UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA CIVIL
MESTRADO EM ENGENHARIA CIVIL

JOÃO HENRIQUE BRUNOW BARBOSA

DISSERTAÇÃO DE MESTRADO

META-HEURÍSTICA PARA PLANEJAMENTO DA LOGÍSTICA REVERSA DE PNEUS INSERVÍVEIS PARA ATENDER UMA PLANTA GERADORA DE ENERGIA COM BASE NO MODELO TWO-ECHELON CAPACITATED VEHICLE ROUTING PROBLEM

VITÓRIA
2021

JOÃO HENRIQUE BRUNOW BARBOSA

DISSERTAÇÃO DE MESTRADO

META-HEURÍSTICA PARA PLANEJAMENTO DA LOGÍSTICA REVERSA DE PNEUS INSERVÍVEIS PARA ATENDER UMA PLANTA GERADORA DE ENERGIA COM BASE NO MODELO TWO-ECHELON CAPACITATED VEHICLE ROUTING PROBLEM

Dissertação apresentada ao Curso de Mestrado em Engenharia Civil do Programa de Pós-Graduação em Engenharia Civil da Universidade Federal do Espírito Santo.

Orientador: Prof. Dr. Rodrigo de Alvarenga Rosa.

VITÓRIA

2021

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

META-HEURÍSTICA PARA PLANEJAMENTO DA LOGÍSTICA REVERSA DE PNEUS INSERVÍVEIS PARA ATENDER UMA PLANTA GERADORA DE ENERGIA COM BASE NO MODELO TWO-ECHELON CAPACITATED VEHICLE ROUTING PROBLEM

João Henrique Brunow Barbosa

Dissertação apresentada ao Curso de Mestrado em Engenharia Civil do Programa de Pós-Graduação em Engenharia Civil da Universidade Federal do Espírito, como requisito parcial para obtenção do título de Mestre em Engenharia Civil, área de Construção Civil.

Aprovada no dia **28 de setembro** por:

_____

Prof. Dr. Rodrigo de Alvarenga Rosa
Doutor em Engenharia Elétrica
Orientador – UFES

_____

Prof. Dr. Macksuel Soares de Azevedo
Doutor em Engenharia Civil
Examinador Interno – UFES

_____

Prof. Dr. Renato Elias Nunes de Moraes
Doutor em Computação
Examinador Externo - UFES

Vitória – ES, setembro de 2021

Dedico este trabalho a meus pais Augusto e Patricia que me deram todo suporte para eu chegar até aqui.

AGRADECIMENTOS

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) – Código de Financiamento 001

Agradeço primeiramente a Deus pelas oportunidades que me foram dadas e pelas pessoas que auxiliaram nessa jornada.

Ao meu orientador Rodrigo de Alvarenga Rosa por me acompanhar desde a iniciação científica, passando pelo projeto de graduação e o mestrado, por me permitir fazer parte de seu laboratório, pelos ensinamentos, paciência e suporte dado ao longo de todo o desenvolvimento do trabalho.

Aos professores Macksuel Soares de Azevedo e Renato Elias Nunes de Moraes por aceitarem participar deste trabalho, pelas sugestões e contribuições.

Aos meus pais Augusto e Patrícia pela ajuda durante o desenvolvimento do trabalho, pela compreensão e por me apoiarem durante todos os momentos.

E por fim, aos companheiros do LAMMEP, que me acompanharem nessa jornada de aprendizado, com quais troquei muitas experiências e aprendizados

RESUMO

Anualmente são descartados aproximadamente 800 milhões de pneus no mundo. Esse resíduo causa grande impacto ambiental quando descartado inadequadamente, tendo em vista que eles possuem tempo de decomposição ainda desconhecido. Em centros urbanos uma das estratégias de coleta é realizada em duas etapas. Na primeira etapa pontos de coleta como lojas de pneus e oficinas recebem os pneus inservíveis diretamente dos usuários onde veículos fazem a coleta e os transportam para armazéns intermediários. Na segunda etapa veículos coletam a carga locada nos armazéns intermediários e entregam a carga na empresa recicladora para que seja feito o descarte de forma adequada. O problema na literatura que mais se assemelha com o problema descrito anteriormente é o Two-echelon Capacitated Vehicle Routing Problem (2E-CVRP), é um problema de roteamento de veículos em dois níveis. No primeiro nível, o transporte é realizado por veículos de grande porte e partem de um depósito central com destino aos centros de distribuição e consolidação de cargas. No segundo nível, o transporte é realizado dos centros de distribuição aos clientes finais por meio de veículos de menor porte. Tendo em vista suas similaridades, pode-se considerar o problema da logística reversa de pneus como um 2E-CVRP. Esta dissertação propõe um algoritmo inspirado em *Simulated Annealing* para o problema. O desenvolvimento do algoritmo se mostrou necessário, uma vez que o modelo matemático proposto anteriormente para este problema não se mostrou capaz de encontrar soluções num tempo de execução razoável para uma aplicação prática. O algoritmo foi desenvolvido utilizando linguagem C, utilizando a ferramenta de desenvolvimento Dev-C++.

Palavra-chave: Logística Reversa de Pneus, 2E-CVRP, Meta-heurística, *Simulated Annealing*

ABSTRACT

Approximately 800 million tires are discarded annually in the world. This residue causes a great environmental impact when improperly disposed of, considering that they have an unknown decomposition time. In urban centers, one of the collection strategies is carried out in two stages. In the first stage, collection points such as tire stores and workshops receive waste tires directly from users, where vehicles collect and transport them to intermediate warehouses. In the second stage, vehicles collect the cargo leased from the intermediary warehouses and deliver the cargo to the recycling company so that it can be properly disposed of. The problem in the literature that most resembles the problem described above is the Two-echelon Capacitated Vehicle Routing Problem (2E-CVRP), which is a two-level vehicle routing problem. On the first level, transport is carried out by large vehicles and departs from a central warehouse to cargo distribution and consolidation centers. On the second level, transport is carried out from distribution centers to end customers using smaller vehicles. In view of their similarities, one can consider the problem of reverse tire logistics as a 2E-CVRP. This thesis proposes an algorithm inspired by Simulated Annealing for the problem. The development of the algorithm proved to be necessary, since the mathematical model proposed above for this problem was not capable of finding solutions in a reasonable execution time for a practical application. The algorithm was developed using C language, using the Dev-C++ development tool.

Keyword: Tire Reverse Logistics, 2E-CVRP, Meta-heuristics, Simulated Annealing

LISTA DE SIGLAS

2E-VRP – Two-echelon Vehicle Routing Problem

2E-CVRP – Two-echelon Capacitated Vehicle Routing Problem

2E-CVRPD – Two-echelon Capacitated Vehicle Routing Problem with Drones

2E-CVRP-HFSDTW – Two-echelon Capacitated Vehicle Routing Problem with Heterogeneous Fleet, Site Dependence with Time Windows

2E-CVRP-TW – Two-Echelon Capacitated Vehicle Routing Problem with Time Windows

2E-MTVRP-SS – Two-Echelon Multiple-trip Vehicle Routing Problem with Sattelite Synchronization

2E-CVRP-SD – Two-Echelon Capacitated Vehicle Routing Problem with Stochastic Demands

A2E-CVRP - Adaptive Two-Echelon Capacitated Vehicle Routing Problem

ALNS – Adaptive Large Neighborhood Search

BP – Branch-and-price

CCP – Centros de Consolidação de Pneus

CVRP – Capacitated Vehicle Routing Problem

DTRC – Drone Truck Route Construction

LNS – Large Neighborhood Search

GRASP – Greedy Randomized Adaptive Search Procedure

KPI - Indicadores Chave de Perfomance

GA – Algoritmo Genético

SAW – Simple Addictive Weighting

MDCVRP – Multidepot Capacitated Vehicle Routing Problem

PLIM – Programação Linear Inteira Mista

SA – Simulated Annealing

VND – Variable Neighborhood Descent

VRP – Vehicle Routing Problem

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

O descarte inadequado de resíduos é um problema ambiental que, apesar de diversas propostas para solucionar o problema já haverem sido feitas, ainda não foi resolvido. Um dos principais resíduos sólidos gerados por automóveis durante sua vida útil são os pneus. Há décadas se estuda uma forma de adequadamente se desfazer de pneus inservíveis, porém esse material ainda á colocado em aterros, lixões a céu aberto ou queimados.

Como o progressivo aumento da frota automotiva mundial, o número de pneus inservíveis que são necessários serem descartados também aumentou. Ao contrário de outros resíduos sólidos, o pneu possui um tempo de decomposição ainda desconhecido e causam impactos ambientais devido ao seu grande número e destinação incorreta. Dentre estes impactos citam-se: o assoreamento de rios, facilitando o risco de enchentes, e a poluição atmosférica devido à liberação, após sua queima, de gases como o monóxido de carbono, óxidos de nitrogênio e enxofre, metais pesados, dioxinas e furanos (VELOSO, 2016). Pneus inservíveis descartados a céu aberto podem ainda vir a ser tornar criadouro de insetos transmissores de doenças, especialmente a dengue e chikungunya.

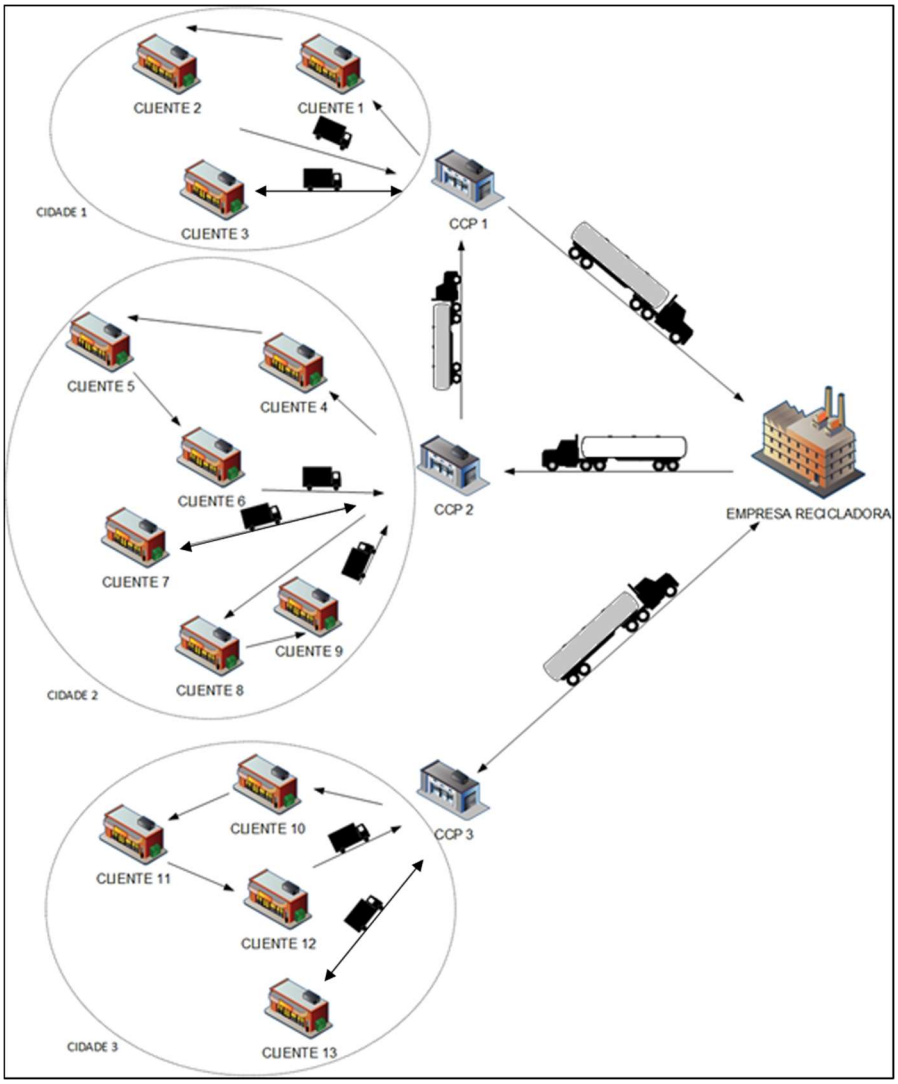
A partir dos anos 1990, começaram-se as tentativas de minimizar os impactos ambientais causados por pneus inservíveis no Brasil. O Conselho Nacional do Meio Ambiente (CONAMA) atribuiu a responsabilidade da destinação final dos pneus inservíveis aos fabricantes e importadores de pneus novos (BRASIL, 2010; CONAMA, 1999, 2002, 2009). Segundo a Resolução nº 416/2009 o prazo máximo de armazenamento dos pneus inservíveis antes do descarte é de 12 meses, destacando que os armazéns devem garantir as condições necessárias para que os pneus não se tornem uma ameaça à saúde pública ou apresente riscos ambientais.

Um dos métodos utilizados para o correto descarte dos pneus inservíveis é a sua queima em usinas devidamente preparadas para esse fim. Nessas usinas, os pneus são utilizados como matéria prima para a produção de energia elétrica. Para que haja o fornecimento contínuo de pneus para as usinas de geração de

energia, é necessária uma malha logística que faça a coleta, estocagem e entrega dos pneus inservíveis nas usinas. Podemos considerar que se trata da logística reversa de pneus, sendo que lojas de pneus ou centros de coleta recebem os pneus inservíveis, caminhões coletam esses pneus e os concentram em armazéns intermediários denominados Centros de Consolidação de Pneus (CCP), que por sua vez tem seus pneus coletados por caminhões que tem por destino as usinas de geração de energia. Tal abordagem de coleta em dois níveis com armazéns intermediários é denominada na literatura como *Two Echelon Vehicle Routing Problem (2E-CVRP).*

O surgimento de novas leis e normas ambientais vem mostrando a importância das questões ambientais. Assim, o interesse no campo da logística envolvendo fatores ambientais tem crescido nos meios empresariais e acadêmicos (JABALI et al., 2012). Esta dissertação de mestrado aborda o problema da logística reversa estruturada em Centros de Consolidação de Pneus (CCP). Os CCP são centros de armazenamento temporários, responsáveis por consolidar cargas recebidas de veículos de pequeno porte, que coletam os pneus nos centros urbanos, para serem coletadas por veículos de maior porte. Como esquematizado na Figura 1.

Figura 1 - Centro de consolidação de pneus em um CCP



Fonte: Adaptado de Vitorugo e Caliman (2017)

São características comumente encontradas em centros urbanos: (1) restrições dos tipos de veículos permitidos devido a características geométricas das vias; (2) restrições quanto ao horário permitido para recebimento de veículos de carga e restrição quanto ao horário em que os veículos são liberados para trafegar. O problema 2E-CVRP consiste em, dado cargas de múltiplos clientes, pontos de distribuição ou consolidação intermediários, pontos onde se originam ou se destinam todas as cargas e uma frota de veículos, deseja se encontrar rotas para o transporte das cargas, tal que o custo total seja minimizado, considerando as capacidades de carga dos veículos e dois níveis de operação distintos. Apesar de serem parte essencial da realidade dos centros urbanos,

considerações de frota heterogênea e as restrições de tempo e acesso não são contempladas no *Two-echelon Capacitated Vehicle Routing Problem* (2E-CVRP).

Devido a sua relevância e aplicabilidade na vida real, o Two-echelon Capacitated Vehicle Routing Problem (2E-CVRP) vem atraindo cada vez mais atenção na literatura. Uma de suas diversas aplicações é a logística reversa de cargas utilizando de centros de consolidação. Apesar do crescente interesse do tema, o 2E-CVRP ainda é um problema relativamente pouco explorado, principalmente quando se considera o fator tempo. Poucos autores estudaram variantes do 2E-CVRP com janela de tempo ou com limite de duração em cada rota (CUDA; GUASTAROBA; SPERANZA, 2015).

Neste contexto, esta dissertação de mestrado propõe um algoritmo baseado em *Simulated Annealing* (SA) para a variação do 2E-CVRP denominada Two-echelon Capacitated Vehicle Routing Problem with Heterogeneous Fleet, Site Dependence with Time Windows (2E-CVRP-HFSDTW), para o planejamento da logística reversa de pneus inservíveis estruturada em centros de consolidação de pneus (CCP), proposto por Vitorugo e Caliman (2017) que desenvolveram um modelo matemático para o mesmo.

O algoritmo proposto tem como objetivo elaborar o planejamento da logística reversa de pneus que tenha o menor custo, reduzindo o número de caminhões utilizados e as distancias percorridas no primeiro nível e no segundo nível. Para validar o algoritmo, os resultados serão comparados com os resultados obtidos pelo modelo matemático proposto por Vitorugo e Caliman (2017).

1.1. OBJETIVOS

1.1.1. OBJETIVO GERAL

Desenvolver um algoritmo para resolver o problema de planejamento da logística reversa de pneus inservíveis estruturada em vários centros de consolidação de pneus e um armazém central, que foi estruturado como o modelo matemático 2E-CVRP-HFSDTW por Vitorugo e Caliman (2017).

1.1.2. OBJETIVO ESPECIFICO

- Analisar e adaptar as instâncias propostas por Vitorugo e Caliman (2017).
- Comparar os resultados obtidos pelo algoritmo proposto com os resultados obtidos por Vitorugo e Caliman (2017) para analisar a eficiência (tempo de execução) e eficácia (qualidade dos resultados).

1.2. JUSTIFICATIVA

Não se pode negar o impacto gerado pelo descarte irregular de pneus ao ser humano e ao meio ambiente. É estimado que 800 milhões de pneus são descartados no mundo anualmente. No Brasil são produzidos mais de 40 milhões de pneus por ano, sendo que praticamente metade desse número é descartado de forma não sustentável no meio ambiente (FRAGMAQ, 2012).

Devido a crescente importância que as questões ambientais vêm adquirindo, legislações ambientais se tornaram mais pesadas e vem cada vez mais responsabilizando os produtores pela destinação final dos produtos após o consumo. Sendo assim, a logística reversa vem ganhando cada vez mais importância no meio empresarial e acadêmico.

Tal cenário tornou necessário propor melhorias no planejamento do fluxo reverso de pneus inservíveis, auxiliando fabricantes e importadores a dar destinação correta a este produto, além de estimular soluções para seu reaproveitamento. Exemplo disso é a indústria que irá se instalar em Cariacica no Espírito Santo com o objetivo de gerar energia elétrica e artefatos de borracha utilizando pneus inservíveis.

Tendo em vista que a maioria das pesquisas sobre os problemas de roteamento de veículos em dois níveis concentram-se em sua versão básica, o 2E-CVRP (CUDA; GUASTAROBA; SPERANZA, 2015), o estudo de suas variantes apresenta um potencial de publicação ainda pouco explorado, principalmente ao problema de distribuição em dois níveis com frota heterogênea, restrições de acesso e janela de tempo.

Do mesmo modo, segundo Crainic et.al. (2009) o conceito de otimização no contexto de logística urbana ainda não é muito desenvolvido, sendo que existem muito poucos modelos formais e métodos dedicados ao seu design, avaliação, planejamento, gerenciamento e controle.

Apesar de imprescindíveis para a formalização de problemas e para benchmarks, modelos matemáticos apresentam limitações, principalmente quanto a tempo de processamento e no tamanho das instâncias que os tornam difíceis de serem aplicados na prática. Portanto, a elaboração de um algoritmo eficiente, que realize o planejamento da roteirização dos veículos, visando situações comumente encontradas na logística urbana, como restrições de acesso e horários de atendimentos dos clientes e ainda seja capaz de lidar com instâncias maiores em tempos de processamento razoáveis poderá se provar uma importante ferramenta no planejamento logístico urbano.

2. REFERENCIAL TEÓRICO

O roteamento de veículos é um problema considerado NP-hard que apresenta profunda ligação com a área logística, visto que a minimização dos custos é uma das principais preocupações da área (COELHO, 2017). Na logística urbana o problema de roteamento de veículos é comumente tratado como sendo de múltiplos níveis, uma vez que esta abordagem melhor se adequa as características encontradas nos centros urbanos. Os problemas de múltiplos níveis mais comumente encontrados na literatura são os de dois níveis, comumente denominados *two-echelon vehicle routing problems* (2E-VRP). No 2E-VRP, o primeiro nível trata do frete realizado por veículos de grande porte que partem de um depósito central com destino a instalações satélites, que são os centros de distribuição e consolidação de cargas. Já o segundo nível veículos de menor porte realizam o frete partindo dos satélites com destino aos clientes finais (HEMMELMAYR; CORDEAU; CRAINIC, 2012; MANCINI 2013).

Essa abordagem é muito aplicada à logística urbana porque além de determinar rotas ótimas no atendimento de demandas de clientes, reduzindo os custos globais de transporte, mantem de forma eficiente os veículos de grande porte fora dos centros urbanos (Crainic et al., 2008). Com isso é possível aplicar algum grau de segregação ao transporte de cargas dentro dos centros urbanos, diminuindo transtornos no trânsito.

2.1. REVISÃO BIBLIOGRAFICA

Crainic *et al*. (2008) abordaram o problema 2E-CVRP e compararam os resultados de diferentes metodologias. O problema foi dividindo em duas partes. A primeira parte foi tratar o primeiro nível como um VRP e encontrar sua solução. A segunda parte foi solucionar o segundo nível considerando-o como um *multi depot* VRP. As meta-heurísticas utilizadas foram *Split-large-route, Add e Exchange.* Os testes utilizaram de dois conjuntos de seis instâncias contendo 21 e 32 clientes. Segundo os autores, métodos exatos conseguem resolver instâncias pequenas, porém à medida que o tamanho das instâncias aumenta o uso de heurísticas se torna necessário.

Crainic *et al.* (2010) propuseram uma família de heurísticas *Multi-Start* e separou o 2E-CVRP em sub-problema de primeiro nível e sub-problema de segundo

nível, sendo que cada problema foi resolvido separadamente se maneira sequencial num processo iterativo utilizando heurísticas *clustering* e abordagem de busca local. As heurísticas propostas foram comparadas com duas *math-heutistics* apresentadas por Perboli, Tadei e Vigo (2018). Os autores concluíram que as *Multi-Start heuristics* se mostraram superiores as *math-heurístics* tanto em eficiência quanto em eficácia.

Perboli, Tadei e Vigo (2010) propuseram novas classes de inequações na formulação do 2E-CVRP. Os autores utilizaram o método *Branch & Cut* na resolução do problema. Foram utilizadas o mesmo conjunto de instâncias de Perboli, Tadei e Vigo (2008) e Mancini, Perboli e Tadei (2008) para que fosse possível comparar os resultados obtidos com a literatura. O artigo concluiu que a abordagem utilizando o método *Branch & Cut* apresentou maior precisão e maior número de soluções ótimas quando comparado ao modelo proposto por Perboli, Tadei e Vigo (2008), além de ser capaz de otimizar todas as instâncias.

Perboli, Tadei e Vigo (2011) utilizaram um modelo PLIM e desigualdades válidas para solucionar o 2E-CVRP. As instâncias de teste foram baseadas no trabalho de Christofides e Eilon (1969) e Crainic et al. (2010). Os resultados mostraram um bom desempenho em instâncias de pequeno e médio porte. Os autores também se basearam no modelo PLIM para apresentar uma heurística *Clustering-Based* e *Math-Based,* sendo que ambas apresentaram bom desempenho computacional e soluções de boa qualidade.

Crainic *et al.* (2012) adotaram a meta-heurística GRASP utilizando um procedimento de busca local para pós otimização, combinado a um procedimento *Path Relinking*. O artigo separou o problema em transferência entre depósito para satélite e entrega de satélite para cliente, resolvendo iterativamente os dos subproblemas. Os cenários de teste utilizados foram baseados em Perboli e Tadei (2010) e Perboli, Tadei e Vigo (2011) para que seus resultados pudessem ser comparados. O artigo concluiu que apesar de a meta-heurística proposta apresentar melhor eficiência na obtenção de resultados que Perboli, Tadei e Vigo (2011), o desempenho computacional se mostrou inferior à de Perboli e Tadei (2010) e Perboli, Tadei e Vigo (2011).

Meihua *et al.* (2011) propôs um *Ant Colony Optimization* que combina três

heurísticas, *Ant Colony Optimization, Multiple Neighborhood Descent* e *Threshold-Based Local Search.* O problema foi resolvido em três partes. A primeira parte consiste em dividir o problema em subproblemas de roteamento CVRP. Na segunda parte foi aplicada a meta-heurística *Ant Colony Optimization* melhorada pela heurística *Multiple Neighborhood Descent.* Na terceira parte foi aplicada a meta-heurística *Threshold-Based Local Search* para melhorar a solução encontrada na segunda parte. O autor comparou os resultados encontrados com os trabalhos de Feliu *et al.* (2007) e Crainic *et al.* (2008). O algoritmo proposto obteve 12 soluções melhores e 4 soluções idênticas no total de 21 instâncias se comparado a Feliu *et al.* (2007) e 6 soluções melhores e uma solução igual a Crainic *et al.* (2008) em um total de 12 instâncias. Foi concluído que a meta-heurística proposta melhorou a qualidade da solução e acelerou a convergência do algoritmo.

Hemmelmayr, Cordeau e Crainic (2012) consideraram o contexto da logística urbana e propuseram uma heurística ALNS para o problema 2E-CVRP e o *Location Routing Problem*. Eles utilizaram instâncias de Perboli e Tadei (2010), Perboli, Tadei e Vigo (2011) e Crainic et al. (2010) e seus resultados mostraram que dentre as 93 instâncias testadas, 59 apresentaram soluções melhores.

Jepsen, Spoorendonk e Ropke (2013) estudou o *Symmetric Two-Echelon Capacitated Vehicle Routing Problem.* Essa variação do 2E-CVRP considera que todos os satélites se localizam a mesma distância entre si e do depósito central. Os autores propuseram um algoritmo *branch- and-cut* baseado em um modelo matemático não trivial. Os grupos de teste com 21, 32 e 50 clientes foram introduzidas por Feliu et al. (2007) e por Crainic et al. (2010). Os resultados superaram os de Perboli, Tadei, and Vigo (2011) e Perboli, Tadei, and Masoero (2010), resolvendo 47 das 93 instâncias, dentre as quais 34 foram resolvidas pela primeira vez.

Baldacci, Mingozzi e Roberti (2013) propuseram um algoritmo que consiste de um modelo matemático com base em programação dinâmica, um método *dual-ascent* e um algoritmo exato que decompõe o 2E-CVRP em um conjunto limitado de *Multi depot Capacitated Vehicle Routing Problem (MDCVRP)* com restrições laterais. O algoritmo foi testado em 207 instâncias, sendo 153 da

literatura e 54 criadas pelos autores. Comparações com algoritmos introduzidos por Perboli, Tadei e Vigo (2011) e Jepsen, Spoorendonk e Ropke (2013) mostraram que o novo método supera os demais em termos de tamanho, número de instâncias resolvidas e tempo de execução do modelo.

Santos, Da Cunha e Mateus (2013) utilizaram um modelo de programação linear inteira e duas implementações da heurística *branch-and-price* para a resolução do 2E-CVRP. A primeira heurística satisfaz a condição que cada cliente é visitado apenas uma vez e foi chamada de BP-E. A segunda heurística não satisfaz essa condição e foi chamada de BP-NE. Foram utilizadas 114 instâncias da literatura para um único deposito e até 51 clientes. O BP-E se mostrou capaz de encontrar melhores soluções inteiras enquanto o BP-NE apresentou menores GAPs quando um tempo limite era estipulado. Os resultados encontrados também mostraram a heurística BP-NE apresentou menores tempos de execução que a heurística BP-E e a heurística *Branch-and-cut* proposta por Perboli, Tadei e Vigo(2011).

Mancini (2013) revisou os problemas de roteamento de veículo de múltiplos níveis no contexto da logística urbana, inclusive o 2E-CVRP. O autor também apresenta os métodos de otimização utilizados e analisa a adaptabilidade das classes mais comuns de meta-heurísicas aos problemas apresentados.

Zheng et al. (2014) abordaram o 2E-CVRP no contexto da logística urbana, utilizando uma heurística hibrida composta por um GRASP com procedimento de *Route-First Cluster-Second* e um *Variable Neighborhood Descent* (VND). O algoritmo cria um número de soluções iniciais e as otimiza, retornando ao final a melhor solução encontrada. Foram utilizados três conjuntos de instâncias publicados por Baldacci, Mingozzi e Roberti (2013) e comparados com as soluções obtidas pelo algoritmo ALNS proposto por Hemmelmayr, Cordeau e Crainic (2012). Os resultados mostraram que o algoritmo se mostrou eficiente e eficaz, pois o mesmo ora se igualou ora superou o ALNS, tanto em qualidade de soluções quanto em tempo de execução. Os autores observaram também que a aplicação da heurística hibrida é mais simples que outras heurísticas e concluem que ela é a mais adequada para se lidar com o 2E-CVRP no contexto da logística urbana.

Soysal, Bloemhof-Ruwaard, Bektas (2015) estudaram o caso de uma cadeia de suprimentos de um supermercado nos países baixos, apresentando uma formulação de programação linear inteira mista para o 2E-CVRP. Foram criadas diferentes versões do modelo, com diferentes funções objetivos de produzir um conjunto relevante de indicadores de performance (KPI) em relação à distância, tempo, consumo de combustível e custo. Os autores concluíram que em relação ao caso de estudo, a solução mais ambientalmente correta foi obtida através do uso do *two-echelon distribution system,* embora um *single-echelon distribution system* forneça uma solução com menor custo.

Cuda, Guastaroba e Speranza (2015) estudaram o *Two-echelon Routing Problems*, classificando em *two-echelon location routing problems,* o *two-echelon capacitated vehicle routing problems* e o *truck and trailer routing problems*, dando uma descrição geral do problema, identificando as principais variantes e revisando as principais soluções exatas e heurísticas propostas.

Dellaert et al. (2016) propuseram dois modelos matemáticos *path-based* para o 2E-CVRP-TW, sendo um algoritmo *branch-and-price* foi desenvolvido para cada modelo. No primeiro as rotas foram definidas como sendo de primeiro e de segundo nível. No segundo, as rotas de primeiro e segundo nível foram decompostas. Os autores utilizaram um conjunto de instâncias de teste com até 5 satélites e 100 clientes, sendo o primeiro artigo na literatura a resolver instâncias deste tamanho.

Grangier et al (2016) utilizaram a heurística ALNS para resolver uma variante do 2E-CVRP, o *two-echelon multiple-trip vehicle routing problem with satellite synchronization* (2E- MTVRP-SS). Os autores utilizaram instâncias propostas por Solomon (1987) adaptadas para o 2E-MTVRP-SS. Resultados mostraram boas soluções com tempo de execução razoáveis. Os autores concluíram que as janelas de tempo apresentavam maior influência no custo total do que a sincronização de satélites.

Wang, Lan e Zhao (2017) abordaram uma variante do 2E-CVRP, o *Two-Echelon Capacitated Vehicle Routing Problem With Stochastic Demands* (2E-CVRP-SD), aplicado a logística urbana. O problema abordado permite entregas fracionadas somente no primeiro nível. Foi proposto um modelo matemático

Route-based para descrever o problema estocástico e um algoritmo genético (GA) para a solução do problema. As instâncias utilizadas para teste foram baseadas nas instâncias propostas por Breunig et al (2015). Devido à falta de pesquisas sobre o 2E-CVRP-SD, os resultados obtidos foram comparados com soluções conhecidas do 2E-CVRP na literatura, permitindo analisar os benefícios de se considerar demandas estocásticas. Resultados encontrados validaram a eficiência e eficácia do algoritmo proposto.

Esmaili e Sahraeian (2017) estudaram pela primeira vez o 2E-CVRP levando em consideração a satisfação de clientes em entregas de produtos perecíveis e fatores ambientais. Os autores propuseram um modelo bi-objetivo, que minimiza o tempo de espera dos clientes e o custo total das viagens. Foi considerado também uma restrição na emissão máxima de dióxido de carbono em cada rota. O modelo é resolvido utilizando o método *Simple Addictive Weighting* (SAW). O modelo foi aplicado a uma rede de supermercados nos Países Baixos, que possuíam um depósito, dois satélites e 16 clientes. Resultados mostraram que o modelo é adequado para instâncias com até dois satélites e a análise de sensibilidade indicou que politicas menos restritivas quanto as emissões de carbono levaram a uma maior emissão de CO2, porém, levaram a um menor custo total e menor espera dos clientes. Os autores também alegaram que o uso de janelas de tempo poderia levar o modelo a resultados ainda mais realistas.

Song, Gu e Huang (2017) resolveram o *adaptive two-echelon capacitated vehicle routing problem* (A2E-CVRP), uma variante do 2E-CVRP, que considera múltiplos depósitos e permite que o depósito atenda diretamente o cliente. Os autores introduziram uma formulação matemática para o A2E-CVRP onde derivaram um *lower bound* baseado no método proposto por Baldacci et al (2013), e depois utilizaram para derivar um *upper bound* que também é a solução aproximada do A2E-CVRP. Os resultados computacionais mostraram que o A2E- CVRP tem melhores resultados que 2E-CVRP quanto ao custo total de rotas.

JIE, Wanchen et al (2019) abordou o problema *two-echelon capacitated electric vehicle routing problem with battery swapping stations* (2E-EVRP-BSS),

considerando que o veículo de cada nível tem uma capacidade de carga, duração de bateria, taxa de consumo de bateria e custo de troca de bateria. Por eles foi proposto uma formulação de programação inteira e um algoritmo que combina *columm generation* e um *adaptative large neighborhood search*.

Redi et al. (2020) propuseram um modelo de otimização utilizando programação linear inteira e um algoritmo simulated annealing para o problema two-echelon vehicle routing problem with locker facilities (2EVRP-LF). Este problema considera a utilização de sistemas de armários como armazéns intermediários em áreas metropolitanas, onde veículos fazem o transporte de mercadorias até armários, onde os clientes então buscam suas mercadorias. Segundo Redi et al.(2020), o objetivo é minimizar o custo de transporte das mercadorias, considerando os custos de viajem dos veículos, o aluguel dos armários e o custo adicional necessário para compensar o deslocamento do cliente. Os autores afirmam a efetividade do 2EVRP-LF quando comparado com uma otimização em duas etapas e a efetividade e performance de seu algoritmo simulated annealing ao resolver o 2EVRP-LF.

Kitjacharoenchai et al.(2020) abordou o problema Two echelon vehicle routing problem with drones in last mile delivery considerando uma operação sincronizada entre drones e vans. Os autores resolveram o problema utilizando programação inteira mista e duas heurísticas, uma Drone Truck Route Construction (DTRC) e uma Large Neighborhood Search (LNS). Segundo os autores, a LNS gera resultados com melhor qualidade que o DTRC quando testado utilizando vários cenários de benchmark do problema CVRP e sua análise de sensibilidade mostrou que o modelo proposto tem tempos de entrega melhores que os modelos truck-drone routing anteriormente propostos.

Mühlbauer et al. (2021) propôs uma heurística Large Neighbourhood Search paralelizada que é então aperfeiçoada utilizando uma heurística de primeiro nível para o Problema Two-Echelon Vehicle Routing Problem With Swap Containers For Cargo-Bicycles. Segundo os autores, foi considerado o uso de cross-docking entre vans e bicicletas nos chamados satélites e matrizes de distâncias assimétricas baseadas em dados reais para as bicicletas. Os resultados permitem um estudo dos custos e das diminuições de emissões de

CO_2 comparando os cenários propostos utilizando bicicletas com as entregas convencionais utilizando vans.

A Tabela 1 apresenta o resumo da revisão bibliográfica, apresentando o nome dos autores, a variação do problema estudado e o método utilizado no artigo.

Tabela 1 - Resumo da revisão bibliográfica

Autor	Variação	Método			
		Mod. Mat	Heurística	Meta-Heurística	Revisão
Cranic et al. (2008)	2E-VRP			x	
Cranic et al. (2010)	2E-VRP		x		
Perboli, Tadei e Tadei (2010)	2E-VRP	x			
Meihua et al. (2011)	2E-VRP			x	
Perboli,Tadei e Vigo (2011)	2E-CVRP	x		x	
Crainic et al. (2012)	2E-VRP			x	
Hemmelmayr, Cordeau e Crainic (2012)	2E-VRP			x	
Jepsen, Spoorendonk e Ropke (2013)	2E-CVRP		x		
Baidacci, Mingozzi e Roberti (2013)	2E-CVRP	x	x		
Mancini (2013)	2E-VRP				x
Santos, Da Cunha e Mateus (2013)	2E-CVRP	x			
Zheng et al. (2014)	2E-VRP			x	
Cuda, Guastaroba e Speranza (2015)	2E-VRP				x
JIE, Wanchen et al. (2019)	2E-EVRP-BSS	x	x		

Redi et al. (2020)	2E-VRP-LF	x		x
Kitjacharoenchai et al.(2020)	2E-VRPD	x	x	
Mühlbauer et al. (2021)	2E-CVRP		x	
Este Trabalho				

Fonte: Do Autor

2.2. MODELO MATEMÁTICO PROPOSTO POR VITORUGO E CALIMAN (2017)

O modelo matemático proposto por Vitorugo e Caliman (2017), denominado como *Two-echelon Capacitated Vehicle Routing Problem with Heterogeneous Fleet, Site Dependence with Time Windows* (2E-CVRP-HFSDTW), é um problema de roteamento de veículos em dois níveis que realiza a coleta de mercadorias de um número conhecido de clientes à um armazém, com o auxílio de um número fixo de depósitos intermediários, chamados satélites. O 1º nível de um sistema de coleta em dois níveis é composto pelo depósito e pelos satélites e o 2º nível é composto pelos satélites e pelos clientes. O depósito e os satélites possuem uma frota heterogênea de veículos, chamados de veículos de 1º nível e de 2º nível, respectivamente. A localização dos satélites é conhecida e diferentes satélites podem ter diferentes capacidades, conseguindo-se mensurar a capacidade de um satélite por meio da capacidade total de sua frota.

As rotas do 2º nível iniciam-se com veículos do 2º nível partindo dos satélites em direção aos clientes, a fim de coletar produtos de pós-consumo. Cada cliente deve ser visitado somente uma vez. Após todos os clientes serem visitados, os veículos do 2º nível retornam ao satélite de origem. A coleta de pneus nos clientes ocorre durante o dia, considerando uma jornada de trabalho do motorista de 8 horas, de 8 às 16 horas. Já as rotas do 1º nível iniciam-se com veículos do 1º nível partindo do depósito em direção aos satélites para a coleta dos produtos recolhidos no 2º nível. O recolhimento nos satélites para usina geradora de energia (depósito) ocorre durante a noite, no período de 18 às 2 horas da manhã, considerando uma jornada de trabalho de 8 horas

do motorista. Assim, o tempo de ciclo diário da operação é de 16 horas.

A operação nos satélites é realizada da seguinte forma: os veículos do 2º nível são descarregados possibilitando o carregamento dos veículos do 1º nível. Essa operação é realizada com um custo proporcional à quantidade de carga manuseada, uma vez que está sendo considerado a coleta de apenas um tipo de produto. Por fim, após os veículos do 1º nível serem carregados, estes devem retornar ao depósito para que a destinação final dos produtos coletados seja realizada adequadamente. O 2E-CVRP-HFSDTW apresenta restrições referentes a capacidade dos veículos que pertencem as frotas dos dois níveis, restrições referentes a capacidade dos satélites, restrições de acesso de certos tipos de veículos a alguns pontos de coleta, restrições referentes a janela de tempo no 2º nível que controla o horário que cada cliente pode ser atendido e restrição referentes ao limite de tempo máximo de viagem permitido para os veículos nos dois níveis.

O modelo matemático proposto por Vitorugo e Caliman (2017) tem como objetivo determinar as rotas ótimas para os dois níveis do problema e quais serão os satélites utilizados, de forma que as restrições de capacidade dos veículos e dos satélites, as restrições de acesso, as restrições de janela de tempo e de tempo máximo de viagem nos dois níveis sejam respeitadas, com a minimização do custo de coleta nos dois níveis. Este custo é composto pelo custo de transporte fixo e variável das rotas do 1º e do 2º nível e do custo de operação em cada satélite utilizado. A partir deste ponto neste subcapitulo, o texto escrito a seguir é integralmente de autoria de Vitorugo e Calimam (2017).

Convencionalmente, o 2E-CVRP-HFSDTW pode ser definido como um grafo não orientado $G = (V, E)$, onde o conjunto de vértices V é definido como sendo $V = \{D \cup S \cup C\}$. O conjunto $D = \{0\}$ representa o depósito, o conjunto S representa os satélites e o conjunto C representa os clientes. O conjunto E é definido como sendo $E = \{W \cup X\}$ e representa as arestas que interligam os vértices, sendo $W = \{D \cup S\}$ e $X = \{S \cup C\}$.

O modelo é, ainda, composto pelos seguintes conjuntos:

Y: conjunto dos veículos do 1º nível;
Z: conjunto dos veículos do 2º nível;

K: representa os diversos tipos de veículos do 2º nível.

São parâmetros do modelo:

a: quantidade de satélites;
b: quantidade de clientes;
ψ: o número de tipos de veículos $z \in Z$ do 2º nível;
f_z : tipo $k \in K$ do veículo $z \in Z$;
g_{ik} : parâmetro que tem o valor de 1 caso o cliente $i \in C$ possa ser atendido pelo veículo do tipo $k \in K$ e 0, caso contrário;
l : capacidade dos veículos $y \in Y$, sendo que no 1º nível a frota é homogênea e, portanto, a capacidade é igual para todos os veículos;
λ_z : capacidade de cada veículo $z \in Z$;
d_i : demanda por coleta de cada cliente $i \in X$;
M : parâmetro para a lógica do modelo que assume um valor suficientemente grande;
o_{ij} : custo em R$ de percorrer o arco $(i,j) \in W$ do 1º nível;
p_{ij} : comprimento, em quilômetro, do arco $(i,j) \in X$ do 2º nível;
q_z : custo por quilômetro percorrido no 2º nível do veículo $z \in Z$;
h_s : custo de operação no satélite $s \in S$;
e : custo fixo de transporte para cada veículo $y \in Y$;
u_z : custo fixo de transporte do veículo $z \in Z$;
r_z : indica a qual satélite $s \in S$ pertence o veículo $z \in Z$, cada satélite possui uma frota de veículos dedicada que só pode servir aos clientes desse satélite;
t_s : número máximo de veículos $z \in Z$ utilizados pelo satélite $s \in S$, cada satélite possui uma frota de t_s veículos, definida para atender seus clientes e que pode ser compartilhada com outros satélites;
$\bar{y}_{ij}$: tempo de viagem do veículo $y \in Y$ no arco $(i,j) \in W$ do 1º nível;
$\hat{y}_{ij}$: tempo de viagem de um veículo $z \in Z$ em um arco $(i,j) \in X$ do 2º nível;
tws_i : término da janela de tempo do cliente $i \in C$ do 2º nível;
twe_i : início da janela de tempo do cliente $i \in C$ do 2º nível;
Tmx : tempo máximo de viagem em uma rota do veículo $y \in Y$ do 1º nível;
$Tmax$: tempo máximo de viagem em uma rota do veículo $z \in Z$ do 2º nível;
op : tempo de carga/descarga de um pneu.
O modelo possui dois grupos de variáveis de decisão: um grupo de variáveis para

cada nível do problema. A seguir, são apresentadas as variáveis do primeiro grupo relativas ao 1º nível.

α_{ijy}: variável binária que possui valor igual a 1 se um veículo $y \in Y$ percorrer um arco $(i,j) \in W$, caso contrário seu valor é 0;
η_{sy}: quantidade de carga coletada no satélite $s \in S$ por um veículo $y \in Y$ vindo do depósito;
μ_{sy}: define a posição do satélite $s \in S$ na rota do veículo $y \in Y$.

Continuando a definição das variáveis, são apresentadas as variáveis do segundo grupo relativas ao 2º nível.

ϕ_{ijsz}: representa a carga de um veículo $z \in Z$ ao chegar no nó $j \in X$ vindo do nó $i \in X$. Refere-se à rota do veiculo $z \in Z$ iniciada no satélite $s \in S$;
β_{ijsz}: variável binária que possui valor igual a 1 se o veículo $z \in Z$ percorre um arco $(i,j) \in X$, considerando que ele inicia sua rota em um satélite $s \in S$, e possui valor 0, caso contrário;
τ_s: total de carga coletada nos clientes do 2º nível que são direcionadas para o satélite $s \in S$;
ν_y: variável binária que possui valor igual a 1 se um veículo $y \in Y$ é utilizado e possui valor 0 em caso contrário;
σ_z: variável binária que possui valor igual a 1 se um veículo $z \in Z$ é utilizado e possui valor 0 em caso contrário;

T_{iz}: tempo acumulado do serviço de um veículo $z \in Z$ em um cliente $i \in C$ do 2º nível.

Com base nos conjuntos, parâmetros e variáveis de decisão apresentados, a função objetivo e as restrições do modelo 2E-CVRP-HFSDTW proposto são apresentadas a seguir.

Função Objetivo

$$Minimizar: \sum_{y \in Y} \sum_{s \in S} o_{ij} \alpha_{ijy} + \sum_{s \in S} \sum_{(i,j) \in X} \sum_{z \in Z} q_z p_{ij} \beta_{ijsz} + \sum_{s \in S} h_s \tau_s + \sum_{y \in Y} \sum_{s \in S} \nu_y \alpha_{0s} + \sum_{z \in Z} \sigma_z u_z \quad (1)$$

Restrições

$$\sum_{j \in W} \alpha_{sjy} - \sum_{i \in W} \alpha_{isy} = 0 \qquad \forall s \in S, y \in Y \quad (2)$$

$$\sum_{j \in W} \alpha_{sjy} \leq 1 \qquad \forall s \in W, y \in Y \quad (3)$$

$$\mu_{iy} + 1 \leq \mu_{jy} + (1 - \alpha_{ijy}) M \qquad \forall (i,j) \in S, y \in Y \quad (4)$$

$$\mu_{sy} \geq 0 \qquad \forall s \in S, y \in Y \quad (5)$$

$$\mu_{sy} \leq a \sum_{j \in W} \alpha_{sjy} \qquad \forall s \in S, y \in Y \quad (6)$$

$$\eta_{sy} \leq l \sum_{j \in W} \alpha_{sjy} \qquad \forall s \in S, y \in Y \quad (7)$$

$$\sum_{s \in S} \eta_{sy} \leq l \qquad \forall y \in Y \quad (8)$$

$$\sum_{y \in Y} \eta_{sy} = \tau_s \qquad \forall s \in S \quad (9)$$

$$\sum_{i \in X} \beta_{sisz} \leq \sum_{j \in W} \sum_{z \in Y} \alpha_{sjz} \qquad \forall s \in S, z \in Z \quad (10)$$

$$\sum_{s \in S} \sum_{\{j \in X \mid j \neq (i+a)\}} \sum_{\{z \in Z \mid r_z = s\}} \beta_{(i+a)jsz} = 1 \qquad \forall i \in C \quad (11)$$

$$\sum_{\{j \in X \mid j \neq (i+a)\}} \beta_{j(i+a)sz} - \sum_{\{j \in X \mid j \neq (i+a)\}} \beta_{(i+a)jsz} = 0 \qquad \forall i \in C, s \in S, z \in Z \quad (12)$$

$$\sum_{\{m \in S \mid m \neq s\}} \left(\sum_{i \in C} \beta_{s(i+a)mz} + \sum_{i \in C} \beta_{(i+a)smz} \right) = 0 \qquad \forall s \in S, z \in Z \quad (13)$$

$$\sum_{i \in C} \sum_{z \in Z} \beta_{s(i+a)sz} \leq t_s \qquad \forall s \in S \qquad (14)$$

$$\sum_{\{s \in S \mid s \neq j\}} \sum_{j \in X} \sum_{z \in Z} \beta_{sjsz} \leq \psi \qquad (15)$$

$$\sum_{\{s \in S \mid s \neq j\}} \sum_{j \in X} \beta_{sjsz} \leq 1 \qquad \forall z \in Z \qquad (16)$$

$$\sigma_z = \sum_{\{s \in S \mid s \neq j\}} \sum_{j \in X} \beta_{sjsz} \qquad \forall z \in Z \qquad (17)$$

$$\sum_{\{s \in S \mid r_z = s\}} \sum_{\{j \in X \mid j \neq (i+a)\}} \phi_{(i+a)jsz}$$
$$\geq \left(\left(\sum_{\{s \in S \mid r_z = s\}} \sum_{\{j \in X \mid j \neq (i+a)\}} \beta_{(i+a)jsz} \right) - 1 \right) M \qquad \forall i \in C, z \in Z \qquad (18)$$
$$+ \Upsilon d_{(i+a)} + \sum_{\{s \in S \mid r_z = s\}} \sum_{\{j \in X \mid j \neq (i+a)\}} \phi_{j(i+a)sz}$$

$$\phi_{ijsz} \leq \lambda_z \beta_{ijsz} \qquad \forall s \in S, (i,j) \in X, z \in Z, r_z = s \qquad (19)$$

$$\tau_s = \sum_{i \in C} \sum_{z \in Z} \phi_{(i+a)ssz} \qquad \forall s \in S \qquad (20)$$

$$\sum_{s \in S} \sum_{j \in X} \beta_{j(i+a)sz} = 0 \qquad \forall i \in C, z \in Z, k \in K, g_{if_z} = 0 \qquad (21)$$

$$\mathrm{T}_{(j+a)z} \geq \mathrm{T}_{iz} + \left(\left(\hat{y}_{i(j+a)} + op\ \theta d_i \right) \beta_{i(j+a)sz} \right) - \left(M \left(1 - \beta_{i(j+a)sz} \right) \right) \qquad \forall s \in S, i \in X, j \in C, k \in K, r_z = s,\ i \neq j \qquad (22)$$

$$\mathrm{T}_{sz} \geq 0 \qquad \forall s \in S, z \in Z \qquad (23)$$

$$\mathrm{T}_{(i+a)z} \leq twe_i \sum_{s \in S \mid r_z = s} \sum_{j \in X} \beta_{(i+a)jsz} \qquad \forall i \in C, z \in Z \qquad (24)$$

$$\mathrm{T}_{(i+a)z} \geq tws_i \sum_{s \in S \mid r_z = s} \sum_{j \in X} \beta_{(i+a)jsz} \qquad \forall i \in C, z \in Z \qquad (25)$$

$$\sum_{i \in X} \sum_{j \in C} (\hat{y}_{i(j+a)} + 2\ op\ \theta d_i)\ \beta_{i(j+a)sz} \leq Tmax \qquad \forall s \in S, z \in Z, r_z = s \qquad (26)$$

$$\sum_{(i,j) \in W} \bar{y}_{ij}\ \alpha_{ijy} + \sum_{s \in S} op\ \theta \eta_{sy} \leq Tmx \qquad \forall z \in Z \qquad (27)$$

$$\eta_{sy} \geq 0 \qquad \forall s \in S, y \in Y \qquad (28)$$

$$\mu_{sy} \geq 0 \qquad \forall s \in S, y \in Y \qquad (29)$$

$$\phi_{ijsz} \geq 0 \qquad \forall s \in S, (i,j) \in X, z \in Z \qquad (30)$$

$$\tau_s \geq 0 \qquad \forall s \in S \qquad (31)$$

$$\alpha_{ijy} \in \{0,1\} \qquad \forall (i,j) \in W, y \in Y \qquad (32)$$

$$\beta_{ijsz} \in \{0,1\} \qquad \forall s \in S, (i,j) \in X, z \in Z \qquad (33)$$

A função objetivo, Equação (1), representa o custo total de distribuição do sistema e deve ser minimizada. A primeira parcela dessa Equação (1) representa o custo das viagens realizadas através dos arcos $(i,j) \in W$, enquanto a segunda parcela da Equação (1) representa as viagens realizadas através dos arcos $(i,j) \in X$. A terceira parcela da Equação (1) quantifica o custo de carregamento e descarregamento h_s das cargas em cada satélite $s \in S$. Por fim, a quarta e quinta parcela da Equação (1) representam o custo fixo de transporte total dos veículos de 1º nível e de 2º nível, respectivamente.

As Restrições (2) asseguram a conservação de fluxo de veículos $y \in Y$ em cada satélite $s \in S$. As Restrições (3) garantem que um veículo $y \in Y$ visita um satélite $s \in S$ no máximo uma vez. As Restrições (4), (5) e (6) evitam a formação de *subtour* no 1º nível. As Restrições (7) e (8) impedem que a capacidade l dos veículos $y \in Y$ seja excedida. As Restrições (9) estabelecem que toda carga que chega a um satélite $s \in S$ deve ser distribuída por esse satélite. As Restrições (10) impedem que um satélite $s \in S$ não utilizado no roteamento do 1º nível seja utilizado no roteamento do 2º nível.

As Restrições (11) garantem que todo cliente $i \in C$ seja atendido. As Restrições (12) determinam a conservação de fluxo de veículos $z \in Z$ em cada cliente $i \in C$. As

Restrições (13) eliminam o tráfego entre satélites $s \in S$ no roteamento do 2º nível. As Restrições (14) restringem a quantidade de veículos $z \in Z$ utilizados para a distribuição de cargas de cada satélite $s \in S$ e a Restrição (15) garante que a quantidade total de veículos $z \in Z$ utilizados em todo o roteamento do 2º nível seja menor ou igual à quantidade ψ de veículos $z \in Z$.

As Restrições (16) limitam que cada veículo $z \in Z$ pode fazer no máximo uma viagem a partir de um satélite. As Restrições (17) indicam quais veículos do 2º nível foram utilizados no roteamento.

As Restrições (18) e (19) garantem que a capacidade λ_z dos veículos $z \in Z$ não seja excedida. As Restrições (20) obrigam que toda carga coletada por veículos $z \in Z$ pertencentes a um determinado satélite $s \in S$ deve ser entregue a esse satélite. As Restrições (21) asseguram que um veículo do 2º nível só pode atender um cliente se esse cliente estiver em um local que esse veículo pode percorrer.

As restrições (24), (25) e (26) garantem que cada cliente $i \in C$ seja atendido dentro de sua janela de tempo (tws_i, twe_i). As restrições (27) e (28) garantem que o limite máximo de viagem nas rotas dos veículos do 1º nível e do 2º nível, respectivamente, sejam respeitadas.

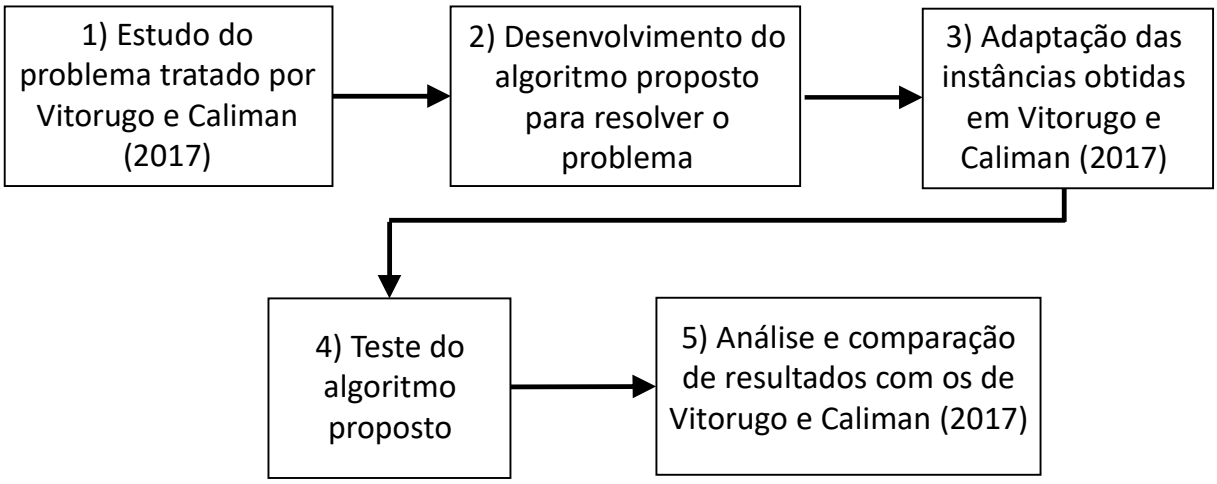
As Restrições (22) a (27) determinam o domínio das variáveis, sendo η_{sy}, μ_{sy}, ϕ_{ijsz} e τ_s variáveis maiores ou iguais a zero e α_{ijy} e β_{ijsz} variáveis binárias.

3. MÉTODO DE PESQUISA

Este trabalho foi desenvolvido a partir do estudo do modelo matemático proposto por Vitorugo e Caliman (2017) e da elaboração de um algoritmo baseado na meta-heurística *Simulated Annealing* para o planejamento da logística reversa de pneus inservíveis para atender uma planta geradora de energia com base no modelo *Two-Echelon Capacitaded Vehicle Routing Problem With Time Windows*.

O estudo foi realizado em 5 etapas distintas e sequenciais, mostrados na Figura 2.

Figura 2 - Etapas do Projeto



Fonte: Do autor.

A primeira etapa consistiu do estudo das características e particularidades do problema abordado por Vitorugo e Caliman (2017) e que serviram de base para seu modelo matemático. Essas informações serviram de base para o prosseguimento do projeto.

Na segunda etapa foi desenvolvido um algoritmo baseado em SA para resolver o problema proposto por Vitorugo e Caliman (2017). O algoritmo foi escrito em linguagem C e executada por meio do software Dev-C++ versão 5.11 (BLOODSHED, 2018).

A terceira etapa consistiu da adequação das instâncias de teste propostas por Vitorugo e Caliman (2017) ao formato necessário (retirada de blocos de instruções do CPLEX, conversão de .dat para .txt, etc) para utilização das mesmas como arquivos de entrada para o algoritmo proposto.

Na quarta etapa foram realizados os primeiros testes do algoritmo desenvolvido utilizando instâncias selecionadas. Estes testes permitiram avaliar o

comportamento do algoritmo, calibrar os parâmetros de entrada e a qualidade dos resultados encontrados.

Por fim, na quinta etapa foram executadas todas as instâncias propostas por Vitorugo e Caliman (2017) e os resultados obtidos pelo algoritmo foram comparados com os obtidos pelo modelo matemático. Devido ao componente aleatório presente na metodologia *Simulated Annealing*, o algoritmo foi executado 10 vezes para cada instância e a média dos resultados foi utilizada para fins de comparação.

3.1. PROBLEMA ESTUDADO

É estudado problema de roteamento em dois níveis da cadeia logística reversa de pneus inservíveis para o estado do Espírito Santo, Brasil. Algumas possíveis instâncias foram levantados levando em consideração a futura instalação de uma empresa recicladora em Cariacica, no Espírito Santo, no condomínio da Marca Ambiental.

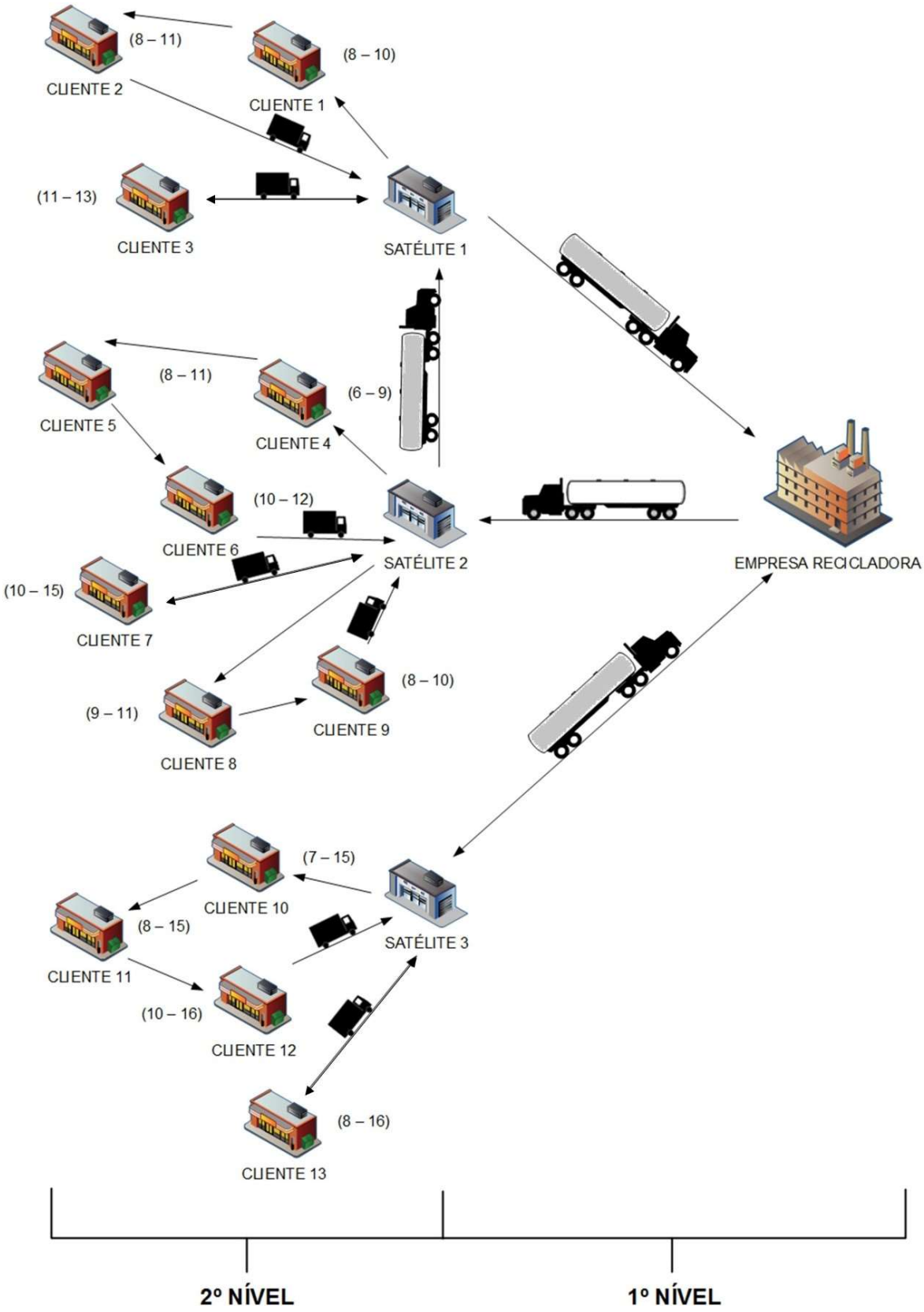
A logística reversa de pneus proposta tem início quando um pneu chega ao fim de sua vida útil e ele deve ser deixado em um local apropriado. O descarte apropriado pode ocorrer com o encaminhamento do pneu para lojas de pneus, borracheiros ou oficinas automotivas, aqui denominados como pontos de coleta. Os pontos de coleta são responsáveis por encaminhar os pneus inservíveis recebidos para centros de consolidação, da onde serão conduzidos à empresa recicladora de pneus. Esta empresa está em fase de implantação e utilizará os pneus para serem queimados e, assim, gerar energia elétrica (FRAGA, 2016).

A cadeia da logística reversa foi considerada como tendo início nos pontos de coleta e seu fim sendo a empresa recicladora, sendo que existem Centros Consolidação de pneus (CCP) para a consolidação dos pneus antes de serem levados a empresa. Os postos de coleta são locais onde o público pode descartar os pneus inservíveis e os CCP são responsáveis pela consolidação e pelo transporte de pneus coletados em vários pontos de coleta até a empresa recicladora.

Comparando a logística reversa de pneus inservíveis com o 2E-CVRP, a empresa recicladora é o armazém, os CCPs são os satélites, os postos de coleta são os clientes e ao lado do cliente, está janela de tempo que o cliente pode ser atendido.

A cadeia proposta é ilustrada na **Figura 3**.

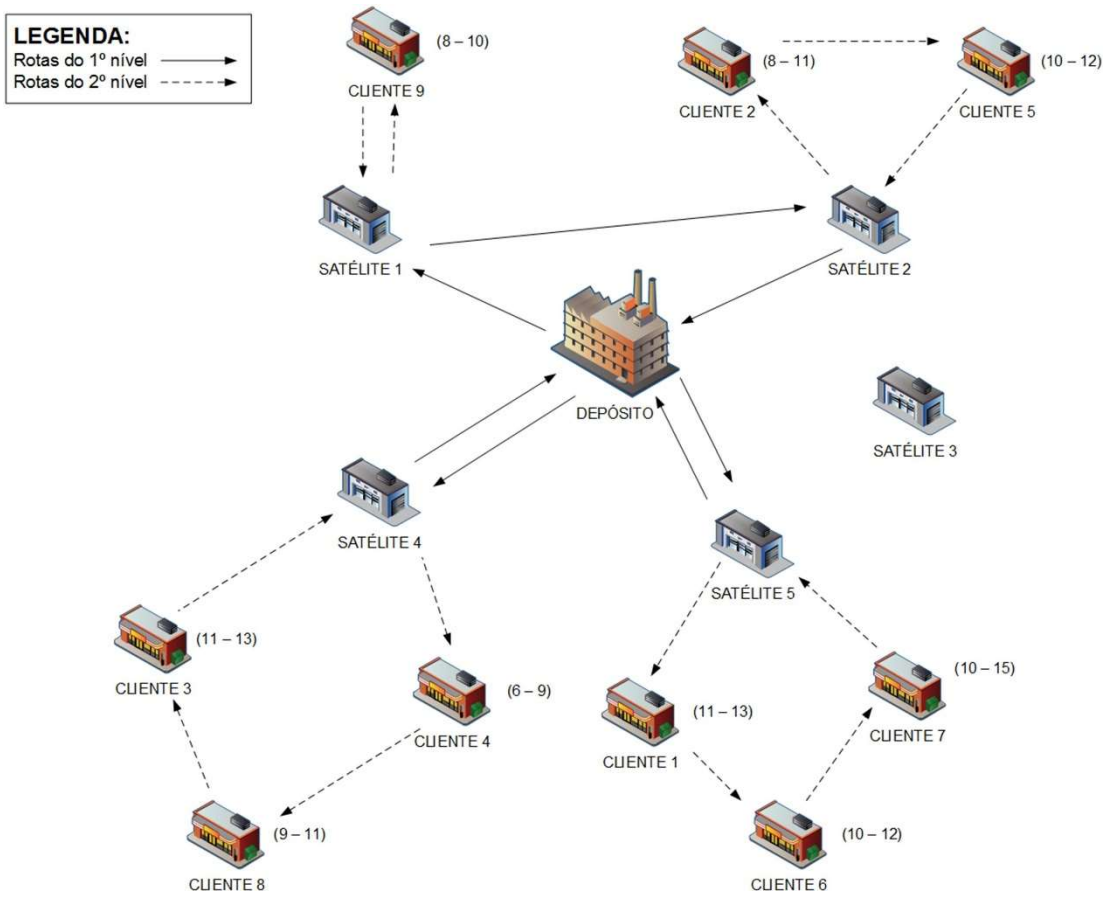
Figura 3 - Representação da rede de logística reversa do pneu



Fonte: Adaptado de Vitorugo e Caliman (2017)

O primeiro nível é composto pela empresa recicladora e pelos CCPs enquanto o segundo nível é composto pelos CCPs e os pontos de coleta. Veículos vazios pertencentes ao segundo nível saem de seus respectivos satélites e visitam cada cliente dentro de uma janela de tempo onde os pneus são coletados e levados para o satélite de onde originou o veículo. Após a finalização do segundo nível, o primeiro nível é iniciado, onde veículos pertencentes ao primeiro nível partem da empresa recicladora em direção aos satélites e coletam os pneus processados e, após o carregamento, retornam a empresa recicladora. Nos satélites os pneus coletados pelos veículos do segundo nível são consolidados pelos veículos do primeiro nível (FRAGA, 2016). A Figura 4 apresenta a esquematização do problema.

Figura 4 - Distribuição 2E-CVRP-HFSDTW



Fonte: Vitorugo e Caliman (2017)

3.2. Instâncias

Essa dissertação utiliza instâncias que foram propostas e executadas por Vitorugo e Caliman (2017), portanto, o detalhamento das instâncias é feito abaixo com base em seu trabalho. Essa prática permite a avaliação dos resultados da meta-heurística proposta quando comparados com um método exato, pois o problema tratado é o mesmo.

Neste trabalho foram utilizadas 14 instâncias, divididas em 2 grupos. Grupo 1, com instâncias de 1 a 7 e Grupo 2 com instâncias de 9 a 14. O Grupo 1 foram criados com o intuito de realizar a planejamento da logística reversa do estado do Espírito Santo. No Grupo 2 a demanda de cada cliente foi acrescida de 50% em relação ao Grupo 1, prevendo um aumento de demandas futuras.

A Tabela 2 apresenta os principais parâmetros das instâncias. A coluna 'Demanda Total' representa a somatória da demanda de todos os cliente da instância. A coluna 'Qtd. Sat.' indica a quantidade de satélites disponíveis em cada instância e a coluna 'Qtd. Clientes', a quantidade de clientes considerados em cada instância.

Por fim, as colunas 'Veículos 1º nível' e 'Veículos 2º nível' subdividem-se em duas colunas cada, sendo elas 'Tipo veículo' e 'Qtd. Veículos', que indicam o tipo de veículo e a quantidade de cada tipo de veículo disponível no roteamento do 1º nível e do 2º nível, respectivamente.

Tabela 2 - Dados das Instâncias

Grupo	Inst.	Demanda total (un)	Qtd. Sat. (un)	Qtd. Clientes (un)	Veículos 1º nível		Veículos 2º nível	
					Tipo veículo	Qtd. Veículos (un)	Tipo veículo	Qtd Veículos (un)
1	1	3840	9	6	1	4	2	7
					2	3	3	7
					3	1	4	7
					4	1		
	2	4800	9	8	1	4	2	7
					2	3	3	7
					3	1	4	7
					4	1		
	3	5760	9	10	1	4	2	7
					2	3	3	7
					3	1	4	7
					4	1		
	4	6720	9	12	1	4	2	7
					2	3	3	7
					3	1	4	7

				4	1		
	5	7680	9	14	1 2 3 4	4 3 1 1	2 3 4	7 7 7
	6	8640	9	20	1 2 3 4	5 3 1 1	2 3 4	7 7 7
	7	8640	9	30	1 2 3 4	5 3 1 1	2 3 4	10 7 5
2	8	5760	9	6	1 2 3 4	4 3 1 1	2 3 4	7 7 7
	9	7680	9	8	1 2 3 4	4 3 1 1	2 3 4	7 7 7
	10	8640	9	10	1 2 3 4	4 3 1 1	2 3 4	7 7 7
	11	9600	9	12	1 2 3 4	4 3 1 1	2 3 4	7 7 7
	12	10560	9	14	1 2 3 4	4 3 1 1	2 3 4	7 7 7
	13	12480	9	20	1 2 3 4	5 3 1 1	2 3 4	7 7 7
	14	13440	9	30	1 2 3 4	4 3 1 1	2 3 4	10 7 5

Fonte: Do autor.

Segundo Vitorugo e Caliman(2017), as instâncias foram criadas pensando na oferta de pneus inservíveis por município do Espírito Santo. Foram feitas variações no número de clientes de acordo com o incremento da capacidade de absorção de pneus

da empresa recicladora, já que uma maior capacidade de reciclagem de pneus possibilita atender um número maior de clientes, utilizando ao todo 30 clientes localizados em 26 municípios do Espírito Santo.

A instância 1 representa a capacidade real da empresa recicladora. A instância 2 considera um aumento de 25% na capacidade real da empresa recicladora. As instâncias 3 e 8 consideram um incremento de 50% na capacidade real. A instância 4, um incremento de 75% na capacidade real. As instâncias 5 e 9 consideram que a capacidade real foi dobrada. Para as instâncias 6, 7 e 10, a capacidade real foi incrementada em 125%. Para a instância 11, um acréscimo de 150% em relação a capacidade real. Finalmente, para as instâncias 12, 13 e 14 a capacidade real foi acrescida de 175%, 225% e 250% respectivamente.

De acordo com Vitorugo e Caliman (2017), as escolhas da localização dos satélites foi feita de acordo com a localização dos pontos de coleta de pneu inservíveis contidos no Espirito Santo conforme os dados da RECICLANIP (2017). A quantidade máxima de satélites existentes foi utilizada em todas as instâncias, pois conforme os dados da RECICLANIP (2017), estes postos já existem e não há a necessidade de verificar qual a melhor localização e custo para os satélites.

Ainda segundo Vitorugo e Caliman (2017), a elaboração das instâncias foi realizada de forma que todas elas respeitassem a Resolução nº 416/2009 do CONAMA que determina a implantação de um ponto de coleta de pneus inservíveis em todos os municípios do Espírito Santo com mais de 100 mil habitantes.

Os 4 tipos de veículos considerados nas instâncias representam 4 tipos reais de veículos e suas características. Os veículos do tipo 1 representa os caminhões pesados com capacidade de 27,5 toneladas, o veículo do tipo 2 representa os caminhões semipesados com capacidades de 16,0 toneladas, o veículo do tipo 3 representa os caminhões médios com capacidade de 10,0 toneladas e por fim o veículo do tipo 4 representa os caminhões semi-leves com capacidade de 3,5 toneladas. Para encontrar a capacidade dos veículos em número máximo de pneus, a capacidade média dos veículos foi dividida pelo peso médio de um pneu e com isso foi obtida uma estimativa da capacidade média do veículo. Uma vez encontrada, esta capacidade foi utilizada como a capacidade do veículo. O custo variável do veículo foi obtido se fazendo a média dos custos de manutenção do veículo e o custo fixo é calculado de acordo com a média que compreende depreciação, remuneração de capital, licenciamento, seguro obrigatório, IPVA, salário do motorista, encargos sociais

e seguro do casco. Vitorugo e Caliman (2017)

A Tabela 3 mostra os dados dos veículos agrupados por tipo.

Tabela 3 - Dados dos Veículos

Veículo (tipo)	Custo do km rodado (R$)	Capacidade (pneus)	Custo fixo (R$)
1	4,12	3.000	496
2	2,06	1.800	280
3	1,76	700	262
4	1,24	280	185

Fonte: Do Autor

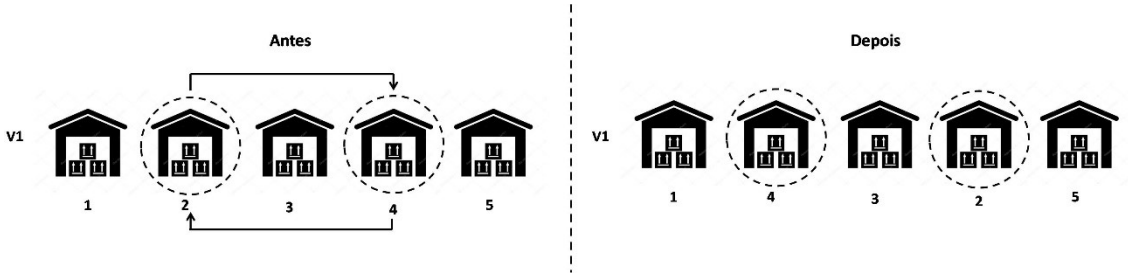
4. Algoritmo Proposto

A meta-heurística Simulated Annealing proposta por Kirkpatrick et al. (1983) consiste numa técnica de busca local probabilística fundamentada no processo de tempera do aço e se mostra muito eficaz na resolução de problemas de otimização. Esta meta-heurística cria uma solução inicial como ponto de partida de sua busca e sucessivamente cria soluções vizinhas a solução atual por meio de um único movimento. Sendo assim foi utilizada a meta-heurística Simulated Annealing como base para a criação do algoritmo proposto por esta dissertação. Para atender as características do 2E-CVRP-HFSDTW, foram criados movimentos para a geração de soluções e foram feitias mudanças na função de cálculo do valor da solução. Uma característica importante é que foi elaborado um algoritmo baseado em SA para cada nível, sendo que os dois níveis devem ser integrados pelo total coletado em cada CCP (satélite).

4.1. Movimentos Propostos

Movimentos são perturbações aplicadas a uma solução com o objetivo de criar soluções vizinhas. Cada um dos níveis possui seu próprio conjunto de movimentos. Para gerar novas soluções (soluções vizinhas) do segundo nível, foram propostos quatro movimentos, o movimento 1, o movimento 2, movimento 3. Caso a solução criada não satisfaça alguma das restrições presentes no algoritmo, ela é descartada e uma nova solução é gerada. O movimento 1 consiste na troca de posições de atendimento de dois clientes diferentes atendidos pelo mesmo veículo. Neste movimento, um veículo é escolhido de forma aleatória e, também de forma aleatória, dois clientes que sejam atendidos por este veículo (Figura 5).

Figura 5 - Movimento 1 do 2° nível

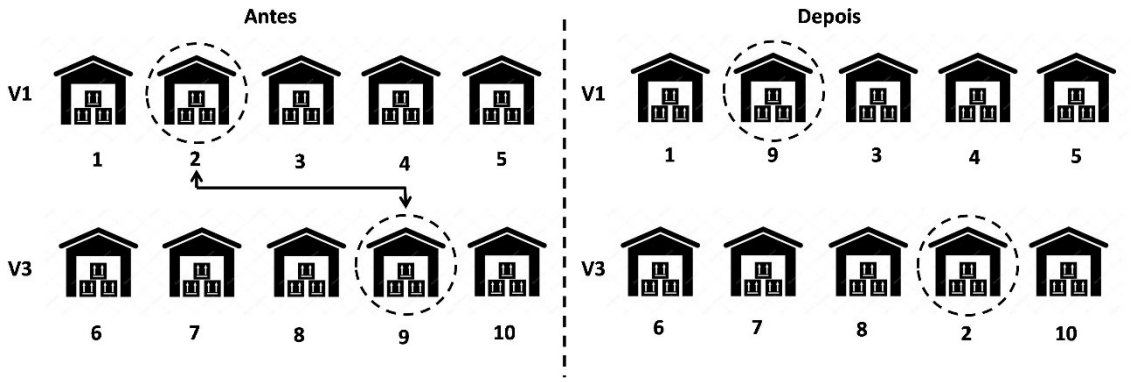


Fonte: Do autor

Na Figura 5**Erro! Fonte de referência não encontrada.** o veículo escolhido aleatoriamente foi o V1, que atendia em sua rota os clientes 1, 2, 3, 4 e 5, representados pela figura de armazéns pois são locais onde o veículo irá buscar cargas. A ordem que os clientes foram mostrados na figura representa as posições de atendimento de cada cliente, ou seja, a ordem em que o veículo passará em cada cliente. Os clientes circulados representam os dois clientes escolhidos de forma aleatória, sendo eles os clientes 2 e 4. As setas mostram o movimento sendo feito, onde o cliente 4 é levado para posição ocupada pelo cliente 2 e o cliente 2 é levado para a posição anteriormente ocupada pelo cliente 4. Por fim é mostrado a nova ordem de atendimento do veículo V1, onde os clientes 2 e 4 agora ocupam suas novas posições de atendimento.

O movimento 2 realiza a troca as posições de atendimento entre dois clientes distintos atendidos por dois veículos distintos. Neste movimento um veículo é escolhido de forma aleatória e também de forma aleatória, um cliente atendido por esse veículo é escolhido. Em seguida um veículo diferente do primeiro é escolhido de forma aleatória e também de forma aleatória, um cliente atendido por esse veículo é escolhido. Após essas escolhas, o cliente do primeiro veículo é retirado de sua posição de atendimento no primeiro veículo e colocado na posição de atendimento do cliente do segundo veículo. Analogamente, o cliente do segundo veículo é retirado de sua posição de atendimento no segundo veículo e colocado na posição de atendimento do cliente do primeiro veículo (Figura 6).

Figura 6 - Movimento 2 do 2° nível

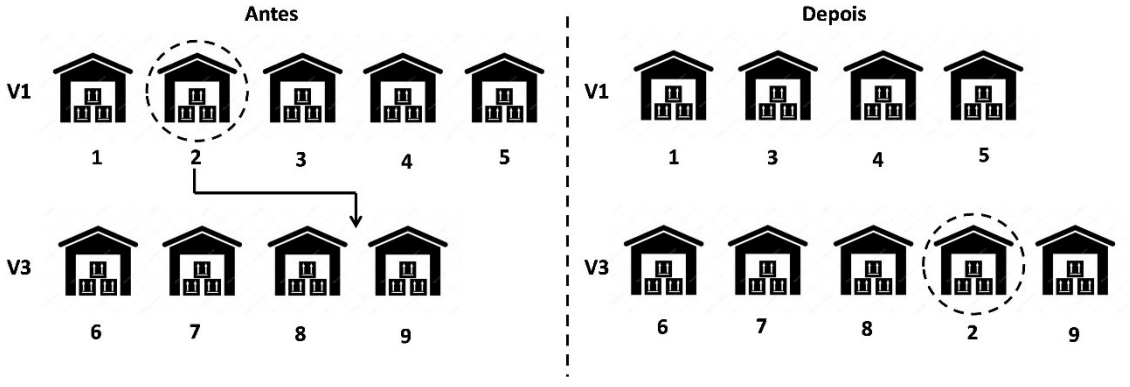


Fonte: Do autor

Na Figura 6**Erro! Fonte de referência não encontrada.** os dois veículos escolhidos aleatoriamente foram o V1 e o V3. O veículo V1 atende em sua rota os clientes 1, 2, 3, 4 e 5. Já veículo V3 atende em sua rota os clientes 6, 7, 8, 9 e 10. Em ambos os veículos os clientes são representados por figuras de armazéns pois são locais onde os veículos irão buscar cargas. A ordem que os clientes foram mostrados na figura representa as posições de atendimento de cada cliente, ou seja, a ordem em que o veículo passará em cada cliente. Os clientes circulados representam os dois clientes escolhidos de forma aleatória, um em cada veículo, sendo eles o cliente 2 pertencente ao veículo V1 e o cliente 9 pertencente ao veículo V3. As setas mostram o movimento sendo feito, onde o cliente 2 é retirado do veículo V1 e levado para o veículo V3 ocupando a posição do pertencente ao cliente 9 e o cliente 9 é retirado do veículo V3 e levado para o veículo V1 ocupando a posição anteriormente ocupada pelo cliente 2. Por fim são mostradas as novas ordens de atendimento do veículo V1 e do veículo V3, onde os clientes 2 e 9 agora ocupam suas novas posições de atendimento em seus novos veículos.

O movimento 3 é a transferência de um cliente de um veículo para o outro veículo diferente do primeiro. Neste movimento um veículo é escolhido de forma aleatória e também de forma aleatória, um cliente atendido por esse veículo é escolhido. Em seguida um veículo diferente do primeiro é escolhido de forma aleatória. Após essas escolhas, o cliente do primeiro veículo é retirado de sua posição de atendimento no primeiro veículo e colocado em uma posição de atendimento aleatória no segundo veículo (Figura 7).

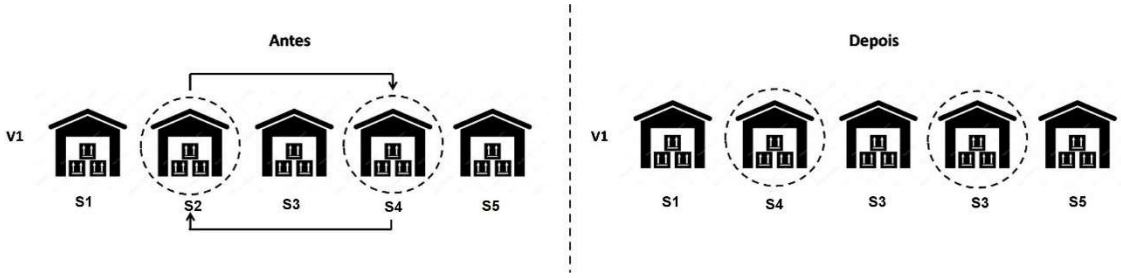
Figura 7 - Movimento 3 do 2° nível



Fonte: Do autor

Na Figura 7 os dois veículos escolhidos aleatoriamente foram o V1 e o V3. O veículo V1 atende em sua rota os clientes 1, 2, 3, 4 e 5. Já veículo V3 atende em sua rota os clientes 6, 7, 8 e 9. Em ambos os veículos os clientes são representados por figuras de armazéns pois são locais onde os veículos irão buscar cargas. A ordem que os clientes foram mostrados na figura representa as posições de atendimento de cada cliente, ou seja, a ordem em que o veículo passará em cada cliente. O cliente circulado representa o cliente escolhido de forma aleatória, sendo ele o cliente 2 pertencente ao veículo V1. A seta mostra o movimento sendo feito, onde o cliente 2 é retirado do veículo V1 e levado para o veículo V3 onde ocupará uma posição escolhida aleatoriamente, que na figura é a posição de atendimento entre o cliente 8 e o cliente 9. Por fim são mostradas as novas ordens de atendimento do veículo V1 e do veículo V3, onde o cliente 2 não mais existe na rota do veículo V1, passando agora a ocupar uma posição na rota do veículo V3.
Para gerar novas soluções (soluções vizinhas) do primeiro nível, serão criados três movimentos, o movimento 1, o movimento 2 e movimento 3. Devido a sua similaridade com o segundo nível, os movimentos do primeiro nível serão muito parecidos com os de segundo nível. A diferença entre eles é que enquanto no segundo nível os veículos atendem os clientes, os veículos de primeiro nível atendem os CCP (satélites) O movimento 1 consiste na troca de posições de atendimento de dois satélites diferentes atendidos pelo mesmo veículo. Neste movimento, um veículo é escolhido de forma aleatória e, também de forma aleatória, dois satélites que sejam atendidos por este veículo (Figura 8).

Figura 8 - Movimento 1 do 1° nível



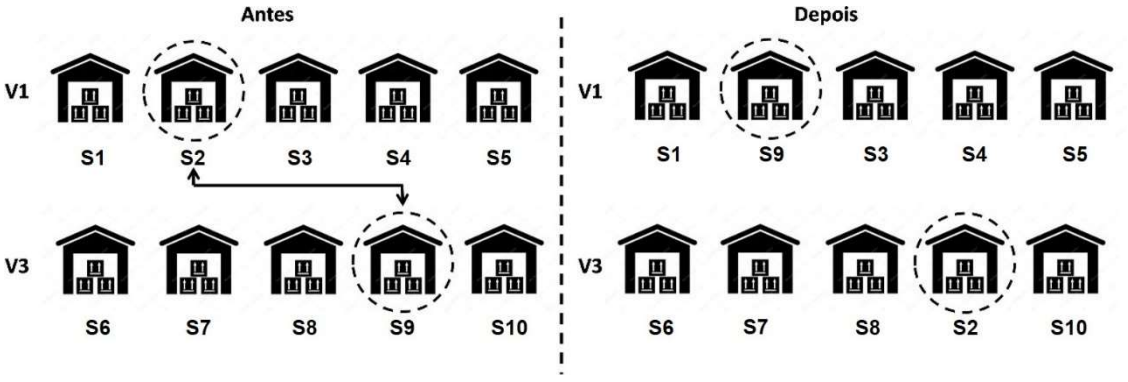
Fonte: Do autor

Na Figura 8 o veículo escolhido aleatoriamente foi o V1, que atendia em sua rota os satélites S1, S2, S3, S4 e S5, representados pela figura de armazéns pois são

locais onde o veículo irá buscar cargas. A ordem que os satélites foram mostrados na figura representa as posições de atendimento de cada satélite, ou seja, a ordem em que o veículo passará em cada satélite. Os satélites circulados representam os dois satélites escolhidos de forma aleatória, sendo eles os satélites S2 e S4. As setas mostram o movimento sendo feito, onde o satélite S4 é levado para posição ocupada pelo satélite S2 e o satélite S2 é levado para a posição anteriormente ocupada pelo satélite S4. Por fim é mostrado a nova ordem de atendimento do veículo V1, onde os satélites S2 e S agora ocupam suas novas posições de atendimento.

O movimento 2 realiza a troca as posições de atendimento entre dois satélites distintos atendidos por dois veículos distintos. Neste movimento um veículo é escolhido de forma aleatória e também de forma aleatória, um satélite atendido por esse veículo é escolhido. Em seguida um veículo diferente do primeiro é escolhido de forma aleatória e também de forma aleatória, um satélite atendido por esse veículo é escolhido. Após essas escolhas, o satélite do primeiro veículo é retirado de sua posição de atendimento no primeiro veículo e colocado na posição de atendimento do satélite do segundo veículo. Analogamente, o satélite do segundo veículo é retirado de sua posição de atendimento no segundo veículo e colocado na posição de atendimento do satélite do primeiro veículo (Figura 9).

Figura 9 - Movimento 2 do 1° nível



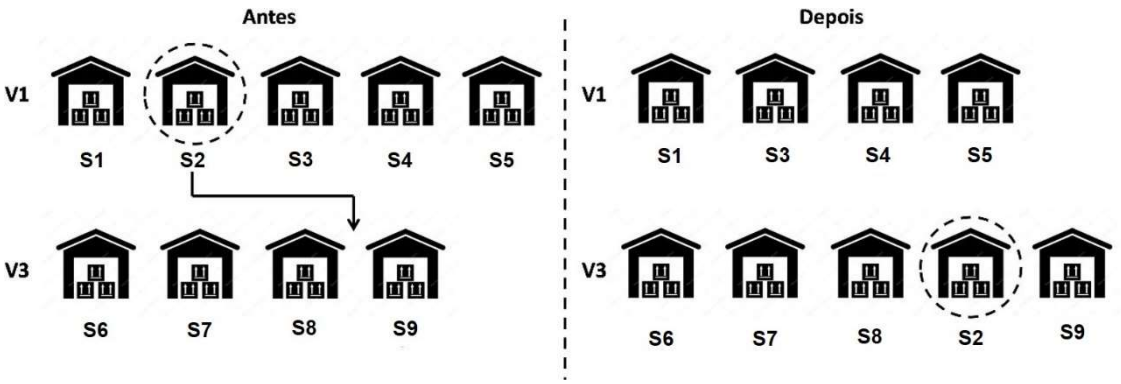
Fonte: Do autor

Na Figura 9**Erro! Fonte de referência não encontrada.** os dois veículos escolhidos aleatoriamente foram o V1 e o V3. O veículo V1 atende em sua rota os satélites S1, S2, S3, S4 e S5. Já veículo V3 atende em sua rota os satélites S6,

S7, S8, S9 e S10. Em ambos os veículos os satélites são representados por figuras de armazéns pois são locais onde os veículos irão buscar cargas. A ordem que os satélites foram mostrados na figura representa as posições de atendimento de cada satélite, ou seja, a ordem em que o veículo passará em cada satélite. Os satélites circulados representam os dois satélites escolhidos de forma aleatória, um em cada veículo, sendo eles o satélite S2 pertencente ao veículo V1 e o satélite S9 pertencente ao veículo V3. As setas mostram o movimento sendo feito, onde o satélite S2 é retirado do veículo V1 e levado para o veículo V3 ocupando a posição do pertencente ao satélite S9 e o satélite S9 é retirado do veículo V3 e levado para o veículo V1 ocupando a posição anteriormente ocupada pelo satélite S2. Por fim são mostradas as novas ordens de atendimento do veículo V1 e do veículo V3, onde os satélites S2 e S9 agora ocupam suas novas posições de atendimento em seus novos veículos.

O movimento 3 é a transferência de um satélite de um veículo para o outro veículo diferente do primeiro. Neste movimento um veículo é escolhido de forma aleatória e também de forma aleatória, um satélite atendido por esse veículo é escolhido. Em seguida um veículo diferente do primeiro é escolhido de forma aleatória. Após essas escolhas, o satélite do primeiro veículo é retirado de sua posição de atendimento no primeiro veículo e colocado em uma posição de atendimento aleatória no segundo veículo (Figura 10).

Figura 10 - Movimento 3 do 1° nível



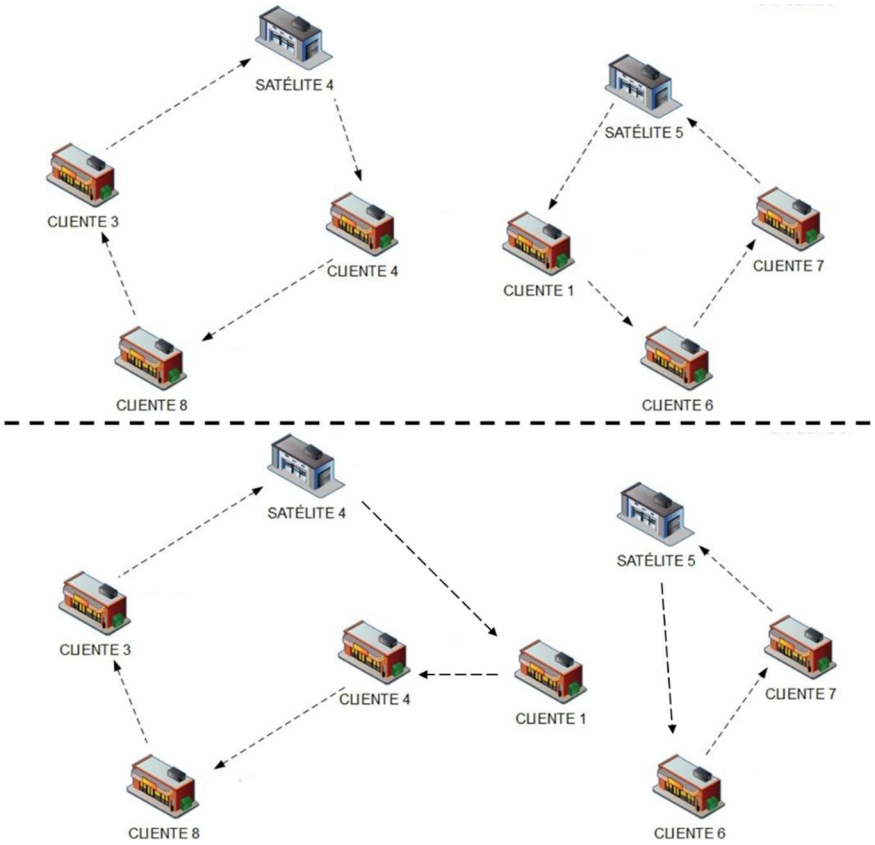
Fonte: Do autor

Na Figura 10 os dois veículos escolhidos aleatoriamente foram o V1 e o V3. O veículo V1 atende em sua rota os satélites S1, S2, S3, S4 e S5. Já veículo V3

atende em sua rota os satélites S6, S7, S8 e S9. Em ambos os veículos os satélites são representados por figuras de armazéns pois são locais onde os veículos irão buscar cargas. A ordem que os satélites foram mostrados na figura representa as posições de atendimento de cada satélite, ou seja, a ordem em que o veículo passará em cada satélite. O satélite circulado representa o satélite escolhido de forma aleatória, sendo ele o satélite S2 pertencente ao veículo V1. A seta mostra o movimento sendo feito, onde o satélite S2 é retirado do veículo V1 e levado para o veículo V3 onde ocupará uma posição escolhida aleatoriamente, que na figura é a posição de atendimento entre o satélite S8 e o satélite S9. Por fim são mostradas as novas ordens de atendimento do veículo V1 e do veículo V3, onde o satélite S2 não mais existe na rota do veículo V1, passando agora a ocupar uma posição na rota do veículo V3.

Para integrar os dois níveis, foi criada uma lógica que determinará se será necessário fazer o roteamento do primeiro nível. Está lógica é utilizada sempre que uma nova solução viável for encontrada para o segundo nível. A lógica é composta de duas condições com o objetivo de fazer com que o primeiro nível seja roteado somente quando necessário. A primeira condição que tem de ser atendida para que haja o novo roteamento do primeiro nível é que tem de haver mudança na quantidade total de carga presente nos satélites. Para que isso ocorra um cliente precisa ter seu destino final alterado devido a um movimento de 2° nível (Figura 11).

Figura 11 - Cliente tendo seu destino final alterado devido a um movimento de 2° nível



Fonte: Do Autor

Na Figura 11, o cliente 1 que inicialmente pertence a uma rota que tem como destino o satélite 5 tem o veículo que faria seu atendimento mudado devido a um movimento no 2° nível, sendo que após este movimento ele teve seu destino final modificado para o satélite 4. A segunda condição é que após a primeira condição ser satisfeita, o veículo que atende o satélite onde o cliente foi colocado não tenha capacidade suficiente para transportar o acréscimo de carga que foi gerado pela adição deste novo cliente em sua rota.

4.2. Visão Geral do Algoritmo Proposto

Esse subcapítulo apresenta a visão geral do algoritmo proposto, com o pseudocódigo exibido pela Figura 12.

Figura 12 - Pseudocódigo do Algoritmo

1	INÍCIO
2	Parametros(iterMax2, T2, TC2, α2, NR2)

```
3  IniciarVariaveis2()
4  IniciarVariaveis1()
5  LeituraDeDados()
6  interacao2 = 0
7  T = T0
8  reannealing2 = 0;
9  SolucaoInicial2(sAtual2)
10 RoteamentoPrimeiroNivel(sAtual2)
11 IgualaSolucao(sArmazenadaAtual1, sBest1)
12 IgualaSolucao(sArmazenadaBest1, sBest1)
13 CalculaFOGeral(sAtual2)
14 IgualaSolucoes2(sBest2, sAtual2)
15 /*Loop de Reannealing /*
16 ENQUANTO (reannealing2 < NR)
17     /*Loop principal – Verifica se foram atendidas as condições de termino do
   algoritmo*/
18     ENQUANTO (temperatura2 > Tc)
19         /*Loop Interno – Realização de perturbação em uma iteração*/
20         PARA (i = 1) ATÉ (iterMax) INCREMENTANDO (1)
21             IgualaSolucao(sViz2, sAtual2)
22             retorno = Gera_Vizinho(sViz2)
23             SE(retorno == 0)
24                 demandaSatelite(sViz2)
25                 retorno = PrecisaPrimeiroNivel(sViz2)
26             SE(retorno == 1)
27                 CalculaFOGeral(sViz2, sBest1)
28             SE NÃO
29                 CalculaFOGeral(sViz2, sArmazenadaAtual1)
30             FIM-SE NÃO
31             SE (sViz2.FO < sAtual2.FO)
32                 IgualaSolucoes2(sAtual2, sViz2)
33                 SE (retorno == 1)
34                     IgualaSolucoes1(sArmazenadaAtual1, sBest1)
35                 FIM-SE
36                 SE(sViz2.FO < sBest2.FO)
37                     IgualaSolucoes2(sBest2, sViz2)
38                     SE(retorno == 1)
39                         IgualaSolucoes1(sArmazenadaBest1, sBest1)
40                     FIM-SE
41                 FIM-SE
42             SENÃO
43                 x = ((rand()%100)/99.0)
44                 Δ = sViz2.FO – sAtual2.FO
45                 /*Teste de aceitação de uma solução pior*/
46                 SE (x < e^(-Δ/T))
47                     IgualaSolucao(sAtual2, sViz2)
48                     SE(retorno == 1)
49                         IgualaSolucoes1(sArmazenadaAtual1, sBest1)
```

```
50                                  FIM-SE
51                              FIM SE
52                          FIM-SENÃO
53                  FIM-PARA
54                  /*Atualização da Temperatura*/
55                  T = α×T
56                  FIM-ENQUANTO /*Loop principal*/
57                  r = r + 1
58                  T =  T0/e^(log((T0/100)/(NR - r + 1) ))    /*Calculo da temperatura para o
reannealing*/
59                  T0 = T
60          FIM-ENQUANTO /*Lood Reannealing*/
61  /*Saída do Algoritmo*/
62  Imprima(sBest1)
63  FIM-ALGORITMO
```

Fonte: Do Autor

No início, os parâmetros do algoritmo número de iterações, temperatura inicial, temperatura de congelamento, coeficiente de resfriamento e o número de reannealing, são inicializados com valores pré-determinados pela função *Parametros*. As variáveis necessárias para o funcionamento do SA de segundo nível são inicializadas com valor ‘0’ pela função *InicializarVariaveis2* e para o SA de segundo nível são inicializadas com valor ‘0’ pela função *InicializarVariaveis1*. Por fim, acontece a leitura da instância pela função *LeituraDeDados* e a atribuição de valores às variáveis do segundo nível $Interacao2$, $temperatura2$ e $reanealing2$, que serão incrementadas ao longo do pseudocódigo.

O algoritmo então cria uma solução possível ($\mathrm{sAtual2}$) para o segundo nível utilizando a função *SolucaoInicial2*. Esta solução é criada ordenando-se os clientes de acordo com o início de seus horários de atendimento e alocando-os sequencialmente nos veículos de forma que todas as necessidades do cliente sejam atendidas. Esta solução viável será o ponto de partida para o algoritmo procurar novas soluções de segundo nível.

Depois, é necessário que seja feito um roteamento de primeiro nível para que seja obtido uma solução para a instância. Esse roteamento de primeiro nível é feito através da função *RoteamentoPrimeiroNivel,* que recebe da solução de segundo nível quais satélites estão sendo utilizados, suas demandas e cria uma solução de primeiro nível.

A solução de primeiro nível é então copiada para as estruturas de solução $sArmazenadaAtual1$ e $sArmazenadaBest1$ utilizando a função *IgualaSolucoes1.* Esta cópia é feita para que a solução de primeiro nível atual não seja perdida quando um novo roteamento de primeiro nível é feito, pois a nova solução viável encontrada pode ser descartada devido a um maior valor de FO que o atual.

Uma vez de posse de soluções de primeiro e segundo nível ($sAtual2$ e $sArmazenadaBest1$), o algoritmo calcula a FO (função objetivo) da instância utilizando a função *CalculaFOGeral* e copia a estrutura $\mathrm{sAtual2}$ para $\mathrm{sBest2}$ usando a função *IgualaSolucoes2.*

O algoritmo entra no *loop* de *reannealing,* incrementando de 1 até chegar no número máximo de *reannealings* estipulado (*NR*), logo após, entra no *loop* de *temperatura,* onde a temperatura é reduzida a um fator de decréscimo α até que seja atingida a temperatura de congelamento ($T0$). Após entrar no *loop* de *temperatura,* o algoritmo entra no *loop* de *iterações*, incrementando de 1 até chegar no número máximo de iterações ($\mathrm{IterMax2}$). No início deste *loop*, a estrutura solução $sAtual2$ e copiada para a estrutura solução $sViz2$ e a demanda dos satélites na solução $sViz2$ é calculada com base na carga dos veículos que chegam no satélite utilizando a função *demandaSatelite*.

Uma vez definida a demanda dos satélites, a função *GeraVizinho* gera uma perturbação na solução armazenada na estrutura $sViz2$ para gerar uma nova solução. Se a função obteve sucesso em gerar uma nova solução, ela retorna ‘0’, caso contrário, retorna ‘1’.

Caso *GeraVizinho* retorne uma solução viável, o algoritmo utiliza a função *demandaSatelite* para calcular a demanda nos satélites para a nova solução $sViz2$ encontrada e utiliza a função *PrecisaPrimeiroNivel* para verificar se a nova solução de segundo nível encontrada, $sViz2$, faz necessário o roteamento do primeiro nível. Se é necessário fazer o roteamento de primeiro nível, a função *PrecisaPrimeiroNivel* retorna ‘1’, uma nova $sBest1$ é criada a partir do novo roteamento de primeiro nível e o cálculo da FO é feito pela função *CalculaFOGeral* utilizando $sViz2$ e $sBest1$. Caso contrário, calculo da FO é feito pela função *CalculaFOGeral* utilizando $sViz2$ e $sArmazenadaAtual1$.

Se a FO de $sViz2$ possuir um valor menor que a FO de $sAtual2$, a função *IgualaSolucoes2* copia a estrutura $sViz2$ para $sAtual2$. Se além de atender a condição anterior, o retorno da função *PrecisaPrimeiroNivel* feita anteriormente for '1', *IgualaSolucoes2* copia a estrutura $sBest1$ para $sArmazenadaAtual1$.

Se a FO de $sViz2$ possuir um valor menor que a FO de $sBest2$, a função *IgualaSolucoes2* copia a estrutura $sViz2$ para $sBest2$. Se além de atender a condição anterior, o retorno da função *PrecisaPrimeiroNivel* feita anteriormente for '1', *IgualaSolucoes2* copia a estrutura $sBest1$ para $sArmazenadaBest1$.

Caso a FO de $sViz2$ não possua um valor menor que a FO de $sAtual2$, o algoritmo verifica a possibilidade de aceitar uma solução $sViz2$ pior que a $sAtual2$ através de uma rotina de cálculo característica do *Simulated Annealing*. Primeiramente é calculado um valor 'x' que dá aleatoriedade a aceitação de solução com valor de FO pior, sendo $x = \frac{(rand()\%100)}{99.0}$. Depois, um valor $(\Delta) = sViz2.FO - sAtual2.FO$, sendo $sViz2.FO$ o valor da FO de $sViz2$ e $sAtual2.FO$ o valor da FO de $sAtual2$. Então, o algoritmo faz o teste, se o valor de x for menor que $e^{(\frac{-(\Delta)}{T})}$, sendo T o valor da temperatura no momento em que a solução foi encontrada, a solução pior é aceita. Sendo assim, soluções piores tem maior probabilidade de serem aceitas quando a temperatura tem valores elevados, como por exemplo, no início do algoritmo ou logo após um *reannaling*, quando as temperaturas estão mais elevadas.

Se a solução pior foi aceita, função *IgualaSolucoes2* copia a estrutura $sViz2$ para $sAtual2$. Se além de atender a condição anterior, o retorno da função *PrecisaPrimeiroNivel* feita anteriormente for '1', *IgualaSolucoes2* copia a estrutura $sBest1$ para $sArmazenadaAtual1$.

Após o fim do *loop* de iterações a temperatura é atualizada ($T = T \times \alpha$) e o *loop* de iterações recomeça. Após a temperatura alcançar a temperatura de congelamento (Tc), a temperatura sobe de acordo com a equação $T = \frac{T0}{e^{\frac{\log(\frac{T0}{100})}{(NR-r+1)}}}$, se o número de *reannealings* (r) for menor que o número máximo de *reannealinigs* (NR), o *loop* de temperatura recomeça tendo a nova temperatura calculada T como ponto de partida. Caso o número máximo de *reannealinigs* (NR) tenha sido atingido, o *loop*

de *reannealing* se encerra e o algoritmo termina.

4.3. Algoritmo de Primeiro Nível e Interação Entre Níveis

Este algoritmo de roteamento de veículos em dois níveis é composto por dois algoritmos baseados em *Simulated Annealing*, um para o primeiro nível, que realiza o roteamento dos veículos que partem do armazém e atendem os satélites, e um para o segundo nível, que realiza o roteamento dos veículos que partem dos satélites e atendem os clientes.

O roteamento de veículos de primeiro nível é diretamente afetado pelo resultado do roteamento do segundo nível, pois seu resultado afeta as demandas dos satélites e quais satélites precisam ser atendidos no primeiro nível.

A função *PrecisaPrimeiroNivel* recebe uma estrutura solução de segundo nível e verifica se os veículos de primeiro nível que atendem os satélites possuem capacidade suficiente para absorver a mudança nas demandas dos satélites. Caso a capacidade de algum dos veículos de primeiro nível não seja suficiente para acomodar o aumento de demanda do satélite, o roteamento de primeiro nível é feito através da função *RoteamentoPrimeiroNível*. Caso seja necessário fazer o roteamento de primeiro nível, a função *PrecisaPrimeiroNivel* retorna ‘1’, caso contrário, retorna ‘0’. O pseudocódigo da função *PrecisaPrimeiroNivel* exibido na Figura 13.

Figura 13 - Pseudocódigo do PrecisaPrimeiroNivel

```
1  INÍCIO
2  PARA (i=0) ATÉ (Sol1.NCCusados) INCREMENTANDO (1)
3      PARA (j=0) ATÉ (Sol1.NCC[Sol1.VeicUsados[i]]) INCREMENTANDO (1)
4          necessario = necessario + satelite[Sol1.matriz[Sol1.VeicUsados[i]][j]].demanda
5      FIM-PARA
6      SE(veiculo[i].capacidade < necessario)
7          RoteamentoPrimeiroNivel()
8          retorna 1
9      FIM-SE
10 FIM-PARA
11 retorna 0
12 FIM DA FUNCAO
```

Fonte: Do autor.

A função *RoteamentoPrimeiroNivel* realiza o roteamento de primeiro nível e retorna uma nova estrutura solução de primeiro nível $sBest1$. O pseudocódigo da função

RoteamentoPrimeiroNivel exibido na Figura 14.

Figura 14 - Pseudocódigo do Roteamento de Primeiro Nível

```
1  INÍCIO
2  Parametros(iterMax1, T1, TC1, α1, NR1)
3  SolucaoInicial1(sAtual1)
4  CriaVetorVeiculosUsados(sAtual1)
5  CalculaFO1(sAtual1)
6  IgualaSolucoes1(sBest1, sAtual1)
7  Interacao1 = 0
8  T1 = T01
9  Reannealing1 = 0;
10 /*Loop de Reannealing /*
11 ENQUANTO (r1 < NR1)
12     /*Loop principal – Verifica se foram atendidas as condições de termino do algoritmo*/
13     ENQUANTO (T1 > Tc1)
14         /*Loop Interno – Realização de perturbação em uma iteração*/
15         PARA (Iteracao1 = 1) ATÉ (iterMax1) INCREMENTANDO (1)
16             IgualaSolucao1(sViz1, sAtual1)
17             retorno = GeraVizinho1(sViz1)
18             SE(retorno == 0)
19                 CalculaFO1(sViz1)
20                 SE (sViz1.FO < sAtual1.FO)
21                     IgualaSolucoes1(sAtual1, sViz1)
22                     SE(sViz1.FO < sBest1.FO)
23                         IgualaSolucao1(sBest1, sViz1)
24                     FIM-SE
25                 SENÃO
26                     x = ((rand()%100)/99.0)
27                     Δ = sViz1.FO – sAtual1.FO
28                     /*Teste de aceitação de uma solução pior*/
29                     SE (x < e^(-Δ/T1))
30                         IgualaSolucao1(sAtual2, sViz2)
31                     FIM SE
32                 FIM-SENÃO
33             FIM-SE
34         FIM-PARA
35         /*Atualização da Temperatura*/
36         T1 = α1×T1
37     FIM-ENQUANTO /*Loop principal*/
38     r1 = r1 + 1
39     T1 =  T01/e^(log((T01/100)/(NR1 – r1 + 1) ))   /*Calculo da temperatura para o
       reannealing*/
40     T01 = T1
41 FIM-ENQUANTO /*Lood Reannealing*/
42 FIM DA FUNCAO
```

Fonte: Do Autor

O algoritmo primeiramente cria uma solução possível ($\mathrm{sAtual1}$) para o primeiro nível utilizando a função *SolucaoInicial1*. Esta solução é criada ordenando-se os satélites de acordo com o início de seus horários de atendimento e alocando-os sequencialmente nos veículos de forma que todas as necessidades do satélite sejam atendidas. Esta solução será o ponto de partida para o algoritmo procurar novas soluções de primeiro nível.

Uma vez que o algoritmo encontrou uma solução inicial, a função *CriaVetorVeiculosUsados* é feita. Esta função recebe uma estrutura solução e cria um vetor contendo todos os veículos que estão sendo utilizados. Tal vetor será utilizado na escolha dos veículos dentro da função *GeraVizinho1*.

O algoritmo então utiliza a função *CalculaFO1* para calcular a função objetivo (FO) da estrutura solução ($sAtual1$) e copia a estrutura solução $sAtual1$ para a estrutura solução $sBest1$ utilizando a função *IgualaSolucoes*.

São então atribuídos valores às variáveis do primeiro nível $Iteracao1$, $T1$ e $r1$, que serão incrementadas ao longo do pseudocódigo.

O algoritmo entra no *loop* de *reannealing,* incrementando de 1 até chegar no número máximo de *reannealings* estipulado ($NR1$), logo após, entra no *loop* de *temperatura,* onde a temperatura é reduzida a um fator de decréscimo $\alpha 1$ até que seja atingida a temperatura de congelamento ($Tc1$). Após entrar no *loop* de *temperatura,* o algoritmo entra no *loop* de *iterações*, incrementando de 1 até chegar no número máximo de iterações ($\mathrm{IterMax1}$). No início deste *loop*, a estrutura solução $sAtual1$ e copiada para a estrutura solução $sViz1$.

Em seguida a função *GeraVizinho1* utiliza a solução armazenada na estrutura $sViz1$ para gerar uma nova solução. Se a função obteve sucesso em gerar uma nova solução, ela retorna ‘0’, caso contrário, retorna ‘1’.

Caso *GeraVizinho1* retorna uma solução viável, o valor da função objetivo (FO) de $sViz1$ é calculada utilizando a função *CalculaFO1*.

Se a FO de $sViz1$ possuir um valor menor que a FO de $sAtual1$, a função *IgualaSolucoes1* copia a estrutura $sViz1$ para $sAtual1$.

Se a FO de $sViz1$ possuir um valor menor que a FO de $sBest1$, a função

IgualaSolucoes1 copia a estrutura $sViz1$ para $sBest1$.

Caso a FO de $sViz1$ não possua um valor menor que a FO de $sAtual1$, o algoritmo verifica a possibilidade de aceitar uma solução $sViz1$ pior que a $sAtual1$ através de uma rotina de cálculo característica do *Simulated Annealing*. Primeiramente é calculado um valor 'x' que dá aleatoriedade a aceitação de solução com valor de FO pior, sendo $x = \frac{(rand()\%100)}{99.0}$. Depois, um valor $(\Delta) = sViz1.FO - sAtual1.FO$, sendo $sViz1.FO$ o valor da FO de $sViz1$ e $sAtual1.FO$ o valor da FO de $sAtual1$. Então, o algoritmo faz o teste, se o valor de x for menor que $e^{(\frac{-(\Delta)}{T})}$, sendo T o valor da temperatura no momento em que a solução foi encontrada, a solução pior é aceita. Sendo assim, soluções piores tem maior probabilidade de serem aceitas quando as temperaturas têm valor elevado, como por exemplo, no início do algoritmo ou logo após um *reannealing*, quando as temperaturas estão mais elevadas.

Se a solução pior foi aceita, função *IgualaSolucoes1* copia a estrutura $sViz1$ para $sAtual1$.

Após o fim do *loop* de iterações a temperatura é atualizada ($T1 = T1 \times \alpha 1$) e o *loop* de iterações recomeça. Após a temperatura alcançar a temperatura de congelamento ($Tc1$), a temperatura sobe de acordo com a equação $T1 = \frac{T01}{e^{\frac{\log(\frac{T01}{100})}{(NR-r1+\ )}}}$, se o número de *reannealings* ($r1$) for menor que o número máximo de *reannealinigs* ($NR1$), o *loop* de temperatura recomeça tendo a nova temperatura calculada $T1$ como ponto de partida. Caso o número máximo de *reannealinigs* ($NR1$) tenha sido atingido, o *loop* de *reannealing* se encerra e a função termina, retornando uma nova estrutura de solução de primeiro nível $sBest1$.

4.4. Funções *GeraVizinho* e *GeraVizinho1*

A função *GeraVizinho* tem o objetivo de criar uma solução $sViz2$. Ela recebe uma estrutura de solução $sViz2$ e as informações da instância e retorna uma estrutura de solução $sViz2$ modificada. O pseudocódigo da função *GeraVizinho* exibido na Figura 15

Figura 15 - Pseudocódigo função *GeraVizinho*

1	INICIO

```
2  movimento = 1 + rand() % 3
3  SE (movimento == 1)
4      aleVeic1 = rand() % sViz2.NCCusados;
5      retornoMov = Movimento1(sViz2, aleVeic1)
6      SE (retornoMov == 0)
7          retornoMov = JanelaTempo(sViz2, aleVeic1)
8          SE (retornoMov == 1)
9              retorna 1
10         FIM-SE
11         CalculaDistTotal2(sViz2, aleVeic1)
12         retorna 0
13     FIM-SE
14     retorna 1
15 FIM-SE
16 SE (movimento == 2)
17     aleVeic1 = rand() % sViz2.NCCusados;
18     aleVeic1 = aleVeic2
19     ENQUANTO ( aleVeic2 == aleVeic1) FAZ
20         aleVeic2 = rand() % sViz2.NCCusados;
21     FIM-ENQUANTO
22     retornoMov = Movimento2(sViz2, aleVeic1, aleVeic2)
23     SE (retornoMov == 0)
24         retornoMov = JanelaTempo(sViz2, aleVeic1)
25         SE (retornoMov == 1)
26             retorna 1
27         FIM-SE
28         retornoMov = JanelaTempo(sViz2, aleVeic2)
29         SE (retornoMov == 1)
30             retorna 1
31         FIM-SE
32         CalculaDistTotal2(sViz2, aleVeic1)
33         CalculaDistTotal2(sViz2, aleVeic2)
34         retorna 0
35     FIM-SE
36     retorna 1
37 FIM-SE
38 SE (movimento == 3)
39     aleVeic1 = rand() % sViz2.NCCusados
40     aleVeic1 = aleVeic2
41     ENQUANTO ( aleVeic2 == aleVeic1) FAZ
42         aleVeic2 = rand() % numVeiculos2
43     FIM-ENQUANTO
44     retornoMov = Movimento3(sViz2, aleVeic1, aleVeic2)
45     SE (retornoMov == 0)
46         retornoMov = JanelaTempo(sViz2, aleVeic1)
47         SE (retornoMov == 1)
48             retorna 1
49         FIM-SE
```

```
50                  retornoMov = JanelaTempo(sViz2, aleVeic2)
51              SE (retornoMov == 1)
52                  retorna 1
53              FIM-SE
54              CalculaDistTotal2(sViz2, aleVeic1)
55              CalculaDistTotal2(sViz2, aleVeic2)
56              CriarVetorVeiculosUsados2(sViz2)
57              retorna 0
58      FIM-SE
59      retorna 1
60  FIM-SE
61  retorna 1
62  FIM DA FUNÇÃO
```

Fonte – Do Autor

Primeiramente a função *GeraVizinho* atribui a variável ‘movimento’ um número aleatório de 1 a 3.

Se ‘movimento’ for igual a 1, o algoritmo escolhe aleatoriamente um veículo que esteja sendo utilizado ($aleVeic1$). A função *Movimento1*, que é responsável pela troca da posição de atendimento entre dois clientes do mesmo veículo, é então chamada. Ela recebe a estrutura solução $sViz2$ e $aleVeic1$ e retorna ‘0’ caso o movimento seja possível e retorna ‘1’ caso não seja possível. Se *Movimento1* retorna ‘0’, a função *JanelaTempo* é chamada para calcular se a rota do $aleVeic1$ na estrutura solução $sViz2$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função *JanelaTempo* retorna ‘1’ e o que faz a função *GeraVizinho* também retornar ‘1’. Se *JanelaTempo* retornar ‘0’, a distância total percorrida por $aleVeic1$ na estrutura solução $sViz2$ é calculada utilizando a função *CalculaDistTotal*. Uma vez calculadas as distâncias a função *GeraVizinho* retorna ‘0’.

Se ‘movimento’ for igual a 2, o algoritmo escolhe aleatoriamente dois veículos que estejam sendo utilizados ($aleVeic1$ e $aleVeic2$). A função *Movimento2*, que é responsável pela troca de dois clientes entre veículos, é então chamada. Ela recebe a estrutura solução $sViz2$ e os veículos escolhidos anteriormente e retorna ‘0’ caso o movimento seja possível e retorna ‘1’ caso não seja possível. Se *Movimento2* retorna ‘0’, a função *JanelaTempo* é chamada para calcular se a rota de $aleVeic1$ contida na estrutura solução $sViz2$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função

JanelaTempo retorna ‘1’ e o que faz a função *GeraVizinho* também retornar ‘1’. Se *JanelaTempo* retornar ‘0’, a função *JanelaTempo* é chamada para calcular se a rota de $aleVeic1$ contida na estrutura solução $sViz2$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função *JanelaTempo* retorna ‘1’ e o que faz a função *GeraVizinho* também retornar ‘1’. Se *JanelaTempo* retornar ‘0’, distância total percorrida por $aleVeic1$ contida na estrutura solução $sViz2$ é calculada utilizando a função *CalculaDistTotal*, seguido pelo cálculo da distância total percorrida por $aleVeic2$ contido estrutura solução $sViz2$ utilizando a função *CalculaDistTotal.* Uma vez calculadas as distâncias a função *GeraVizinho* retorna ‘0’.

Se ‘movimento’ for igual a 3, o algoritmo escolhe aleatoriamente dois veículos ($aleVeic1$ e $aleVeic2$) que estejam sendo utilizados. A função *Movimento3*, que é responsável pela retirada de um cliente do primeiro veículo e alocação deste cliente no segundo veículo, é então chamada. Ela recebe a estrutura solução $sViz2$, os veículos escolhidos anteriormente e retorna ‘0’ caso o movimento seja possível e retorna ‘1’ caso não seja possível. Se *Movimento3* retorna ‘0’, a função *JanelaTempo* é chamada para calcular se a rota de $aleVeic1$ contida na estrutura solução $sViz2$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função *JanelaTempo* retorna ‘1’ e o que faz a função *GeraVizinho* também retornar ‘1’. Se *JanelaTempo* retornar ‘0’, a função *JanelaTempo* é chamada para calcular se a rota de $aleVeic2$ contida na estrutura solução $sViz2$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função *JanelaTempo* retorna ‘1’ e o que faz a função *GeraVizinho* também retornar ‘1’. Se *JanelaTempo* retornar ‘0’, distância total percorrida por $aleVeic1$ contida na estrutura solução $sViz2$ é calculada utilizando a função *CalculaDistTotal*, seguido pelo cálculo da distância total percorrida por $aleVeic2$ contido na estrutura solução $sViz2$ é calculada utilizando a função *CalculaDistTotal* Uma vez calculadas as distâncias a função *GeraVizinho* retorna ‘0’.

A função *GeraVizinho1* é muito similar a função *GeraVizinho*, com a diferença que a função *GeraVizinho1* atua no roteamento de primeiro nível enquanto a função *GeraVizinho* atua no roteamento de segundo nível.

A função *GeraVizinho1* tem o objetivo de criar uma solução $sViz1$. Ela recebe uma estrutura de solução $sViz1$ e as informações da instância e retorna uma estrutura de solução $sViz1$ modificada. O pseudocódigo da função *GeraVizinho1* é exibido na Figura 16

Figura 16 - Pseudocódigo função *GeraVizinho1*

```
1  INICIO
2  movimento = 1 + rand() % 3
3  SE (movimento == 1)
4      aleVeic1 = rand() % sViz1.NCCusados;
5      retornoMov = Movimento11(sViz1, aleVeic1)
6      SE (retornoMov == 0)
7          retornoMov = JanelaTempo1(sViz1, aleVeic1)
8          SE (retornoMov == 1)
9              retorna 1
10         FIM-SE
11         CalculaDistTotal1(sViz1, aleVeic1)
12         retorna 0
13     FIM-SE
14     retorna 1
15 FIM-SE
16 SE (movimento == 2)
17     aleVeic1 = rand() % sViz1.NCCusados
18     aleVeic1 = aleVeic2
19     ENQUANTO (aleVeic2 == aleVeic1) FAZ
20         aleVeic2 = rand() % sViz1.NCCusados;
21     FIM-ENQUANTO
22     retornoMov = Movimento21(sViz1, aleVeic1, aleVeic2)
23     SE (retornoMov == 0)
24         retornoMov = JanelaTempo1(sViz1, aleVeic1)
25         SE (retornoMov == 1)
26             retorna 1
27         FIM-SE
28         retornoMov = JanelaTempo1(sViz1, aleVeic2)
29         SE (retornoMov == 1)
30             retorna 1
31         FIM-SE
32         CalculaDistTotal1(sViz1, aleVeic1)
33         CalculaDistTotal1(sViz1, aleVeic2)
34         retorna 0
35     FIM-SE
36     retorna 1
37 FIM-SE
38 SE (movimento == 3)
39     aleVeic1 = rand() % sViz1.NCCusados
40     aleVeic1 = aleVeic2
41     ENQUANTO (aleVeic2 == aleVeic1) FAZ
```

```
42              aleVeic2 = rand() % numVeiculos1
43          FIM-ENQUANTO
44          retornoMov = Movimento31(sViz1, aleVeic1, aleVeic2)
45          SE (retornoMov == 0)
46                  retornoMov = JanelaTempo(sViz1, aleVeic1)
47                  SE (retornoMov == 1)
48                      retorna 1
49                  FIM-SE
50                      retornoMov = JanelaTempo1(sViz1, aleVeic2)
51                  SE (retornoMov == 1)
52                      retorna 1
53                  FIM-SE
54                  CalculaDistTotal1(sViz1, aleVeic1)
55                  CalculaDistTotal1(sViz1, aleVeic2)
56                  retorna 0
57                  FIM-SE
58          retorna 1
59          FIM-SE
60  retorna 1
61  FIM DA FUNÇÃO
```

Fonte – Do Autor

Primeiramente a função *GeraVizinho1* atribui a variável ‘movimento’ um número aleatório de 1 a 3.

Se ‘movimento’ for igual a 1, o algoritmo escolhe aleatoriamente um veículo que esteja sendo utilizado ($aleVeic1$). A função *Movimento11*, que é responsável pela troca da posição de atendimento entre dois clientes do mesmo veículo, é então chamada. Ela recebe a estrutura solução $sViz1$ e $aleVeic1$ e retorna ‘0’ caso o movimento seja possível e retorna ‘1’ caso não seja possível. Se *Movimento11* retorna ‘0’, a função *JanelaTempo1* é chamada para calcular se a rota do $aleVeic1$ na estrutura solução $sViz1$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função *JanelaTempo1* retorna ‘1’ e o que faz a função *GeraVizinho1* também retornar ‘1’. Se *JanelaTempo1* retornar ‘0’, a distância total percorrida por $aleVeic1$ na estrutura solução $sViz1$ é calculada utilizando a função *CalculaDistTotal1*. Uma vez calculadas as distâncias a função *GeraVizinho1* retorna ‘0’.

Se ‘movimento’ for igual a 2, o algoritmo escolhe aleatoriamente dois veículos que estejam sendo utilizados ($aleVeic1$ e $aleVeic2$). A função *Movimento21*, que é responsável pela troca de dois clientes entre veículos, é então chamada. Ela

recebe a estrutura solução $sViz1$ e os veículos escolhidos anteriormente e retorna ‘0’ caso o movimento seja possível e retorna ‘1’ caso não seja possível. Se *Movimento21* retorna ‘0’, a função *JanelaTempo1* é chamada para calcular se a rota de $aleVeic1$ contida na estrutura solução $sViz1$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função *JanelaTempo1* retorna ‘1’ e o que faz a função *GeraVizinho1* também retornar ‘1’. Se *JanelaTempo1* retornar ‘0’, a função *JanelaTempo1* é chamada para calcular se a rota de $aleVeic2$ contida na estrutura solução $sViz1$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função *JanelaTempo1* retorna ‘1’ e o que faz a função *GeraVizinho1* também retornar ‘1’. Se *JanelaTempo1* retornar ‘0’, distância total percorrida por $aleVeic1$ contida na estrutura solução $sViz1$ é calculada utilizando a função *CalculaDistTotal1*, seguido pelo cálculo da distância total percorrida por $aleVeic2$ contido estrutura solução $sViz1$ utilizando a função *CalculaDistTotal1.* Uma vez calculadas as distâncias a função *GeraVizinho1* retorna ‘0’.

Se ‘movimento’ for igual a 3, o algoritmo escolhe aleatoriamente dois veículos ($aleVeic1$ e $aleVeic2$), sendo que $aleVeic1$ é um veículo em uso e $aleVeic2$ não é necessariamente um veículo em uso. A função *Movimento31*, que é responsável pela retirada de um cliente do primeiro veículo e alocação deste cliente no segundo veículo, é então chamada. Ela recebe a estrutura solução $sViz1$, os veículos escolhidos anteriormente e retorna ‘0’ caso o movimento seja possível e retorna ‘1’ caso não seja possível. Se *Movimento31* retorna ‘0’, a função *JanelaTempo1* é chamada para calcular se a rota de $aleVeic1$ contida na estrutura solução $sViz1$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função *JanelaTempo1* retorna ‘1’ e o que faz a função *GeraVizinho1* também retornar ‘1’. Se *JanelaTempo1* retornar ‘0’, a função *JanelaTempo1* é chamada para calcular se a rota de $aleVeic2$ contida na estrutura solução $sViz1$ respeita os horários de atendimento de todos os clientes. Se algum horário de atendimento não é respeitado, a função *JanelaTempo1* retorna ‘1’ e o que faz a função *GeraVizinho1* também retornar ‘1’. Se *JanelaTempo1* retornar ‘0’, distância total percorrida por $aleVeic1$ contida na estrutura solução $sViz1$ é calculada utilizando a função *CalculaDistTotal1*, seguido pelo cálculo da distância

total percorrida por $aleVeic2$ contido na estrutura solução $sViz1$ utilizando a função *CalculaDistTotal1.* Uma vez calculadas as distâncias a função *GeraVizinho1* retorna ‘0’.

4.5. Movimentos

As funções de movimentos têm como objetivo realizar uma mudança específica na solução e com isso criar uma solução vizinha.

A função *Movimento1* é responsável pela troca da posição de atendimento entre dois clientes do mesmo veículo e recebe a estrutura de solução $sViz2$ e o veículo $aleVeic1$. O pseudocódigo da função *Movimento1* é exibido na Figura 17

Figura 17 - Pseudocódigo Movimento1

```
1  INICIO
2  SE (sViz2.NCC[aleVeic1] > 1)
3        aleCliente1 = rand() % sViz2.NCC[aleVeic1]
4        Aux = aleCliente1
5        aleCliente2 = aleCliente1
6        ENQUANTO (aleCliente2 == aleCliente1)
7              aleCliente2 = rand() % sViz2.NCC[aleVeic1]
8        FIM-ENQUANTO
9        sViz2.matriz[aleVeic1][aleCliente1] = sViz2.matriz[aleVeic1][aleCliente2];
10       sViz2.matriz[aleVeic1][aleCliente2] = Aux;
11       retorna 0
12 FIM-SE
13 SE NÃO
14       retorna 1
15 FIM-SE-NÃO
16 FIM DA FUNÇÃO
```

Fonte: Do Autor

Primeiramente a função verifica se $aleVeic1$ possui mais de um cliente em sua rota de atendimento. Se $aleVeic1$ possui mais de um cliente, a função escolhe um cliente aleatório $aleCliente1$ da rota de $aleVeic1$. Depois, a função escolhe um novo cliente $aleCliente2$ diferente do primeiro cliente $aleCliente1$ escolhido anteriormente. Por fim a função realiza a troca da posição de atendimento dos clientes na rota e a função retorna ‘0’. Caso $aleVeic1$ possuir apenas um cliente ou menos, a função retorna ‘1’.

A função *Movimento2*, é responsável pela troca de dois clientes entre veículos e recebe a estrutura de solução $sViz2$, o veículo $aleVeic1$ e o veículo $aleVeic2$. O

pseudocódigo da função *Movimento2* é exibido na Figura 18

Figura 18 - Pseudocódigo Movimento2

```
1  INICIO
2  SE(sViz2.NCC[aleVeic1] != 0)
3          aleCliente1 = rand() % sViz2.NCC[aleVeic1]
4          SE(sViz2.NCC[aleVeic2] != 0)
5                  aleCliente2 = rand() % sViz2.NCC[aleVeic2]
6          FIM-SE
7          SENÃO
8                  retorna 1
9          FIM-SENÃO
10 SENÃO
11 retorna 1
12 FIM-SENÃO
13 acesso = restricaoAcesso(aleCliente1, aleVeic2)
14 SE(acesso = 1)
15     retorna 1
16 FIM-SE
17 acesso = restricaoAcesso(aleCliente2, aleVeic1)
18 SE(acesso == 1)
19     retorna 1
20 FIM-SE
21 retiraCargaCliente2(aleVeic1, aleCliente1)
22 retiraCargaCliente2(aleVeic2, aleCliente2)
23 alocado = alocaCliente(aleVeic1, aleCliente2)
24 SE( alocado == 1)
25     retorna 1
26 FIM-SE
27 alocado = alocaCliente(aleVeic2, aleCliente1)
28 SE( alocado == 1)
29     retorna 1
30 FIM-SE
31 Cliente2 = sViz2.matriz[aleVeic2][aleCliente2]
32 Cliente1 = sViz2.matriz[aleVeic1][aleCliente1]
33 sViz2.matriz[aleVeic1][aleCliente1] = Cliente2
34 sViz2.matriz[aleVeic2][aleCliente2] = Cliente1
35 retorna 0
36 FIM DA FUNÇÃO
```

Fonte: Do Autor

Primeiramente a função verifica se $aleVeic1$ possui ao menos um cliente em sua rota de atendimento. Se $aleVeic1$ possui mais de um cliente, a função escolhe um cliente aleatório $aleCliente1$ da rota de $aleVeic1$. Depois é verificado se $aleVeic2$ possui ao menos um cliente em sua rota de atendimento. Se $aleVeic2$ possui mais

de um cliente, a função escolhe um cliente aleatório $aleCliente2$ da rota de $aleVeic2$. Se $aleVeic1$ ou $aleVeic2$ não possuir clientes em sua rota de atendimento, a função retorna ‘1’. A função *restriçãoAcesso* é então utilizada para verificar se $aleCliente1$ pode ser atendido por $aleVeic2$ e se $aleCliente2$ pode ser atendido por $aleVeic1$. Se o cliente não pode ser atendido por aquele tipo de veículo a função retorna ‘1’, caso contrário, retorna ‘0’. Caso a função retorne ‘1’ quando verificando algum dos dois veículos, a função *Movimento2* retorna ‘1’. Se passar pela verificação, a função *retiraCargaCliente2* é utilizada para retirar a carga do $aleCliente1$ do $aleVeic1$ e retirar a carga do $aleCliente2$ do $aleVeic2$. Após a retirada, função *alocaCliente* aloca a carga do $aleCliente1$ no $aleVeic2$ e a carga do $aleCliente2$ no $aleVeic1$. Caso a alocação do cliente em seu novo veículo tenha sucesso, a função *alocaCliente* retorna ‘0’, caso contrário, retorna ‘1’. Caso a função não obtenha sucesso em alocar algum dos clientes, a função *Movimento2* retorna ‘1’. Se ambos os clientes conseguem ser alocados em seus novos veículos, é realizada a troca de clientes entre os veículos e a função *Movimento2* retorna ‘0’.

A função *Movimento3*, é responsável pela retirada de um cliente do primeiro veículo e alocação deste cliente no segundo veículo. O pseudocódigo da função *Movimento3* é exibido na Figura 19

Figura 19 - Pseudocódigo Movimento3

```
1  INICIO
2  SE(sViz2.NCC[aleVeic1] != 0)
3      aleCliente1 = rand() % sViz2.NCC[aleVeic1];
4  FIM-SE
5  SENÃO
6      retorna 1
7  FIM-SENÃO
8  acesso = restricaoAcesso(aleVeic2, aleCliente1);
9  SE(acesso == 1)
10     retorna 1
11 FIM-SE
12 retiraCargaCliente2(aleVeic1, aleCliente1);
13 alocado = alocaCliente(aleVeic2, aleCliente1);
14 SE(alocado == 1)
15     retorna 1
16 FIM-SE
17 aleCliente2 = rand() % sViz2.NCC[aleVeic2];
18 PARA(i= sViz2.NCC[aleVeic2]) ATE (aleCliente2) INCREMENTA (1)
19     sViz2.matriz[aleVeic2][i+1] = sViz2.matriz[aleVeic2][i];
```

```
20  FIM-PARA
21  sViz2.matriz[aleVeic2][aleCliente2] = aleCliente1;
22  retorna 0
23  FIM DA FUNÇÃO
```

Fonte: Do Autor

Primeiramente a função verifica se $aleVeic1$ possui ao menos um cliente em sua rota de atendimento. Se $aleVeic1$ possui mais de um cliente, a função escolhe um cliente aleatório $aleCliente1$ da rota de $aleVeic1$. Se $aleVeic1$ não possuir clientes em sua rota de atendimento, a função retorna '1'. A função *restriçãoAcesso* é então utilizada para verificar se $aleCliente1$ pode ser atendido por $aleVeic2$. Se o cliente não pode ser atendido por aquele tipo de veículo a função retorna '1', caso contrário, retorna '0'. Caso a função retorne '1', a função *Movimento3* retorna '1'. Se passar pela verificação, a função *retiraCargaCliente2* é utilizada para retirar a carga do $aleCliente1$ do $aleVeic1$. Após a retirada, função *alocaCliente* aloca a carga do $aleCliente1$ no $aleVeic2$. Caso a alocação do cliente em seu novo veículo tenha sucesso, a função *alocaCliente* retorna '0', caso contrário, retorna '1'. A função não obtenha sucesso em alocar $aleCliente1$, a função *Movimento3* retorna '1'. Se o cliente consegue ser alocados em seu novo veículo, é realizada a colocação de $aleCliente1$ na rota de atendimento do $aleVeic2$e a função *Movimento3* retorna '0'.

As funções de movimento utilizadas no primeiro nível são similares as funções de movimento utilizadas no segundo nível. A diferença está que no primeiro nível não existe a restrição de acesso.

A função *Movimento11* é responsável pela troca da posição de atendimento entre dois satélites do mesmo veículo e recebe a estrutura de solução $sViz1$ e o veículo $aleVeic1$. Seu pseudocódigo é mostrado na Figura 20.

Figura 20 - Pseudocódigo Movimento11

```
1  INICIO
2  SE (sViz1.NCC[aleVeic1] > 1)
3        aleSatelite1 = rand() % sViz1.NCC[aleVeic1]
4        Aux = aleSatelite1
5        Satelite2 = Satelite1
6        ENQUANTO (Satelite2 == Satelite1)
7              aleCliente2 = rand() % sViz1.NCC[aleVeic1]
8        FIM-ENQUANTO
```

```
9       sViz1.matriz[aleVeic1][aleSatelite1] = sViz1.matriz[aleVeic1][aleSatelite2];
10      sViz1.matriz[aleVeic1][aleSatelite2] = Aux;
11      retorna 0
12  FIM-SE
13  SE NÃO
14      retorna 1
15  FIM-SE-NÃO
16  FIM DA FUNÇÃO
```

Fonte: Do Autor

Primeiramente a função verifica se $aleVeic1$ possui mais de um satélite em sua rota de atendimento. Se $aleVeic1$ possui mais de um satélite, a função escolhe um satélite aleatório $aleSatelite1$ da rota de $aleVeic1$. Depois, a função escolhe um novo satélite $aleSatelite2$ diferente do primeiro satélite $aleSatelite1$ escolhido anteriormente. Por fim a função realiza a troca da posição de atendimento dos satélites na rota e a função retorna '0'. Caso $aleVeic1$ possuir apenas um satélite ou menos, a função retorna '1'.

A função *Movimento21*, é responsável pela troca de dois satélites entre veículos e recebe a estrutura de solução $sViz1$, o veículo $aleVeic1$ e o veículo $aleVeic2$. Seu pseudocódigo é mostrado na Figura 21.

Figura 21 - Pseudocódigo Movimento21

```
1   INICIO
2   SE (sViz1.NCC[aleVeic1] != 0)
3       aleSatelite1 = rand() % sViz1.NCC[aleVeic1]
4       SE (sViz1.NCC[aleVeic2] != 0)
5           aleSatelite2 = rand() % sViz1.NCC[aleVeic2]
6       SE-FIM
7       SENÃO
8           retorna 1
9       FIM-SENÃO
10  FIM-SE
11  SENÃO
12      retorna 1
13  FIM-SENÃO
14  retiraCargaSatelite(aleVeic1, aleSatelite1)
15  retiraCargaSatelite(aleVeic2, aleSatelite2)
16  alocado = alocaSatelite(aleVeic1, aleSatelite2)
17  SE (alocado == 1)
18      retorna 1
19  FIM-SE
20  alocado = alocaSatelite(aleVeic2, aleSatelite1)
```

```
21  SE(alocado == 1)
22      retorna 1
23  FIM-SE
24  Satelite2 = sViz1.matriz[aleVeic2][aleSatelite2]
25  Satelite1 = sViz1.matriz[aleVeic1][aleSatelite1]
26  sViz1.matriz[aleVeic1][aleSatelite1] = Satelite2
27  sViz1.matriz[aleVeic2][aleSatelite2] = Satelite1
28  retorna 0
29  FIM DA FUNÇÃO
```

Fonte: Do Autor

Primeiramente a função verifica se $aleVeic1$ possui ao menos um satélite em sua rota de atendimento. Se $aleVeic1$ possui mais de um satélite, a função escolhe um satélite aleatório $aleSatelite1$ da rota de $aleVeic1$. Depois é verificado se $aleVeic2$ possui ao menos um satélite em sua rota de atendimento. Se $aleVeic2$ possui mais de um satélite, a função escolhe um satélite aleatório $aleSatelite2$ da rota de $aleVeic2$. Se $aleVeic1$ ou $aleVeic2$ não possuir satélites em sua rota de atendimento, a função *Movimento21* retorna '1'. Se passar pela verificação, a função *retiraCargaSatelite* é utilizada para retirar a carga do $aleSatelite1$ do $aleVeic1$ e retirar a carga do $aleSatelite2$ do $aleVeic2$. Após a retirada, a função *alocaSatelite* aloca a carga do $aleSatelite1$ no $aleVeic2$ e a carga do $aleSatelite2$ no $aleVeic1$. Caso a alocação do satélite em seu novo veículo tenha sucesso, a função *alocaSatelite* retorna '0', caso contrário, retorna '1'. Caso a função não obtenha sucesso em alocar algum dos satélites, a função *Movimento2* retorna '1'. Se ambos os satélites conseguem ser alocados em seus novos veículos, é realizada a troca de satélites entre os veículos e a função *Movimento2* retorna '0'.

A função *Movimento31*, é responsável pela retirada de um satélite do primeiro veículo e alocação deste satélite no segundo veículo. Seu pseudocódigo é mostrado na Figura 22.

Figura 22 - Pseudocódigo Movimento31

```
1  INICIO
2  SE (sViz1.NCC[aleVeic1] != 0)
3      aleSatelite1 = rand() % sViz1.NCC[aleVeic1]
4  FIM-SE
5  SENÃO
6      retorna 1
7  FIM-SENÃO
8  retiraCargaSatelite(aleVeic1, aleSatelite1);
9  alocado = alocaSatelite(aleVeic2, aleSatelite1);
```

```
10  SE(alocado == 1)
11      retorna 1
12  FIM-SE
13  aleSatelite2 = rand() % sViz1.NCC[aleVeic2];
14  PARA(i= sViz1.NCC[aleVeic2]) ATE (aleCliente2) INCREMENTA (1)
15      sViz1.matriz[aleVeic2][i+1] = sViz1.matriz[aleVeic2][i];
16  FIM-PARA
17  sViz1.matriz[aleVeic2][aleCliente2] = aleSatelite1;
18  CriarVetorVeiculosUsados(sViz1)
19  retorna 0
20  FIM DA FUNÇÃO
```

Fonte: Do Autor

Primeiramente a função verifica se $aleVeic1$ possui ao menos um satélite em sua rota de atendimento. Se $aleVeic1$ possui mais de um satélite, a função escolhe um satélite aleatório $aleSatelite1$ da rota de $aleVeic1$. Se $aleVeic1$ não possuir satélites em sua rota de atendimento, a função Movimento31 retorna '1'. Se passar pela verificação, a função *retiraCargaSatelite* é utilizada para retirar a carga do $aleSatelite1$ do $aleVeic1$. Após a retirada, função *alocaSatelite* aloca a carga do $aleSatelite1$ no $aleVeic2$. Caso a alocação do satélite em seu novo veículo tenha sucesso, a função *alocaSatelite* retorna '0', caso contrário, retorna '1'. A função não obtenha sucesso em alocar $aleSatelite1$, a função *Movimento31* retorna '1'. Se o satélite consegue ser alocados em seu novo veículo, é realizada a colocação de $aleSatelite1$ na rota de atendimento do $aleVeic2$, a função *CriarVetorVeiculosUsados* é chamada para criar um vetor contendo os veículos utilizados e a função *Movimento31* retorna '0'.

4.6. Cálculo das Funções Objetivo

O cálculo do custo total da solução é feito em partes, primeiramente é calculado o custo do primeiro nível, depois será calculado o custo do segundo nível e pôr fim o custo obtido para cada nível será somado para obter o custo total. O cálculo da solução de cada nível é calculado somando os valores fixos de cada um dos veículos usados, com o produto da distância percorrida por cada veículo utilizado com seu respectivo custo variável, como mostrado nas equações.

$$S1 = \sum_{i=1}^{\sigma} C1f_i + \sum_{i=1}^{\sigma} (C1v_i\ d1_i)$$

$$S2 = \sum_{j=1}^{\varphi} C2f_j + \sum_{j=1}^{\varphi} (C2v_j\ d2_j) + \sum_{s=1}^{\omega} \sum_{v=1|stvc_{k\,s}=1}^{\varphi} (Cs_s\ k_v)$$

$$S = S1 + S2$$

S1 representa o custo da solução do primeiro nível, S2 o custo da solução do segundo nível e S o custo total da solução. Cf é o valor do custo fixo do veículo, Cv é o custo por quilometro do veículo e d é a distância percorrida pelo veículo, Cs é o custo de operação por pneu do satélite o qual o veículo pertence, k é a carga do veículo (número de pneus) σ é o número de veículos presentes no primeiro nível e φ é o número de veículos presentes no segundo nível.

O cálculo da função objetivo de primeiro nível é feito pela função *CalculaFO1*. Essa função recebe uma estrutura de solução $sViz1$, e devolve a estrutura solução com o valor da função objetivo calculada. Seu pseudocódigo é mostrado na Figura 23.

Figura 23 - Pseudocódigo CalculaFO1

1	INICIO
2	sViz1.FO = 0
3	PARA(i=0) ENQUANTO (i < numVeiculos1) INCREMENTA (1)
4	sViz1.FO = sViz1.FO + sViz1.usado[i] * veiculo1[i].custoVeiculo + sViz1.usado[i] * sViz1.distTotal[i] * veiculo1[i].custoVariavel
5	FIM-PARA
6	FIM DA FUNCAO

Fonte: Do Autor

Primeiramente a função *CalculaFO1* zera o valor de $sViz1.FO$ para que nenhum valor residual presente interfira no cálculo.

Depois, a função passa por todos os veículos de primeiro nível e, se o veículo é utilizado, soma o valor do custo fixo do veículo e o custo variável do veículo multiplicado pela distância total percorrida por ele. Uma vez terminados os veículos, a função termina devolvendo a estrutura solução $sViz1$ com sua função objetivo calculada.

O cálculo da função objetivo de segundo nível é feito pela função *CalculaFOGeral*. Essa função recebe uma estrutura de solução $sViz2$ e uma estrutura solução $Sol1$ e devolve a estrutura solução com o valor da função objetivo calculada. A estrutura solução de primeiro nível recebida pode variar dependendo do local onde a função *CalculaFOGeral* é chamada, assim, generalizamos o nome dessa estrutura solução para $Sol1$. Seu pseudocódigo é mostrado na Figura 24.

Figura 24- Pseudocódigo CalculaFOGeral

```
1  INICIO
2  sViz2.FO = 0
3  PARA(i=0) ENQUANTO (i < numVeiculos2) INCREMENTA (1)
4      sViz2.FO = sViz2.FO + sViz2.usado[i] * veiculo2[i].custoVeiculo + sViz2.usado[i] *
       sViz2.distTotal[i] * veiculo2[i].custoVariavel + sViz2.usado[i] * sViz2.qtdCargaVeicl[i] *
       satélites[veiculo2[i].satelite].custoCargaDescarga
5  FIM-PARA
6  sViz2.FO = sViz2.FO + Sol1.FO
7  FIM DA FUNCAO
```

Fonte: Do Autor

Primeiramente a função *CalculaFOGeral* zera o valor de $sViz2.FO$ para que nenhum valor residual presente interfira no cálculo.

Depois, a função faz passa por todos os veículos de segundo nível e, se o veículo é utilizado, soma o valor do custo fixo do veículo, o custo variável do veículo multiplicado pela distância total percorrida por ele e o custo do satélite multiplicado pelo número de pneus.

Por fim, a função soma o valor da função objetivo de primeiro nível ao valor da função objetivo de segundo nível.

Uma vez terminados os veículos, a função termina devolvendo a estrutura solução $sViz2$ com sua função objetivo calculada.

5. RESULTADOS E ANÁLISE

5.1. Definição de Parâmetros

Os parâmetros a serem definidos para o algoritmo proposto são, a temperatura inicial do primeiro nível $T1$, a temperatura inicial do segundo nível $T2$, número de iterações do primeiro nível $IT1$, número de iterações do segundo nível $IT2$, coeficiente de resfriamento do primeiro nível $\alpha 1$, coeficiente de resfriamento do segundo nível $\alpha 2$, temperatura de congelamento do primeiro nível $Tc1$ e temperatura de congelamento do segundo nível $Tc2$, o número de *reannealing* do primeiro nível $NR1$ e o número de *reannealing* segundo nível $NR2$.

Para privilegiar instância maiores, onde o CPLEX se mostra menos eficaz, foi utilizado a instância 6 com base para a definição dos parâmetros para o algoritmo. A instância 6 possui um dos maiores números de clientes entre todas as instâncias de teste, porém se encontra na média quando consideramos sua demanda total de atendimento.

Primeiramente os valores de $T1$ e $T2$ foram avaliados. Esses valores, juntamente com o coeficiente de resfriamento, tem um impacto direto no número de ciclos de resfriamento necessários até que se atinja a temperatura de congelamento. Elas também influenciam diretamente o aumento de temperatura durante o ciclo de *reannealing* e indiretamente na aceitação de soluções piores para a fuga de mínimos locais. O parâmetro número de iterações do primeiro nível $IT1$ foi fixado em 150, o número de iterações de segundo nível $IT2$ em 400, os coeficientes de resfriamento de primeiro e segundo nível $\alpha 1$ e $\alpha 2$ em 0,975, as temperaturas de congelamento de primeiro e segundo nível $Tc1$ e $Tc2$ em 0,1 e o número de *reannealing* do primeiro e segundo níveis, $NR1$ e $NR2$, como 1. Esses parâmetros foram definidos empiricamente com base nos testes feitos ao longo do desenvolvimento do algoritmo.

Para compreender a influência das temperaturas no algoritmo iniciamos variando $T2$ enquanto estipulamos um valor para $T1$. A Tabela 1 mostra a temperatura e a média das FO, o tempo de execução para cada $T2$ testado e o desvio da FO em relação ao CPLEX.

Tabela 4 - Teste dos Valores de T2

FO (R$)	Tempo (s)	$T2$	$IT2$	$T1$	$IT1$	Desvio (%)
15.520,00	3.188	100.000	400	100	150	0,01
15.559,77	3.252	90.000	400	100	150	0,01
15.575,58	3.159	80.000	400	100	150	0,02
15.575,97	3.311	70.000	400	100	150	0,02
15.575,97	3.200	50.000	400	100	150	0,02
15.581,00	2.394	30.000	400	100	150	0,02
15.611,00	2.294	20.000	400	100	150	0,02
15.639,63	2.151	10.000	400	100	150	0,02
15.641,04	2.073	6.000	400	100	150	0,02
15.644,28	1.703	1.200	400	100	150	0,02
15.656,00	1.777	1.000	400	100	150	0,02
15.670,15	1.786	700	400	100	150	0,02
15.675,00	1.699	600	400	100	150	0,02
15.704,46	1.992	500	400	100	150	0,02

Fonte: Do Autor

A variação da temperatura $T2$ mostrou que apesar de ter impacto no tempo de execução, sendo que quanto maior a temperatura maior o tempo de execução, esta influência foi relativamente pequena. Um aumento de 19.900% em relação ao menor $T2$ testado implicou em um aumento de apenas 60% no tempo de execução. As diferenças percentuais entre a média da FOs de cada teste e a FO do modelo se mantiveram entre 0,01% e 0,02%.

Para compreender a influência de $T1$ no algoritmo, estipulamos agora um valor a $T2$. A Tabela 5 mostra a média das FO, o tempo de execução para cada $T1$ testado e o desvio da FO em relação ao CPLEX.

Tabela 5 - Teste dos Valores de T1

FO (R$)	Tempo (s)	$T2$	$IT2$	$T1$	$IT1$	Desvio (%)
15.576,97	5.015	50.000	400	2.500	150	0.02
15.579,74	4.993	50.000	400	2.000	150	0.02
15.604,36	4.649	50.000	400	1.500	150	0.02
15.640,04	4.591	50.000	400	1.200	150	0.02
15.644,28	4.481	50.000	400	1.000	150	0.02
15.644,28	4.344	50.000	400	800	150	0.02
15.645,34	3.938	50.000	400	500	150	0.02
15.653,82	3.775	50.000	400	300	150	0.02
15.796,45	3.617	50.000	400	200	150	0.03

Fonte: Do Autor

Assim como $T2$, a influência da variação de $T1$ foi relativamente pequena nos tempos de execução. Um aumento de 1.150% em relação ao menor $T2$ testado implicou em um aumento de 38,7% nos tempos de execução. As diferenças percentuais entre a média da FOs de cada teste e a FO do modelo se mantiveram entre 0,02% e 0,03%.

Agora testamos a influência que iterações $IT1$ e $IT2$ tem sobre os algoritmos. Para isso mantemos os valores de $\alpha 1$, $\alpha 2$, $Tc1$, $Tc2$, $NR1$ e $NR2$ fixados e agora também fixamos os valores de $T1$ e $T2$.

Para avaliar o comportamento do algoritmo quando variamos $IT2$, estipulamos um valor a $IT1$ e variamos os valores de $IT2$. A Tabela 6 mostra a média das FOs, o tempo de execução do algoritmo para cada valor de $IT2$ testado e o desvio da FO em relação ao CPLEX.

Tabela 6 - Teste dos Valores de IT2

FO (R$)	Tempo (s)	$T2$	$IT2$	$T1$	$IT1$	Desvio (%)
15.765,45	1.483	50.000	100	1.000	150	0,03
15.617,93	2.138	50.000	200	1.000	150	0,02
15.602,22	3.243	50.000	300	1.000	150	0,02
15.515,78	4.323	50.000	400	1.000	150	0,01
15.500,04	5.590	50.000	500	1.000	150	0,01
15.478,82	6.822	50.000	600	1.000	150	0,01

Fonte: Do Autor

A variação de $IT2$ mostrou ter uma grande influência no tempo de execução do algoritmo. Um aumento de 500% no valor de $IT2$ em relação ao menor valor de $IT2$ testado resultou no aumento de 360% no tempo de execução. As diferenças percentuais entre a média da FOs de cada teste e a FO do modelo se mantiveram entre 0,01% e 0,03%.

Para compreender a influência de $IT1$ no algoritmo, estipulamos agora um valor a $IT2$. A Tabela 7 mostra a média das FO e o tempo de execução para cada $IT1$ testado.

Tabela 7 - Teste dos Valores de IT1

FO (R$)	Tempo(s)	$T2$	$IT2$	$T1$	$IT1$	Desvio (%)
15.647,24	1.471	50.000	400	1.000	50	0,02
15.637,81	2.925	50.000	400	1.000	100	0,02
15.515,78	4.323	50.000	400	1.000	150	0,01
15.562,35	5.148	50.000	400	1.000	200	0,01

15.551,42	8.611	50.000	400	1.000	300	0,01
15.500,04	11.619	50.000	400	1.000	400	0,01

Fonte: Do Autor

A variação de $IT1$ mostrou ter a maior influência no tempo de execução dentre todos os parâmetros testados. Um aumento de 700% em relação ao menor valor de $IT1$ testado resultou em um aumento de 689,9% no tempo de execução. As diferenças percentuais entre a média das FOs de cada teste e a FO do modelo se mantiveram entre 0,01% e 0,02%.

Podemos atribuir este comportamento do algoritmo em relação aos parâmetros à própria construção do algoritmo e do funcionamento dos ciclos e resfriamento do *Simulated Annealing.* Como a forma de decréscimo das temperaturas é a multiplicação por um coeficiente de resfriamento, o valor decrescido ao fim do ciclo é maior quando as temperaturas estão mais elevadas. Assim quanto maior a temperatura inicial, maior o aumento de temperatura necessário para que haja o aumento do número de ciclos de resfriamento (supondo um coeficiente de resfriamento constante). Por outro lado, os ciclos de iteração sempre decrescem o número de iterações em 1, significando que o acréscimo de uma unidade no número máximo de iterações implica no aumento de 1 iteração a ser realizada durante o ciclo de iterações.

Pelo ponto de vista da construção do algoritmo, por estar compreendido dentro do roteamento de segundo nível, o roteamento de primeiro nível é executado múltiplas vezes durante o algoritmo. Aumentos nos parâmetros de segundo nível podem acarretar um aumento no número de movimentos executados e no número de vezes que o primeiro nível será roteado. Porém, aumentos nos parâmetros de primeiro nível, levam a aumentos no tempo de execução do primeiro nível, que por ser executado múltiplas vezes, tem seu efeito multiplicado quando analisamos o tempo de execução total do algoritmo.

O número de *reannealings* representa o número de vezes que ocorrerá o aumento da temperatura do algoritmo uma vez que ele tenha atingido a temperatura de congelamento. Ele é utilizado principalmente como um método para fugir de mínimos locais. Para exemplificar o comportamento do algoritmo durante dos testes, foi escolhido os dados de uma execução do algoritmo. A Tabela 8 mostra o valor da FO, a temperatura e o ciclo em que uma nova melhor solução foi encontrada.

Tabela 8 - Lista Dos Valores De *FO* Das Melhores Soluções Encontradas Durante A Execução

FO (R$)	Temperatura	Iteração
17527,76	50.000,00	1
17526,95	50.000,00	4
17419,65	50.000,00	150
17319,07	32.512,37	5211
17039,05	32.512,37	5213
16959,37	32.512,37	5233
16958,54	32.512,37	5238
16957,79	32.512,37	5240
16944,48	6.596,903	24074
16921,98	565,9639	53175
16918,67	565,9639	53180
16713,54	565,9639	53183
16712,5	565,9639	53214
16711,75	565,9639	53216
16703,94	565,9639	53224
16703,87	565,9639	53238
16682,86	565,9639	53244
16672,9	245,4378	63281
16657,1	245,4378	63283
16656,93	245,4378	63286
16647,17	245,4378	63291
16527,61	123,8986	71193
16342,28	98,65244	73854
16106,31	98,65244	73959
16105,56	98,65244	73965
16082,47	98,65244	73977
16077,22	98,65244	73981
15981,24	27,12338	89233
15980,52	27,12338	89240
15979,78	27,12338	89247
15964,82	27,12338	89256
15963,9	27,12338	89264
15963,46	27,12338	89266
15963,19	27,12338	89286
15962,44	27,12338	89302
15962	26,4453	89449
15948,22	26,4453	89477
15787,91	26,4453	89554
15778,4	25,13956	90035
15641,49	25,13956	90100
15640,74	25,13956	90109
15639,61	25,13956	90112
15639,25	25,13956	90164
15639,11	25,13956	90173

15637,97	25,13956	90193
15636,87	24,51107	90433
15635,96	24,51107	90434
15627,56	22,71832	91230
15626,01	22,71832	91238
15625,32	22,15036	91651
15618,68	22,15036	91655
15618,68	22,15036	91659
15616,05	22,15036	91671
15615,3	22,15036	91675
15611,58	22,15036	91678
15611,21	22,15036	91686
15608,25	21,59661	91999
15607,88	21,59661	92018
15607,88	21,59661	92021
15602,22	19,51659	93228

Fonte: Do Autor

Nos testes rodados, não foram encontradas novas soluções melhores com temperaturas e número de ciclos maiores que soluções anteriormente encontradas. Isso indica que os ciclos de *reannealing* não impactaram na melhora dos valores de FO, não sendo encontradas novas soluções durante os ciclos de *reannealing.*

Assim, para execução do algoritmo, foram escolhidos os valores dos parâmetros de forma a balancear o tempo de execução e o valor da FO além de contemplar futuras execuções de instâncias maiores ou mais complexas. Como os valores de $T1$ e $T2$ apresentaram as menores influencias nos tempos de execução, não foram adotados nem os maiores valores testado nem os menores valores, sendo assim escolhidos os valores de $T2$ como 50.000, $T1$ como 1.000.

Na escolha dos valores das iterações, que apresentam a maior influência no tempo de execução do algoritmo, foram escolhidos os valores que apresentaram os menores tempo de execução ainda mantendo o desvio percentual no entorno de 0,01%. Assim, foram escolhidos $IT1$ como 150 e $IT2$ como 400. Como o número de *reannealings* não apresentou ganhos em relação a FO, os valores de $NR1$ e $NR2$ foram adotados como 1. Foi decidido que o coeficiente de resfriamento utilizado seria o mesmo adotado nos testes, sendo assim $\alpha 1$ e $\alpha 2$ foram adotados como 0,975. Assim como os coeficientes de resfriamento, foi decidido adotar as temperaturas mesmas temperaturas de

congelamento utilizadas nos testes, assim $Tc1$ e $Tc2$ foram adotados como 0,1.

5.2. Resultados e Análises

O modelo matemático foi resolvido com o solver IBM® ILOG® CPLEX® Versão 12.6 em computador com processador Intel® Xeon® com 8 núcleos e 64 GB de memória RAM. A meta-heurística proposta foi desenvolvida em linguagem C e executada no mesmo computador utilizado para executar o modelo matemático. Foi definido um tempo limite para execução do CPLEX de 28.800 segundos, equivalente a 8 horas. Para obter os resultados da meta-heurística, cada instância foi executada 10 vezes, possibilitando a verificação de sua estabilidade a partir da diferença entre os 10 resultados gerados.

A Tabela 9 exibe os resultados gerados pelo algoritmo e pelo CPLEX para cada instância indicada na coluna Instância.

Tabela 9 - Comparação dos Resultados do CPLEX e Algoritmo

Grupo	Instância	Modelo de Vitorugo e Caliman (2017)					Algoritmo Proposto					Diferença	
		FO (R$)	UB (R$)	LB (R$)	Tempo de execução (s)	GAP (%)	Média (R$)	Melhor (R$)	Pior (R$)	Desv. Pad. Rel. (%)	Tempo de execução (s)	Dif FO (%)	Tempo de execução (s)
1	1	5.790,26			4.571,47	0,0	5.790,26	5.790,26	5.790,26	0,00	8.645,8	0,0	4.074,33
	2	6.806,07			533,89		6.806,07	6.806,07	6.806,07	0,00	8.026,8	0,0	7.492,91
	3	9.479,36	9.479,36	9.362,49	28.800,00	1,23	9.675,256	9.672,93	9.681,27	0,04	5.133,6	0,02	-23.666,40
	4	10.731,33	10.731,33	10.000,83	28.800,00	6,81	11.056,34	11.056,34	11.056,34	0,00	5.387,2	0,03	-23.412,80
	5	12.084,23			28.800,00		12.493,92	12.476,6	12.500,49	0,08	5.045,2	0,03	-23.754,80
	6	15.339,88	15.339,88	11.971,54	28.800,00	21,96	15.607,87	15.463,35	15.764,29	0,71	4.323,4	0,02	-24.476,60
	7	-	-	-	28.800,00	-	17.717,11	17.525,48	17.837,65	0,76	5.538,2	-	-23.261,80
2	8	11.327,27			1.361,36		11.327,27	11.327,27	11.327,27	0,00	3.912,6	0,0	2.551,24
	9	12.755,26			27.967,08		12.755,26	12.755,26	12.755,26	0,00	4.155,2	0,0	-23.811,88
	10	16.298,93	16.298,93	14.584,17	28.800,00	10,52	16.298,93	16.298,93	16.298,93	0,00	3.857,8	0,0	-24.942,20
	11	17.704,17	17.704,17	15.166,97	28.800,00	14,33	17.704,17	17.704,17	17.704,17	0,00	3.976,8	0,0	-24.823,20
	12	20.113,91	20.113,91	16.436,25	28.800,00	18,28	19.511,55	19.278,88	19.703,30	1,00	3.483,4	-0,03	-25.316,60
	13	23.762,47	23.762,47	18.653,39	28.800,00	21,50	21.908,21	21.711,42	22.162,34	0,87	3.418,0	-0,08	-25.382,00
	14	26.451,45	26.451,45	21.137,52	28.800,00	20,09	25.089,42	24.802,13	25.405,72	0,95	4.658,6	-0,05	-24.141,40

Fonte: Do Autor

As colunas de 2 a 6 contêm informações do desempenho do CPLEX. *FO* exibe a função objetivo encontrada pelo modelo; *LB* e *UB* são, respectivamente, *Lower Bound* e *Upper Bound* da solução e não são exibidos quando o CPLEX encontra a solução ótima, pois são iguais à própria *FO*. A coluna *GAP* é maior que 0 quando o modelo não encontra o ótimo, sendo calculado como $(UB - LB)/UB$; *Tempo de Execução*. (s), coluna 4, mostra o tempo que o CPLEX levou para encontrar a solução ótima, porém limitado a 28800 segundos (8 horas).

As colunas de 7 a 11 contêm informações sobre o desempenho do algoritmo proposto. *Média* é a média da função objetivo das 10 execuções da instância; *Desv. Pad. Rel* indica o desvio padrão relativo da Função objetivo, calculado como a divisão do desvio padrão da FO pela média. A coluna *Melhor* apresenta o menor valor encontrado nas 10 execuções; *Tempo de Execução* mostra (em segundos) a quantidade de tempo média que o algoritmo demandou em toda a execução.

As colunas 12 e 13 mostram comparações entre os resultados do algoritmo proposto e do CPLEX. *Dif. FO* é a diferença percentual média entre as funções objetivo, calculado como: a diferença entre a FO Média do algoritmo e a FO do CPLEX, dividida pela FO do CPLEX. Portanto, quando o valor é positivo, o CPLEX teve melhor desempenho, caso contrário, o algoritmo foi melhor, e quando for zero, ambas tiveram o mesmo resultado. O mesmo raciocínio se aplica à *Dif. Tempo Exec.(s)*, que indica a diferença em segundos entre o tempo de execução do algoritmo proposto e do CPLEX.

Analisando os resultados encontrados obtidos pelo algoritmo e comparando com os resultados obtidos pelo CPLEX, o algoritmo obteve valores FO idênticos ao CPLEX em 4 das 5 instâncias onde o CPLEX obteve uma solução ótima. Em 6 das 14 instâncias não houve diferença entre a FO média do algoritmo e a FO do CPLEX, sendo que em 3 das 14 instâncias o algoritmo obteve valores de FO melhores, observando que o resultado do algoritmo está contido ente o *UB* e o *LB* do CPLEX.

Observa-se que o algoritmo é estável, com desvio padrão relativo médio de 0,32% e máximo de 1,0%, Instância 12. Algumas instâncias apresentaram resultados piores, 5 das 14 instâncias obtiveram valores médios de FO superiores aos encontrados pelo CPLEX, porém a diferença percentual se

manteve inferior a 0,05%.

O algoritmo obteve ganhos no tempo de execução em 11 das 14 instâncias, sendo que as 3 instâncias onde o algoritmo não superou o CPLEX são as menores entre todas as instâncias testadas. Um dos fatores que contribuem para isso é a necessidade de o algoritmo passar por todos os ciclos de resfriamento, iterações, e busca de soluções estipulados mesmo caso já tenha encontrado a solução ótima do problema. O tempo médio de execução do algoritmo foi de 1,4 horas, porém vale ressaltar que o algoritmo obteve um ganho médio de 6,6 horas nas instâncias em que teve tempos de execução menores em relação ao CPLEX.

Os resultados mostraram que o tempo de execução do algoritmo decresce conforme as instâncias se tornam maiores e mais restritivas. Isso ocorre pois, nas instâncias menores e menos restritas, o número de movimentos bem-sucedidos aumenta, fazendo com que o algoritmo tenha de passar por um número maior de rotinas antes de aceitar ou rejeitar uma possível solução. Vale ressaltar que um movimento bem-sucedido de segundo nível pode acarretar todo um novo roteamento do primeiro nível, o que eleva ainda mais o tempo de execução do algoritmo. Nas instâncias maiores e mais restritivas ocorre o contrário, o número de possíveis soluções diminui e devido a violações das restrições, o algoritmo consegue descartar possíveis soluções sem ter de passar por todas as rotinas de aceitação de possível solução.

Este comportamento quanto ao tempo de execução ocorre devido ao número de veículos em ambos níveis serem os mesmos nas instâncias de 1 a 6 e 8 a 13, isso faz que a diferença entre capacidade de demanda e a capacidade de carga disponível seja grande em instâncias com menor número de clientes ou menor demanda, como as instâncias 1, 2, 8 e 9.

A Tabela 10 mostra as distâncias percorridas em cada nível, o número de veículos utilizados por nível e o número de satélites utilizados.

Tabela 10 - Resultados das Instâncias

Grupo	Instância	Dist, 1°Nivel (Km)	Dist, 2°Nivel (Km)	N° Veic 1° Nivel	N° Veic 2° Nivel	N° de Satelites Utilizados
1	1	96,61	76,88	2	3	3
	2	108,20	94,76	2	3	3
	3	520,48	420,50	4	5	5

	4	406,47	759,78	3	5	5
	5	447,98	836,18	4	6	5
	6	748,92	1325,94	4	10	8
	7	844,82	2244,63	4	11	6
	8	689,80	579,63	4	6	6
	9	694,25	586,64	4	6	6
	10	1146,48	751,62	5	7	7
2	11	1150,22	769,00	5	8	8
	12	735,88	1484,42	5	9	7
	13	745,78	1896,83	5	12	7
	14	738,06	2911,23	5	14	8

Fonte: Do Autor

Comparando os pares de instâncias 1 e 8, 2 e 9, 3 e 10, 4 e 11, 5 e 12, 6 e 13, 7 e 14 temos que a instância pertencente ao grupo 2 sempre possui um maior valor de FO. Apesar de ambos possuírem o mesmo número de cientes, satélites, número de veículos por tipo no primeiro nível e segundo nível, o aumento da demanda por cliente fez necessário o uso de um número maior de veículos em cada nível. Assim, o aumento das demandas dos clientes faz com o que diversas rotas que eram utilizadas no grupo 1 tenham suas restrições de capacidade violada, tornando necessária a utilização de mais veículos. O aumento do número de veículos utilizados nas instâncias se mostrou proporcional ao aumento da demanda, sendo que um aumento na demanda por cliente ocasionou um aumento no número de veículos utilizados em cada nível. A exceção foram as Instâncias 1 e 8, 2 e 9, onde o aumento de 50% na demanda dobrou a quantidade de veículos necessários. Esse grande aumento no número de veículos se deve ao pequeno número de clientes nesses cenários, e a grande distância entre os clientes, o que torna mais vantajoso aumentar o número de veículos utilizados ao invés de colocar mais clientes na rota do veículo.

Para observar o comportamento de cada nível do algoritmo proposto durante a sua execução, obteve-se a temperatura em que se encontrava o nível no momento em que ele encontrava uma nova melhor solução. No **Apêndice 1** estão traçados os gráficos do comportamento de cada nível do algoritmo para cada instância de teste, relacionando a Temperatura no momento que a solução foi encontrada e o valor da solução encontrada. Como cada instância foi executada 10 vezes, os dados escolhidos para traçar o gráfico de cada instância foi a que apresentou a menor FO, ou no caso de apresentarem FOs iguais, a

escolha foi feita e maneira aleatória. Como cada instância executa o roteamento de primeiro nível múltiplas vezes, os dados escolhidos para traçar o gráfico pertencem ao roteamento de primeiro nível que gerou a melhor solução da instância. Não foi observado nenhum padrão para o comportamento das soluções em relação a temperatura, provavelmente devido ao fator aleatório presente na construção dos resultados.

6. CONSIDERAÇÕES FINAIS

O problema 2E-CVRP tem se mostrado como relevante e atual por lidar com logística urbana e questões ambientais, o que tem despertado o interesse na literatura, Apesar de estar recebendo cada vez mais atenção ainda existem poucos estudos sobre ele, menos ainda sobre suas variantes, o que o torna um excelente nicho para pesquisa e publicações, O trabalho de Vitorugo e Caliman (2017) se mostrou uma boa base para a pesquisa, pois identificou uma variante e a resolveu por meio de modelo matemático, Porém, devido à complexidade do problema, somente o modelo não é o suficiente pois não suporta instâncias muito grandes e apresenta tempos de processamento elevados, o que o torna inviável para uso em uma situação real de planejamento.

Esse trabalho propôs um algoritmo baseado na meta-heurística *Simulated Annealing (SA)* aplicado ao problema de logística reversa de pneus inservíveis estruturado em dois níveis, visando minimizar os custos de transporte entre os clientes e a unidade recicladora, considerando múltiplos níveis, janela de tempo, restrição de acesso e frota heterogenia. Este problema Two-echelon Capacitated Vehicle Routing Problem with Heterogeneous Fleet, Site Dependence with Time Windows para a logística reversa de pneus inservíveis estruturada em dois níveis foi proposto por Vitorugo e Caliman (2017), que propôs um modelo matemático para a solução do mesmo.

Este modelo proposto por Vitorugo e Caliman (2017) encontrou soluções ótimas para as menores instâncias de teste, com 6 e 8 clientes, e uma instância com 14 clientes, levando até 1,3 horas de tempo de processamento. Para instâncias com 10 ou mais clientes, somente uma instância resultou em solução ótima, sendo que as outras instâncias apresentaram GAPs de até 21,96%, levando a

tempos de processamento superiores a 6,4 horas. Mesmo que o modelo matemático consiga superar o tempo de processamento do algoritmo em instâncias pequenas, esse tempo se degrada rapidamente com o crescimento do número de clientes o que o torna inviável seu uso quando projetamos os crescimentos da planta recicladora e da demanda. Tais tempos de processamento que podem superar o horário de um expediente de trabalho não são compatíveis com uma ferramenta de planejamento logístico, sendo mais indicado o uso de uma meta-heurística para obtenção de resultados em tempos viáveis.

Considerando os ganhos nos tempos de processamento, que em muitas instâncias se mostrou superior a 7 horas, e a pequena deterioração da qualidade da solução, inferior a 0,05%, o uso do algoritmo se mostra atraente. Portanto, o algoritmo se mostra uma ferramenta viável para o roteamento de veículos em dois níveis.

O algoritmo proposto pode ser aplicado não só a logística reversa de pneus inservíveis como também para outros produtos como granéis, onde não é necessária uma arrumação de carga. Sua aplicação vai além do recolhimento de cargas, podendo também ser utilizado para o recolhimento e distribuição de mercadorias a partir de uma indústria, ou matriz, por meio de múltiplos centros de distribuição.

Como recomendação de trabalhos futuros, propõe-se o desenvolvimento novos algoritmos baseados em diferentes meta-heurísticas para buscar melhores tempos de execução sem sacrificar a qualidade dos resultados e comparar com os resultados do algoritmo utilizado nesta dissertação.

REFERENCICAS

BALDACCI, R,; MINGOZZI, A,; ROBERTI, R, An Exact Algorithm for the Two-Echelon Capacitated Vehicle Routing Problems, **Operations Research**, 61, p, 298-314, 2013,

BRASIL, Lei Nº 12,305, de 2 de agosto de 2010, Institui a Política Nacional de Resíduos Sólidos; altera a Lei no 9,605, de 12 de fevereiro de 1998; e dá outras providências, **Diário Oficial da União**, Brasília, 2010, Disponível em <http://www,planalto,gov,br/ccivil_03/_ato2007-2010/2010/lei/l12305,htm>, Acesso em: 15 mai, 2017,

BREUNIG, U,; SCHMID, V,; HARTL, R, F; VIDAL, T, A fast large neighbourhood based heuristic for the two-echelon vehicle routing problem, **Computers and Operation Research**, 76, p, 208–225, 2016,

CHRISTOFIDES, Nicos; EILON, Samuel, **An algorithm for the vehicle-dispatching problem**, Or, p, 309-318, 1969,

COELHO, L,C, Série Pesquisa Operacional – Problema de Roteamento de Veículos, Disponível em: <http://www,logisticadescomplicada,com/serie-pesquisa-operacional-%E2%80%93-problema-de-roteamento-de-veiculos/>, Acesso em: 09 de jul, 2017,

CONAMA – CONSELHO NACIONAL DO MEIO AMBIENTE, Resolução n° 258, de 26 de agosto de 1999, **Diário Oficial da União**, 2 dez, 1999,

CONAMA – CONSELHO NACIONAL DO MEIO AMBIENTE, Resolução n° 301, de 21 de março de 2002, **Diário Oficial da União**, 28 ago, 2003,

CONAMA – CONSELHO NACIONAL DO MEIO AMBIENTE, Resolução N°416, de 30 de setembro de 2009, **Diário Oficial da União**, 1 dez, 2009,

CRAINIC, T,G,; MANCINI, S,; PERBOLI, G,; TADEI, R, GRASP with path relinking for the two-echelon vehicle routing problem, **CIRRELT**, 45, 2012,

CRAINIC, T, G,; MANCINI, S,; PERBOLI, G,; TADEI, R, Clustering-Based Heuristic for the Two-echelon Vehicle Routing Problem, **CIRRELT,** 46, 2008,

CRAINIC, T,G,; MANCINI, S,; PERBOLI, G,; TADEI, R, Multi-start heuristics for the two- echelon vehicle routing problem, **CIRRELT**, 30, 2010**,**

CRAINIC, T, G,; RICCIARDI, N,; STORCHI, G, Models for Evaluating and Planning City Logistics Systems, **CIRRELT**, 11, 2009,

CUDA, R,; GUASTAROBA, G,; SPERANZA, M, G, A survey on two-echelon routing problems, **Computers & Operations research**, 55, p, 185-199, 2015,

DELLAERT, N,; SARIDARQ, F, D,; WOENSEL, T, V,; CRAINIC, T, G, Branch & Price Based Algorithms for the Two-Echelon Vehicle Routing Problem with Time Windows,**CIRRELT**, 45, 2016,

ESMAILI, M,; SAHRAEIAN, R, A new Bi-objective model for a Two-echelon Capacitated Vehicle Routing Problem for Perishable Products with the Environmental Factor, **IJE Transactions**, 30, p, 523-531, 2017,

FELIU, J, G,; PERBOLI, G,; TADEI, R,; VIGO, D, The Two-echelon Capacitated Vehicle Routing Problem, Technical **Report, Control and Computer Department Politecnico de Torino**, University of Bologna, Bologna, Italy, 2007,

FRAGA, K, **Modelo matemático para planejamento da logística reversa de pneus inservíveis com base no modelo *Two-Echelon Capacitated Vehicle Routing Problem*,** 96 f, Dissertação (Mestrado em Engenharia Civil), Programa de Pós-Graduação em Engenharia Civil, Universidade Federal do Espirito Santo, Vitória-ES, 2016,

GRANGIER, P,; GENDREAU, M,; LEHUÉDÉ, F,; ROUSSEAU, L, M, An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization, **European Journal of Operational Research**, 50, p, 80-91, 2016,

HEMMELMAYR, V, C,; CORDEAU, J, F,; CRAINIC, T, G, An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics, **Computers & Operations Research**, 39, p, 3215–28, 2012,

JABALI, O,, VAN WOENSEL, T,, KOK, A,G, Analysis of Travel Times and CO2 Emissions in Time-Dependent Vehicle Routing, **Production and Operations Management**, 21, p, 1060-1074, 2012,

JEPSEN, M,; ROPKE, S,; SPOORENDONK, S, A branch-and-cut algorithm for the symmetric two-echelon capacitated vehicle routing problem, **Transportation**

Science, 47, p, 23–37, 2013,

JIE, Wanchen et al, The two-echelon capacitated electric vehicle routing problem with battery swapping stations: Formulation and efficient methodology, European Journal of Operational Research, v, 272, n, 3, p, 879-904, 2019,

KIRKPATRICK, Scott; GELATT, C. Daniel; VECCHI, Mario P. Optimization by simulated annealing. science, v. 220, n. 4598, p. 671-680, 1983

KITJACHAROENCHAI, Patchara; MIN, Byung-Cheol; LEE, Seokcheon. Two echelon vehicle routing problem with drones in last mile delivery. **International Journal of Production Economics**, v. 225, p. 107598, 2020.

MANCINI, Simona, Multi-Echelon Distribution Systems in City Logistics, **European Transport**, 54, n, 2, 2013,

MEIHUA, W,; XUHONG, T,; SHAN, C,; SHUMIN, W, Hybrid ant colony optimization algorithm for two echelon vehicle routing problem, **Procedia Engineering**, 15, p, 3361 – 3365, 2011,

MÜHLBAUER, Ferdinand; FONTAINE, Pirmin. A parallelised large neighbourhood search heuristic for the asymmetric two-echelon vehicle routing problem with swap containers for cargo-bicycles. **European Journal of Operational Research**, v. 289, n. 2, p. 742-757, 2021

PERBOLI, G,; TADEI, R,; VIGO, D, The Two-Echelon Capacitated Vehicle Routing Problems, **CIRRELT**, 55, 2008,

PERBOLI, G,; TADEI, R,; VIGO, D, The Two-Echelon Capacitated Vehicle Routing Problems: Models and Math-Based Heuristics, **Transportation Science**, 45, n, 3, p, 364-380, 2011,

RECICLANIP. **Pontos de coleta no Brasil**. Disponível em:< http://www.reciclanip.org.br/v3/pontos-coleta/brasil> Acesso em: 01 jul. 2017.

REDI, A. A. N. et al. A Simulated Annealing Algorithm for Solving Two-Echelon Vehicle Routing Problem with Locker Facilities. Algorithms, v. 13, n. 9, p. 218, 2020

SANTOS, F, A,; DA CUNHA, A, S,; MATEUS, G, R, Branch-and-price algorithms

for the Two-Echelon Capacitated Vehicle Routing Problem, **Optimization Letters,** 7, p, 1537–1547, 2013,

SONG, L,, GU, H,; HUANG, H, A lower bound for the adaptive two-echelon capacitated vehicle routing problem, **Springer Science + Business Media New York**, 33, p, 1145-1167**,** 2017,

SOYSAL, M,, BLOEMHOF-RUWAARD, J, M,; BEKTAS, T, The time-dependent Two-Echelon Capacitated Vehicle Routing Problem With Environmental Considerations, **International Journal of Production Economics**, 164, p, 366-378**,** 2015,

VELOSO, Z, M, F, **Ciclo de Vida dos Pneus**, Disponível em: <http://www.inmetro.gov.br/painelsetorial/palestras/Zilda-Maria-Faria-Veloso-Ciclo-Vida-Pneus.pdf>, Acesso em: 20 jul, 2017,

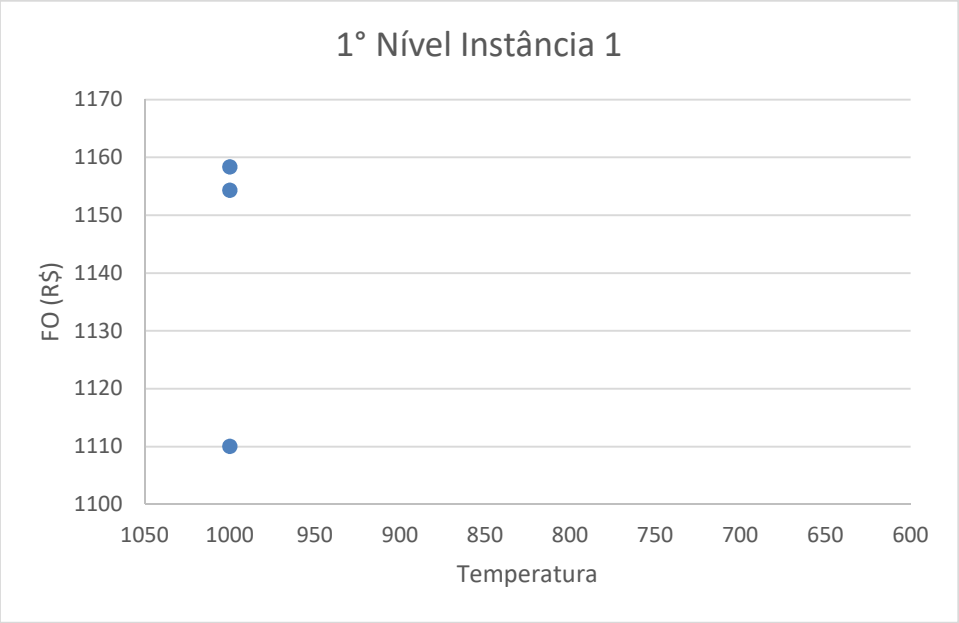
VITORUGO, L, R, e CALIMAN, R,L, Planejamento da logística reversa de pneus inservíveis para atender uma planta geradora de energia com base no modelo two-echelon capacitated vehicle routing problem with time windows**,**, Projeto de Graduação (Engenharia de Produção)- Universidade Federal do Espírito Santo, Espírito Santo, 2017

ZHENG-YANG, Z,; WEI-SHENG, X,; ZHI-YU, X,; WEI-HUI, S, A Hybrid GRASP+VND Heuristic for the Two-Echelon Vehicle Routing Problem Arising in City Logistics, **Mathematical Problems in Engineering**, 2014, p, 11, 2014,

WANG, K; LAN, S,; ZHAO, Y, A genetic-algorithm-based approach to the two-echelon capacitated vehicle routing problem with stochastic demands in logistics service, **Journal of the Operational Research Society,** 2017,

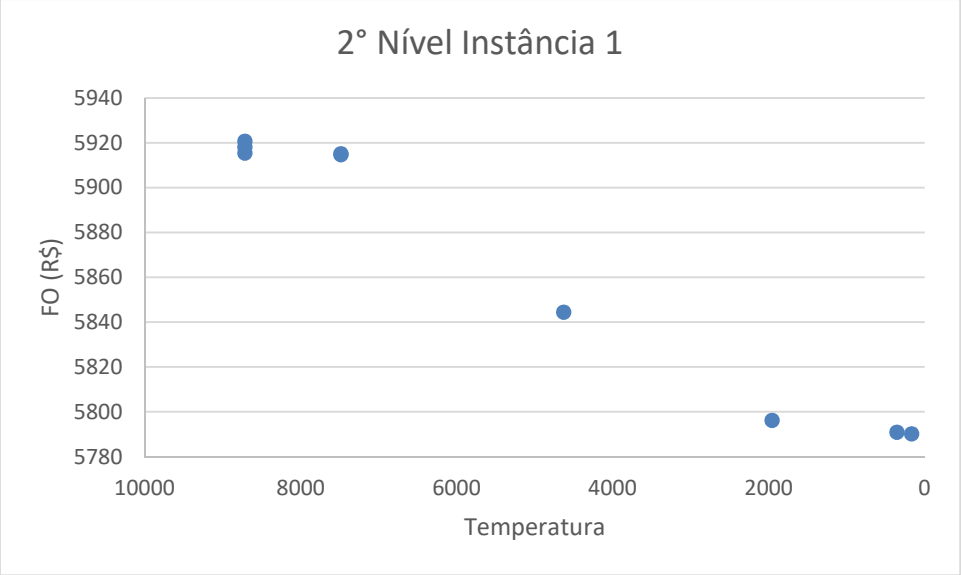
APENDICE 1

Figura 25 - Gráfico Instância 1 Nível 1



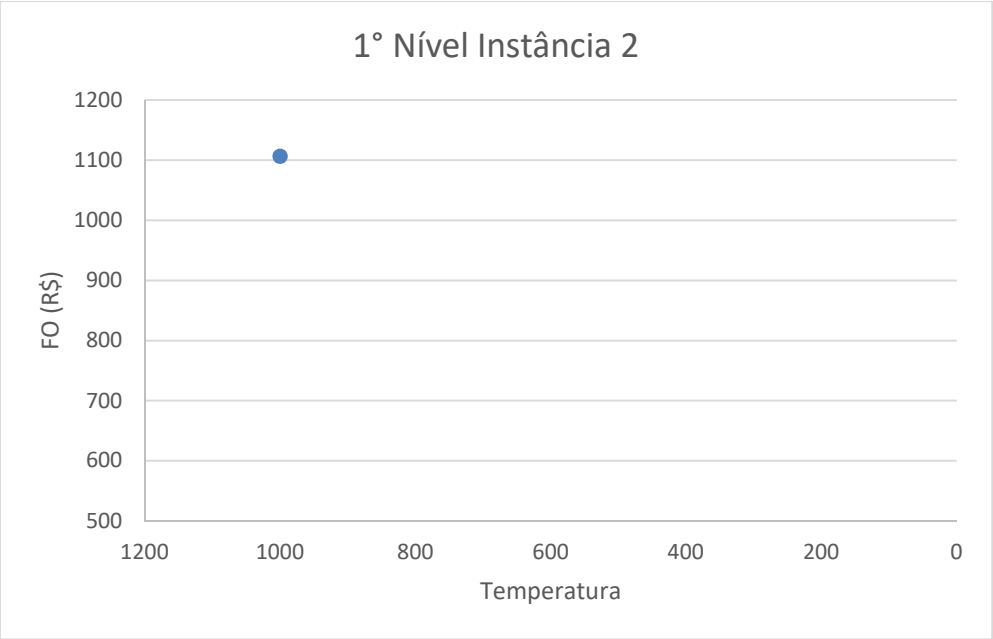
Fonte: Do Autor

Figura 26 - Gráfico Instância 1 Nível 2



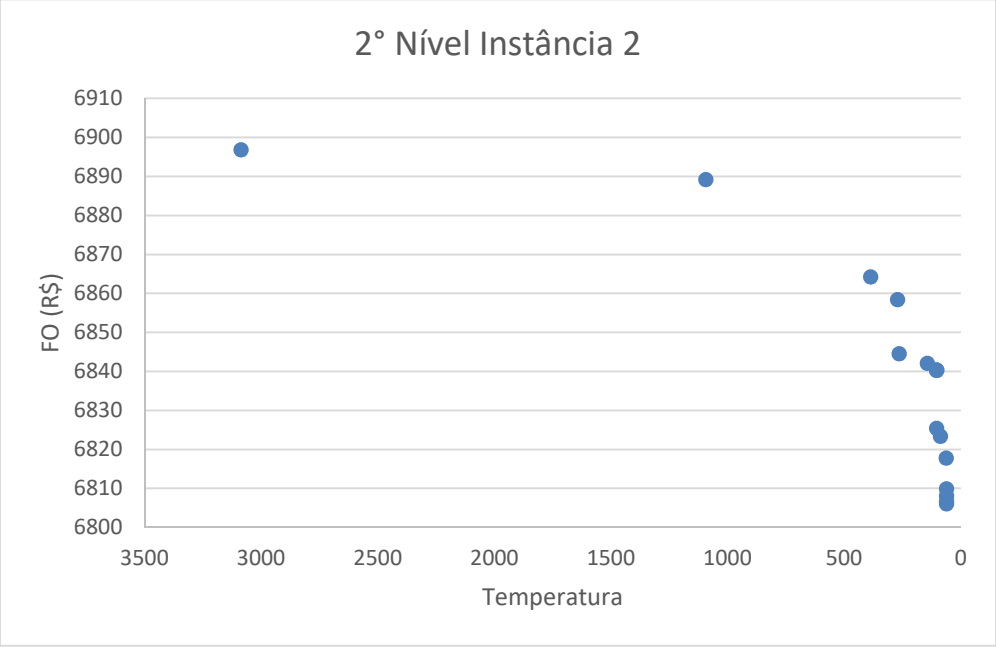
Fonte: Do Autor

Figura 27 - Gráfico Instância 2 Nível 1



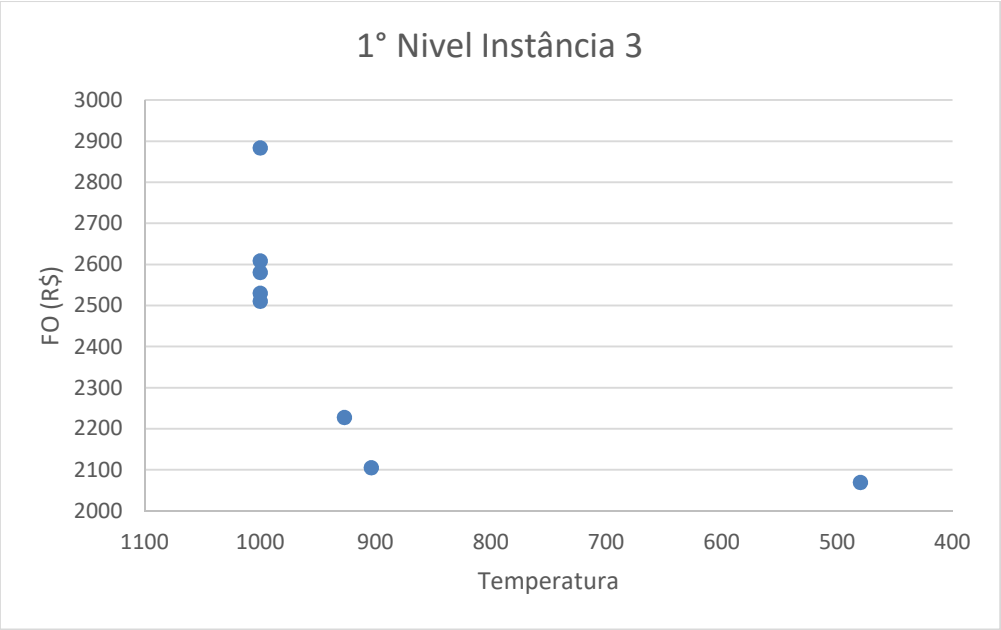
Fonte: Do Autor

Figura 28 - Gráfico Instância 2 Nível 2



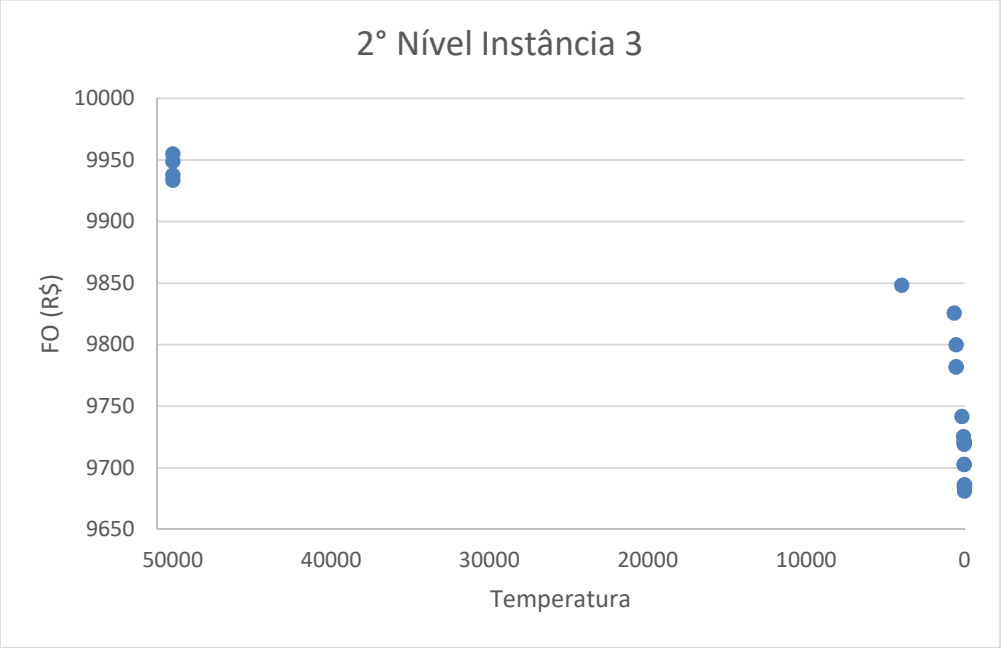
Fonte: Do Autor

Figura 29 - Gráfico Instância 3 Nível 1



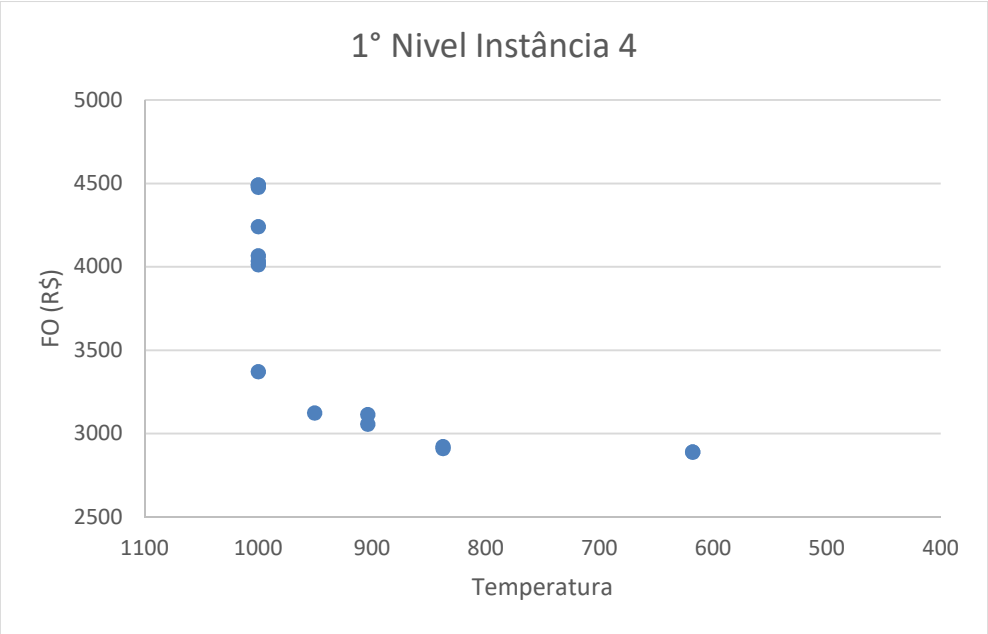
Fonte: Do Autor

Figura 30 - Gráfico Instância 3 Nível 2



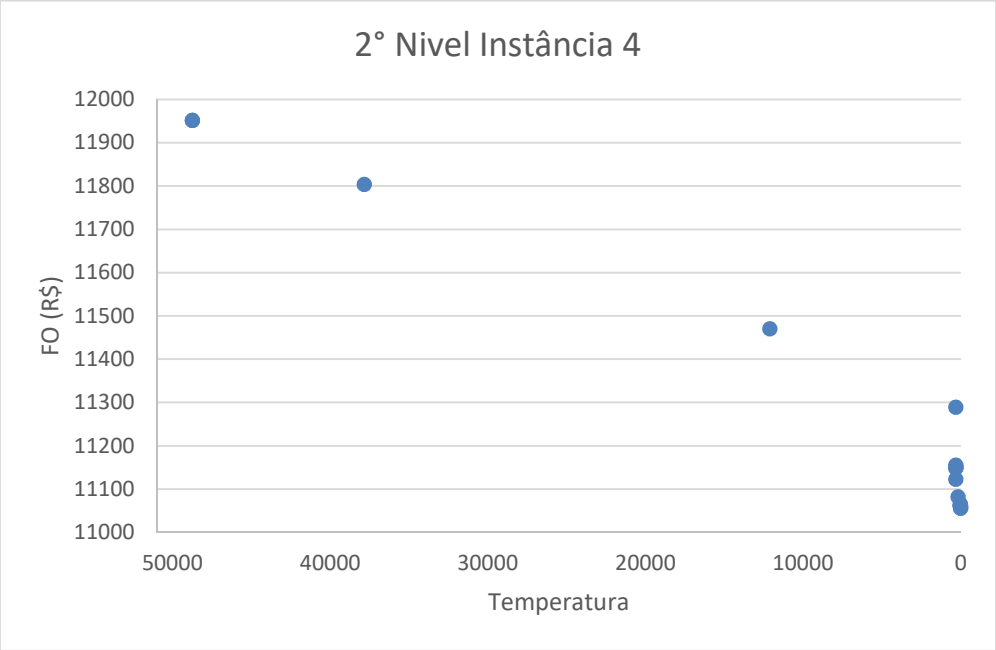
Fonte: Do Autor

Figura 31 - Gráfico Instância 4 Nível 1



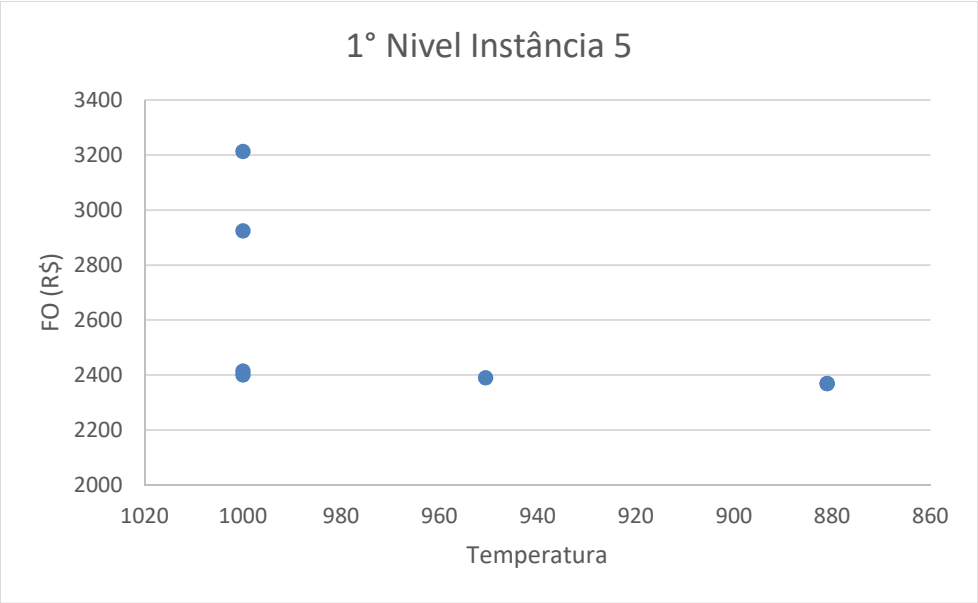
Fonte: Do Autor

Figura 32 - Gráfico Instância 4 Nível 2



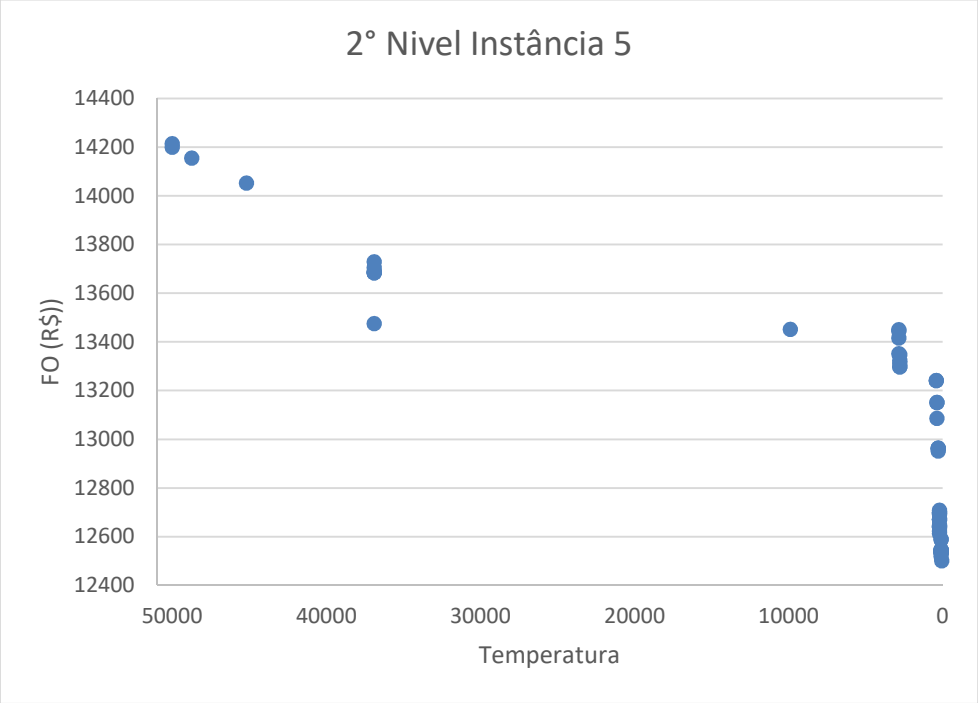
Fonte: Do Autor

Figura 33 - Gráfico Instância 5 Nível 1



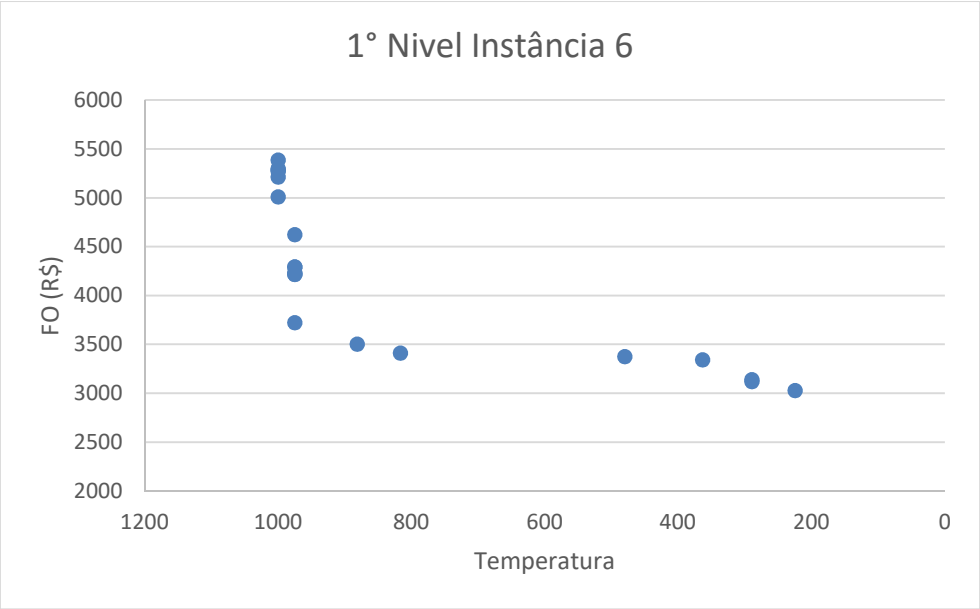
Fonte: Do Autor

Figura 34 - Gráfico Instância 5 Nível 2



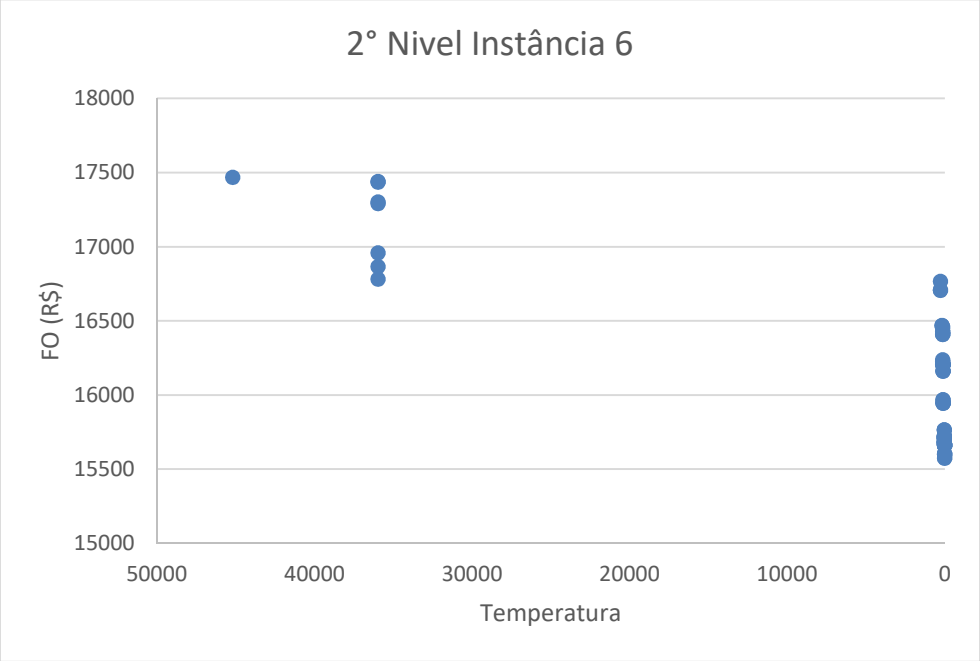
Fonte: Do Autor

Figura 35 - Gráfico Instância 6 Nível 1



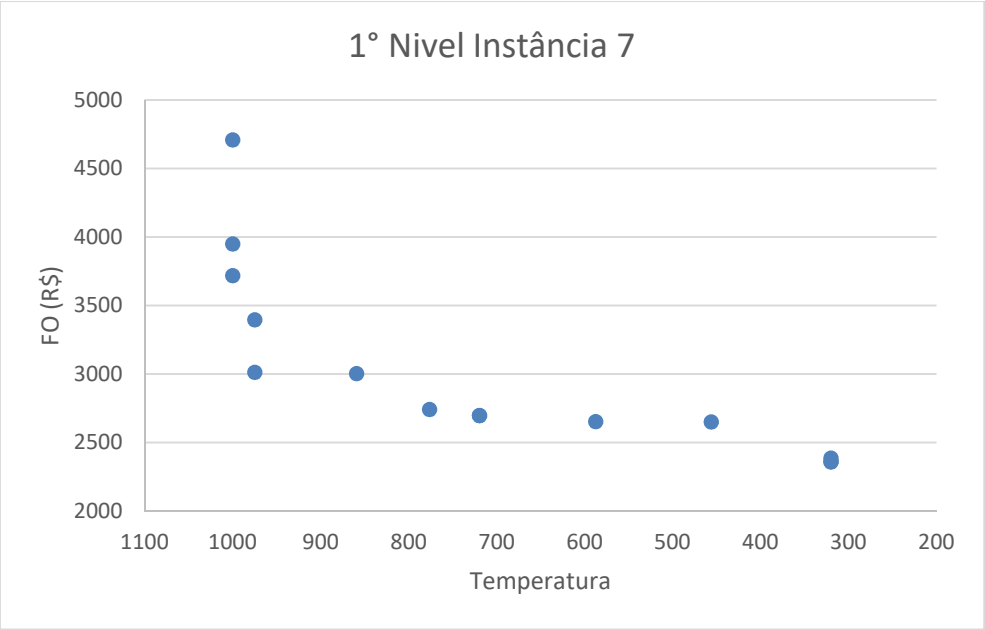
Fonte: Do Autor

Figura 36 - Gráfico Instância 6 Nível 2



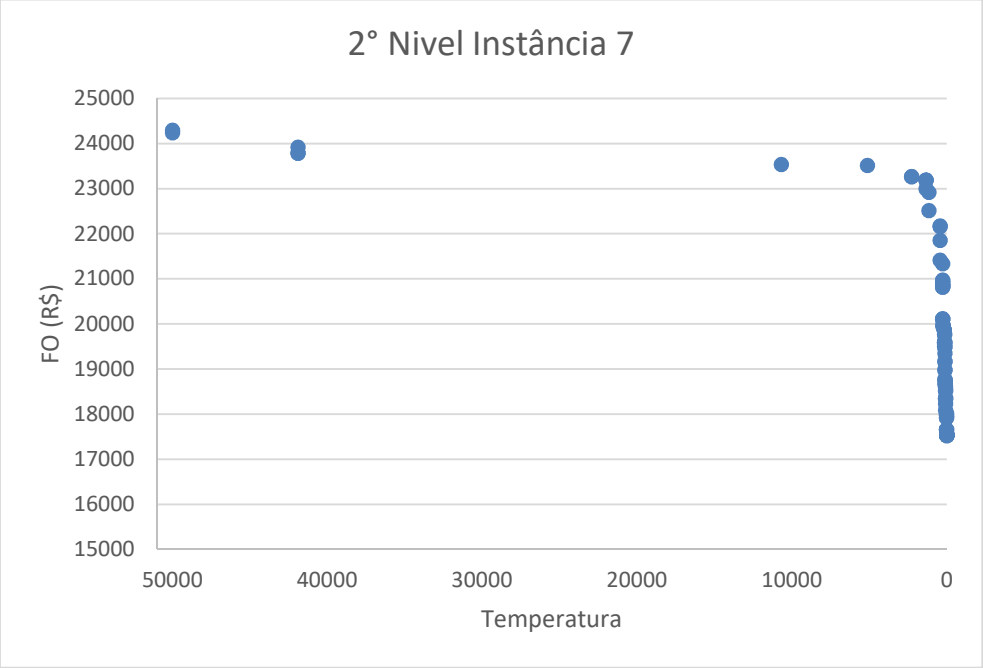
Fonte: Do Autor

Figura 37 - Gráfico Instância 7 Nível 1



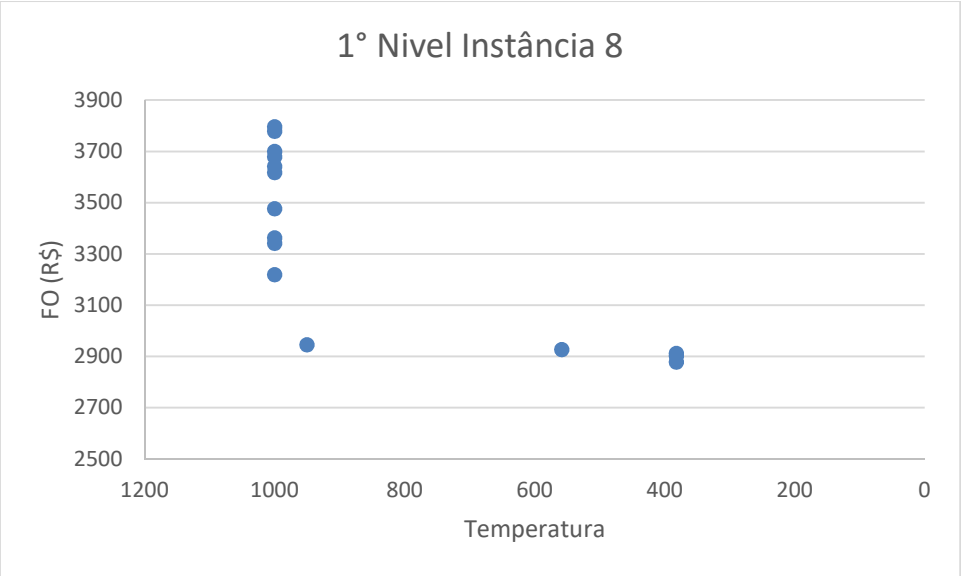
Fonte: Do Autor

Figura 38 - Gráfico Instância 7 Nível 2



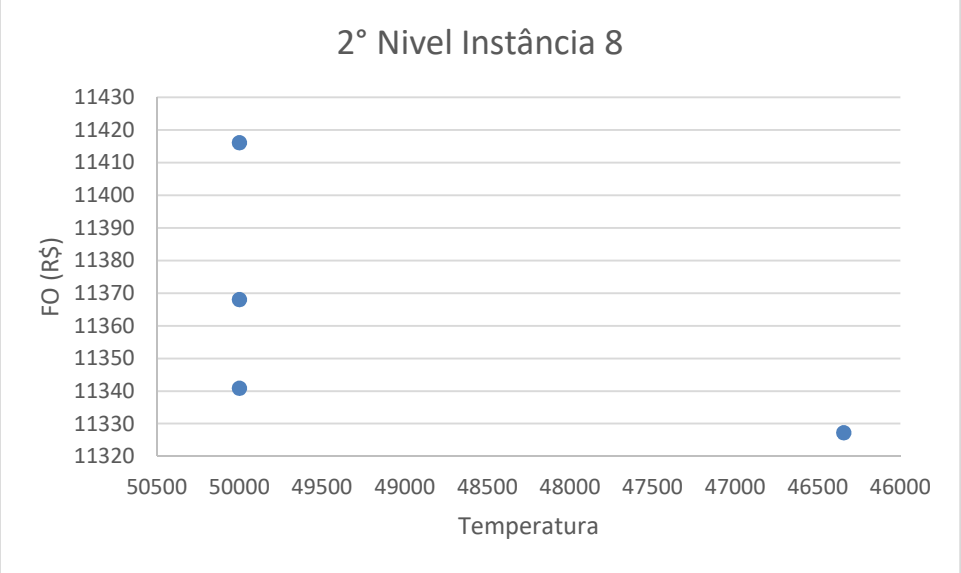
Fonte: Do Autor

Figura 39 - Gráfico Instância 8 Nível 1



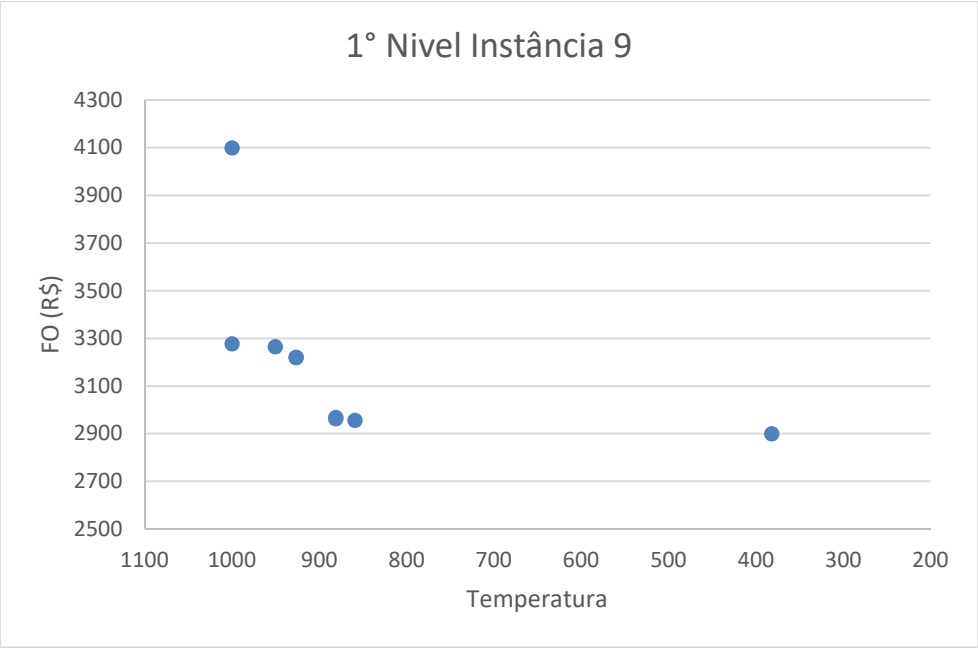
Fonte: Do Autor

Figura 40 - Gráfico Instância 8 Nível 2



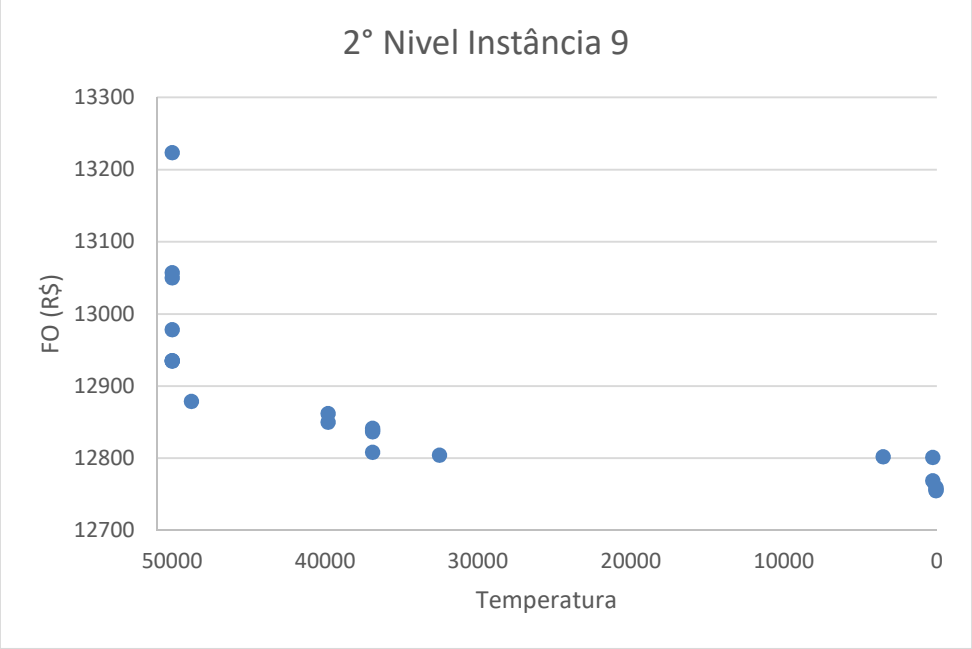
Fonte: Do Autor

Figura 41 - Gráfico Instância 9 Nível 1



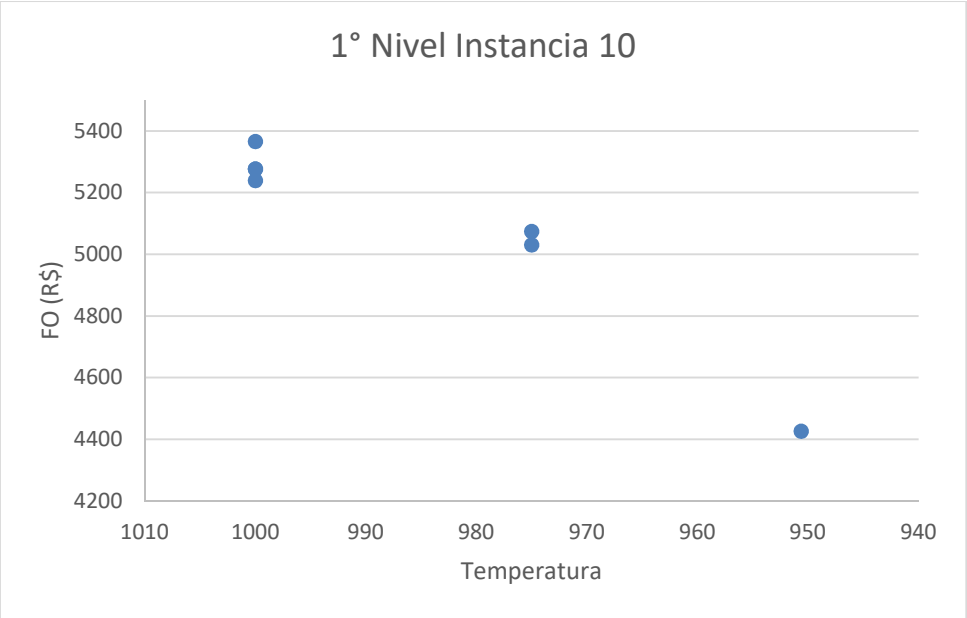
Fonte: Do Autor

Figura 42 - Gráfico Instância 9 Nível 2



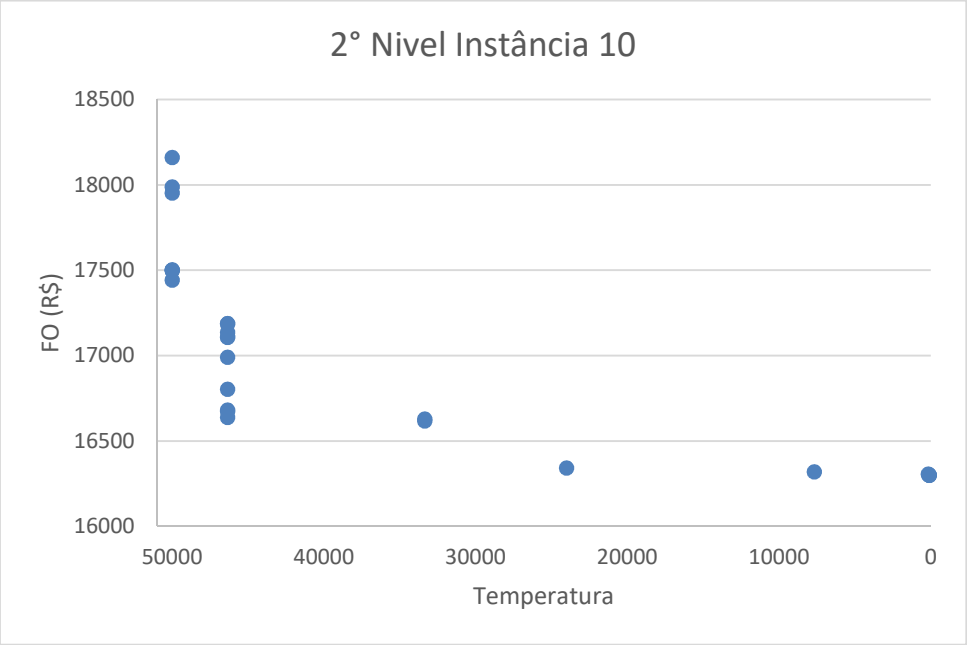
Fonte: Do Autor

Figura 43 - Gráfico Instância 10 Nível 1



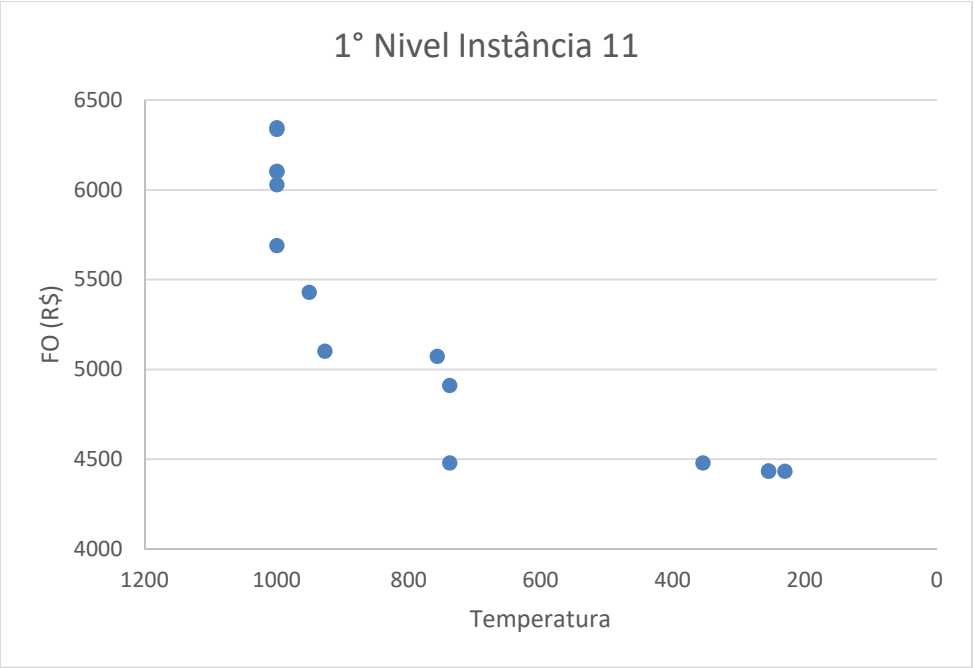
Fonte: Do Autor

Figura 44 - Gráfico Instância 10 Nível 2



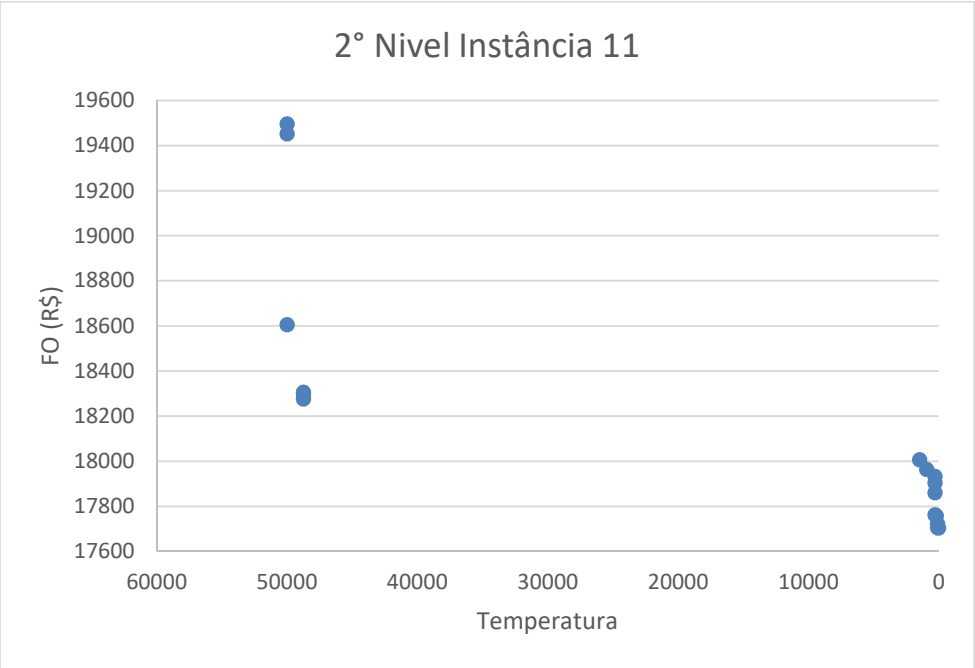
Fonte: Do Autor

Figura 45 - Gráfico Instância 11 Nível 1



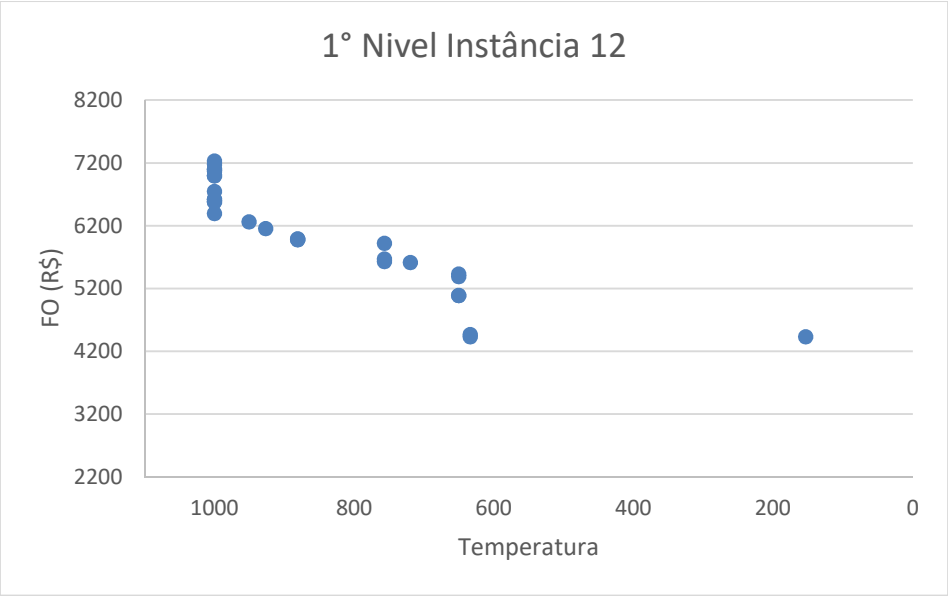
Fonte: Do Autor

Figura 46 - Gráfico Instância 11 Nível 2



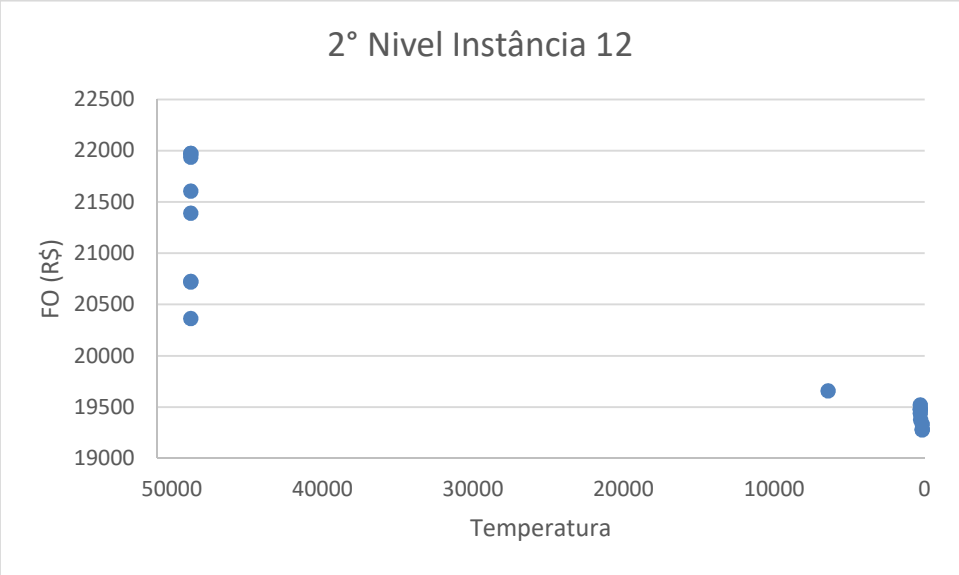
Fonte: Do Autor

Figura 47 - Gráfico Instância 12 Nível 1



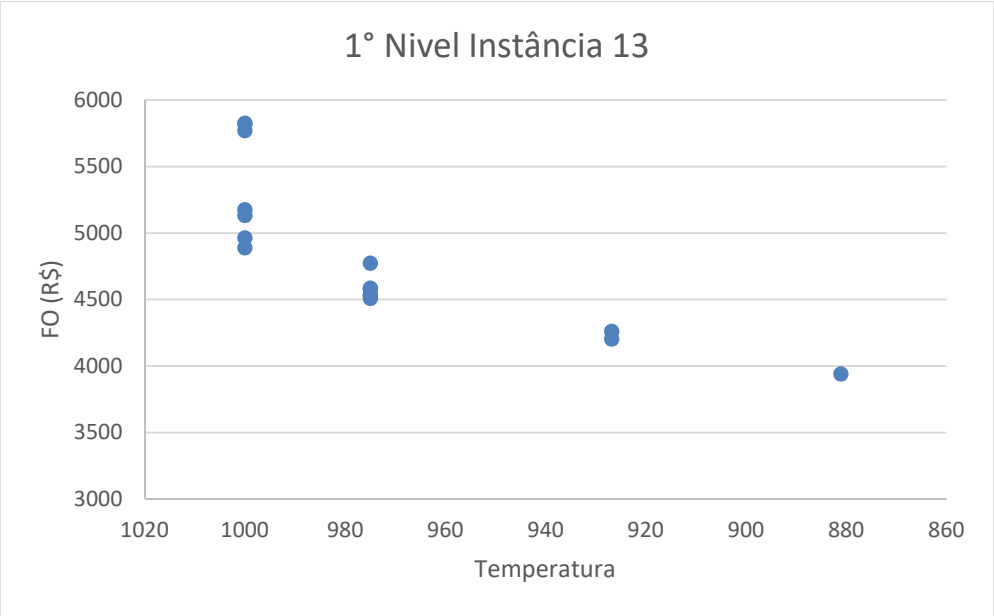
Fonte: Do Autor

Figura 48 - Gráfico Instância 12 Nível 2



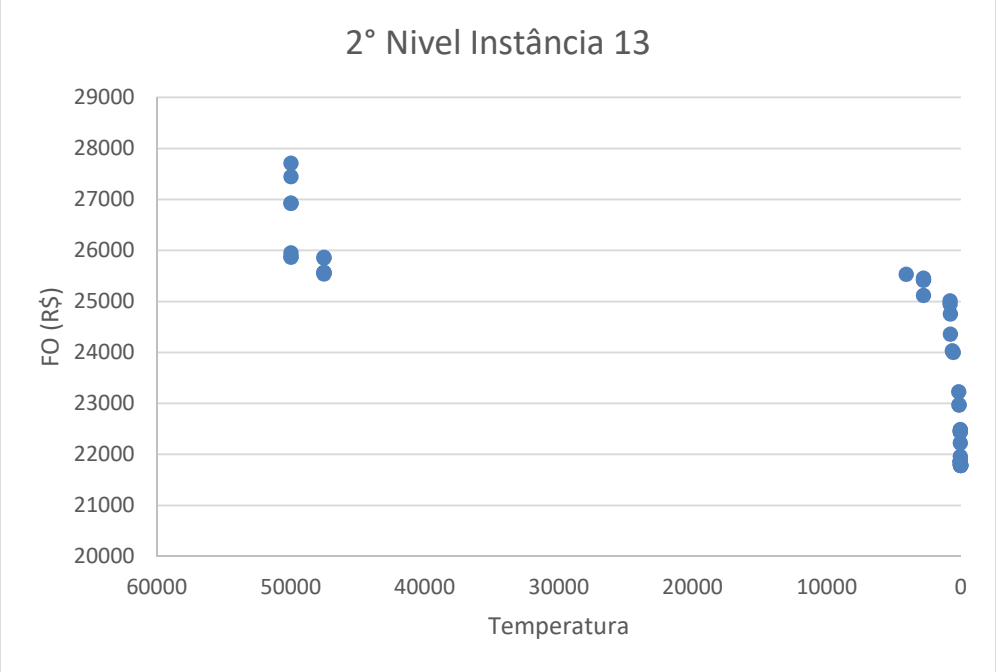
Fonte: Do Autor

Figura 49 - Gráfico Instância 13 Nível 1



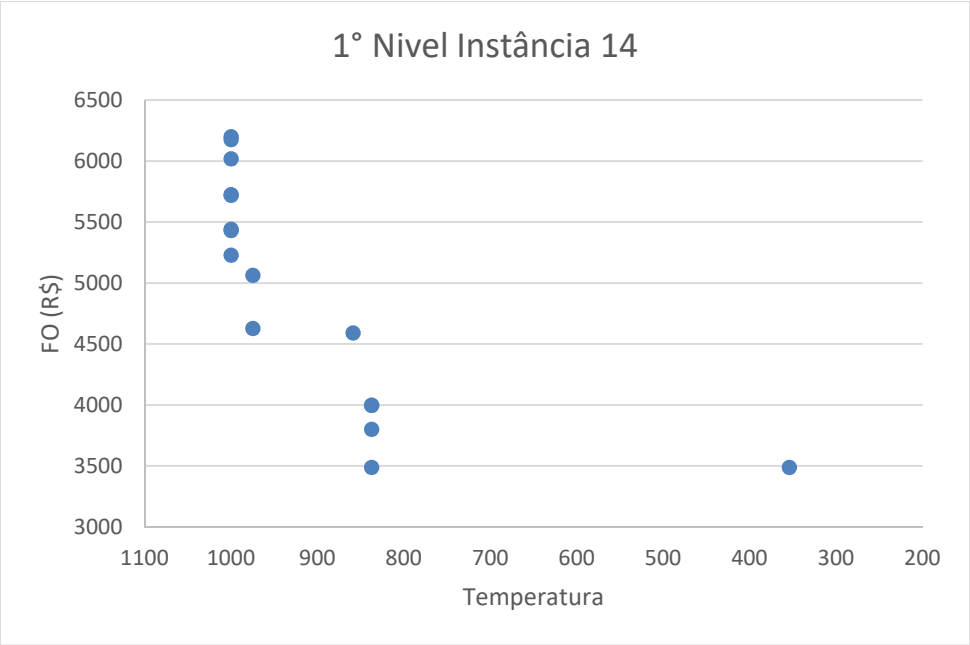
Fonte: Do Autor

Figura 50 - Gráfico Instância 13 Nível 2



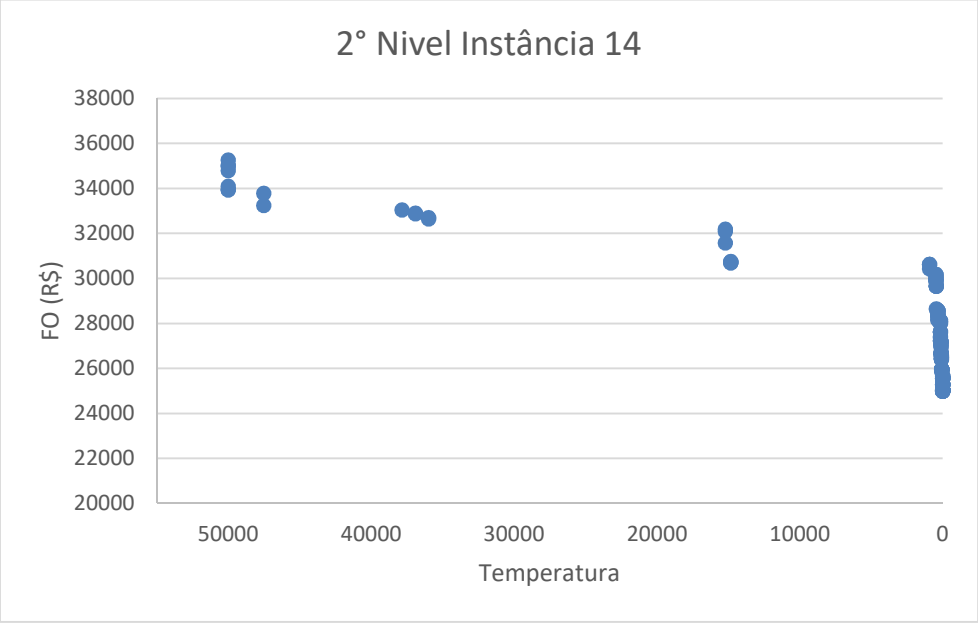
Fonte: Do Autor

Figura 51 - Gráfico Instância 14 Nível 1



Fonte: Do Autor

Figura 52 - Gráfico Instância 14 Nível 2



Fonte: Do Autor