UMA ABORDAGEM MATHEURÍSTICA PARA O PROBLEMA DE MÍNIMA LATÊNCIA

Resumo: Este trabalho estuda o Problema de Mínima Latência (PML), que é uma variação do Problema do Caixeiro Viajante. Ele é resolvido por meio da implementação das matheurísticas RENS, FP e *Local Branching*. O algoritmo final proposto é uma hibridização das técnicas RENS e *Local Branching* sendo testado e comparado com problemas teste da literatura, a fim de demonstrar a eficácia da técnica. As implementações demonstraram resultados ótimos para problemas de até 25 elementos, sendo que para tais problemas foram obtidos resultados médios com um *gap* do ótimo de 4.84% que é aproximadamente 16% melhor que a execução padrão do *solver* com um tempo limite predeterminado.

Palavras Chaves: Problema de Mínima Latência, Matheurísticas, RENS, FP, *Local Branching*.

Abstract: This paper studies the Minimum Latency Problem (PML), which is a variation of the Traveling Salesman Problem. It is solved through the implementation of RENS, FP and Local Branching matheuristics. The proposed algorithm is a hybridization of RENS and Local Branching techniques and was tested and compared with literature instances to demonstrate the effectiveness of the technique. The implementations found optimal results for problems up to 25 elements, and for these problems we obtained average results with an optimum gap of 4.84% which was approximately 16% better than the standard execution of the solver with a predetermined time-out.

Keywords: Minimum Latency Problem, Matheuristics, RENS, FP, *Local Branching*.

1. Introdução

Problemas de análise combinatória estão presentes em várias áreas e são aplicáveis a inúmeros problemas do nosso cotidiano, desde uma simples escolha de qual caminho seguir para chegar a um destino, até decisões complexas, como por exemplo, a correta organização de um quadro de horários para alocação de recursos.

A resolução de tais problemas, obtendo um resultado ideal ou o que normalmente é chamado de resultado ótimo, podem requerer um tempo de processamento muito alto do ponto de vista do demandante, até mesmo quando se tem poucos elementos dependendo da estrutura do problema.

A análise e estudo de técnicas que possam trazer resultados próximos de um resultado dito ideal com um tempo aceitável para o demandante, se faz muito importante, visto que, em algumas áreas, a diferença de segundos pode justificar ou não o uso da informação gerada. Para os chamados problemas de otimização combinatória, o domínio é, tipicamente, finito. Dada a impossibilidade computacional de se examinar todos os possíveis elementos de um domínio, algum tipo de abordagem na qual apenas uma fração dos elementos é examinada, deve ser utilizada.

O campo de pesquisa de métodos heurísticos para aplicações de problemas de otimização combinatória é amplo. Isto é devido à importância de problemas de otimização para a comunidade científica, bem como para as áreas de produção e operações (BLUM; ROLI, 2003). Em sua forma geral, problemas de otimização têm como objetivo maximizar ou minimizar uma função definida sobre um determinado domínio (GENDREAU; POTVIN, 2005). As heurísticas exploram uma área limitada do espaço de soluções e, em teoria, produzem respostas aceitáveis em um tempo computacional inferior ao tempo de resposta de uma solução exata (LOOCK; HINNEN, 2015). De acordo com OSMAN; LAPORTE (1996) e NORONHA; DA SILVA; ALOISE (2001), uma metaheurística é definida como um processo de geração iterativa, que conduz ao desenvolvimento de novas classes de heurísticas que melhoram consideravelmente a eficiência de algoritmos aproximados. Elas são projetadas para atacar os problemas complexos de otimização onde os métodos heurísticos e clássicos de otimização não conseguiram ser eficazes e eficientes (AGARWAL; COLAK; ERENGUC, 2015). "Uma metaheurística constitui uma estrutura mais genérica baseada em princípios ou conceitos sobreposta a uma determinada heurística" (LISBOA, 2007).

Dado o avanço tecnológico tanto em software quanto em hardware, aliado aos avanços dos métodos exatos, modelos que utilizam programação linear inteira mista, do inglês *mixed integer linear programming* (MILP), apresentam resultados ótimos ou limites próximos ao ótimo em tempo computacional razoavelmente aceitável. Dado tal fato, pesquisadores passaram a incorporar métodos heurísticos nas etapas de solução de um MILP, procedimento esse que é definido como matheurística (ARCHETTI; SPERANZA, 2014).

As matheurísticas são algoritmos heurísticos desenvolvidos através da capacidade de mesclar técnicas de programação matemática com metaheurísticas. A principal característica que difere a técnica de uma metaheurística é a de utilizar características derivadas do modelo matemático do problema em questão (BOSCHETTI et al., 2009).

Esse trabalho estuda o Problema de Mínima Latência (PML), que é uma variação do Problema do Caixeiro Viajante (PCV), apresentando uma formulação matemática e métodos matheurísticos para a resolução do problema, testando e comparando os resultados obtidos na resolução de problemas teste disponíveis na literatura, a fim de demonstrar a eficácia da técnica. Esta é uma pesquisa aplicada, quantitativa e no que se refere ao procedimento técnico utilizado, trata-se de uma pesquisa experimental.

Esse trabalho está organizado da seguinte forma: a seção 2 define o problema. A seção 3 apresenta de forma detalhada os procedimentos metodológicos adotados, a seção 4 apresenta matheurísticas implementadas, uma séria de experimentos e seus respectivos resultados são apresentados na seção 5 e, finalmente na seção 6, são expostas as conclusões e considerações finais.

2. Definição do problema

O Problema de Mínima Latência (PML), do inglês Minimum Latency Problem (MLP) é uma variante do Problema do Caixeiro Viajante (PCV). Nesse problema, o objetivo é encontrar um circuito hamiltoniano iniciando em um único depósito que minimize a soma dos tempos de espera (latência) dos consumidores (SARUBBI, 2008). Esse problema também é conhecido na literatura como *Traveling Repairman Problem* (TSITSIKLIS, 1992), *Delivery Man Problem* (FISCHETTI et al., 1993), *Cumulative Traveling Salesman Problem* (BIANCO et al., 1993) e *School Bus Driver Problem* (CHAUDHURI et al., 2003).

O PML pode ser aplicado, por exemplo, a um serviço de entrega de produtos a clientes em que o foco está em reduzir o tempo de espera para o recebimento e não no custo operacional da entrega. De forma geral, se diz que o PML é orientado ao cliente enquanto o PCV é orientado ao fornecedor (ARCHER; WILLIAMSON, 2003).

O problema definido a seguir exemplifica o PML: um distribuidor que precisa fazer a entrega de várias mercadorias, para diversos clientes em localidades distintas, visando atendê-los no menor tempo possível, este tempo é calculado da saída do depósito até o destino e o tempo de retorno para o depósito (WU et al., 2004).

A grande diferença entre o PML e o PCV está justamente na função objetivo, pois ambos podem ser mapeados facilmente para um grafo, gerando um problema NP-Completo. Para exemplificar as diferenças entre os problemas, uma analogia é feita levando em consideração um grafo não direcionado conforme a Figura 1. Cada nodo exibe o tempo total de espera e as arestas correspondem a uma unidade de tempo entre um nodo e outro. O caminho destacado com a cor azul representa uma solução viável tanto para o PCV quanto para o PML, porém a função de avaliação para ambos resulta em valores diferentes. Enquanto o objetivo do PCV é minimizar o tempo total de visita dos nodos o PML visa minimizar a soma das latências de cada nodo e essa latência varia de acordo com uma dada solução.

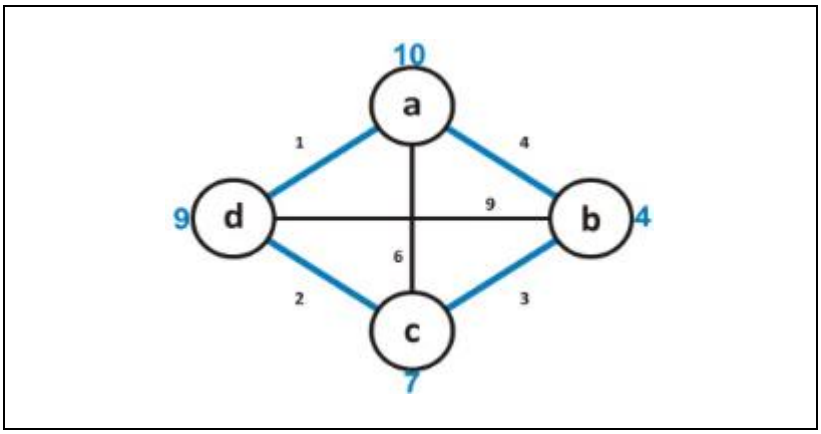

FIGURA 1 - Grafo não direcionado. Fonte: elaborado pelo autor.

O caminho C destacado (C = {a,b,c,d,a}) é uma solução viável para o PCV e também para o PML, porém o resultado da função objetivo para cada problema é calculado de forma diferente. Para o PCV o resultado total t é calculado conforme a expressão (1.1):

$$t = (4 + 3 + 2 + 1)$$
$$t = 10 \qquad (1.1)$$

Levando-se em consideração que os valores de cada aresta expressam uma unidade de tempo para a conclusão do percurso, o caminho C visa minimizar o tempo de retorno ao nodo inicial, ou depósito. Entretanto, analisando isoladamente cada nodo e o seu tempo de espera, o problema toma outras proporções:

- Nodo b: 4 unidades de tempo para concluir o percurso A-B;
- Nodo c: 7 unidades de tempo (4 unidades de tempo para concluir a visita ao nodo b e 3 unidades de tempo para concluir o percurso B-C);
- Nodo d: 9 unidades de tempo (7 unidades de tempo para concluir a visita ao nodo c e 2 unidades de tempo para concluir o percurso C-D);
- Nodo a: 10 unidades de tempo (9 unidades de tempo para concluir a visita ao nodo d e 1 unidade de tempo para concluir o percurso D-A);

Sendo a função objetivo do PML a soma do tempo de espera (latência) de cada nodo, temos o seguinte resultado (latência total lt), na expressão (1.2), utilizando esse caminho:

$$\begin{aligned} lt &= 4 + (4+3) + (4+3+2) + (4+3+2+1) \\ lt &= (4*4) + (3*3) + (2*2) + (1*1) \\ lt &= 30 \end{aligned} \tag{1.2}$$

Através de uma observação superficial na maneira como o cálculo é realizado, pode-se tentar otimizar o caminho deslocando o nodo com menor latência para o início do percurso, pois é ele quem implica no maior acúmulo de tempo. Um caminho seguindo essa visão pode ser configurado como {a,d,c,b,a}. Entretanto, esse caminho acarretará em uma latência total de 43 unidades de tempo.

O resultado desse novo caminho é maior que o resultado anterior, pois aplicamos o deslocamento apenas no primeiro elemento. Aplicando-se essa abordagem em todos os elementos teremos o caminho {a, b, c, b, a} com uma latência total de 20 unidades de tempo. Esse deslocamento não é a única técnica, e nem de longe a técnica perfeita, para a resolução do problema.

SARUBBI (2008) apresenta uma formulação matemática para o problema. Considere o grafo $G = (V, A)$, onde V é um conjunto de vértices e A é um conjunto de arcos. Suponha que exista uma origem $1 \in V$ e, para cada vértice $k \in V$, que inclui também o vértice 1, uma demanda única deve ser entregue por um circuito que minimize a soma dos tempos de espera de cada vértice. Um modelo de programação linear inteira mista pode ser definido utilizando como parâmetros de entrada os tempos necessários para percorrer os arcos (variável c_{ij}).

$$c_{ij} = tempo\ gasto\ para\ percorrer\ o\ arco\ (i,j)$$

Além das variáveis de decisão indicando se um arco foi utilizado (variável x_{ij}) e o fluxo total de produtos que passam pelo arco (variável f_{ij}).

$$x_{ij} = \begin{cases} 0, & se\ o\ veículo\ atravessa\ o\ arco\ (i,j) \\ 1, & caso\ contrário \end{cases}$$
$$f_{ij} = fluxo\ total\ de\ produtos\ que\ passam\ pelo\ arco\ (i,j)$$

A formulação para o Problema de Mínima Latência proposta por SARUBBI (2008) é dada por:

$$\text{minimize} \quad \sum_{(i,j)\in A} c_{ij} f_{ij} \tag{2.1}$$

sujeito a

$$\tag{2.2}$$

$$\sum_{i\in V|i\neq j} x_{ij} = 1 \qquad \forall j \in V \tag{2.3}$$

$$\sum_{j\in V|i\neq j} x_{ij} = 1 \qquad \forall i \in V \tag{2.4}$$

$$\sum_{j\in V|j\neq 1} f_{1j} = n \tag{2.5}$$

$$\sum_{i\in V} f_{ik} - \sum_{j\in V} f_{kj} = 1 \qquad \forall k \in V - \{1\} \tag{2.6}$$

$$\sum_{i,j\in V} f_{ij} = \frac{n(n+1)}{2} \tag{2.7}$$

$$\tag{2.8}$$

$$f_{ij} \leq |V| x_{ij} \qquad \forall i,j \in V \tag{2.9}$$

$$f_{ij} \geq 0 \qquad \forall i,j \in V$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in V$$

A função objetivo (2.1) minimiza o tempo total de espera para entrega dos produtos. As restrições (2.2) e (2.3) garantem que só chega e sai apenas um arco de cada vértice. O conjunto de restrições (2.4) assegura que o fluxo que sai do vértice origem é igual ao número de vértices. As restrições (2.5) forçam que em cada vértice seja entregue sua demanda unitária. A restrição (2.6) é uma restrição redundante que restringe o somatório de todos os fluxos na rede em $\frac{n(n+1)}{2}$, sendo n a cardinalidade de V. Segundo SARUBBI (2008) tal restrição é responsável por uma melhora nos *gaps* de relaxação linear da formulação. As restrições (2.7) e (2.8) garantem que não pode haver fluxo nos vértices que não foram escolhidos na solução ótima e que esse fluxo é positivo. E, finalmente, restrições (2.9) asseguram a integralidade da solução. O autor ainda ressalta que são as variáveis f_{ij} que asseguram a eliminação de subciclos inválidos.

3. Método de pesquisa

Do ponto de vista de sua natureza, esta é uma pesquisa aplicada. Quanto a abordagem ao problema é quantitativa. No que se refere ao procedimento técnico utilizado, trata-se de uma pesquisa experimental. Os experimentos foram realizados através do processamento dos problemas testes, coletando os dados da melhor rota e tempo de execução. Ao todo foram processados 50 problemas teste, divididos em 5 grupos de acordo com a quantidade de elementos: 10, 12, 15, 20 e 25 elementos. Cada grupo possuía 10 problemas testes.

Cada problema teste foi processado por meio de uma construção inicial através da matheurística *Relaxation Enforced Neighborhood Search* (RENS) e *Feasibility Pump* (FP), a fim de verificar qual construção gerava melhores resultados. Para o método FP a estratégia *Local Search* foi utilizada em caso de ciclagem. Após a construção da solução inicial a matheurística *Local Branching* é aplicada como refinamento, com o objetivo de verificar qual construção apresenta melhores resultados finais em relação ao *gap* (definido na seção 5.1) do ótimo e também ao atempo de execução. As metaheurísticas foram implementadas utilizando a linguagem de programação C++ e o *solver* IBM Cplex na versão 12.63. Para cada problema teste era conhecido o resultado ótimo (SARUBBI, 2008).

4. Métodos heurísticos utilizados

Para desenvolvimento do trabalho em questão, conforme descrito na seção 3, foram utilizados os seguintes métodos: *Relaxation Enforced Neighborhood Search* (RENS), *Feasibility Pump* (FP) e *Local Branching* (LB). Os procedimentos são descritos nas seções subsequentes.

4.1 Relaxation Enforced Neighborhood Search (RENS)

O algoritmo de busca RENS é um método que integra as heurísticas *Large Neighborhood Search* (LNS), que por sua vez são baseadas no método *Local Search* (LS). Muito utilizado na solução de problemas MIP considerados NP-difíceis, o método LNS tem como critério de busca, a definição de uma vizinhança de tamanho consideravelmente grande em um determinado ponto e então, com apenas uma iteração, a vizinhança é totalmente e/ou parcialmente explorada na busca da solução ótima ou de uma solução viável. Segundo BERTHOLD (2009) o método RENS investiga o conjunto de todos os arredondamentos possíveis de uma solução relaxada do LP (*Linear Problem*).

Após relaxar o problema inicial, resolver o problema relaxado e de posse da solução, que pode conter valores fracionários que vão de 0 (zero) a 1 (um), a matheurística RENS realiza um processo de arredondamento dos valores encontrados, para o inteiro mais próximo, ou seja, gera um conjunto solução pertencente ao conjunto binário. Por consequência é criado um subproblema reduzido, contendo os números inteiros de valor 1 (um), que será utilizado para realizar o processo de busca da solução ótima, viável ou em alguns casos de uma solução inicial.

4.2 *Feasibility Pump* (FP)

FISCHETTI; GLOVER; LODI (2005) propuseram uma matheurística para construir soluções factíveis de forma geral para um MIP, chamado *Feasibility Pump* (FP). Segundo BERTACCO et. al (2007) método FP trabalha com um par de pontos $(x^*, \bar{x})$ com $x^* \in P$ e $\bar{x}$ inteiro, que são atualizados iterativamente com o objetivo de reduzir tanto quanto possível a distância entre eles dada por $\Delta(x^*, \bar{x})$. Para ser mais específico, inicia-se com qualquer $x^* \in P$, e uma solução inteira tipicamente infactível $\bar{x}$ como arredondamento de x^*. Em cada iteração de FP, chamada *pumping cycle*, $\bar{x}$ é fixado e se encontra através da programação linear o ponto $x^* \in P$ que é o mais próximo possível de $\bar{x}$. Se $\Delta(x^*, \bar{x}) = 0$, então x^* é uma solução factível para o MIP, e o método finaliza. Do contrário, $\bar{x}$ é substituído pelo arredondamento de x^* de modo a reduzir ainda mais $\Delta(x^*, \bar{x})$, e o processo é iterado.

4.3 *Local Branching* (LB)

Conforme FISCHETTI e LODI (2005) a matheurística *Local Branching* (LB) pode ser descrita supondo uma dada solução de referência factível $\bar{x}$ que um dos objetivos seja encontrar uma solução melhorada que não esteja "tão distante" de $\bar{x}$. No método é adicionada uma restrição de *branching* ($\Delta(x, \bar{x}) \leq k$) que visa definir a distância máxima k entre as duas soluções $(x, \bar{x})$ e, além disso, a heurística

realiza a busca, de acordo com uma dada solução, em uma vizinhança de tamanho (k) pré-definido denominada *k-OPT* que passa a ser considerada como o conjunto de soluções que atendem a restrição adicional, onde k é um número de valor inteiro que expressa o raio de busca entre a solução $\bar{x}$ e a solução a ser encontrada, para que então seja minimizado o tempo de busca e o "esforço" computacional para achar a solução.

Para FISCHETTI e LODI (2005) ao definir o valor do tamanho do parâmetro de vizinhança k, é possível tornar a vizinhança suficientemente pequena para que o tempo de busca seja minimizado, mas ao mesmo tempo grande o suficiente para provavelmente conter soluções melhores do que $\bar{x}$. Para esse trabalho foi adotada uma vizinha de tamanho 1 (k=1).

5. Experimentos e resultados

Todos os experimentos foram realizados em uma máquina dotada de um processador Intel Core i7 de 2.9 GHz, 16GB de memória RAM com o sistema operacional macOS High Sierra. As metaheurísticas foram implementadas utilizando a linguagem de programação C++ e o *solver* IBM Cplex na versão 12.63.

Os problemas teste utilizados para os experimentos são os mesmos utilizados por SARUBBI (2008). Eles foram gerados obtendo custos inteiros aleatórios entre 1 e 100, armazenando-os em uma matriz de adjacência, seguindo os mesmos passos que FISCHETTI; LAPORTE; MARTELLO (1993). Os problemas teste foram arranjados em 5 grupos de acordo com o número de elementos, e para cada grupo foram gerados 10 problemas teste. Os grupos problemas teste analisados no trabalho possuem 10, 12, 15, 20 e 25.

5.1 Avaliação dos problemas teste

Para cada problema teste avaliado era conhecido o valor ótimo, o que permitiu o processo de verificação do desempenho do algoritmo proposto. Ao final do processamento de cada problema teste foram registrados o custo da melhor solução encontrada, o tempo de execução para encontrar tal solução, média dos valores das soluções encontradas e a média do tempo de execução. A métrica utilizada para quantificar o desempenho do resultado encontrado é o *gap* entre o resultado obtido e o resultado do melhor valor conhecido. Esse valor é dado em porcentagem e é calculado pela fórmula (2):

$$GAP(\%) = 100 * \frac{(resultado\ encontrado - resultado\ ótimo)}{resultado\ encontrado} \quad (2)$$

5.2 Experimentos

Dado o objetivo principal do trabalho, que é avaliar uma melhor combinação entre uma matheurística para geração de uma solução inicial, refiná-la e realizar a comparação com uma execução padrão do *solver* com tempo definido, é necessário estabelecer os seguintes grupos de experimentos:

1. **RENS**: execução utilizando apenas o método RENS para geração da solução inicial.
2. **FP**: execução utilizando apenas o método FP para geração da solução inicial.
3. **RENS+LB**: execução do método RENS refinando a solução gerada através do método *Local Branching* com tempo limite para resolução do MIP de 10s.
4. **FP+LB**: execução do método RENS refinando a solução gerada através do método *Local Branching* com tempo limite para resolução do MIP de 10s.
5. **Solver Padrão**: execução padrão do solver limitando o tempo em 200s.

Após realizar esses grupos de experimentos será possível aferir a melhor combinação, gerando assim um grupo final de experimentos para comparação de resultados com a execução padrão do *solver* limitada em 200s. Dado o fato de existirem vários problemas testes em um mesmo grupo de experimentos, será utilizada como métrica de desempenho de uma determinada implementação o *gap* médio para aquele conjunto de problemas teste.

5.3 Métodos de construção da solução inicial

Conforme descrito na seção 3, foram analisadas 2 estratégias de construção da solução inicial: RENS e FP, porém dos 50 problemas teste selecionados o método RENS não conseguiu gerar uma solução inicial para 15, falhando assim em 30% dos casos. Em contrapartida o FP conseguiu gerar solução inicial para todos os problemas testes selecionados, porém com um tempo superior. Dessa forma, para realizar os comparativos entre as implementações dos métodos, os problemas testes que não obtiveram solução inicial através do método RENS foram removidos dos grupos, reduzindo assim para 35 o número total de problemas teste submetidos à matheurística de refinamento LB.

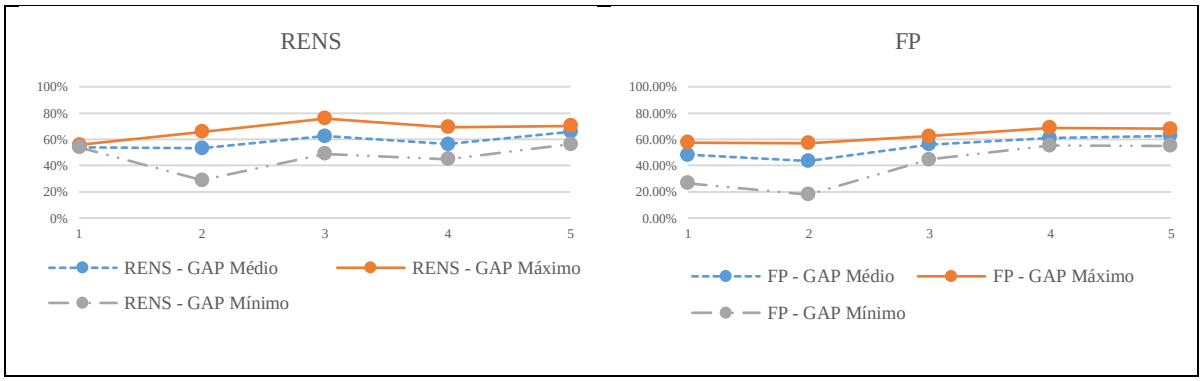

FIGURA 2 – Comparação entre os *gaps* dos resultados gerados pelos métodos construtivos RENS e FP. Fonte: elaborado pelos autores.

Conforme destacado pela Figura 2, os *gaps* médios dos métodos foram similares, porém o tempo gasto pelo FP é elevado em relação ao RENS.

5.4 Método de refinamento de solução

Os resultados apresentados nessa seção demonstram a relação dos *gaps* encontrados ao refinar as soluções inicias geradas pelos métodos RENS e FP executando a matheurística *Local Branching*.

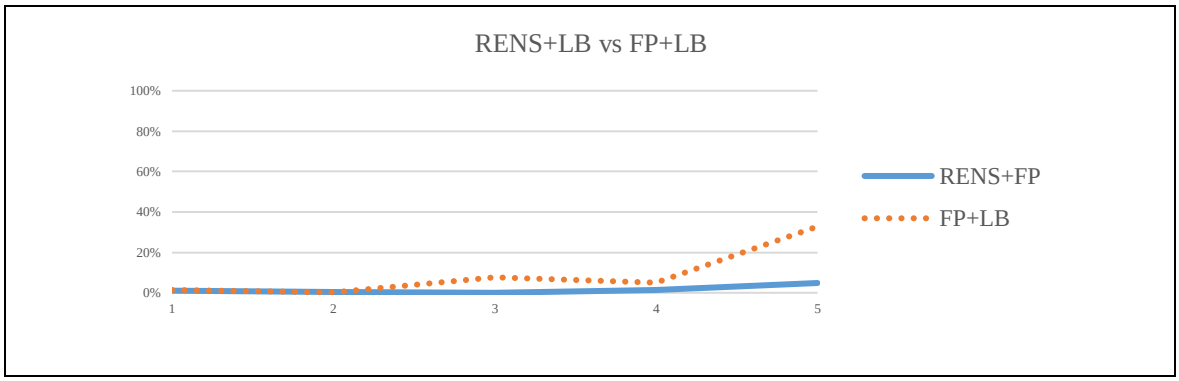

FIGURA 3 – Comparação entre os *gaps* dos resultados gerados pelo método de refinamento *Local Branching* aos resultados apresentados pela implementação do RENS e FP. Fonte: elaborado pelos autores.

TABELA 3 – Resultados comparativos entre o método RENS+LB e FP+LB.

Grupo	**RENS+FP**		**FP+LB**	
	GAP Médio (%)	**Temo Médio (s)**	**GAP Médio (%)**	**Temo Médio (s)**
1	0.90%	1.99	1.40%	3.90
2	0.34%	4.69	0.19%	13.80
3	0.00%	61.75	7.58%	193.00
4	1.48%	118.78	4.87%	333.47
5	4.84%	135.06	32.79%	440.34

5.5 Comparação de resultados

Conforme destacado na seção anterior a melhor combinação foi a do grupo de experimentos RENS+LB, sendo assim, esse grupo foi comparado com os resultados obtidos pelo *solver* padrão limitado em 200s para a resolução do MIP. A Tabela 3 destaca os resultados obtidos. Observa-se que apenas para o grupo 2 o *gap* médio foi superior, entretanto com um tempo computacional consideravelmente inferior.

TABELA 3 – Resultados comparativos entre o método RENS+LB e a execução do solver padrão limitado em 200s.

Grupo	**RENS+LB**		***Solver* Padrão**	
	GAP Médio (%)	**Temo Médio (s)**	**GAP Médio (%)**	**Tempo Médio (s)**
1	0.90%	1.99	0.90%	2.46
2	0.34%	4.69	0.19%	103.78
3	0.00%	61.75	0.10%	200.01
4	1.48%	118.78	3.12%	200.01
5	4.84%	135.06	5.65%	200.01

6. Conclusões e trabalhos futuros

A implementação da matheurística RENS apresentou resultados similares na construção de soluções iniciais em relação à implementação do FP com um tempo inferior, porém falhou em gerar resultados para 30% dos problemas testes analisados. Sugere-se utilizar a implementação FP quando houver falha no RENS. A combinação que apresentou melhores resultados para o conjunto de problemas teste foi RENS+LB.

Ao comparar a utilização da técnica proposta para o PML, observa-se resultados superiores que a execução padrão do *solver* em um tempo limitado em 200s.

Como trabalhos futuros, tanto as implementações do RENS quanto do FP devem ser avaliadas para se adequar melhor ao PML gerando soluções iniciais melhores. No caso do FP a estratégia *Local Search* pode ser substituída por outra disponível na literatura para tentar fugir de ciclagem. Mais testes devem ser executados a fim de explorar outros possíveis parâmetros das matheurísticas, como tempo para resolução do MIP e soluções exploradas. Diversos trabalhos publicados utilizam problemas testes da TSPLIB como critério de comparação, desta forma trabalhos futuros deverão analisar tais problemas testes.

Agradecimentos:

Os autores agradecem ao programa de pós-graduação em Modelagem Matemática Computacional, que permitiu o desenvolvimento deste trabalho.

Referências

AGARWAL, Anurag; COLAK, Selcuk; ERENGUC, Selcuk. Metaheuristic methods. In: Handbook on Project Management and Scheduling Vol. 1. Springer International Publishing, 2015. p. 57-74.

ARCHETTI, C., SPERANZA, M.G., 2014. A survey on matheuristics for routing problems. EURO Journal on Computational Optimization, 2(4), pp.223-246.

ARCHER, A., WILLIAMSON, D. P. Faster approximation algorithms for the minimum latency problem. Proceedings of the 40th Annual ACM- SIAM Symposium on Discrete algorithms. pp. 88–96, 2003.

BERTACCO, L., FISCHETTI, M. and LODI, A., 2007. A feasibility pump heuristic for general mixed-integer problems. Discrete Optimization, 4(1), pp.63-76.

BERTHOLD, T., 2007. RENS-relaxation enforced neighborhood search. Konrad-Zuse-Zentrum für Informationstechnik.

BIANCO, L, S.The traveling salesman problem with cumulative costs. Networks 23 (2), 81–91, 1993.

BLUM, C.; ROLI, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. Acm Computing Surveys, v. 35, n. 3, p. 268-308, Sep 2003.

BOSCHETTI, Marco A., et al. Matheuristics: Optimization, simulation and control. International Workshop on Hybrid Metaheuristics. Springer, Berlin, Heidelberg, 2009.

CHAUDHURI, Kamalika, et al. Paths, trees, and minimum latency tours. Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on. IEEE, 2003.

FISCHETTI, M.; LAPORTE, G.; MARTELLO, S. The delivery man problem and cumulative matroids. Operations Research, v. 41, n. 6, p. 1055-1064, 1993.

FISCHETTI, M., GLOVER, F. and LODI, A., 2005. The feasibility pump. Mathematical Programming, 104(1), pp.91-104.

GENDREAU, M.; POTVIN, J. Y. Metaheuristics in combinatorial optimization. Annals of Operations Research, v. 1., MINGOZZI, A., RICCIARDELLI 40, n. 1, p. 189-213, Nov 2005.

LISBOA, Fabrício da Silveira. GRASP para o Problema de Roteamento de Veículos com Multi-compartimentos e Restrição de Janela de Tempo, 2007. Disponível em <http://www.uenf.br/Uenf/Downloads/POS-ENGPRODUCAO_2397_1211921263.pdf>

LOOCK, M.; HINNEN, G. Heuristics in organizations: A review and a research agenda. Journal of Business Research, v. 68, n. 9, p. 2027-2036, Sep 2015.

NORONHA, T.F.; DA SILVA, M.M.; ALOISE, D.J. - Uma Abordagem sobre Estratégias Metaheurísticas. Revista Eletrônica de Iniciação Científica (REIC), Ano I, Vol I, No. I, 2001.

OSMAN, I. H.; LAPORTE, G. Metaheuristics: A bibliography. Annals of Operations Research, v. 63, p. 513-623, 1996.

SARUBBI, J. F. M. Problemas de roteamento com custos de carga. Tese (Doutorado em Ciência da Computação) - Programa de Pós-Graduação em Ciência da Computação, UFMG, Belo Horizonte. 2008. 168p.

TSITSIKLIS, J. N. Special cases of traveling salesman and repairman problems with time windows. Networks 22 (3), 263–282, 1992.

WU, B. Y.; HUANG, Z.-N.; ZHAN, F.-J. Exact algorithms for the minimum latency problem. Information Processing Letters, v. 92, n. 6, p. 303-309, 2004.