em caso afirmativo, começa-se a montagem de uma nova rota (novo caminhão) e retorna-se ao passo 2 [19].
4. As rotas nas quais nenhum cliente foi alocado (caminhões não utilizados) são formadas simplesmente pelo arco depósito-depósito. Essa operação pode parecer redundante, porém será importante na avaliação da solução.

De modo a garantir a factibilidade do cromossomo, é permitido ao caminhão que chegue ao cliente antes do limite inferior da janela de tempo determinado. Porém, o cliente somente será atendido e o caminhão só partirá para o próximo ponto após o limite inferior da janela de tempo ser alcançado.

3.2 AVALIAÇÃO DA SOLUÇÃO

O objetivo essencial do algoritmo é minimizar a distância total percorrida pelos caminhões (ou um custo relacionado aos percursos). Ao mesmo tempo, também é desejável a utilização da menor frota possível de caminhões, uma vez que a utilização do caminhão também acarreta um custo fixo.

Deste modo, procede-se da seguinte maneira para o cálculo do fitness (que constitui a função objetivo do problema de minimização): uma vez decodificado o cromossomo, somam-se as distâncias (custos) a serem percorridas em cada rota por cada caminhão (as distâncias

entre os pontos são conhecidas). Ao arco que liga o depósito ao próprio depósito (o caminhão que percorre esse arco portanto não é utilizado) é atribuído um custo (distância) negativo relacionado à economia associada a não utilização daquele caminhão da frota. Este custo negativo serve de incentivo à minimização da quantidade de caminhões paralelamente à minimização da distância percorrida. Para saber-se a distância real a ser percorrida deve-se somar de volta os custos negativos relacionados aos caminhões não utilizados.

3.3 O ALGORITMO

A seguir será descrito o algoritmo para resolução do PRVJT utilizando-se o algoritmo genético com listas tabu. O algoritmo pode ser organizado através dos seguintes passos, cujas etapas serão descritas em detalhes após. Sem a lista-tabu, os passos 5 e 9 não são realizados.

1. Entrada de dados
2. Gerar população inicial
3. Selecionar pais para cruzamento
4. Gerar soluções filhos através do crossing-over
5. Caso algum dos filhos estiver na lista tabu, substituí-lo por uma solução aleatória
6. Classificar a população e os filhos da melhor para a pior solução
7. Inserção das soluções filhas na população
8. Realizar Mutação
9. Inserir população na lista tabu e retirar os elementos mais antigos da mesma
10. Caso o critério de parada tenha sido atingido, apontar a melhor solução da população como sendo a solução final. Caso contrário, voltar ao passo 3

3.3.1 Entrada de Dados e geração da população inicial

Há informações características do problema específico que precisam ser informadas antes de rodar o algoritmo. São elas: distância entre os clientes; tempo de percurso entre os pontos; janelas de tempo de atendimento de cada cliente; demanda dos clientes; capacidade do caminhão. Cabe observar que a frota na resolução deve ser bem maior do que a necessária, para factibilizar todos os cromossomos. Em um caso prático podem ser utilizados "caminhões imaginários", pois o algoritmo tende, com o passar das gerações a minimizar a quantidade de caminhões utilizados, desta forma inutilizando esses caminhões extras na solução final.

Quanto à geração da população inicial, neste trabalho ela é gerada aleatoriamente.

3.3.2 Seleção de pais e cruzamento

Baseado na seleção natural, a escolha dos indivíduos como pais é feita com base na sua aptidão, ou seja, os melhores indivíduos têm maiores chances de serem escolhidos para transmitirem suas características adiante. Nesse trabalho utilizou-se um dos métodos mais utilizados, o da roleta, no qual a probabilidade de escolha de cada indivíduo é proporcional ao índice de aptidão do mesmo. Este índice é igual ao inverso do valor do fitness (de modo a fazer o melhor indivíduo ter o maior índice de aptidão). Deste modo, a probabilidade pode ser calculada dividindo o índice de aptidão do indivíduo pela soma dos índices de aptidão de todos os indivíduos da população. A quantidade de pais escolhidos para cruzamento (podendo um mesmo indivíduo ser escolhido mais de uma vez) é definida pela taxa de cruzamento.

O cruzamento, também chamado de *crossing-over* é a operação através da qual os indivíduos pais transmitem suas características para os indivíduos filhos. O operador de cruzamento clássico é o cruzamento de um ponto. No entanto, para o caso do PRVJT e qualquer outro problema no qual os cromossomos são permutações dos mesmos genes, este operador gera muitos filhos infactíveis [18].

Para resolver esse problema, diversos operadores de cruzamento foram desenvolvidos, entre os de maior destaque estão: PMX (*partially matched crossover),* CX (*cycle crossover*) e

OX (*order crossover*) [6], Neste trabalho optou-se pela utilização do operador PMX (*Partially Matched Crossover)*, proposto por Golberg e Lindle, em 1985 [16]. Esta modalidade de cruzamento é descrita pelo seguinte procedimento:

1. Sobre os cromossomos pais (p1 e p2) realiza-se dois pontos de corte aleatórios.
2. Os filhos (f1 e f2) herdam integralmente as porções entre os pontos de corte de p2 e p1 respectivamente.
3. No restante das posições, f1 e f2 herdam os genes de p1 e p2 desde que não gere infactibilidade, ou seja, desde que não se repitam com os genes já alocados no passo 2. Estas permanecem vazias.
4. As posições ainda vazias são preenchidas da seguinte forma: utilizando como exemplo f1, uma vez não podendo herdar um gene (no caso uma cliente) de p1 no passo 3, procura-se a posição em que esse cliente se encontra em p2 e preenche-se o espaço vazio de f1 com o elemento de p1 da posição encontrada. Caso este cliente também já faça parte de f1 repete-se o processo com esse novo gene até gerar uma solução factível. Com f2 realiza-se a operação análoga (trocando todos f1 por f2, p1 por p2 e p2 por p1) [7].

3.3.3 Lista tabu

Um dos problemas mais comuns associado aos algoritmos genéticos é a convergência prematura a ótimos locais. Estes podem gerar "superindivíduos", que dominam a reprodução e, caso isso não seja controlado por algum artifício, toda a população acaba descendendo dele. Com isso, após algumas gerações, a população é formada por indivíduos idênticos ou muito semelhantes [20], fazendo com que o operador de cruzamento não cause qualquer alteração. Caso isso ocorra após um número pequeno de iterações, este "superindivíduo" pode representar um ótimo local muito distante do ótimo global.

Entre as medidas para combater a perda de diversidade pode-se citar o aumento na taxa de mutação, diminuição na pressão de seleção (mecanismos que permitam com que indivíduos ruins sejam selecionados para cruzamento) e controlar o número de filhos do "superindivíduo". Contudo, no caso particular do PRVJT, no qual boas soluções estão em pequenas partes do espaço de busca [14], o sucesso dessas medidas nem sempre é satisfatório.

O presente trabalho propõe uma metodologia para solucionar esse problema através de listas tabu. Estas são características da meta-heurística Busca Tabu e tem como intuito justamente evitar a convergência para ótimos locais. No caso da Busca Tabu, a lista representa os movimentos (alterações na solução) realizados nas últimas iterações. Aqui, a lista tabu será composta pelos indivíduos das últimas gerações do algoritmo genético. Assim, todos os filhos gerados em uma geração são comparados com os indivíduos presentes na lista tabu. Caso estejam presentes na lista, estes são abortados e substituídos por soluções geradas aleatoriamente. Desta forma, preserva-se a diversidade da população e evita-se a convergência para ótimos locais.

O tamanho da lista tabu é outro parâmetro a ser definido. Listas muito pequenas podem possibilitar ao algoritmo contorná-la e acabar apontando para os ótimos locais da mesma forma que sem as listas. Já listas muito grandes podem fazer com que indivíduos bons deixem de ser gerados e atrasa o progresso do algoritmo.

3.3.4 Classificação, inserção das soluções filhos na população e mutação

Este passo consiste no ranqueamento da população e dos filhos gerados separadamente, do melhor para o pior indivíduo, baseado no fitness de cada um.

Após isso se efetua a atualização da população através da entrada de cromossomos filhos e a saída de cromossomos já presentes na população. Neste caso, foi garantida a permanência dos 10% melhores indivíduos da população da geração anterior e assegurada a entrada do melhor filho gerado. A escolha dos demais filhos a entrarem na população e

cromossomos a serem retirados foi feita de forma aleatória como mecanismo de preservação de diversidade. A taxa de renovação da população varia bastante de estudo para estudo.

Uma vez atualizada a população, cada cromossomo pode ou não sofrer mutação. Em relação ao genótipo, a mutação representa a troca de posições no cromossomo entre dois genes aleatórios, operador chamado de *swap* (troca) [19].

O único indivíduo que não deve sofrer mutação é o melhor elemento da população, de modo a sempre manter a melhor solução até então encontrada na população.

3.3.5 Atualização da lista tabu e critério de parada

Ao final de cada geração (iteração) a lista tabu deve ser atualizada. Essa operação dá-se de maneira bastante elementar. A população ao final da iteração é inserida na lista tabu. Em contrapartida, os elementos da população a mais tempo presente na lista tabu são retirados da mesma. Deste modo, a lista tabu contém sempre os elementos das últimas X gerações, com X sendo o parâmetro de tamanho da lista determinado pelo usuário.

Terminada a iteração é verificado o critério de parada do algoritmo. Neste trabalho foi utilizado o critério do número máximo de iterações. Portanto, caso esse número máximo de iterações tenha sido atingido, o algoritmo é finalizado e o melhor cromossomo encontrado até então apontado como solução do problema. Caso contrário retorna-se ao processo de seleção de indivíduos para cruzamento.

3.4 TESTES REALIZADOS

Para testar o algoritmo descrito e principalmente comparar seu desempenho com e sem a lista-tabu, foi utilizado um problema real. Este é um problema de uma cooperativa que oferece, entre outros serviços, o de entrega de refeições aos seus associados.

Cada cliente tem uma demanda específica e uma janela de tempo na qual deve ser atendido. O atendimento pleno das especificações de horário é importante, pois tratam-se de funcionários que devem realizar as refeições em seus intervalos determinados pelo trabalho e há reclamações caso o atendimento não ocorra neste período. Além de assegurar a entrega dentro das janelas de tempo, a minimização da distância total a ser percorrida é importante, pois os custos de transporte associados são de extrema relevância, especialmente por tratar-se de uma associação de pequeno porte e orçamento apertado.

A associação atende diariamente até 53 clientes com localizações definidas, bem como é conhecida a localização da central, de onde partem os carros que realizam as entregas. As distâncias entre os clientes são conhecidas. Cada cliente tem uma demanda de refeições, que variou de 1 até 16, tendo cada carro a capacidade de armazenar 200 refeições. Os horários das entregas de refeições variam entre 11:30 e 13:30. Sendo assim, foram geradas janelas de tempo aleatórias para cada cliente dentro deste intervalo de 120 minutos, as quais devem ser atendidas com precisão. Para o cálculo do tempo de viagem, foi considerada a velocidade média de 60km/h. Deste modo, tendo a distância entre os locais pode-se estimar o tempo de viagem. Por tratar-se de uma entrega simples que não exige qualquer tipo de operação complexa de descarregamento, o tempo de atendimento do cliente foi desconsiderado.

Para efeito de testes, além do problema completo, o algoritmo também foi testado para somente 9 destes clientes (neste caso sendo possível a comparação com a solução exata) e 14 clientes.

Os parâmetros utilizados estão na tabela a seguir e foram obtidos a partir de testes.

Tabela 1: parâmetros do algoritmo.

Taxa de cruzamento	80%
Taxa de renovação	80%
Tamanho da população	30
Taxa de mutação	5%

A quantidade de iterações variou conforme o tamanho do problema. Para o de 9 clientes o algoritmo era interrompido após 200 iterações. Para os problemas com 14 e 53 clientes o número de iterações foi de 2000 e 15000 respectivamente.

O tamanho da lista-tabu foi definido como 5, ou seja, a lista contém os indivíduos das 5 últimas gerações (iterações do algoritmo).

Para cada uma das situações a seguir a metodologia foi testada com e sem a lista, cada vez com 50 replicações. Além da distância total, foi observado também o tempo computacional gasto.

Os testes foram realizados no programa MatLab e rodados em uma máquina com processador intel core i5 2,30 GHz com 4GB de memória RAM. Para resolução do modelo exato foi utilizado o programa LINGO 12.0 no mesmo computador.

4. RESULTADOS

A seguir são apresentados os resultados obtidos nos testes realizados. Para o problema com 9 clientes foi possível obter a solução exata (ótima). Para os demais, há somente os resultados obtidos através da heurística proposta (com e sem lista – tabu), demonstrando a importância do desenvolvimento de métodos heurísticos para a resolução de problemas de maiores dimensões. Na tabela os problemas de 9, 14 e 53 clientes são representados, respectivamente como P9, P14 e P53. As melhores e piores soluções correspondem as melhores e piores rotas finais definidas para todos os veículos (soma das distâncias de cada um).

Tabela 2: Resultados encontrados.

		P9	P14	P53
Método exato	solução encontrada (m)	29155		
	tempo computacional (seg)	185		
Heurística com lista-tabu	média das soluções (m)	29263,6	34672	197065,6
	melhor solução (m)	29155 (48 vezes)	33605 (22 vezes)	150175
	pior solução (m)	31955	37135	243330
	tempo computacional médio por replicação (seg)	4,51	59,50	1092,20
Heurística sem lista-tabu	média das soluções (m)	31106,9	37760,9	212612,4
	melhor solução (m)	29155 (13 vezes)	33605 (3 vezes)	174325
	pior solução (m)	33310	51285	271100
	tempo computacional médio por replicação (seg)	3,97	49,28	1012,18

Analisando os resultados constata-se a melhoria obtida na utilização da lista-tabu.

No problema de 9 clientes, as distâncias mínimas encontradas no algoritmo com a lista-tabu foram, em média 5,93% menores do que as observadas sem a mesma. Neste caso específico, no qual é conhecida a solução ótima, observa-se que quando é utilizada a lista-tabu o algoritmo encontrou a solução ótima em 96% das replicações, frente a somente 26% sem a lista. Além disso, observa-se que a pior solução entre as 50 replicações também foi menor naquelas com a lista-tabu, comprovando que a lista-tabu ajuda na fuga de ótimos locais ruins. Finalmente, o tempo computacional aumentou em 13,51% quando a lista-tabu foi utilizada.

No problema de 14 clientes, as distâncias a serem percorridas foram em média 8,18% menores com a lista-tabu. O algoritmo chegou 22 vezes em uma mesma solução (33605 m), sendo esta a melhor encontrada, sugerindo que seja a solução ótima, ainda que não comprovado. Assim como no problema de 9 clientes, a pior replicação entre as "com-lista" foi bastante melhor do que sem a lista e houve certo incremento no tempo computacional.

Finalmente, no caso completo (53 clientes), houve melhora de 7,31% em média nas replicações com a lista-tabu. A melhor solução também foi encontrada em uma replicação

com a presença da lista. Analogamente aos outros casos, as replicações com lista-tabu não permitiram soluções muito ruins, fazendo com que a pior replicação com a lista seja melhor do que a pior sem a lista. Com relação ao tempo computacional, houve aumento de 7,91%.

Os gráficos 1, 2 e 3 representam uma replicação do algoritmo para cada problema, comparando-se a evolução da melhor solução da população a cada iteração com e sem a lista-tabu. O eixo das abscissas representa a quantidade de iterações transcorridas, já nas ordenadas estão as distâncias da rota do melhor indivíduo da população em quilômetros.

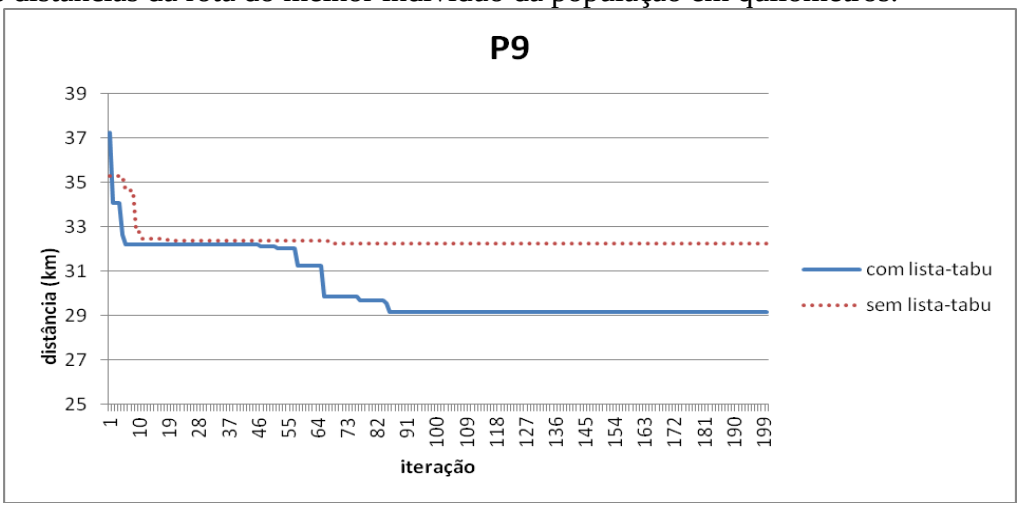

Gráfico 1: Problema de 9 clientes

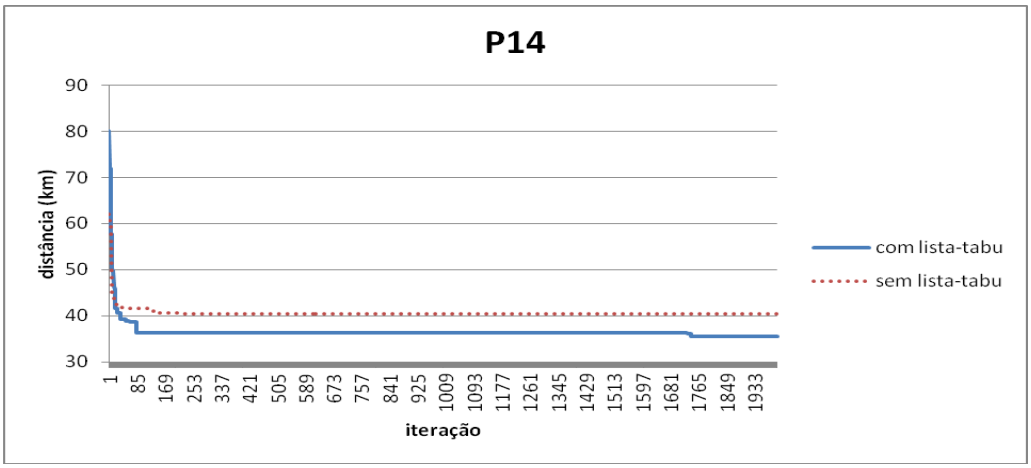

Gráfico 2: Problema de 14 clientes

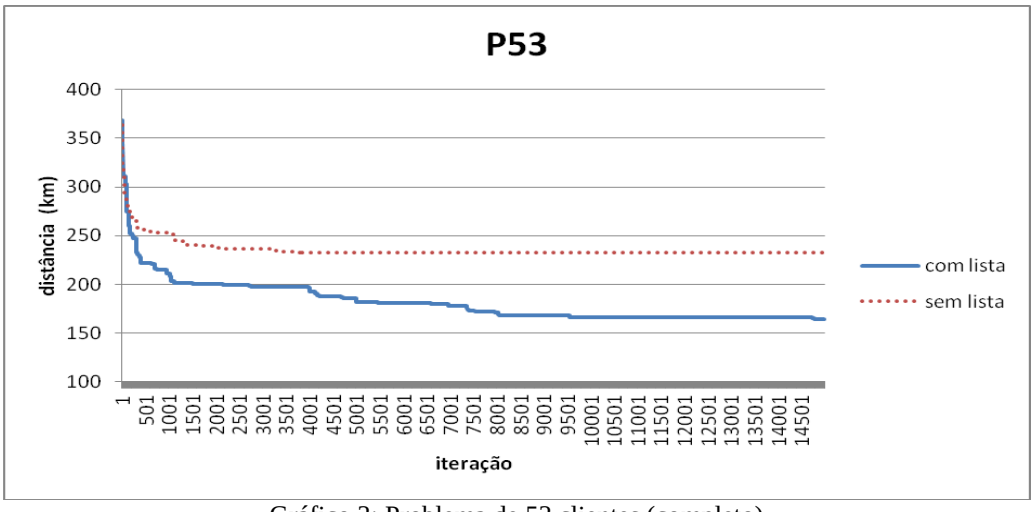

Gráfico 3: Problema de 53 clientes (completo)

Os gráficos comparando a evolução do melhor indivíduo da população com e sem a lista-tabu explicita o efeito da lista-tabu sobre o algoritmo genético e demonstra que a lista-tabu cumpre seu papel de evitar a convergência para ótimos locais. Nos três gráficos (especialmente no de 9 e 53 clientes) é possível observar que, após um pequeno número de iterações, as melhorias na solução são nulas ou quase nulas quando o algoritmo genético opera sem a lista-tabu. Quando a lista-tabu é utilizada, o algoritmo consegue "fugir" de ótimos locais ruins, abranger uma maior área de busca e continuar melhorando a solução por um maior número de iterações.

Conclui-se disso que a probabilidade de encontrar uma solução boa é maior quando o algoritmo inclui a lista-tabu.

5. CONCLUSÕES FINAIS

Conclui-se deste estudo que a introdução de lista-tabu constitui-se em uma alternativa interessante para melhorar o desempenho do algoritmo genético na resolução do problema de roteamento de veículos com janelas de tempo ao evitar convergências prematuras para ótimos locais ruins.

Considerando-se que os custos de transporte representam parcela significativa no orçamento de empresas que realizam entregas ou coletas de produtos, a redução na distância total percorrida é de suma importância. Além disso, casos como o do estudo descrito neste trabalho, nos quais há janelas de tempo a serem atendidas, requerem atenção ao atendimento dos horários, visando um serviço de qualidade.

Dada a inviabilidade da utilização de um modelo matemático exato à um PRVJT de grandes dimensões, a implementação do algoritmo aqui proposto permite elaborar em tempo hábil um roteiro de entrega de refeições que minimize a distância e garanta simultaneamente a qualidade desejada pelos clientes no serviço.

6.REFERÊNCIAS BIBLIOGRÁFICAS

[1] ARENALES, M.; ARMENTANO, V.; MORABITO, R.; YANASSE, H. *Pesquisa Operacional***.** 6ª Ed. Rio de Janeiro: Eslevier, 2007.

[2] BAKER, B.M; AYECHEW, M.A. A genetic algorithm for the vehicle routing problem. *Computers and Operations Research***,** v.30 p. 787-800, 2003.

[3] BALDACCI, R; MINGOZZI, A; ROBERTI, R. Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research***.** v. 218. p. 1-6, 2012.

[4] BERGER, J; BARKAOUI, M. A parallel hybrid genetic algorithm for the vehicle routing problem with time windows. *Computers & Operations Research***.** v. 31. p. 2037-2053, 2004.

[5] CAVALHEIRO, A.F. *GADBMS – Um algoritmo genético minerador de dados para base de dados relacionados***.** 74f. Dissertação (mestrado em informática) – Setor de Ciências Exatas, Universidade Federal do Paraná, Curitiba, 2002.

[6] CHAN, K.C; TANSRI, H. A study of genetic crossover operations on the facilities layout problem. *Computers and Industrial Engineering***.** v.26 p.537-550, 1994..

[7] EL-BAZ, M.A. A genetic algorithm for facility layout problems of different manufacturing environments. *Computers and Industrial Engineering***.** v. 47. P. 233-246, 2004.

[8] HILLIER, F.S.; LIEBERMAN, G.J. *Introdução à pesquisa operacional***.** 8ª Ed. Porto Alegre: AMGH, 2010.

[9] HOMBERGER, J; GEHRING, H; A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *European Journal of Operational Research***.** v. 162 p. 220-238, 2005.

[10] LOPES, F.M. *Algoritmo Genético restrito por Listas Tabu no contexto de mineração de dados*. 77f. Dissertação (mestrado em informática) – Setor de Ciências Exatas, Universidade Federal do Paraná, Curitiba 2001.

[11] MESTER, D; BRÄYSY, O. Active guided evolution strategies for large-scale vehicle routing problems with time windows. *Computers & Operations Research*. v. 32. p. 1593-1614, 2005.

[12] NASCIMENTO, I.Z. *Abordagens determinística e estocástica para o problema de roteirização de veículos na entrega de refeiçõe***s**. 98f. Dissertação de mestrado (programa de pós-graduação em métodos numéricos em engenharia) – Setor de Tecnologia, Universidade Federal do Paraná, Curitiba, 2011.

[13] OLIVEIRA, H.C.B; VASCONCELO, G.C; ALVANEGRA, G.B; *Uma abordagem evolucionária para o problema de roteamento de veículos com janela de tempo.* XXXVII SBPO, Gramado, 2005.

[14] OMBUKI B; ROSS, B. J; HANSHAR, F. Multi-Objective Genetic Algorithms for Vehicle Routing Problem with Time Windows. *Applied Intelligence***.** v.24. p. 17-30, 2006.

[15] PUREZA, V.; MORABITO, R; REIMANN, M. Vehicle routing with multiple deliverymen: Modeling and heuristic approaches for the VRPTW. *European Journal of Operational Research.* v. 218, n. 3, p. 636-647, maio 2011.

[16] SILVA, A.F; OLIVEIRA, A.C; *Algoritmos genéticos: alguns experimentos com os operadores de cruzamento ("Crossover") para o problema do caixeiro viajante assimétrico***.** XXVI ENEGEP, Fortaleza, 2006.

[17] TAILLARD, E; BADEAU, P.; GENDREAU, M.; GUERTIN, F.; POTVIN, J. A Tabu Search Heuristic for the Vehicle Routing Problem with Soft Time Windows. *Transportations Science***.** v.31, n.2, p 170 – 186, 1997.

[18] TAN, K.C; LEE, L.H; OU, K; ZHU, Q.L. Heuristic methods for vehicle routing problem with time windows. *Artificial Intelligence in Engineering***.** V.15 p 281-295, 2001.

[19] VIANA, F.H.F. Um Algoritmo Genético para o Problema de Roteamento de Veículos com Janelas de Tempo. *Revista de Inteligência Computacional Aplicada (RICA)***.** n.8, julho 2010.

[20] ZHU, K. Q. A Diversity-controlling Adaptive Genetic Algorithm for the Vehicle Routing Problem with Time Windows. *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'03).* p. 176-183, 2003.