UNIVERSIDADE FEDERAL FLUMINENSE

UM ESTUDO DE PROBLEMAS DE CLUSTERIZAÇÃO COM RESTRIÇÕES DE CAPACIDADE E DE CONEXIDADE

NÁDIA MENDES DOS SANTOS

Niterói
2014

NÁDIA MENDES DOS SANTOS

UM ESTUDO DE PROBLEMAS DE CLUSTERIZAÇÃO COM RESTRIÇÕES DE CAPACIDADE E DE CONEXIDADE

Tese de Doutorado submetida ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Doutor. Área de Concentração: Algoritmos e Otimização.

Prof. Luiz Satoru Ochi D.Sc. (Orientador)
Universidade Federal Fluminense (IC-UFF)

Niterói
2014

NÁDIA MENDES DOS SANTOS

UM ESTUDO DE PROBLEMAS DE CLUSTERIZAÇÃO COM RESTRIÇÕES DE CAPACIDADE E DE CONEXIDADE

Tese de Doutorado submetida ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Doutor. Área de Concentração: Algoritmos e Otimização.

Aprovada por:

BANCA EXAMINADORA

Prof. LUIZ SATORU OCHI, D.Sc. – Presidente (IC-UFF)

Prof. Carlos Alberto de Jesus Martinhon, D. Sc (IC-UFF)

Prof. Fábio Protti, D. Sc (IC-UFF)

Prof. José André de Moura Brito, D. Sc (ENCE-IBGE)

Prof. Nelson Maculan Filho, D. Sc (UFRJ-COPPE)

Prof. Virgílio José Martins Ferreira, D. Sc (UFRJ-COPPE)

Niterói, 25 de Novembro de 2014

A Deus, por ser minha forte rocha e meu refúgio

À minha mãe, por ter me iniciado nos caminhos dos estudos e por seu amor incondicional.

AGRADECIMENTOS

Primeiramente, a Deus que me deu força e proteção durante as inúmeras viagens à cidade de Niterói, sempre presente nas tribulações. A Deus, minha fé hoje e sempre.

À minha mãe, Maria de Jesus dos Santos Nascimento, por ter me dado amor, carinho e conselhos nas horas mais difíceis da minha vida.

À Ravena Bezerra do Nascimento, por sua paciência e conselhos nas dificuldades encontradas durante o desenvolvimento desta tese.

Ao meu orientador, professor Dr. Luiz Satoru Ochi, pela confiança depositada em mim desde minha inscrição para o programa DINTER-IFPI-UFF, em que logo me aceitou como orientanda. Seu apoio durante os meses de estágio na UFF e os períodos distantes, no estado do Piauí, foram de valiosa contribuição no desenvolvimento desta tese.

Ao professor José André de Moura Brito, sempre dedicado e esclarecedor no desenvolvimento deste trabalho. Sua orientação nesta pesquisa foi de fundamental importância.

Aos companheiros do Doutorado em Computação os quais merecem mérito pelo progresso da pesquisa, Eyder Rios e Gustavo S. Semaan.

E ao IFPI (Instituto Federal de Educação, Ciência e Tecnologia do Piauí) que contribui com o apoio financeiro necessário à realização do convênio DINTER (Doutorado InterInstitucional) junto à UFF (Universidade Federal Fluminense).

“Não há limite para nada, podemos chegar onde quisermos se tivermos dedicação. Com trabalho e esforço é possível”. João H.Vogel, 18 anos aceito na Universidade de Harvard.

RESUMO

O presente trabalho de doutorado traz uma proposta de solução para um Problema de Agrupamento que agrega Restrições de Capacidade e de Conexidade. Trata-se de uma aplicação real que pode ser mapeada em um problema de clusterização em grafos com restrições de capacidade e de conexidade. Em geral, a obtenção de um ótimo global para tal problema é uma tarefa muito difícil, tendo em vista o aspecto combinatório. Procurando-se encontrar soluções de boa qualidade para este problema, foram desenvolvidas versões diferentes de algoritmos heurísticos que utilizam os conceitos das metaheurísticas GRASP e VNS. No que concerne ao algoritmo GRASP (*Greedy Randomized Adaptive Search Procedure*), foram implementadas quatro versões diferentes, a saber: a primeira versão utilizando o modelo clássico, já conhecido na literatura, a segunda foi baseada na técnica de reconexão de caminhos, uma terceira aplicando uma busca através de um procedimento VND (*Variable Neighboord Descent*) e uma quarta com uma busca com componentes randômicos através de um módulo RVND (*Random VND*). O algoritmo VNS foi implementado em seu modelo tradicional e com três variações diferentes, ou seja, um VNS (*Variable Neighbourhood Search*) padrão, um GVNS (*General Variable Neighbourhood Search*), um RVNS (*Reduced Variable Neighbourhood Search*) e um SVNS (*Skewed Variable Neighbourhood Search*).

São apresentados testes comparativos com algoritmos da literatura ilustrando a eficiência (tempo) e/ou eficácia dos métodos propostos. Assim, os resultados observados para os experimentos computacionais realizados com instâncias reais do Censo Demográfico 2010 indicam que os algoritmos propostos constituem-se como uma boa alternativa para a resolução do problema que foi o objeto de estudo desta tese.

Palavras-chave: clusterização, capacidade, conexidade, GRASP, VNS.

ABSTRACT

This doctoral work brings a proposed solution to a problem of grouping that adds capacity constraints and connectivity. It is an actual application that is associated with a clustering problem in graphs with capacity constraints and connectivity. In general, to obtain a global optimal for this problem is a very difficult task in view of the combinatorial aspect. Seeking to find good quality solutions to this problem, different versions of heuristic algorithms which use the concepts of metaheuristics GRASP and VNS been developed. With respect to the algorithm GRASP four different versions have been implemented, namely: the first version using the classical model, already known in the literature, the second was based on the path relinking technique, the third by applying an search through VND procedure and the fourth with a search with random components through a module RVND (Random VND). The VNS algorithm is implemented in its traditional model and with three different variations, in other words, a standard VNS (Variable Neighbourhood Search), a GVNS (General Variable Neighbourhood Search), a RVNS (Reduced Variable Neighbourhood Search) and a SVNS (Skewed Variable Neighbourhood Search).

Comparative tests are shown with algorithms from the literature illustrating the efficiency (time) and/or efficacy of the proposed methods. Thus, the results observed for the computational experiments with real instances of the 2010 Population Census indicate that the proposed algorithms constitute themselves as a good alternative to solve the problem that was the object of study of this thesis.

Keywords: clusterizing, capacity, connectedness , GRASP, VNS.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AG	-	Algoritmo Genético
AGM	-	Árvore Geradora Mínima (*The Minimum Spanning Tree*)
AGT	-	Algoritmo Genético Tradicional
GRASP	-	*Greedy Randomized Adaptative Search Procedure*
LRC	-	Lista Restrita de Candidatos (*Restricted Candidate List)*
PR	-	*Path-Relinking*
RC	-	Reconexão por Caminhos
SB	-	Solução Base
SAT	-	Satisfabilidade
SKATER	-	*Spatial 'K'luster Analysist by Tree Edge Removal*
SI	-	Solução Intermediária
SG	-	Solução Guia
VND	-	Descida com Vizinhança Variável (*Variable Neighborhood Descent)*
VNS	-	*Variable Neighbourhood Search*
SVNS	-	*Skewed Variable Neighbourhood Search*
GVNS	-	*General Variable Neighbourhood Search*
RVNS	-	*Reduced Variable Neighbourhood Search*

SUMÁRIO

CAPÍTULO 1 – INTRODUÇÃO

É crescente o destaque que a área de otimização combinatória e mais geralmente a área de pesquisa operacional (PO) vem apresentando no contexto da Ciência da Computação. Este fato em parte é decorrente de que muitos dos problemas reais que são estudados atualmente em Ciência da Computação, podem ser mapeados como problemas de otimização combinatória, ou de forma mais geral, como problemas de PO pertencentes à classe NP-Difícil.

Em particular, no que diz respeito aos problemas de elevada complexidade computacional, mesmo diante do grande desenvolvimento dos métodos exatos de otimização e da evolução dos computadores, a sua aplicação de forma isolada na solução de muitos problemas NP-Difícil torna-se limitada devido ao dinamismo e a necessidade de, em muitos casos, haver uma resposta em um baixo tempo computacional.

Dentre os vários problemas pertencentes a esta classe, destacamos o problema de agrupamento ou de clusterização. Em linhas gerais, nesse problema deve-se agrupar os *n* objetos associados a uma base de dados em um número fixo de grupos. Estes grupos, por sua vez, são definidos mediante a avaliação de uma função objetivo e, eventualmente, considerando um conjunto de restrições.

Assim como em outros problemas da área de otimização combinatória, busca-se a implementação de métodos eficientes que produzam soluções viáveis de boa qualidade em tempo computacional factível. Neste caso, soluções de melhor qualidade são obtidas a expensas de métodos de computação mais intensivos, disponíveis na literatura.

Em suma, esses métodos encontram soluções viáveis que são boas o suficiente para serem consideradas úteis para a aplicação em questão. Dentre esses métodos, destacamos as heurísticas e as metaheurísticas.

Neste contexto, no presente trabalho de tese, foram estudadas e utilizadas as seguintes metaheurísticas: GRASP e VNS. Também foram utilizados os procedimentos de RC (Reconexão por Caminhos) e o método VND (Descida em Vizinhança Variável) para a construção de métodos híbridos que estão associados aos algoritmos que serão propostos.

Neste trabalho, esses métodos foram aplicados para a resolução de um problema de clusterização que agrega dois tipos de restrições adicionais, a saber: de capacidade e de conexidade (contiguidade). A restrição de capacidade está associada com um limite inferior *L* pré-estabelecido, com base em um atributo (característica) dos objetos. Assim, o somatório desse atributo nos objetos pertencentes a um mesmo cluster deve ser menor ou igual a *L*. E,

quanto à restrição de conexidade do cluster, devem ser observadas relações de vizinhança entre os objetos.

Mais especificamente, quaisquer dois objetos alocados a um mesmo cluster, devem ser vizinhos (fronteira) ou deve existir, pelo menos, um caminho entre dois objetos, passando apenas por objetos pertencentes ao mesmo cluster.

1.1 MOTIVAÇÃO E JUSTIFICATIVA

O problema de categorização de objetos é uma das mais antigas e comuns atividades humanas. Na teoria do desenvolvimento cognitivo de Piaget, o comportamento é controlado através de organizações mentais denominadas "esquemas", que o indivíduo utiliza para representar o mundo, classificar os grupos e para designar as ações. Supondo que um arquivo de dados possa representar a mente humana, os esquemas são análogos aos registros desse arquivo, ou seja, são as estruturas mentais ou cognitivas pelas quais os indivíduos intelectualmente organizam o meio.

Sendo a ação de agrupar objetos uma atividade própria do ser humano, pode-se verificar claramente que, nos seus primeiros anos de vida, dispõe-se de poucos esquemas, os quais apresentam informações bastante generalizadas acerca do universo ao qual estamos inseridos [PIAGET 1996]. Quando uma criança tem novas experiências (vendo imagens novas ou ouvindo palavras novas), ela tenta adaptar esses novos estímulos às estruturas cognitivas que já possui. Esse processo ocorre através da assimilação, que é o processo cognitivo pelo qual uma pessoa integra (classifica) um novo dado às estruturas cognitivas prévias [WADSWORTH 1996].

Portanto, a assimilação permite que a criança adapte os novos estímulos aos esquemas que ela já possui até aquele momento. Por exemplo, imaginemos que uma criança está aprendendo a reconhecer animais e, até o momento, o único animal que ela conhece e tem organizado esquematicamente é o cachorro. O fato é que o esquema de cachorro é a única estrutura cognitiva conhecida pela criança. Então, quando apresentada a esta criança outro animal que possua alguma semelhança, como um cavalo, ela responderá como se fosse um cachorro (quadrúpede, rabo, pescoço, nariz). Assim, o processo de assimilação, ou seja, a similaridade entre o cavalo e o cachorro ocorre em função da proximidade dos estímulos e da pouca variedade e da qualidade dos esquemas acumulados pela criança até o momento.

A diferenciação do cavalo para o cachorro deverá ocorrer por um processo chamado de acomodação. Assim, gradativamente novos estímulos são assimilados e acomodados na mente humana.

Portanto, a capacidade de diferenciar objetos, e no sentido inverso, processar o agrupamento (ou clusterização) de objetos, é iniciada ainda nos primeiros anos de nossas vidas.

A Figura 1.1 apresenta um conjunto de 4 (quatro) objetos para o qual pode-se formar grupos a partir das similaridades existentes entre eles. É possível agrupá-los conforme os atributos e critérios desejados. Caso o critério utilizado seja o atributo forma, serão formados quatro grupos: um de retângulos, um de elipses, um de pentágonos e um de triângulos. Se o critério utilizado for o atributo numeração obtém-se três grupos: número 1, número 2 e número 3.

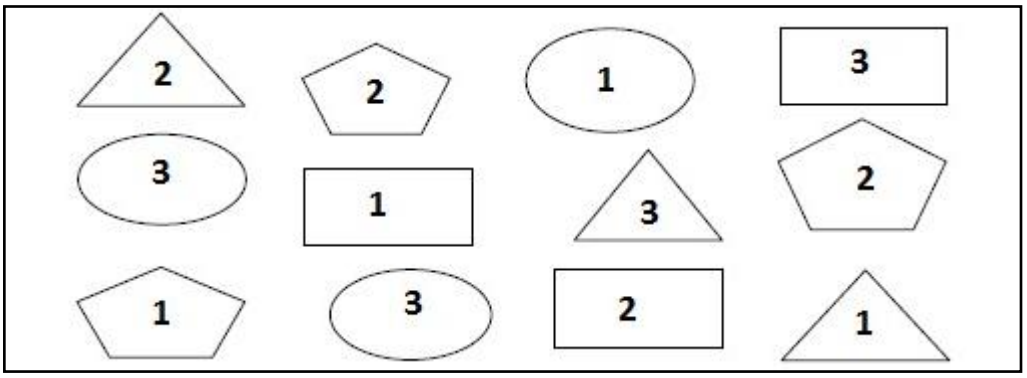


Figura 1.1 Conjunto de Objetos

Neste exemplo, a resolução do problema de clusterização é uma tarefa muito simples uma vez que temos um conjunto de 12 (doze) objetos com duas dimensões (forma e numeração).

Conforme mencionado na introdução, neste trabalho propõe-se resolver um particular problema de clusterização com restrições de capacidade e de conexidade utilizando uma modelagem baseada em grafos.

Há vários trabalhos na literatura [SEMAAN 2010][SEMAAN 2009][SEMAAN 2008][SHIEH AND MAY 2001] em que é salientado que a obtenção de um ótimo global para este problema torna-se impraticável à medida que o número de objetos a serem agrupados aumenta. Ou seja, caso seja aplicado algum método exato de enumeração implícita ou explícita para a resolução desse problema, seriam necessários meses ou anos para que fosse obtido o ótimo global.

Em decorrência desta última observação, a aplicação de métodos de agrupamento baseados em heurísticas e metaheurísticas é justificada e, acima de tudo, de fundamental importância para a resolução desse problema. Esses métodos tendem a produzir soluções viáveis de boa qualidade em um tempo computacional factível.

1.2 OBJETIVOS

A presente tese traz uma proposta de resolução para o problema de clusterização com restrições de capacidade e de conexidade. Mais especificamente, são propostos algoritmos que utilizam os conceitos das metaheurísticas GRASP e VNS e de heurísticas híbridas que incorporam os procedimentos de Reconexão por Caminhos (RC) e de Descida em Vizinhança Variável (VND).

Os resultados computacionais reportados neste trabalho foram obtidos a partir aplicação das metaheurísticas e das heurísticas em um conjunto de instâncias reais do Censo Demográfico de 2010 e da Contagem Populacional [CENSO DEMOGRAFICO 2010] (ambos dados de uso público).

Foram objetivos específicos deste trabalho:

- revisar a literatura em busca de trabalhos sobre o problema de clusterização e uma visão correlata com o problema de clusterização com restrições de capacidade e de conexidade;

- descrever os novos algoritmos que foram implementados a partir do estudo das metaheurísticas GRASP e VNS e do procedimento de Reconexão por Caminhos (RC) e do método de Descida em Vizinhança Variável (VND);

- programar os algoritmos heurísticos (GRASP e VNS) e híbridos com os módulos de (RC e VND) para resolver o problema de clusterização;

- avaliar e comparar (quanto à qualidade da solução e ao tempo de processamento) os algoritmos heurísticos com os algoritmos híbridos utilizando o conjunto de instâncias reais;

1.3 METODOLOGIA

Neste trabalho, faz-se uso das metaheurísticas como GRASP (*Greedy Randomized Adaptive Search Procedure*) [FEO AND RESENDE, 1995][PITSOULIS E RESENDE, 2002][RESENDE E RIBEIRO 2002] e VNS (*Variable Neigborhood Search*) [MLADENOVIC 1995],[GLOVER AND KOCHENBERGER 2002][MLADENOVIC AND HANSEN 1997].

E do módulo de PR (*Path-Relinking*) ou RC (Reconexão por Caminhos) [GLOVER, 1999], [GLOVER ET AL, 2000], [GLOVER ET AL 2004] e do método de VND (*Variable Neigborhood Descent*)[HANSEN; MIDADENOVIC, 2003][CHAVES E LORENA, 2005][HERNÁNDEZ-PÉREZ ET AL 2009] para desenvolver os algoritmos híbridos**,** visando resolver o problema de clusterização com restrições de capacidade e de conexidade.

Inicialmente, o presente trabalho traz uma descrição do problema de clusterização com suas peculiaridades, quais sejam: medidas de similaridade, critérios de agrupamento, classificação dos algoritmos de clusterização (não hierárquicos e hierárquicos). O problema de clusterização é formalizado com suas restrições de capacidade e de conexidade. Também foi efetuada uma revisão bibliográfica de problemas de clusterização, comparando as ideias e as metodologias propostas nestes trabalhos com as ideias formalizadas neste trabalho de tese.

Foram implementados algoritmos para a resolução problema de clusterização com restrições de capacidade e de conexidade que fizeram uso de metaheurísticas como GRASP e VNS. E para desenvolver as heurísticas híbridas foram feitas combinações das metaheurísticas citadas com o procedimento de Reconexão por Caminhos, com os métodos VND e RVND. Todos as implementações foram feitas na Linguagem C, utilizando o ambiente de desenvolvimento *NetBeans* 7.0.1 com dados reais (de uso público) coletados do Censo Demográfico de 2010 e da Contagem Populacional do IBGE [CENSO DEMOGRAFICO 2010]. De forma a comparar e avaliar os algoritmos propostos foram realizadas execuções sistemáticas e análises probabilísticas.

A parte final desta tese traz um conjunto de resultados computacionais que ratificam a eficácia dos algoritmos heurísticos e híbridos no que se refere à qualidade das soluções produzidas, e quanto à obtenção de soluções que atendessem as restrições de capacidade e conexidade do problema de clusterização.

1.4 CONTRIBUIÇÕES

O presente trabalho de tese tem como contribuição o desenvolvimento de métodos híbridos que combinam metaheurísticas e heurísticas de busca local para resolver o problema de clusterização com restrições de capacidade de conexidade. Assim, os primeiros algoritmos desenvolvidos foram baseados em métodos híbridos (GRASP com *Path-Relinking*, GRASP com Busca Local VND e GRASP com Busca Local RVND) e heurísticas de busca local (VNS, GVNS, RVNS e *Skewed* VNS), introduzindo inovações como novos procedimentos construtores para a solução do problema tratado nesta tese.

É fato conhecido que a combinação de uma metaheurística com outras técnicas de otimização, chamadas de metaheurísticas híbridas, pode proporcionar um comportamento mais eficiente e uma maior flexibilidade quando se trata de problemas do mundo real e em larga escala [BLUM AND ROLI, 2003]. Isto pode ser obtido, por exemplo, combinando uma metaheurística com heurísticas de busca específicas para um problema real, como no caso deste trabalho.

Geralmente, a utilização de diferentes métodos híbridos possibilita a produção de soluções de melhor qualidade quando comparadas às soluções provenientes de algoritmos baseados em apenas metaheurísticas. De fato, a escolha de uma abordagem híbrida adequada, muitas vezes utilizando algoritmos de busca mais especializados que consideram a natureza do problema e/ou informações sobre o espaço de busca, é determinante para alcançar um melhor desempenho computacional na resolução de problemas difíceis [RAIDL, 2006].

Consta também como resultado deste trabalho a realização de experimentos computacionais de modo a comparar o desempenho dos diversos algoritmos desenvolvidos, o que influencia na disponibilização de um algoritmo de solução competitivo para o problema tratado.

Finalmente, propõe-se como trabalhos futuro penalização das soluções; inserir nos algoritmos uma variável referente às eventuais penalidades existentes nos *clusters*; desenvolvimento de uma formulação de programação inteira para o problema de clusterização com restrições de capacidade e de conexidade e implementar uma formulação matemática para utilizar um bom *Upper Bound* gerado a partir de uma solução viável de boa qualidade produzida pela melhor resultado das metaheurísticas aplicadas ao problema de clusterização com restrições de capacidade e de conexidade.

1.5 ORGANIZAÇÃO

Este trabalho de tese está organizado de seguinte forma: O Capítulo 1 traz uma ideia geral do problema tratado nessa tese, considerando a descrição de seus objetivos, da metodologia e contribuições desta pesquisa científica. O Capítulo 2 traz uma descrição detalhada do problema de clusterização e as medidas utilizadas no cálculo da similaridade e possíveis critérios de agrupamento. Apresenta a classificação dos algoritmos de clusterização. Formaliza o problema de clusterização com restrições de capacidade e conexidade, finalizando com uma revisão bibliográfica de problemas de clusterização.

O Capítulo 3 apresenta os algoritmos propostos para a resolução do problema de clusterização com restrições de capacidade e de conexidade que foi o objeto de estudo desta tese. Em particular, tais algoritmos incorporam os conceitos das metaheurísticas como GRASP e VNS. E para construir as heurísticas híbridas foram feitas combinações das metaheurísticas citadas com o procedimento de reconexão por caminhos e também com os métodos VND e RVND.

O Capítulo 4 traz um conjunto de resultados computacionais e faz uma discussão dos algoritmos propostos e de suas versões híbridas, além de fazer uma análise de desempenho

dos algoritmos propostos avaliando-os e comparando-os mediante a utilização de instâncias reais. O Capítulo 5 traz as conclusões do trabalho e aponta algumas direções para os possíveis desdobramentos desta tese.

CAPÍTULO 2 – O PROBLEMA DE CLUSTERIZAÇÃO

Atualmente, muitos pesquisadores da área de computação têm concentrado a sua pesquisa no estudo e o desenvolvimento de várias técnicas de clusterização. Estes estudos têm trazido contribuições significativas para várias áreas, quais sejam: a estatística espacial [MONTEIRO 2009] [RODRIGUES 2001], mineração de dados [SMITH 2003] [TSAI ET. AL 2003], biologia [RECH AND ABSY 2011][BOLSHAKOVA AND AZUAJE 2003], medicina [NEUHAUS 1992][MANLY 2008], ciências ambientais [BROWN AND BOLKER, 2004] [GUO ET. AL 2003], processamento de imagens[DREW AND AU 2003] [FANG ET AL 2003], engenharia de produção industrial [WANG 2003] [TRINDADE E OCHI 2004], dentre outras.

A vantagem desta grande variedade de aplicações reais é que tem-se o envolvimento de muitos pesquisadores e estudiosos com as mais variadas formações e objetivos tratando do problema de clusterização. E como desvantagem destaca-se o uso de diferentes notações e formas de abordagem para as mesmas questões**,** o que tem dificultado o estabelecimento de metodologias genéricas e taxinomias em função do substancial número de trabalhos e notações concernentes a este tema.

De uma forma geral, os pesquisadores têm encontrado limitações computacionais no que se refere à automação dos processos de clusterização. Essas limitações, por sua vez, são decorrentes da dificuldade em produzir soluções de boa qualidade em um tempo computacional factível para diversas aplicações reais que podem ser mapeadas em um problema de agrupamento. Enfatiza-se que a complexidade computacional intrínseca ao problema de clusterização tem na representação dos dados seu maior desafio seja: pelo número de objetos a serem agrupados, pela mensuração de similaridade/dissimilaridade entre eles, e pelas variadas formas, densidades e tamanhos que os grupos podem apresentar.

2.1 DESCRIÇÃO DE CLUSTERIZAÇÃO

A clusterização corresponde ao processo de tomar um conjunto de *n* objetos de uma base de dados e agrupá-los em classes de objetos denominadas *clusters*. Portanto, um *cluster* é um conjunto de objetos similares entre si e, ao mesmo tempo, os elementos pertencentes subconjuntos diferentes apresentem alta dissimilaridade sendo a similaridade avaliada em função dos atributos associados aos objetos. [HAN AND KAMBER 2001].

Formalmente, o problema de clusterização pode ser definido da seguinte forma: Dado um conjunto X formado por n objetos $X= \{x_1, x_2,...,x_i,...,x_n\}$, com cada um dos objetos x_i

possuindo p atributos $x_i = (x_i^1, x_i^2, ..., x_i^p)$ (características do objeto), deve-se construir k partições CL_i (*clusters*) a partir de X, respeitando às seguintes restrições [HAN AND KAMBER 2001] [HRUSCHKA 2001] [DIAS AND OCHI 2003]:

$$CL_i \neq \emptyset \; i = 1, \ldots, k$$

$$CL_i \cap CL_j = \emptyset \; i, j = 1, \ldots, k \; e \; i \neq j$$

$$U_{i=1}^{k} = CL_j = X$$

Quando o número de *clusters* é um dado de entrada para o algoritmo, ou seja, quando k é fixado previamente, o problema é conhecido na literatura como problema de k-clusterização ou simplesmente por Problema de Clusterização (PC) [FASULO 1999]. Neste caso, o número total de soluções viáveis pode ser obtido utilizando-se a fórmula apresentada na equação (1) associada ao número de *Stirling* de segundo tipo, a seguir [CHIOU AND LAN 2001]:

$$N(n,k) = \frac{1}{k!}\sum_{i=0}^{k}(-1)^i \binom{k}{i} (k-i)^n \quad (1)$$

E quando o número de clusters não é definido previamente, tem-se um problema de ainda mais difícil conhecido na literatura como Problema de Clusterização Automática (PCA) [DOVAL ET. AL, 1999]. Neste caso, o total de soluções viáveis é dado pela seguinte equação:

$$N(n,k) = \sum_{k=1}^{n}\frac{1}{k!}\sum_{i=0}^{k}(-1)^i \binom{k}{i} (k-i)n \quad (2)$$

A literatura [WELCH 1983] mostra que estes dois problemas (PC e PCA) de otimização são classificados como NP-Completo.

Então, mediante a descrição do problema de clusterização, é possível concluir que uma boa clusterização é aquela que minimiza a dissimilaridade entre objetos pertencentes a um mesmo cluster (intra-*cluster*) e maximiza a dissimilaridade entre objetos pertencentes a clusters diferentes (inter-*cluster*).

A obtenção de uma boa clusterização depende, ainda, da observação de características importantes por parte dos algoritmos de clusterização, segundo [HAN AND KRAMBER 2001][AGRAWAL ET. AL 1998], quais sejam:

- Escalabilidade: Para a maioria das aplicações reais, exige-se que o algoritmo seja capaz de apresentar boas soluções que independem do tamanho da entrada (número de objetos). Algumas estratégias costumam a ser usadas para diminuir a sensibilidade do algoritmo ao tamanho do conjunto. Portanto, esses algoritmos

podem trabalhar com qualquer massa de dados ou dimensões. Então, possuir uma alta escalabilidade é uma propriedade desejável em um algoritmo de clusterização;

- Seleção de Variáveis: A escolha das variáveis que serão avaliadas para o processo de clusterização implica somente considerá-las ou não. Muitas vezes, a própria aplicação direciona a escolha das variáveis relevantes ao problema;
- Tipos de Variáveis: As variáveis podem ser do tipo quantitativo (discretas ou contínuas) ou do tipo qualitativo (nominais ou ordinais) ou mistas;
- Habilidade para tratar ruídos: Dados com ruídos são comuns em várias aplicações reais. Seja por erros de digitação, ou por uma falha na coleta dos dados, é sempre possível termos em uma base de dados registros cujas informações foram corrompidas, ou seja, consideradas fora do padrão utilizado. Portanto, o desenvolvimento de um algoritmo que seja capaz de tratar os ruídos é um pré-requisito importante.
- Habilidade para processar dados com dimensões elevadas: alguns algoritmos são muito eficientes quando lidam com objetos com duas ou três dimensões (atributos). Não obstante, para bases de dados cujos objetos têm um número elevado de dimensões, a qualidade das soluções apresentadas cai consideravelmente. É importante então, que o número de dimensões da base de dados não interfira negativamente no desempenho do algoritmo.
- Identificação de *clusters* com formas arbitrárias: É desejável que um algoritmo de clusterização seja hábil a encontrar *cluster* de formatos variados como, por exemplo: esféricos, hiper-esféricos, curvilíenos, alongados, já que em muitas aplicações nenhum conhecimento a *priori* sobre a distribuição dos objetos pode ser obtido;
- Existência de restrições: Existem aplicações reais em que será necessário definir os grupos (*clusters*) de forma a satisfazer algumas restrições;
- Mínimo de conhecimento na entrada dos parâmetros: em alguns algoritmos, os resultados são bastante sensíveis aos parâmetros de entradas, tais como as configurações do algoritmo e (os) arquivos de dados. Com frequência, estes parâmetros são difíceis de determinar, especialmente quando se lida com grande volume de dados e com alta dimensionalidade, isto é, com muitos atributos.

- Interpretação dos resultados: Deve ser de fácil interpretação, ou seja, os resultados devem ser simples e de fácil entendimento e úteis para o objetivo ao que ele se propõe.

2.2 MEDIDAS DE SIMILARIDADE

O problema de clusterização requer a definição de uma função objetivo que expresse o grau de homogeneidade dos agrupamentos formados. Esta homogeneidade está diretamente relacionada com a utilização de uma medida de similaridade que permite avaliar se dois objetos são similares entre si no que concerne a um conjunto de atributos. Dessa forma, espera-se que os objetos alocados a um mesmo cluster sejam similares (para estes atributos) e os objetos pertencentes a clusters distintos tenham baixa similaridade. Sendo assim, uma boa clusterização estabelece medidas de qualidade de clusterização, sendo essas medidas correspondentes às distâncias entre dois objetos.

Para se avaliar a semelhança entre dois objetos, verifica-se que, quanto menor for a dissimilaridade entre os mesmos, menor a distância entre eles. Assim, pode-se utilizar uma matriz $D_{n\ x\ n}$ onde são armazenadas as distâncias entre todos os objetos tomados dois a dois, sendo tal matriz construída a partir de uma matriz X. Cada entrada d_{ij} de D (calculada a partir de X) representa a medida de dissimilaridade entre os objetos i e j.

Na maioria dos problemas de clusterização, cada um dos n objetos é descrito por um conjunto de p atributos ou variáveis. Desta forma, o conjunto de dados também pode ser representado por uma matriz $n\ x\ p$:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \cdot\cdot & \cdot\cdot & & \cdot\cdot \\ \cdot\cdot & \cdot\cdot & & \cdot\cdot \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \ddots & & \vdots \\ x_{nl} & x_{n2} & \cdots & x_{np} \end{bmatrix}$$

onde x_{il} é o valor do atributo l para o objeto i.

Esta matriz aqui denominada de matriz de Medida de Dissimilaridade (D), pode ser referenciada como matriz de Dissimilaridade ou ainda de Proximidade [GORDON, 1981]. Se X é o conjunto de objetos que serão agrupados, a medida de dissimilaridade é tal que:

(i) $d_{ij} \geq 0$ para todo x_i e x_j pertencente a X;

(ii) $d_{ij} = 0$ para todo x_i e x_j pertencente a X tal que $x_i = x_j$;

(iii) $d_{ij} = d_{ji}$ para todo x_i e x_j pertencente a X;

(iv) $d_{ij} \leq d_{ih} + d_{hj}$ para todo x_i , x_j e x_h pertencentes a X;

As propriedades (ii) e (iii) implicam que a *Matriz de Dissimilaridade* pode ser especificada por uma matriz diagonal inferior contendo *n(n – 1)/2* entradas:

$$D = \begin{vmatrix} \cdots & \cdots & \cdots & \cdots \\ d_{21} & \cdots & \cdots & \cdots \\ d_{31} & d_{32} & \cdots & \cdots \\ \vdots & \vdots & \ddots & \cdots \\ d_{n1} & d_{n2} & \cdots & d_{n,n-1} \end{vmatrix}$$

Dentre as medidas de distância utilizadas, podemos destacar: a distância Euclidiana, de *Manhattan* e a de *Minkowski* [HAN AND KRAMBER, 2001][DIAS 2004].

A matriz de Dissimilaridade (D) pode ser obtida a partir da utilização das expressões matemáticas que denotam as medidas de dissimilaridade que serão apresentadas a seguir. A medida mais comum é a Distância Euclidiana, dada pela seguinte forma:

$$D_{Euclidiana}(x_i, x_j) = \left[\sum_{l=1}^{p}(x_{il} - x_{jl})^2\right]^{1/2} \quad (3)$$

onde x_{il} é o valor do atributo l para o objeto i e x_{jl} é o valor do atributo l para o objeto j.

Na Distância Euclidiana cada um dos l atributos ($l=1,...,p$) pode ser ponderado por um peso específico w_l relativo a sua importância. Dessa forma, a fórmula fica descrita como a Equação 4 a seguir:

$$D_{EuclidianaPonderada}(x_i, x_j) = \left[\sum_{l=1}^{l} wl(x_{il} - x_{jl})^2\right]^{1/2} \quad (4)$$

Outra medida utilizada é Distância de *Manhattan* (*city block*), dada por:

$$D_{Manhattan}(x_i, x_j) = \sum_{l=1}^{p}|x_{il} - x_{jl}| \quad (5)$$

As três medidas apresentadas anteriormente, são aplicáveis às variáveis do tipo quantitativo. As diferenças entre essas duas distâncias são representadas pela Figura 2.1 (considera-se que seja utilizado um espaço bidimensional). Cada objeto é representado por um ponto no plano, sendo considerada a existência de dois atributos do tipo quantitativo. A

Distância Euclidiana fornece uma medida mais precisa da distância entre os pontos, enquanto que a Distância de *Manhattan* apresenta uma expressão mais simples que pode representar, em alguns casos, alguma vantagem computacional [NEVES 2003].

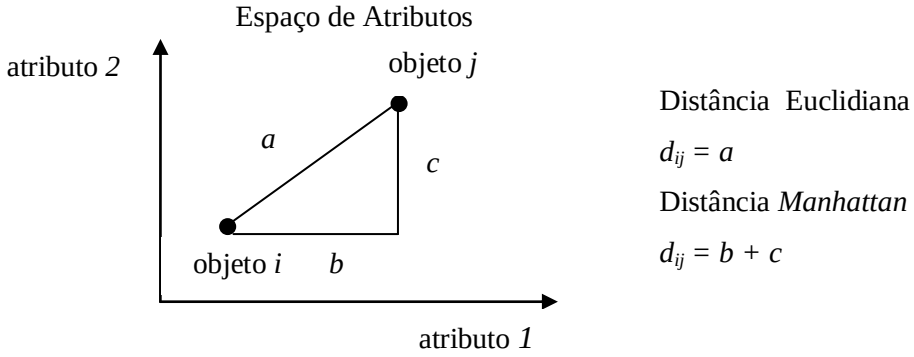


Figura 2.1 Representação da Distância Euclidiana e *Manhattan*

Essas duas medidas de similaridade, Distância Euclidiana e *Manhattan*, podem ser entendidas como sendo casos especiais de uma medida geral, chamada de Distância de *Minkowski*, que é expressa pela equação a seguir:

$$D_{Minkowski}(x_i, x_j) = \left[\sum_{l=1}^{p} |x_{il} - x_{jl}|^q\right]^{1/q} \quad (6)$$

Neste trabalho optou-se pela distância euclidiana, uma vez que todas as instâncias consideradas têm objetos com todos os atributos quantitativos. Com o objetivo de representar os cálculos das Distâncias Euclidiana (concordância) da Tabela 2.1, apresenta-se um exemplo de instâncias e sua ilustração através na Figura 2.2:

Tabela 2.1: Exemplo de Instância

Objeto	Atributo 1	Atributo 2
A	1	1
B	1	3
C	3	3
D	5	3
E	6	4
F	7	1

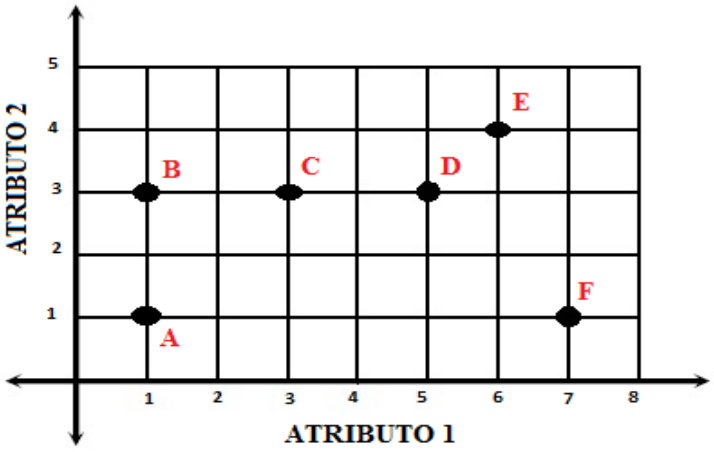


Figura 2.2 Representação Gráfica da Tabela 2.1

A partir da Figura 2.2, a distância Euclidiana ($DEuclidiana(x_i, x_j)$) pode ser definida da seguinte maneira: Sejam $A = (x_1, y_1)$ => $A = (1, 1)$ e $B = (x_2, y_2)$ => $B = (1, 3)$. Então, a $D_{Euclidiana}(A, B) = \sqrt[2]{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ =>$\sqrt[2]{(1-1)^2 + (3-1)^2}$=$\sqrt[2]{(0)^2 + (2)^2}$= $D_{Euclidiana}(A, B) = \sqrt[2]{4}$= 2. As matrizes a seguir trazem a Distância Euclidiana entre todos os pontos da Figura 2.2 e seus respectivos valores.

$$DEuclidiana(x_i, x_j) = \begin{pmatrix} d_{AA} & d_{AB} & d_{AC} & d_{AD} & d_{AE} & d_{AF} \\ d_{BA} & d_{BB} & d_{BC} & d_{BD} & d_{BE} & d_{BF} \\ d_{CA} & d_{CB} & d_{CC} & d_{CD} & d_{CE} & d_{CF} \\ d_{DA} & d_{DB} & d_{DC} & d_{DD} & d_{DE} & d_{DF} \\ d_{EA} & d_{EB} & d_{EC} & d_{ED} & d_{EE} & d_{EF} \\ d_{FA} & d_{FB} & d_{FC} & d_{FD} & d_{FE} & d_{FF} \end{pmatrix}$$

$$DEuclidiana(x_i, x_j) = \begin{pmatrix} 0{,}0 & 2{,}0 & 2{,}8 & 4{,}4 & 5{,}8 & 6{,}0 \\ 2{,}0 & 0{,}0 & 2{,}0 & 4{,}0 & 5{,}0 & 6{,}3 \\ 2{,}8 & 2{,}0 & 0{,}0 & 2{,}0 & 3{,}1 & 4{,}4 \\ 4{,}4 & 4{,}0 & 2{,}0 & 0{,}0 & 1{,}4 & 2{,}8 \\ 5{,}8 & 5{,}0 & 3{,}1 & 1{,}4 & 0{,}0 & 3{,}1 \\ 6{,}0 & 6{,}3 & 4{,}4 & 2{,}8 & 3{,}1 & 0{,}0 \end{pmatrix}$$

2.3 FUNÇÕES OBJETIVO

Existem várias funções objetivo que podem ser utilizadas nos problemas de clusterização, de forma a avaliar a homogeneidade dos grupos formados. No presente trabalho foi adotada a seguinte função objetivo:

$$f(P) = \sum_{c=1}^{k} Som_c \quad (7)$$

Sendo ***P*** uma partição dos *n* objetos em *k clusters* e Som_c correspondente à soma quadrados dos desvios dos *clusters*. Assim, uma partição é uma possível solução para o problema de agrupamento, é na verdade uma divisão de objetos em conjuntos não sobrepostos (*clusters*).

A Equação 7 representa a medida da dispersão dos valores dos atributos dos objetos de um *cluster* em relação aos valores médios de todos os atributos para um *cluster*. Se os *clusters* forem similares, os valores de Som_c serão pequenos. Dessa forma, quanto menor o valor da equação (7), mais homogêneos serão os clusters.

$$Som_c = \sum_{l=1}^{p} \sum_{i=1}^{n_c} (x_{il} - \overline{x_l})^2 \quad (8)$$

e

$$\overline{x_l} = \frac{1}{n_c} \sum_{i=1}^{n_c} x_{il}\,, para\ l = 1, \dots p \quad (9)$$

sendo:

- n_c o número de objetos membros do *cluster c*;
- x_{il} o atributo *l* do objeto *i*;
- *p* o número de atributos considerados na análise;
- $\overline{x_l}$ o valor médio do atributo *l*, para um *cluster*.

2.4 CLASSIFICAÇÃO DOS ALGORITMOS DE CLUSTERIZAÇÃO

A classificação dos algoritmos de agrupamento tanto para o Problema de Clusterização quanto para o Problema de Clusterização Automática é feita de um modo geral sem distinção entre o primeiro ou o segundo tipo de problemas. Segundo [GARAI E CHAUDHURI 2004] os algoritmos de agrupamento podem ser classificados como:

- Hierárquicos: Aglomerativos e Divisivos;
- Não-hierárquico: Particionamento.

Pode-se, ainda, citar outros algoritmos como os baseados em: densidade, em grades, em co-ocorrência de dados de tipo categóricos, em modelos, em restrições, além de métodos híbridos de clusterização [BERKHIN 2002] [HAN E KAMBER 2001][FISHER 1987].

Em particular, no presente trabalho, adotar-se-á a classificação apresentada no trabalho de [SOARES 2004]. Os algoritmos de clusterização são classificados conforme a Figura 2.2:

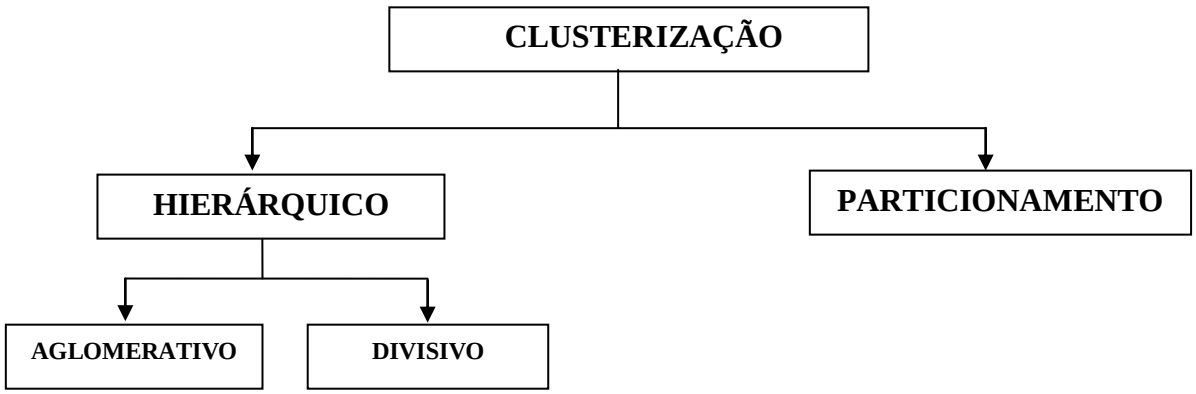


Figura 2.3 Classificação dos algoritmos de clusterização [SOARES 2004]

Os algoritmos Hierárquicos operam sobre um conjunto *X* formado por *n* objetos e procuram construir uma árvore de *clusters* denominada dendrograma. Nessa árvore, cada nó corresponde a um *cluster* que pode ter outros *clusters* filhos, pais ou irmãos de forma que, cada objeto de *n* é associado a um único nó da árvore em um dos seus níveis. Isso permite que os dados sejam manuseados com um nível de granularidade razoavelmente melhor, se a quantidade de objetos tratados for pequena, que os oferecidos por algoritmos de particionamento cuja descrição será apresentada mais à frente.

A literatura apresenta, ainda, uma divisão dos métodos hierárquicos de clusterização em Hierárquico Aglomerativo e Hierárquico Divisivo de acordo com a forma como o dendrograma é construído.

Se a árvore é construída das folhas para a raiz (estratégia *bottom-up*) o algoritmo inicia o processo de clusterização criando *n clusters*, cada um contendo um dos *n* pontos do conjunto *X*. Nesse caso, tem-se o método Hierárquico Aglomerativo. Iterativamente, dois ou mais *clusters* sofrem uma junção para formar um novo *cluster*. Essa junção depende do critério de similaridade que é adotado.

Em um algoritmo dito Hierárquico Divisivo, a árvore é construída da raiz para as folhas (estratégia *top-down*). Ou seja, inicialmente existe um único *cluster* contendo todos os *n* pontos do conjunto *S*, que será dividido a cada nível em dois (ou mais) *clusters* para criar

um novo nível na árvore. A cada nível que vai sendo criado na árvore, os *clusters* são novamente divididos para gerar outros *clusters* de nível inferior.

Comparando os algoritmos de clusterização hierárquico por aglomeração com os de clusterização hierárquico por divisão constata-se que o método divisivo exige um maior número de iterações para ser executado e que dependendo do número de elementos do problema tratado, torna-se inviável sua implementação computacional. Por isso, os algoritmos de clusterização hierárquico por divisão são pouco mencionados na literatura, pois exigem uma maior capacidade computacional que os métodos aglomerativos [KAUFMAN, 1990].

Percebe-se que a Clusterização hierárquica pode ser vista como um problema de particionamento em grafos. Neste sentido, são apresentados trabalhos de excelente nível técnico sobre a utilização de particionamento em grafos para este problema, destacando-se, dentre eles, os trabalhos de [DIAS E OCHI, 2003][SCHRADER 1983].

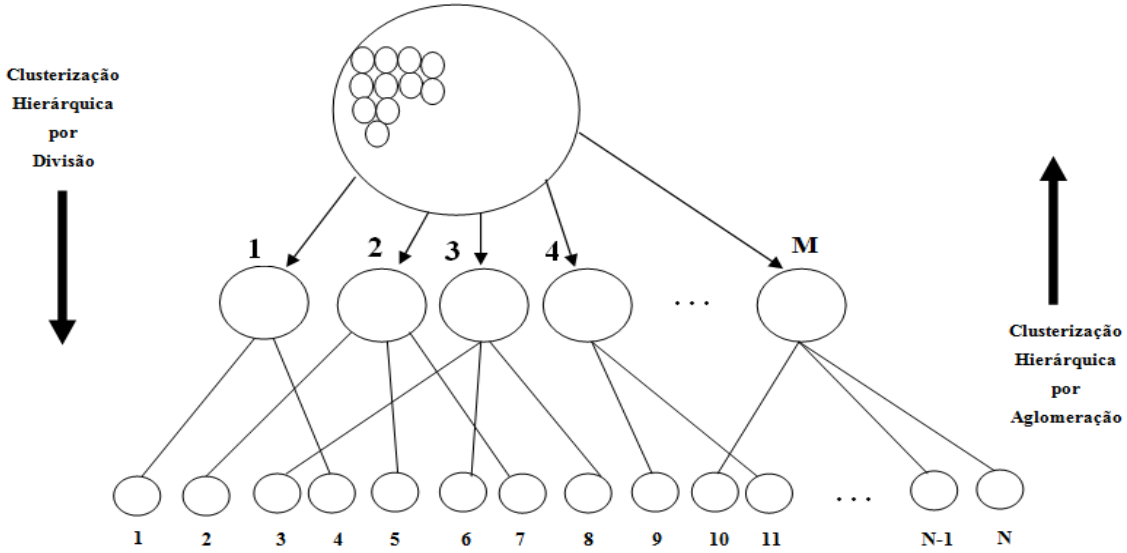


Figura 2.4 Classificação dos algoritmos de clusterização adaptado de SOARES 2004

Diferentemente dos algoritmos hierárquicos, nos algoritmos de particionamento o número de grupos é especificado antes do processo de agrupamento, que se inicia com a divisão dos objetos em *k* grupos. Estes algoritmos melhoram gradualmente os *clusters*. Ou seja, procuram iterativamente agrupar e fazer realocações dos pontos de *X* em *k clusters* de forma a otimizar uma função objetivo *f*. A otimização da função *f* procura assegurar que os pontos pertencentes a um mesmo *cluster* apresentem alta similaridade e os pontos pertencentes a *clusters* distintos tenham baixa similaridade [HAN E KAMBER 2001]. Um algoritmo de particionamento classifica os *n* pontos do conjunto de entrada em *k* grupos de tal

forma que cada grupo contenha, no mínimo, um ponto e que cada ponto pertença a exatamente um grupo. Portanto, o algoritmo de particionamento consiste em agrupar *k* grupos de objetos entre os objetos, ou seja, encontrar grupos "naturais" de objetos para um conjunto de dados não rotulados (tipo de aprendizado não supervisionado). Entretanto, nem todo valor de *k* produz a solução ideal, ou seja, aquela clusterização que reflete fielmente os grupos presentes no conjunto de entrada. [SOARES 2004].

Dentre os algoritmos de particionamento, o mais popular é o algoritmo *K-means*, [JAIN 2010][MACQUENN 1967][STEINLEY AND BRUSCO 2007]. Proposto em 1967, o algoritmo *K-means* representa cada *cluster* pelo centroide do mesmo. O centroide C_m de um *cluster C* é o ponto $C_m = (C_{m1}, C_{m2}, ..., C_{mp})$ onde cada coordenada C_{mj} $(j=1,...,p)$ é dada pela média da *j-e*sima característica dos *t* objetos que pertencem ao cluster C_m, como dado pela Equação 10:

$$C_{mj} = \frac{\sum_i^t x_{ij}}{t} \tag{10}$$

No algoritmo *K-means*, a função objetivo a ser minimizada f leva em consideração a média das distâncias entre cada ponto e o centroide do *cluster* ao qual este pertence. Uma possibilidade para a inicializar o algoritmo consiste em escolher aleatoriamente dentre os *n* pontos para representar o centroide de cada um dos *k clusters*. Cada ponto do conjunto X é associado ao *cluster* cujo centroide seja mais similar, sendo que, cada vez que um ponto é realocado a um *cluster*, o centroide daquele *cluster* é atualizado [SPATH 1980]. Esse processo pode repetir as etapas de associar cada ponto do conjunto X ao centro do *cluster* mais próximo/similar e recalcular/atualizar o centro de cada *cluster* até que nenhum ponto possa ser mudado. Este processo é reiniciado considerando os novos centroides escolhidos até que um critério de parada seja atendido, como um número *i* de iterações sem melhoria na função objetivo.

Uma vez que o algoritmo *k-means* considera os centroides, este método não pode ser aplicado em bases de dados cujos objetos tenham atributos qualitativos. Em geral, o algoritmo *k-means* tradicional tende a produzir ótimos locais de qualidade apenas razoável. Em função disso, são encontradas na literatura algumas propostas de algoritmos (variantes de *k-means*) que trabalham a mesma função objetivo de *k-means*, mas que consideram métodos de busca mais eficientes, sejam eles: o global, *k-means* [MACQUEEN, 1967] [PENA ET AL, 1999], o *j-means* [HANSEN AND MLADENOVIC 2001] e o algoritmo de suavização

hiperbólica [XAVIER 2009]. Estes algoritmos tendem a produzir soluções de qualidade superior àquelas produzidas pelo *k-means*.

2.5 O PROBLEMA DE CLUSTERIZAÇÃO COM RESTRIÇÕES DE CAPACIDADE E CONEXIDADE (PC-RCC)

O Problema de Clusterização também pode ser mapeado em um problema de particionamento de grafos em subgrafos disjuntos, estando cada subgrafo associado a um *cluster* [DOVAL ET. AL, 1999] [DIAS 2004][DIAS AND OCHI 2003][BOLEY ET AL 1999]. Alguns problemas de clusterização têm restrições de capacidade mínima por *cluster* [SHIEH AND MAY 2001][SCHEUERER 2006] e outros agregam a restrição de conexidade [ASSUNÇÃO ET AL, 2006] [NEVES 2003], dentre outras restrições.

Em particular, o presente trabalho traz uma proposta de resolução para o problema de clusterização que agrega tanto a restrição de capacidade, quanto a restrição de conexidade [SEMAAN 2010][SEMAAN 2009][SEMAAN 2008][SHIEH AND MAY 2001]. Soma-se a estes trabalhos o trabalho de BRITO, ET AL 2004 que traz uma proposta de resolução para este problema considerando a aplicação de uma formulação de programação inteira.

A seguir é realizada a descrição do problema de clusterização com as restrições de capacidade e de conexidade onde tem-se a definição das áreas de ponderação (APONDs).

2.5.1 DESCRIÇÃO DO PROBLEMA (PC-RCC)

O problema de clusterização de uma forma geral, ocorre por exemplo, em situações nas quais a regionalização geográfica torna-se necessária, isto é, nos casos dos censos demográficos e de pesquisa socioeconômicas [OPENSHAW 1977].

O problema de Clusterização com Restrição de Capacidade e Conexidade (PC-RCC) pode ser descrito por um grafo $G = (V, A)$. Um grafo não direcionado G consiste de um conjunto $V = \{v_1,...,v_i,...,v_j,...,v_n\}$ de elementos, chamados de vértices, e de uma relação binária A entre eles, denominada aresta. Cada elemento de A corresponde, portanto, a um par de vértices $[v_i, v_j]$ pertencentes ao conjunto V. Por definição, $|V|$ e $|A|$ correspondem, respectivamente, ao número de vértices e ao número de arestas de G. Por exemplo (Figura 2.6): uma rede social pode ser representa através de grafo, onde os vértices representam o nome dos perfis e as arestas representam as ações dos perfis sejam elas, compartilhamentos ou comentários de mensagens. *A*

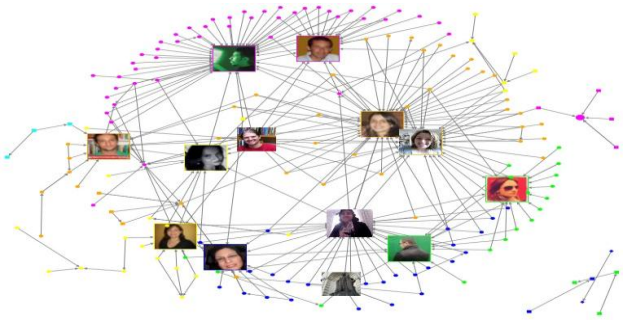
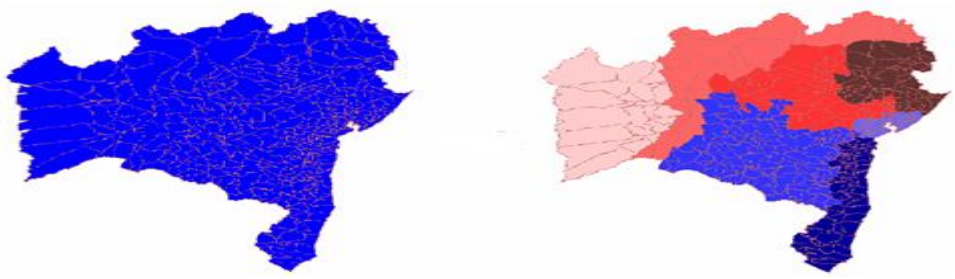


Figura 2.5 Grafo de uma Rede Social [GOOGLE IMAGENS 2014]

Considerando o PC-RCC determinado nesta tese, cada vértice estabelece a relação de vizinhança entre duas arestas é expressa através de uma aresta $[v_i, v_j] \in A$ e associa-se cada vértice v_i o valor correspondente a uma capacidade mínima estabelecida no problema tratado.

O objetivo do PC-RCC é minimizar a função objetivo correspondente à Equação 7 ($f(P) = \sum_{c=1}^{k} Som_c$) que consiste em definir o menor número possível de homogeneidade entre agrupamentos similares. Desta forma, o problema tratado nesta tese, define duas restrições a saber: - Restrição 1 – Conexidade: a restrição de conexidade implica, que cada agrupamento formado, devem existir vizinhos, com pelo menos, um caminho simples, onde dois vértices tomados dois a dois, devem possuir contiguidade; - Restrição 2 – Capacidade Mínima: a restrição de capacidade mínima determina que cada vértices de um grafo possui um valor referente alguma informação de cunho quantitativo.

Dessa forma, o trabalho busca tratar as restrições 1 e 2, simultaneamente. Vejamos uma exemplificação a seguir: inicialmente tem-se um único aglomerado de setores censitários Figura 2.6(a) e a Figura 2.6(b) onde define-se a formação de aglomerados conexos e com capacidade mínima de 200 domicílios por setor, por exemplo, assim tem-se a formação de 07 aglomerados cada um por uma cor diferente.



(a) **(b)**

Figura 2.6 Clusterização com Restrição de Conexidade e Capacidade Mínima – (a) Um único aglomerado de Setores Censitários (b) Aglomerados Conexos e Com Capacidade Mínima [NEVES 2003]

O problema de Clusterização com Restrição de Capacidade e Conexidade (PC-RCC) possui uma formulação matemática definida em Brito et. al, 2004. Nessa formulação são consideradas as seguintes notações e restrições:

V = Quantidade de vértices

K = Quantidade de *clusters*

P_j = População associada a cada vértice *j*

E^* = Arestas da Árvore Gerador Mínima

X_i = Variável Inteira que determina a quantidade de vértices que estão no *i-é*simo *cluster*

Y_{ij} = Variável Binária que assume valor 1 se o *j-é*simo vértice está associado ao *i-é*simo *cluster*

e^i_{mn}= Variável Binária: assume valor 1 se a aresta $(m,n) \in E^*$ está ativa no *i-é*simo *cluster*

t^i_{mn}= Variável Binária: assume valor 1 se a aresta $(m,n) \in E^*$ está ativa no *i-é*simo *cluster*

D_m = Medida que representa o grau de homogeneidade entre dois *clusters* vizinhos.

Função Objetivo está associada ao critério de soma mínima. Ou seja, o valor desta função representa a soma do melhor subconjunto de arestas ativas associadas aos valores D_m .

Restrições 1 e 2: A quantidade de vértices associados a cada um dos *clusters* deve ser igual ao total de vértices da instância.

Restrições 3: Garante que um vértice estará associado a exatamente um *cluster*.

Restrição 4: A soma do atributo referente à capacidade mínima associada a cada um dos vértices, em cada um dos *clusters*, deve ser maior ou igual a capacidade mínima pré-estabelecida.

Restrição 5: Garante que cada uma das arestas associadas à árvore geradora que contém todos os vértices só poderá estar em no máximo um *cluster*. Ou seja, se dois vértices são vizinhos e estão em um mesmo *cluster*, então e^i_{mn}= 1.

Restrição 6: A quantidade de arestas associadas a cada um dos *clusters* deve ser igual a quantidade de vértices em cada um dos *clusters* menos um.

Restrições 7 a 10: Estas restrições garantem que se uma aresta do conjunto E^* estiver vazia, ou seja, pertencer a uma subárvore associada a um *cluster*, então os vértices *m* e *n* desta aresta também farão parte do *cluster*.

Minimizar $\sum_{i=1}^{k} \sum_{\forall (m.n) \in E^*} e^i_{mn}$. D_m

(1) $\sum_{i=1}^{k} X_i$= V

(2) $\sum_{j=1}^{V} Y_{ij} = X_i, \quad i=1,...,k$

(3) $\sum_{j=1}^{k} Y_{ij}=1, \quad j=1,...,V$

(4) $\sum_{j=1}^{V} Y_{ij} \; . P_j, \quad i=1,...,k$

(5) $\sum_{i=1}^{k} e_{mn}^{i} \leq 1, \forall (m.n) \in E^*$

(6) $\sum_{\forall (m.n) \in E^*} e_{mn}^{i} = X_i - 1$, *i=1,...,k*

(7) $t_{mn}^{i} \leq Y_{im}$ *i=1,...,k* $\forall (m.n) \in E^*$

(8) $t_{mn}^{i} \leq Y_{in}$ *i=1,...,k* $\forall (m.n) \in E^*$

(9) $Y_{im} + Y_{in} - 1 \leq t_{mn}^{i} \; \forall (m.n) \in E^*$

(10) $e_{mn}^{i} \leq t_{mn}^{i}$ *i=1,...,k* $\forall (m.n) \in E^*$

$X_i \in Z_+, \quad Y_{ij} \in (0,1), \quad t_{mn}^{i} \in (0,1)$

Na próxima Seção será tratada a modelagem do problema.

2.5.2 MODELAGEM DO PROBLEMA (PC-RCC)

A partir da descrição do problema feita na seção anterior, observa-se que o problema Clusterização com Restrição de Capacidade e Conexidade (PC-RCC), no contexto do censo geográfico, como no caso desta tese, os objetos podem estar associados a domicílios, a setores censitários, às áreas de ponderação (APONDs) e a municípios.

Uma APOND é definida como uma unidade geográfica formada por agrupamentos mutuamente exclusivos de setores censitários, sendo esses, por sua vez, formados por conjuntos de domicílios. Esta unidade é utilizada para se estimar informações para a população e, também, os níveis geográficos mais detalhados da base operacional, sendo tais áreas definidas como forma de atender às demandas por informações em níveis geográficos menores que os municípios.

Em geral, uma APOND é formada por um grupamento de setores censitários que têm, em média, 300 domicílios [CENSO DEMOGRÁFICO 2010]. O tamanho dessas áreas, em termos do número de domicílios e de população, não pode ser muito reduzido, sob a pena de perda de precisão de suas estimativas. As áreas de ponderação foram definidas considerando essa condição e, também, os níveis geográficos mais detalhados da base operacional, como forma de atender a demandas por informações em níveis geográficos menores que os municípios [SILVA 2004] [CENSO DEMOGRÁFICO 2010].

Cada APOND [BRITO e MONTENEGRO 2010] é definida de forma a satisfazer os

seguintes critérios:

Um número mínimo de domicílios por APOND (para permitir a obtenção de estimativas com qualidade estatística em áreas pequenas);

- A contiguidade: os setores que constituem cada uma destas áreas devem possuir fronteira em comum ou deve ser possível sair de um setor A e chegar a um setor B, ambos em uma mesma área, passando apenas por setores que também estejam nesta mesma área;

- Os setores que compõem estas áreas devem ser homogêneos em relação a um conjunto de características (variáveis) populacionais e de infraestrutura. Ou seja, considerando que duas APONDs *i* e *j* são vizinhas são calculadas as distâncias d_{ij} obtido através da aplicação da Equação abaixo:

$$DEuclidiana(x_i, x_j) = \left[\sum_{l=1}^{p}(x_{il} - x_{jl})\right]^{1/2}$$

Estas distâncias fornecem o grau de homogeneidade entre os setores censitários a serem agregados. (x_i,x_j)

Assim, observa-se que o problema de definição de áreas de ponderação corresponde a um problema de agrupamento com uma restrição de capacidade (número mínimo de domicílios) e de conexidade (contiguidade entre os setores). Desta forma, é necessário associar tanto as informações relativas à contiguidade (conexidade) das APONDs, quanto as informações da capacidade dos vértices a uma modelagem matemática que fosse útil para caracterizar esse problema de agrupamento e suas restrições.

Para fins de exemplificação, considere o mapa abaixo (Figura 2.7(a)) com cinco setores censitários e o seus grafos associados na Figura 2.7(b), Grafo Completo, e Figura 2.7(c)), o Grafo Original.

(a) **(b)** **(c)**

Figura 2.7 - Representação Setores Censitários (a), Grafo Completo (b) e Grafo Original (c) [BRITO ET AL, 2010]

O Grafo Completo, Figura 2.7 (b) representa a existência de contiguidade entre dois setores i e j é, portanto, representada em G através de cada aresta A ligando os vértices v_i e v_j. Além disso, o Grafo Completo apresenta cinco vértices acima, onde serão calculadas todas as distâncias (homogeneidade) d_{ij} (distância euclidiana) entre cada dois vértices. A partir do Grafo Completo é obtido um subgrafo que contém subconjunto de arestas de *A*, ou seja, contém as arestas *A'* que indicam a relação de vizinhança entre os vértices, portanto representam a relação de contiguidade entre os setores, denominado neste trabalho como Grafo Original e correspondente ao grafo *G=(V, A')* .Assim, através do Grafo Original *G* , pode-se definir *k* áreas de ponderação mediante a remoção de um subconjunto de *A'* de *G*, de modo que se tenha *n* grafos conexos (contiguidade).

A restrição de conexidade implica que, em cada APOND, todos os setores tomados dois a dois, devem ser vizinhos ou deve existir, pelo menos, um caminho passando apenas pelos vértices pertencentes a mesma APOND. Este trabalho, em seu estudo de caso, utiliza instâncias obtidas a partir dos dados do Censo Demográfico de 2010 (disponíveis para uso público). Essa tese também trata do estudo do problema de clusterização com restrição de capacidade, assim além de garantir a conexidade procura-se conjuntamente garantir a capacidade. Mais especificamente, cada vértice $i \in V$ do grafo corresponderá a uma região e a seus atributos, (inclusive o atributo capacidade). Nesse caso, a restrição de capacidade é uma função de um limite inferior pré-estabelecido, com base em um dos atributos dos vértices do grafo V. Assim, o somatório desse atributo nos vértices pertencentes a um mesmo cluster deve satisfazer um limite inferior.

Além disso, se duas regiões *i* e *j* são vizinhas, existe uma aresta $A' = (i, j)$ cujo valor da distância é d_{ij}, considerando a distância euclidiana, e que existe um atributo associado a cada vértice que é o atributo utilizado como base para se verificar a restrição de capacidade mínima por APOND, nesta tese utiliza-se o atributo total de domicílios particulares permanentes em cada setor censitário. Por domicílio particular permanente denomina-se o domicílio que foi construído a fim de servir exclusivamente para habitação e, na data de referência, tinha a finalidade de servir de moradia a uma ou mais pessoas [CENSO DEMOGRÁFICO 2010].

Dentre as várias abordagens encontradas na literatura para a resolução do problema de clusterização com restrição de conexidade, destaca-se o algoritmo baseado na utilização da Árvore Geradora Mínima (AGM) [NEVES 2003], que também será considerado neste trabalho.

No que concerne à restrição de conexidade do problema abordado neste trabalho, os *clusters* conexos serão produzidos mediante a realização da construção e do particionamento da AGM.

As informações básicas para a construção da AGM são o grafo de conectividade G e as medidas de similaridade entre os objetos. A AGM é construída de forma iterativa. Mais especificamente, o processo tem início a partir de uma árvore T_i, contendo apenas um vértice e, a cada iteração, uma nova aresta e um novo vértice são adicionados à árvore anterior. Na iteração n, a árvore T_n contém todos os n vértices de V e possui um subconjunto de L, contendo $n-1$ arestas, para o qual a soma dos custos associados às arestas é mínima. As etapas utilizadas para a geração da AGM, baseados no algoritmo de *Kruskal*, são:

Etapa 1: Consiste em ordenar, de forma crescente, as arestas (distâncias) do grafo submetido conforme seus custos. Em seguida, a árvore T é iniciada vazia.

Etapa 2: Na árvore T , conforme é realizada a ordenação das arestas, a cada iteração a inserção de uma aresta em T é avaliada uma vez que não pode ocorrer ciclos em T.

Etapa 3: As inserções ocorrem até que a árvore T tenha $|V|-1$ arestas, considerando $|V|$ a quantidade de vértices do grafo submetido ao algoritmo.

Etapa 4: Ao final da execução, o algoritmo retornará uma árvore conexa T com menor custo possível, de acordo com Equação 11 a seguir:

$$min \sum_{A \in T} C_e \tag{11}$$

Ao final da execução das etapas do algoritmo de *Kruskal*, produz-se a árvore geradora mínima, que contém todos os vértices representando todos os objetos. A solução final possui uma AGM com custo mínimo C_e (soma mínima das distâncias associadas às arestas de *G*).

Para encontrar o valor de C_e utiliza-se o exemplo hipotético a seguir, segundo Neves 2003. Tem-se a ilustração (Figura 2.8) da construção de uma árvore geradora mínima para um caso hipotético.

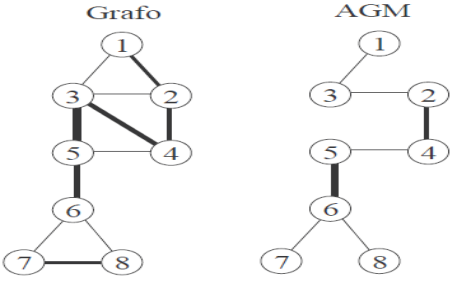


Figura 2.8 - Construção de AGM Adaptado de NEVES 2003

Segundo Neves 2003, as larguras das arestas que ligam os vértices vizinhos são inversamente proporcionais aos coeficientes de similaridades entre os objetos, ou seja, objetos vizinhos semelhantes são ligados por arestas finas (baixo custo). Assim, conclui-se que ao final que a AGM corresponde a um subgrafo desconexo de custo mínimo C_e, contendo somente as arestas mais finas ("menor custo") é obtida através da Equação 3.

A partir deste ponto, procede-se à fase de particionamento da AGM, considerando que cada aresta que for eliminada dividirá a árvore em dois subgrafos conexos, correspondentes a dois *clusters*.

Na Figura 2.9 ilustra uma AGM e os dois grafos conexos que correspondem a duas áreas de ponderação (APONDs):

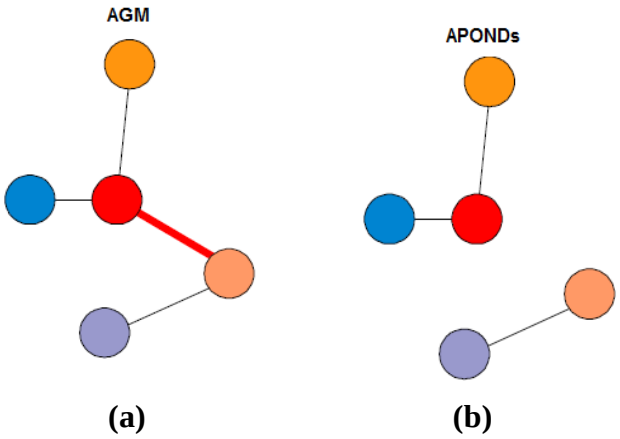


Figura 2.9 - AGM (a) e APONDs (b) [BRITO ET AL 2010]

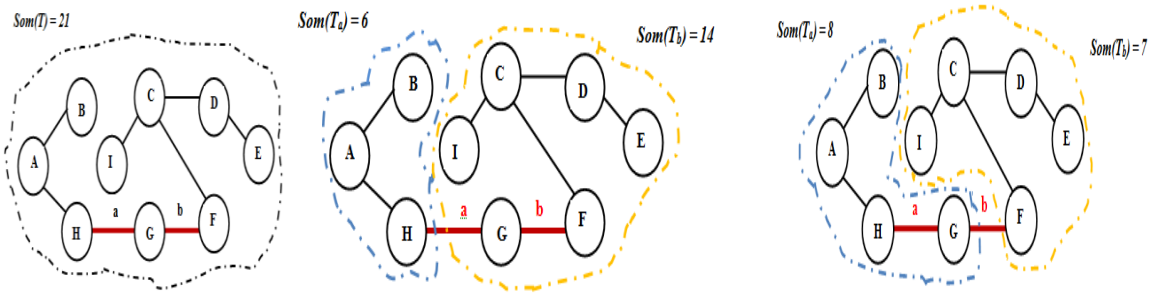
Essa fase consiste em retirar as arestas mais "caras" (custo alto), ou seja, particionar a AGM mediante a remoção de $n-1$ arestas considerando que T tenha $|V|-1$ arestas disponíveis para a obtenção de *k* grafos conexos (*clusters*). Então, considerando $|V|$ a quantidade de vértices do grafo submetido a este algoritmo. A AGM terá, então, $n-1$ arestas. Ou seja, para produzir *k clusters* deve-se remover da AGM as *(k – 1)* arestas que produzem os subgrafos mais homogêneos no que concerne às distâncias.

Existem várias alternativas possíveis para definir o custo de uma aresta. Por exemplo, na etapa da construção da AGM, o custo das arestas do grafo submetido ao algoritmo é calculado através da Equação 3 ($D_{Euclidiana}(x_i, x_j) = \left[\sum_{l=1}^{p}(x_{il} - x_{jl})\right]^{1/2}$). E durante a fase do particionamento do grafo, utiliza-se a Equação 12 para a obtenção de seu valor.

$$Custo_a = Som(T) - (Som(T_a) + Som(T_b) \quad (12)$$

A etapa de particionamento consiste em selecionar um *cluster* T para divisão, buscar a aresta existente entre vértices desse *cluster* que possuir o maior $Custo_a$ e removê-la da árvore. Para isso, deve-se remover cada uma das arestas de T e aplicar à Equação 7 ($f(P) = \sum_{c=1}^{k} Som_c$) em cada um dos novos *clusters* T_a e T_b. Em seguida, com base nos valores da função obtidos em cada um dos novos *clusters*, deve-se utilizar a Equação 12 para o cálculo do custo da aresta. O valor do custo de remoção da aresta corresponde à diferença entre a função objetivo do *cluster* atual e a soma da função objetivo de cada um dos novos *clusters*, sendo a soma da função objetivo dos novos *clusters* correspondente à soma dos quadrados dos desvios das duas subárvores T_a e T_b calculados os valores médios dos p atributos. Os dois novos *clusters* formados serão homogêneos à medida que o custo da aresta ($Custo_a$) a ser removida seja o maior possível. Assim, um custo alto associado à aresta indica que a sua remoção irá gerar *clusters* mais homogêneos.

Para ilustrar o cálculo do custo de remoção de uma aresta ($Custo_a$) da AGM, na Figura 2.10 é ilustrado o particionamento de um *cluster* formando dois novos *clusters*, considerando a remoção das arestas *a* e *b*.



(a) **(b)** **(c)**

Figura 2.10 - Exemplo do Cálculo do Custo de remoção de uma aresta

A Figura 2.10 (a) ilustra a solução formada com um *cluster* associado à subárvore T que, hipoteticamente, possui um valor de função objetivo obtido pela Equação 7 (Som_c) igual a 21.

Já a Figura 2.10 (b) apresenta uma solução, caso a aresta ***a*** seja removida formando os dois novos *clusters* associados às subárvores T_a com valor de função objetivo igual a 6 e T_b com valor de função objetivo igual a 14. Assim, o custo da remoção da aresta *a* é igual a 1, utilizando a Equação 11 ($Custo_a$).

Na Figura 2.10 (c) as subárvores T_a e T_b possuem valores da função objetivo iguais a 8 e 7. Portanto, o custo da remoção da aresta ***b*** é igual a 6, utilizando novamente a Equação 12 ($Custo_b$).

Como se está maximizando o custo da aresta a ser removida, percebe-se que a remoção da aresta ***b*** produz a melhor solução em relação à aresta ***a***.

Este critério de particionamento da AGM foi proposto em um algoritmo denominado SKATER (*Spatial 'K'luster Analysis by Tree Edge Removal*) que foi desenvolvido por um pesquisador da Universidade Federal de Minas Gerais utilizando o método de agrupamento com restrição de contiguidade por árvore geradora mínima.

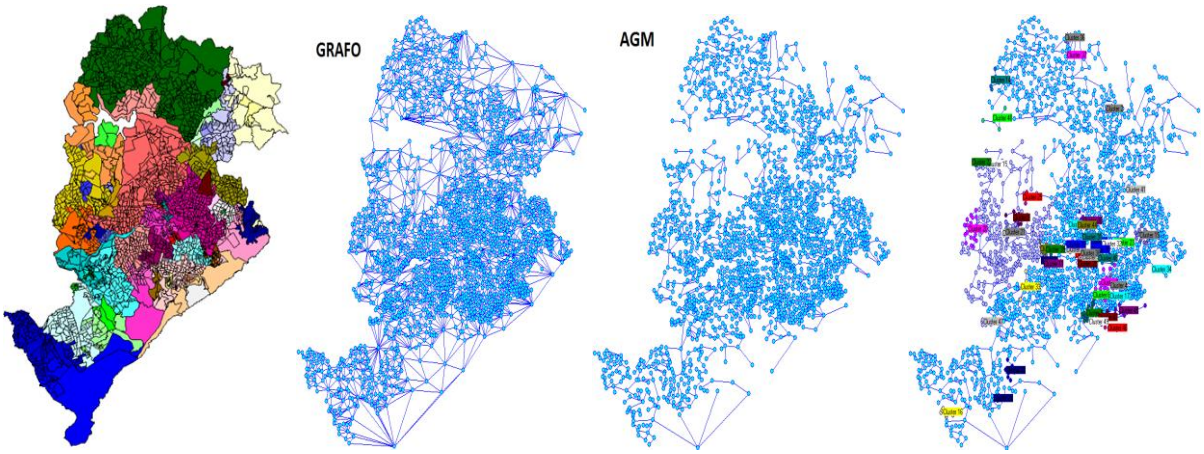
A descrição do algoritmo e do funcionamento do software desenvolvido (elaborado em linguagem C++ para o ambiente *Windows*) pode ser encontrada em [NEVES ET AL 2002].

O SKATER utiliza um dos seguintes três critérios para obtenção de soluções:

- Construir *k clusters* sem estipular a restrição de capacidade;
- A capacidade de cada *cluster* é independente da quantidade de *clusters* construídos;
- O número máximo de *clusters* é denominado por uma capacidade mínima *Cap* informada previamente.

Considerando as opções existentes no SKATER, é possível afirmar que o algoritmo não atende plenamente a demanda do problema de clusterização com restrições de capacidade e conexidade, uma vez que não é possível fixar simultaneamente tanto a quantidade de *clusters*, quanto a capacidade mínima de cada *cluster*.

A Figura 2.11 ilustra o funcionamento do SKATER em quatro estágios diferentes no que se refere à área de Belo Horizonte, mais especificamente, à clusterização de seus setores censitários. Em 2.11(a) considerando aspectos socioeconômicos e condições dos domicílios; a Figura 2.11 (b) o grafo contendo a vizinhança entre os setores censitários; a Figura 2.11 (c) a árvore geradora mínima a partir do grafo original e a Figura 2.11 (d) as subárvores obtidas após o particionamento da AGM com a formação de 50 *clusters*, representados por cores diferentes.



(a) **(b)** **(c)** **(d)**

Figura 2.11 - Representação da região, do grafo, da AGM e do *cluster* de BH [NEVES 2003]

O SKATER pode ser obtido gratuitamente no site do Laboratório de Estatística Espacial (LESTE) do Departamento de Estatística, da Universidade Federal de Minas Gerais [NEVES 2003].

2.6 REVISÃO BIBLIOGRÁFICA

Esta seção traz uma revisão bibliográfica dos principais trabalhos disponíveis na literatura concernente ao problema de clusterização clássico e às suas variantes.

Em CERVEIRA (2014) é apresentado um projeto de otimização para encontrar o *layout* ideal para parques eólicos. Propõe-se um modelo matemático para encontrar a melhor configuração de interconexão elétrica das turbinas de parques eólicos e a subestação. O objetivo é minimizar os custos de instalação, que incluem os custos de custos de cabo e instalação de cabos, tendo em conta limitações técnicas. Este problema corresponde a uma árvore geradora mínima capacitada. A metodologia é aplicada em um estudo de caso real e os resultados são comparados com a solução de solo. Em MURTY ET AL (2014) é criado uma nova medida de similaridade/dissimilaridade para formação dos agrupamentos de *clusters*, chamada de separação por homogeneidade. Essa nova medida é adequada para conjuntos de quaisquer formas, tamanhos e/ou de diferentes densidades. Os principais conceitos da medida proposta são explicadas e resultados experimentais com conjunto de dados reais e artificiais que apoiam a medida proposta são analisados. No trabalho de SANTOS ET AL (2013) é apresentado um algoritmo híbrido baseado na metaheurístca GRASP com o procedimento VND. O GRASP utilizou o VND como forma de intensificação das buscas locais procurando-se, desta forma, melhorar a qualidade das soluções do problema de clusterização com

restrições de capacidade e de conectividade. Ao final do trabalho foram realizados experimentos com instâncias reais obtidas em Censo Demográfico 2010.

No trabalho de CESELLI (2013) considera-se o problema de particionamento de uma região em áreas ligadas e atribuídas a administradores. Foi definido um problema de áreas homogêneas, em que é feita a busca de uma partição adequada de grafo através da atribuição de pesos aos vértices e um subconjunto de pesos pertencentes a um grafo não direcionado com seus componentes conectados. Os autores propõem uma formulação baseada em fluxo multi-produto através da utilização dos procedimentos de Busca Tabu e da Busca em Vizinhança de Grande Porte (VLSN). Em DAMODAR e PRASANTA (2013) é proposta é uma abordagem divisiva para acelerar a clusterização baseada em Árvore Geradora Mínima, e também para identificar os clusters que são formados arbitrariamente durante a forma de uma AGM. Inicialmente, o quadrado do erro do algoritmo de clusterização é usado no pré-processamento de clusterização da AGM. Então, a abordagem usando AGM é aplicada sobre os pontos representativos (centroides) de sub-*clusters* produzidos pelo método de agrupamento usando o quadrado do erro.

VANETIK (2013) trabalha com grafos com restrição de conectividade adaptados para a mineração de dados. Neste artigo, apresenta-se um novo algoritmo para encontrar padrões de grafos com conectividade prescrita em grandes conjuntos de dados a partir de um grafo simples. A autora também demonstra como seu algoritmo pode ser adaptado a um ambiente dinâmico, onde são feitas mudanças nos dados ao longo do tempo. Em BONAMI ET AL (2012) é proposta pela primeira vez a solução do problema de particionamento de grafos com restrições de capacidade para o planejamento de redes ópticas. Trata de uma variação do problema de particionamento de grafos em que o peso de um *cluster* na partição depende das arestas incidentes nos vértices.

No trabalho de SANTOS ET AL (2012) é apresentado um algoritmo híbrido baseado na metaheurístca GRASP com o procedimento *Path-Relinking* para a solução do problema de clusterização com restrições de capacidade e de conectividade.

O trabalho de ZAGOURAS (2012) traz a proposta de um método avançado para classificar os tipos de circulação atmosférica com base em protótipos de grafos conectados entre si. Assim é apresentado um novo método de classificação de circulação atmosférica de mapas isobáricos. O método baseia-se na teoria dos grafos, especificamente na formação de *clusters* com restrição de conectividade.

Em COLGANO (2012) é proposta uma definição formal do problema de clusterização em grafos. Neste trabalho, o problema de clusterização em grafos é descrito como o problema

de encontrar uma solução que satisfaça um conjunto de restrições e que minimize uma função objetivo. O trabalho de CORTEZ ET AL (2012) propõe a resolução do problema de definição das áreas de ponderação do Censo 2012 considerando o seu mapeamento em um problema de clusterização em grafos. Em DENG e BRAD (2011) é proposto um algoritmo baseado na metaheurística (GRASP) e no procedimento *Path-Relinking* (PR) para resolver o problema do agrupamento em um grafo com *n* nós que será particionado em *p* subgrafos associados aos *clusters*. O objetivo é o de maximizar a soma dos pesos das arestas dentro de cada *cluster,* de tal modo que a soma dos pesos dos vértices correspondentes não exceda uma capacidade fixa.

Em SEMAAN (2010) são propostos algoritmos heurísticos (Evolutivos e GRASP) para o problema de clusterização, sendo o número de *clusters* é definido previamente. Neste trabalho o problema de clusterização foi mapeado em um problema de particionamento de grafos com restrições de capacidade e conexidade, em que experimentos foram realizados em instância reais (de uso público) do Censo Demográfico de 2000 e da Contagem Populacional 2007, obtidas no site do IBGE. Ao final do trabalho, observa-se a superioridade do Algoritmo Evolutivo em relação ao GRASP no que concerne à qualidade das soluções. Em BRITO e MONTENEGRO (2010) define-se um novo método de resolução para o problema de definição de áreas de ponderação. O mesmo pode ser mapeado em um problema clássico de agrupamento com restrições de capacidade e conexidade através da aplicação de um algoritmo que combina a construção da árvore geradora com um procedimento baseado na metaheurística VNS.

CAPÍTULO 3 – HEURÍSTICAS PARA O PROBLEMA DE CLUSTERIZAÇÃO

O presente capítulo traz uma descrição dos algoritmos heurísticos propostos para a resolução do problema de clusterização que foi o objeto de estudo desta tese. Conforme já comentado, a resolução do problema abordado não se restringe a uma simples minimização de uma função objetivo, mas também leva em conta o atendimento de restrições de capacidade e conexidade.

Neste contexto, não existem muitos trabalhos relacionados ao problema de clusterização com restrições de capacidade e de conexidade. Mais especificamente, há os trabalhos de [SEMAAN 2010][SEMAAN 2009] [SEMAAN 2008][SHIEH AND MAY 2001] e de [BRITO 2004]. Os dois primeiros autores propuseram abordagens que utilizam metaheurísticas e o último autor propôs uma formulação de programação inteira.

O objeto deste capítulo é propor alguns algoritmos heurísticos para a solução do problema de clusterização com restrições de capacidade e conexidade. Basicamente, são propostos algoritmos que utilizam os conceitos das metaheurísticas GRASP e VNS.

De forma a garantir a produção de soluções válidas para esse problema, o desenvolvimento destes algoritmos levou em consideração as seguintes questões: como representar uma solução, quais procedimentos de construção das soluções iniciais e buscas locais deveriam ser aplicados, qual refinamento deveria ser realizado em relação às soluções produzidas pelas buscas locais e como adaptar o procedimento de Reconexão por Caminhos (RC) e também o método de Descida em Vizinhança Variável (VND) ao problema.

3.1 GRASP

O GRASP (*Greedy Randomized Adaptative Search Procedure*) é uma metaheurística iterativa, gulosa, randômica e adaptativa que foi inicialmente proposta para resolver o problema do conjunto máximo independente [FEO E RESENDE, 1995]. Posteriormente, esta metaheurística foi aplicada ao problema de escalonamento de tarefas [BINATO ET AL, 2001], ao problema de planarização de grafos [RESENDE E RIBEIRO, 1997], ao problema de satisfabilidade (SAT) [RESENDE AND FEO, 1996], ao problema quadrático de alocação [LI ET. AL, 1994], e atualmente o GRASP é um dos métodos heurísticos mais populares da literatura.

O GRASP é composto por duas fases, a saber: a de construção de uma solução inicial e a de aplicação de uma busca local sobre esta solução, com o objetivo de tentar melhorá-la.

Na etapa de construção uma solução é construída iterativamente. Mais especificamente, em cada iteração desta fase, apenas um elemento é adicionado à solução a cada passo. A seleção do próximo elemento é guiada pela aplicação de uma função gulosa que avalia os benefícios ganhos com a inserção deste elemento na solução em construção, também chamada de solução parcial. A cada iteração desta fase, o próximo elemento a ser adicionado à solução parcial é escolhido dentre os elementos de uma lista restrita de candidatos (LRC), selecionado de forma aleatória, o que garante o aspecto probabilístico do método. Essa LRC é formada por um subconjunto dos melhores candidatos, isto é, aqueles cuja incorporação à solução parcial resulta nos melhores ganhos incrementais (aspecto guloso do algoritmo), no que concerne à função objetivo do problema. Uma vez que o elemento selecionado foi incorporado à solução parcial, a LRC é atualizada e os custos incrementais são reavaliados (aspecto adaptativo)

Os elementos contidos na LRC devem estar no intervalo definido pela Equação 13 por:

$$LRC=\{i|\ g_i \leq g_{min} + \alpha\ (g_{max} - g_{min})] \qquad (13)$$

Onde g_{max} *e* g_{min} são, respectivamente, o elemento de maior e o elemento de menor incremento na solução (pior e melhor candidato num problema de minimização) e α um parâmetro de entrada que pode variar de *0* a *1*. Quanto mais próximo de *1* for o valor de α, mais aleatórias são as soluções geradas devido à diversidade de valores que podem ser encontradas na LRC. A partir de uma LRC, seleciona-se um elemento deste conjunto, aleatoriamente, sendo este não necessariamente o melhor candidato. Esta escolha aleatória permite que este procedimento possa ser usado várias vezes para obter soluções distintas.

Na busca local, o objetivo é tentar melhorar a solução inicial *s* produzida na fase de construção. Neste sentido, a cada iteração é gerado um conjunto de soluções na vizinhança de *s*, denotado por *N(s)*. Esse conjunto é produzido através da substituição iterativa da solução corrente por uma solução melhor resultando na nova solução atual *s*. A melhor solução em *N(s)* é escolhida para ser a solução atual, caso esta solução tenha um custo inferior ao custo de *s* (no caso de um problema de minimização).

Nesta fase de busca local, a vizinhança da solução corrente é explorada com o objetivo de encontrar uma solução melhor, ou seja, um ótimo local de qualidade superior (s^+) ou um ótimo global (s^*).

Em um problema de minimização, assim como neste trabalho, entende se que uma solução *s+ representa um ótimo local na vizinhança N(s)* de *s* se $f(s^+) \leq f(s)$, $\forall s \in N(s)$.

Ao buscar-se o desenvolvimento de um bom procedimento de busca local, as seguintes questões devem ser levadas em conta: a definição da vizinhança, a estratégia de busca na vizinhança e o tempo computacional consumido em cada iteração do procedimento. A figura 3.1 traz um pseudocódigo simplificado do GRASP.

Algoritmo 1: GRASP

Entrada: *MaxIter*

1. melhorSolução ← 0;
2. **para** *nIter – 1 , ... , MaxIter* **faça**
3. solução ←AlgoritmoConstrução;
4. melhorSolução ← AlgoritmoBuscaLocal (solução);
5. **fim_para**

Saída: melhorSolução

Figura 3.1 – Pseudo-código do GRASP

3.2 ASPECTOS COMUNS AOS ALGORITMOS

A presente seção traz uma descrição das características comuns aos algoritmos implementados neste trabalho.

Inicialmente, será apresentada a estrutura de dados que foi adotada para a representação de uma solução do problema de agrupamento. Em seguida, faz-se uma descrição dos procedimentos de construção. Esses procedimentos utilizam heurísticas que efetuam o particionamento da árvore geradora mínima (AGM) construída a partir do grafo associado ao problema de clusterização, considerando as restrições de capacidade e de conexidade.

Após a descrição desses procedimentos, são apresentados os procedimentos de busca local que realizam migrações de vértices entre os *clusters* objetivando, simultaneamente, a minimização da função objetivo e o atendimento das restrições de capacidade e de conexidade.

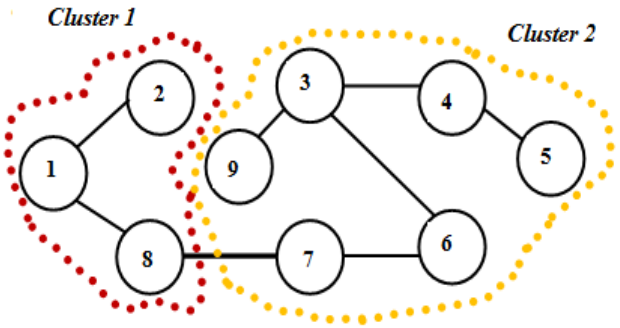
A descrição é finalizada com a apresentação de procedimentos que correspondem a um refinamento nas buscas locais. Mais especificamente, procedimentos que incorporam o método de reconexão por caminhos (RC) e um método VND. Ambos os procedimentos têm a finalidade de intensificar a busca por soluções de melhor qualidade.

3.2.1 REPRESENTAÇÃO DA SOLUÇÃO

É fato conhecido da literatura que a forma de representação da solução tende a impactar na convergência do algoritmo no que diz respeito à obtenção de soluções válidas [HARTUV AND SHAMIR, 1999].

Este trabalho utilizou a representação da solução denominada *group-number* [TRINDADE AND OCHI, 2006]. Nessa estrutura de representação de solução, utiliza-se um vetor (ou *array*) ou, de maneira mais formal, uma estrutura de dados homogênea em que cada índice representa o vértice do grafo e o seu conteúdo é um número inteiro positivo entre *1* e *k*, ou seja, é o *cluster* ao qual o vértice está associado.

A Figura 3.2 ilustra a representação de uma solução através da estrutura *group-number*.



Indice do Vetor/Vértice do Grafo	1	2	3	4	5	6	7	8	9
Cluster inserido vértice do Grafo	1	1	2	2	2	2	2	1	2

Figura 3.2 - Representação da Solução

Nesse exemplo, o grafo acima foi dividido em dois *clusters*, em que o *Cluster* 1 possui os vértices do grafo (1, 2 e 8) e o *Cluster* 2 possui os vértices (3, 4, 5, 6, 7 e 9).

Além deste vetor, há uma estrutura de dados auxiliar que é utilizada para armazenar um atributo referente à capacidade (neste trabalho, *mínima*) do *cluster*, obtida pelo somatório do atributo associado à capacidade dos vértices do grafo. Assim, pode-se definir a capacidade mínima por *clusters*, ou seja, quais *clusters* possuem um valor inferior ao limite pré-determinado. De forma a ilustrar esta última observação, a Tabela 3.1 a seguir traz as capacidades associadas a cada um dos vértices do grafo da Figura 3.2.O atributo de capacidade poderia ser por exemplo, o número de domicílios com pessoas com renda acima de 10 (dez) salários mínimos.

Tabela 3.1: Atributo dos vértices do grafo

Vértices do Grafo	Atributo de Capacidade
1	4
2	1
3	5
4	4
5	3
6	2
7	1
8	2
9	5

Os algoritmos implementados permitem que o valor da capacidade mínima seja informado como um parâmetro de entrada ou que o próprio algoritmo estipule uma capacidade mínima por *Cluster* através da aplicação da Equação 14 proposta por Semaan 2010 e utilizados também em [SANTOS, 2012] [SANTOS, 2013]:

$$Capacidade_{Min\_Cluster} = (\beta/k).\sum_{i=1}^{|V|} x_{il} \qquad (14)$$

sendo:

- β fator de ajuste;
- k quantidade de *clusters*;
- $|V|$ corresponde à quantidade de objetos a serem agrupados;
- x_{il} valor do *l*-ésimo atributo associado ao *i*-ésimo objeto;

Na próxima seção são descritos os procedimentos construtores implementados neste trabalho para construção das soluções iniciais.

3.2.2 ETAPA DE CONSTRUÇÃO

Os procedimentos de construção de soluções iniciais desenvolvidos neste trabalho geram *clusters* através do particionamento de uma AGM T . Esta árvore foi construída a partir do grafo G associado ao problema de clusterização com restrição de capacidade e conexidade, em *k clusters* (subárvore). Portanto, os *k clusters* são definidos através da remoção de *k – 1* arestas de *T*, sendo que a remoção de cada aresta produz duas novas subárvores, ou seja, dois novos *clusters* definidos como T_a e T_b.

Em todos os procedimentos de construção de soluções propostos neste trabalho, a restrição de conexidade é garantida. Porém, apesar dos procedimentos terem como objetivo principal a busca de soluções válidas que atendam concomitantemente às restrições de conexidade e de capacidade mínima por *cluster*, esta última restrição pode não ser atendida. Nesse caso, em algumas situações, pode ser impossível a formação de soluções que atendam ambas as restrições.

Assim, utilizando a ideia de particionamento da AGM, foram implementados cinco estratégias de construtores de soluções iniciais que estão descritas a seguir:

3.2.2.1 CONSTRUTOR 1

A primeira estratégia do procedimento construtor de soluções iniciais realiza o particionamento da AGM respeitando somente a restrição de conexidade, sem considerar a restrição de capacidade, objetivando, portanto, somente a minimização da função objetivo.

O procedimento construtor é aplicado em duas etapas que são executadas (*k – 1*) vezes para a obtenção de *k clusters*:

A primeira etapa consiste na **seleção do cluster a ser particionado** . Nessa etapa, seleciona-se o *cluster* com o menor grau de homogeneidade (considerando a Equação 7), excetuando-se a primeira iteração do procedimento construtor, uma vez que só existe 1 (um) *cluster* composto pela AGM. Voltamos a lembrar de que Equação 7 é representada por ($f(P) = \sum_{c=1}^{k} Som_c$) e a Equação 11 por $Custo_a = Som(T) - (Som(T_a) + Som(T_b)$.

A seguir, define-se qual aresta será **removida** de forma a produzir dois novos *clusters* a partir desta etapa. O valor do custo da remoção da aresta é obtido através da Equação 11. Nesse caso, quanto maior o valor de $Custo_a$ mais homogêneos são os dois novos *clusters* formados. Nesse procedimento construtor foi considerada uma ideia similar à apresentada no procedimento de construção metaheurística GRASP. Ou seja, cria-se uma Lista Restrita de Candidatos (LRC) em que serão incluídas as α arestas que possuem os maiores valores de $Custo_a$. Particiona-se, então, a AGM através da remoção aleatória de uma das arestas inseridas na LRC. E, enquanto a quantidade de *clusters* pré-estabelecida não for alcançada, deve-se repetir os seguintes passos:

- Selecionar o *cluster* com maior valor da função objetivo f (Equação 7)
- Calcular o $Custo_a$ para todas as arestas do *cluster* selecionado;
- Definir a LRC com as α arestas que possuem os maiores valores de $Custo_a$;
- Particionar o *cluster* selecionado através da remoção aleatória de uma aresta de LRC.

A adoção desse procedimento possibilita uma diversificação das soluções, ou seja, encontrar soluções potencialmente válidas, evitando que o algoritmo tenha um comportamento apenas guloso.

A figura a seguir ilustra a aplicação do procedimento Construtor *1* com todas as suas fases e as respectivas explicações sobre cada uma delas, considerando a formação de 3 *clusters*:

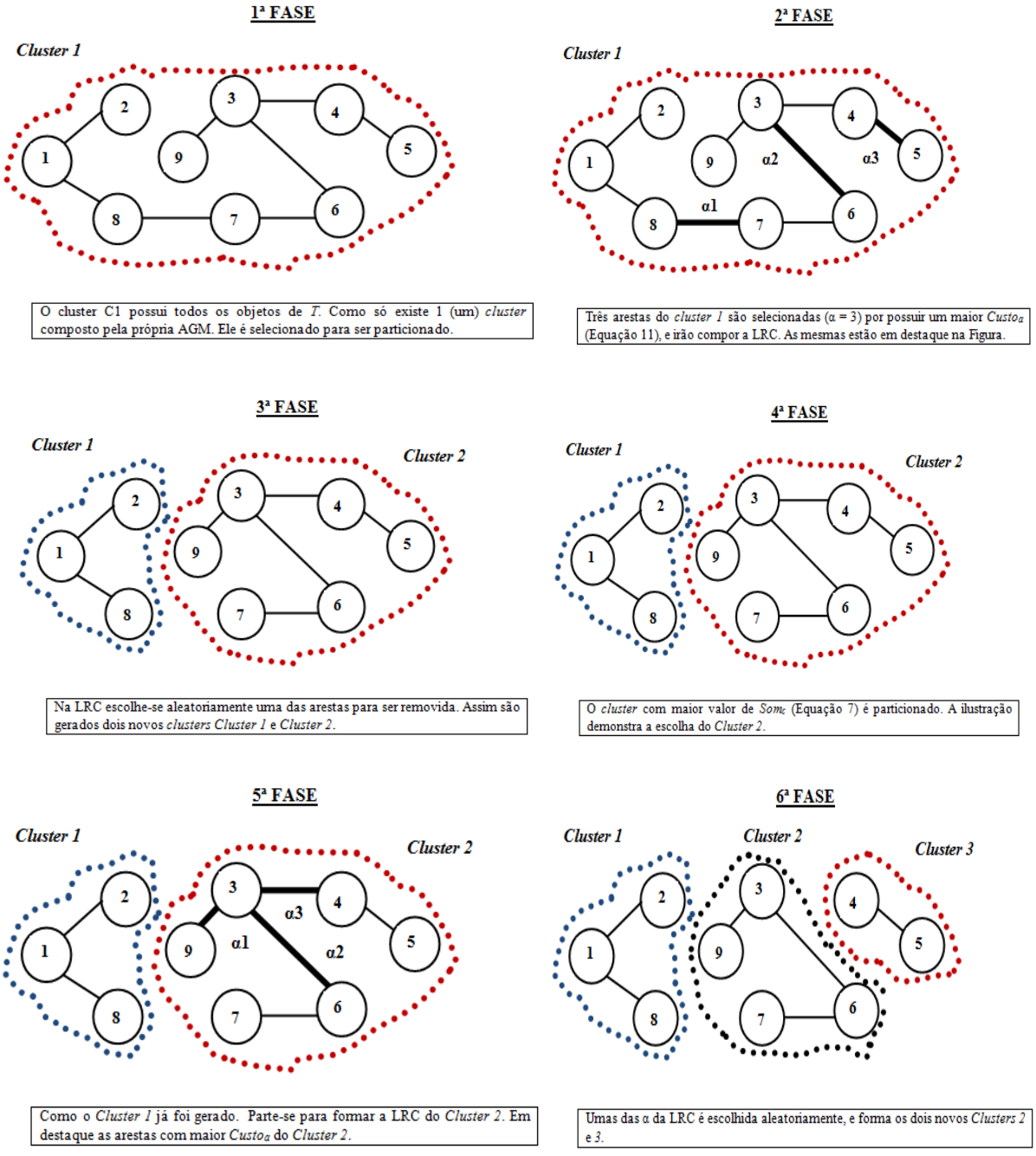


Figura 3.3 - Fases do Construtor 1

Embora exista a restrição de capacidade mínima, ou seja, um limite inferior para o somatório do atributo associado à capacidade dos vértices do *cluster*, o Construtor 1 foca apenas na restrição de conexidade. O procedimento Construtor 2, que será visto a seguir, trata da restrição de conexidade, além de garantir também a capacidade mínima.

3.2.2.2 CONSTRUTOR 2

Neste segundo procedimento, é realizado o particionamento da AGM respeitando as restrições de conexidade e de capacidade mínima. No que se refere à restrição de capacidade mínima, o construtor 2 (dois) trabalha na tentativa de obter *clusters* que atendam essa restrição, ação esta que nem sempre é bem sucedida.

O procedimento construtor 2 (dois) consiste de duas etapas descritas a seguir:

Na primeira etapa é gerada **uma subárvore principal contida na AGM**. A concepção de subárvore principal está associada à existência, na AGM, de um caminho simples sem ciclos, entre dois vértices dessa árvore, ou seja, toma-se um vértice v_i como a origem desse caminho e um vértice v_j como o término. A seleção dos vértices v_i e v_j é feita aleatoriamente entre todos os vértices pertencentes a AGM, para que se possa a partir dessa escolha calcular a distância entre os V_8 e V_9 (Figura 3.4), devido a essa arbitrariedade na escolha dos vértices fica impossível garantir que esse seja o menor caminho entre eles.

A partir da AGM, são selecionados de forma aleatória dois vértices, ilustrados em destaque na Figura 3.4 abaixo:

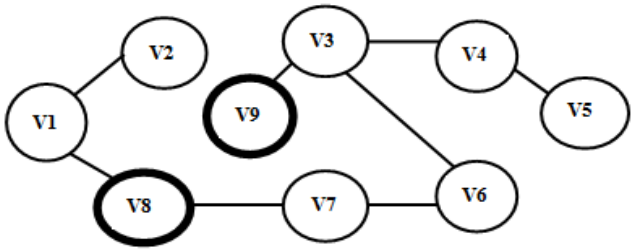


Figura 3.4 - Seleção da subárvore principal da AGM

Os vértices selecionados de forma aleatória (V_8 e V_9) irão formar a subárvore principal da AGM juntamente com os outros vértices que se encontram no caminho existente entre eles. Para a identificação dos vértices que se encontram no caminho existente entre os vértices A e B, utilizou-se a Equação 15.

$$Di(A,B) - (Di(A,verticeX) + Di(B,verticeX)) = 0 \quad (15)$$

sendo:

- A o vértice inicial a subárvore principal da AGM;

- B o vértice final da subárvore principal da AGM;

- $verticeX$ corresponde ao vértice sob consulta. Caso o lado esquerdo da Equação 15 seja igual a 0 (zero), confirma-se que o vértice X pertence ao caminho existente entre os vértices A e B. Esta distância que está sendo referenciada corresponde ao número de arestas em que cada aresta corresponde ao comprimento um.

Caso o lado esquerdo da Equação 15 tenha um resultado diferente de 0 (zero), então o vértice X não pertence à subárvore principal da AGM entre os vértices A e B.

Desta forma, a distância entre os vértices V_8 e V_9 , conforme o caminho existente na AGM, é a quantidade de arestas existentes entre esses dois vértices. A Figura 3.5 ilustra a subárvore principal formada pelos vértices V_8 e V_9 e os vértices que estão no caminho entre V_8 e V_9 .

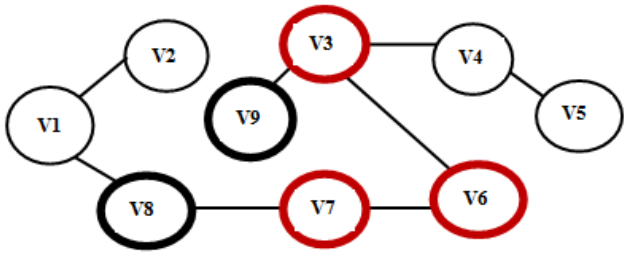
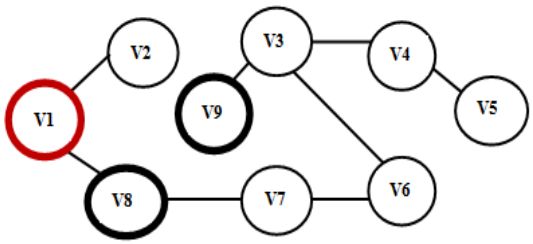


Figura 3.5 - Vértices que estão no caminho entre V_8 e V_9

Exemplificação da Equação 15 para o vértice V_1, é ilustrada na Figura 3.6:



Di (V8,V9) – ((Di (V8, V1) + (Di(V9, V1)) = 0
4 – (1 + 5) = 0
4 – 6 = 0
*– 2 = 0 (**FALSO**)*
Então, comprova-se que o V1 não pertence a subárvore principal (V8 – V9)

Figura 3.6 - Exemplificação do vértice (V_1) na subárvore principal da AGM

Exemplificação da aplicação da Equação 15 para o vértice V_6, é ilustrada na Figura 3.7:

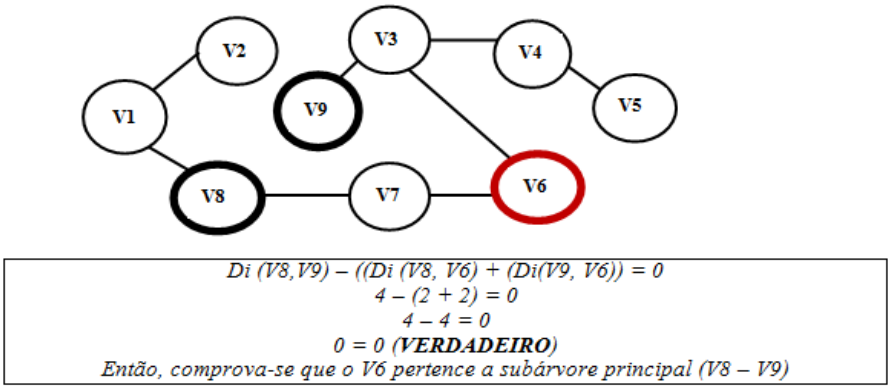


Figura 3.7 - Exemplificação do vértice (V_6) na subárvore principal da AGM

Assim, ao se definir quais vértices pertencem à subárvore principal da AGM através do cálculo da Equação 15, agrupam-se os vértices que não pertencem à subárvore principal serão agrupados ao vértice mais próximo pertencente a esta subárvore. Desta forma, o grafo deve ficar com um formato de uma "barbante" e os vértices pertencentes à subárvore principal possuirão o somatório das capacidades dos vértices a eles agrupados. Então, a subárvore principal toma a forma do grafo apresentado na Figura 3.8:

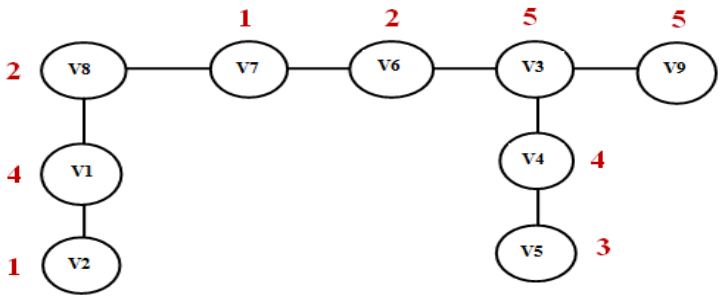


Figura 3.8 - Geração da subárvore principal da AGM

E a Figura 3.9 apresenta os vértices pertencentes à subárvore principal com o somatório dos valores das capacidades dos vértices a eles agrupados, os valores definidos na Tabela 3.1, a ilustração a seguir:

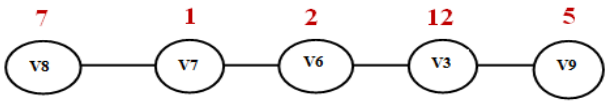


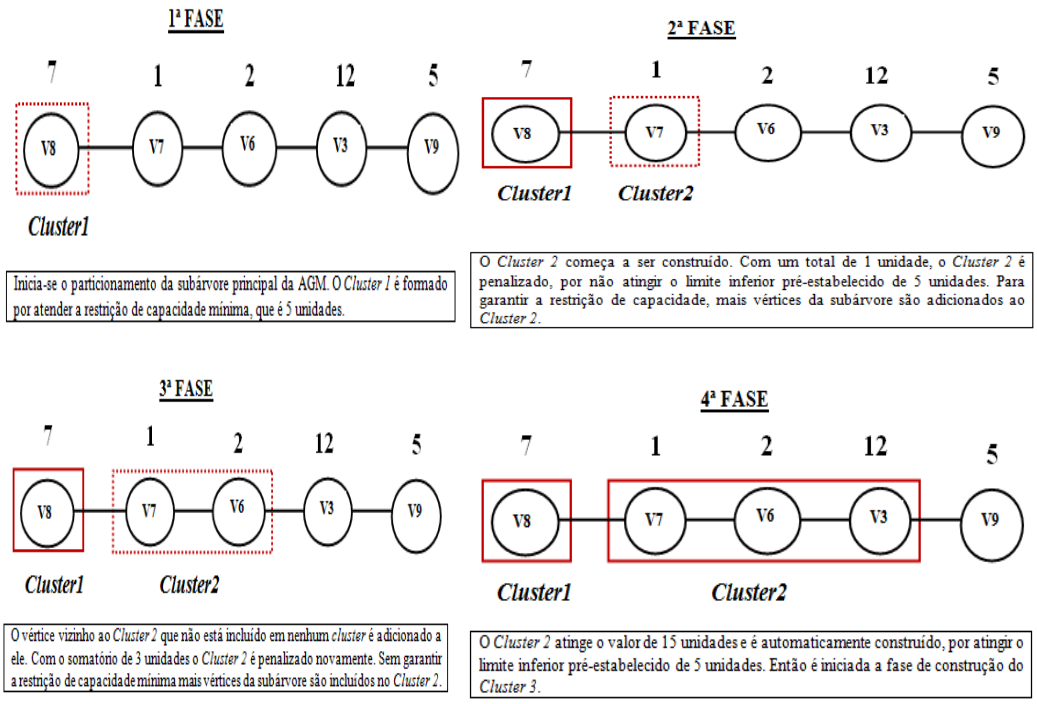
Figura 3.9 - Subárvore principal da AGM com os vértices capacitados

Na segunda etapa do procedimento construtor 2 (dois), é efetuado o particionamento da **subárvore principal considerando a restrição da capacidade mínima até a obtenção de *k clusters***: considerando o exemplo da Figura 3.7, a partir dos vértices (V_8 e V_9) que

constituem a subárvore principal da AGM, são adicionados os grupos de vértices adjacentes, conforme a restrição de capacidade mínima.

Portanto, os *clusters* são formados por vértices ou grupo de vértices, de acordo com a verificação do somatório, ou seja, se a sua capacidade for superior ao limite inferior pré-estabelecido. Assim, a restrição de capacidade mínima é garantida, o *cluster* é formado e um novo *cluster* é obtido a partir da junção dos vértices restantes da subárvore principal da AGM, até que sejam gerados os *k clusters*. Todavia, o particionamento continua sendo executado até que o número (*k*) de *cluster(s)* seja obtido ou, caso ainda exista algum vértice ou grupos de vértices não adicionados a nenhum *cluster* na subárvore principal da AGM. A Figura 3.10 ilustra as fases do procedimento construtor 2 (dois). Nesse caso, definiu-se a capacidade mínima pré-estabelecida com um valor de cinco unidades (pode ser o número de alguma variável definida no problema) e a geração de 3 *clusters*. Assim como a Figura 3.8 os vértices estão capacitados.

A figura a seguir ilustra a aplicação do procedimento Construtor *2* com todas as suas fases e as respectivas explicações sobre cada uma delas, considerando a formação de 3 *clusters e* garantindo a restrição de capacidade mínima por *cluster*:



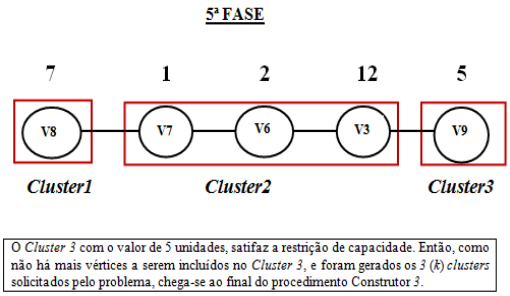


Figura 3.10 - Fases de Construtor 2

A seguir, temos a apresentação do procedimento Construtor 3 que trata também das restrições de conexidade e de capacidade mínima por *cluster*.

3.2.2.3 CONSTRUTOR 3

Assim como o procedimento Construtor 2, o procedimento Construtor 3 trabalha levando em conta as restrições de conexidade e de capacidade mínima por *cluster* para a geração de soluções válidas. Esses procedimentos são semelhantes, diferenciando-se apenas no que concerne à restrição de capacidade. Mais especificamente, esse procedimento também visa maximizar a capacidade dos *clusters* através da delimitação da capacidade máxima por *cluster*.

Assim como o Construtor 2 (dois), o Construtor 3 (três) é aplicado em duas etapas, sendo gerada na primeira etapa uma subárvore principal contida na AGM.

Na segunda etapa do procedimento construtor 3 (três) é particionada a subárvore principal considerando a restrição da capacidade mínima até a obtenção de *k clusters*, e de um fator determinante escolhido aleatoriamente para maximizar a capacidade de um *cluster*. Procura-se maximizar a capacidade dos *clusters* para aumentar a chance de formação de novos *clusters* sem penalização nas iterações seguintes.

Na ilustração das fases do procedimento Construtor 3 (três), na Figura 3.11, exemplifica-se o caso em que a capacidade mínima por *cluster* é de 5 (cinco) unidades e a máxima por *cluster* é de 12 (doze) unidades, ou seja, sendo uma variável *numCap* definida como o valor de somatório de unidades em um *cluster*, fica definido no problema exemplificado que $5 \leq numCap \leq 12$. Portanto, o *cluster* não pode ter menos de 5 (cinco) unidades e nem mais do que 12 (doze) unidades. Para a formação de *3 clusters*.

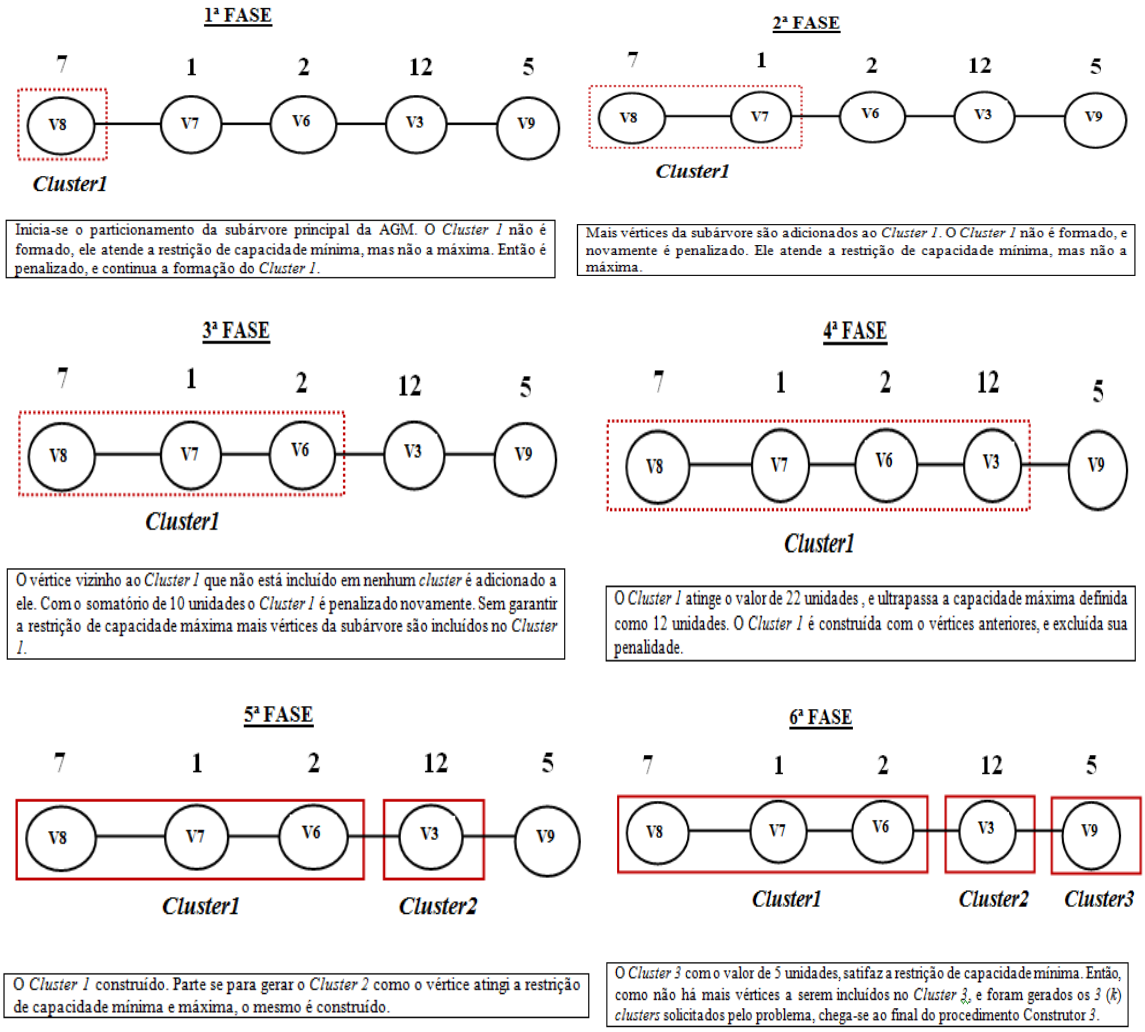


Figura 3.11 - Fases do Construtor 3

Dessa forma, verifica-se que o procedimento Construtor 3 (três) consegue maximizar a capacidade dos *clusters*.

A seguir, temos a apresentação do procedimento Construtor 4 (quatro) que trata também das restrições de conexidade e de capacidade mínima por *cluster*.

3.2.2.4 CONSTRUTOR 4

Neste procedimento, particiona-se a AGM respeitando as restrições de conexidade e de capacidade mínima. O procedimento Construtor 4 (quatro) propicia uma regeneração das soluções inviáveis que eventualmente são produzidas pelo procedimento Construtor 2 (dois), ou seja, quando neste a quantidade de *clusters* gerados é inferior a um número pré-fixado. Neste caso, torna-se necessário regenerar a solução mediante a seleção do *cluster* com o maior número excedente à capacidade mínima sugerida no problema.

O procedimento Construtor 4 (quatro) opera até que a quantidade de *clusters* definida seja atingida, ou até que não seja mais possível dividir o *cluster* em função da violação da restrição de capacidade.

A figura a seguir ilustra a aplicação do procedimento Construtor 4 com todas as suas fases e as respectivas explicações sobre cada uma delas, considerando a formação de 3 *clusters* e garantindo a restrição de capacidade mínima por *cluster* de 5 unidades, a partir de uma solução inválida produzida pelo Construtor 2 (dois):

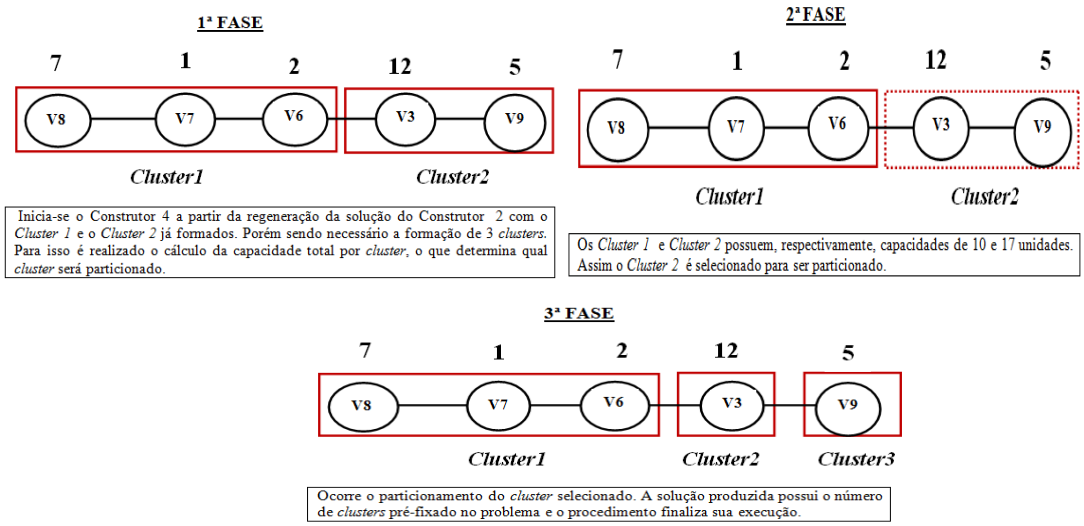


Figura 3.12 - Fases do Construtor 4

3.2.2.5 CONSTRUTOR 5

O procedimento Construtor 5 (cinco) trabalha com as restrições de conexidade e de capacidade mínima por *cluster* visando a geração de soluções válidas.

O Construtor 5 (cinco) é executado em duas etapas, assim como os construtores 2, 3 e 4, sendo gerada na primeira etapa uma subárvore principal contida na AGM.

Na segunda etapa deste procedimento construtor é particionada a subárvore principal**,** considerando a restrição de capacidade mínima. Tal particionamento é aplicado até a obtenção de *k clusters* e de um valor pré-estabelecido para maximizar a capacidade de um cluster considerando a restrição de conexidade existente não só na subárvore principal da AGM**,** mas também no Grafo Original.

Na ilustração das fases do procedimento Construtor 5 (cinco), na Figura 3.13, exemplifica-se o caso em que a capacidade mínima por *cluster* é de 5 unidades.

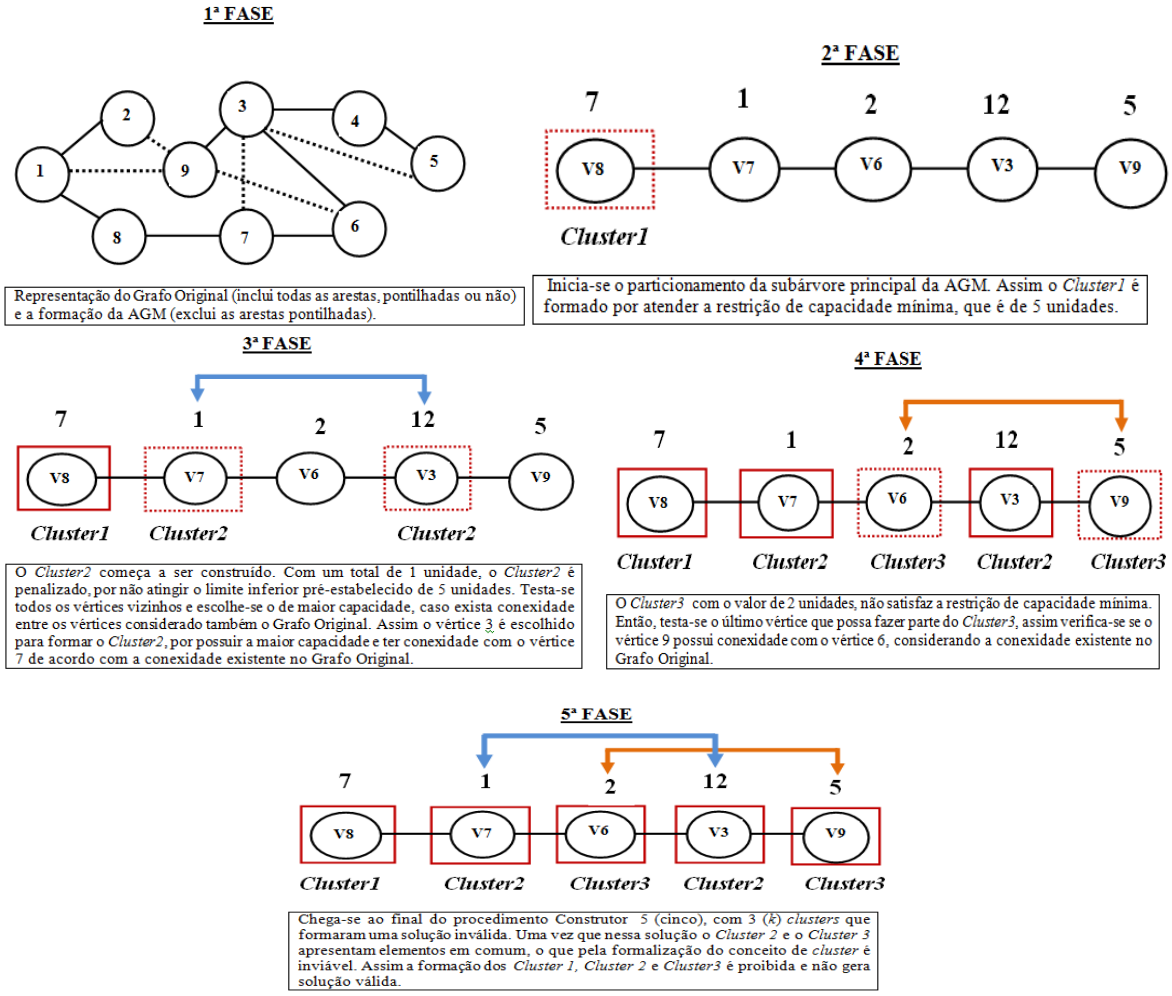


Figura 3.13 - Fases do Construtor 5 – Solução Inválida

A seguir, são apresentados os seis procedimentos de Busca Local que são propostos neste trabalho.

3.2.3 PROCEDIMENTOS DE BUSCA LOCAL

Conforme já comentado, os procedimentos de busca local tendem a melhorar as soluções advindas de procedimentos de construção (em problemas de minimização ou maximização) e propiciar a geração de um número maior de soluções consideradas válidas para o seu problema [DIAS, 2004].

Neste trabalho, foram desenvolvidos seis procedimentos de busca local. E, em todos eles, são realizadas migrações de vértices entre *clusters*, objetivando minimizar a função objetivo. Esses procedimentos de busca local atuam na tentativa de eliminação de penalidades referentes ao não atendimento da restrição de capacidade mínima por *cluster*, e na melhoria da

solução com a redução da função objetivo. Destes, três utilizam a AGM definida pelo(s) procedimento(s) construtor (es) e os outros três usam o Grafo Original.

Para que fosse possível realizar a migração de vértices entre os *clusters* em todas as seis buscas locais propostas neste trabalho, foram respeitados os seguintes requisitos:

- A migração de vértices entre *clusters* acontece, se e somente se, o *cluster* estiver penalizado. Um *cluster* está penalizado quando não gera uma solução válida, seja por não atingir a capacidade mínima pré-estabelecida, ou por não atingir o número de *clusters* definidos previamente no problema;

- Um *cluster* com apenas 1 (um) vértice não deve ser migrado, uma vez que o mesmo ficaria vazio, e não podem existir *clusters* sem vértices;

- Um vértice adjacente a, no mínimo 2 (dois) vértices no mesmo *cluster*, não pode ser migrado, pois essa migração resultaria em um *cluster* desconexo. Dessa forma, deixaria de garantir a restrição de conexidade.

A seguir, tem-se a apresentação dos seis procedimentos de busca local.

3.2.3.1 BUSCA LOCAL 1

Além de melhorar a homogeneidade dos clusters, o procedimento de Busca Local 1 também tem como objetivo principal eliminar penalidades. Mais especificamente, este procedimento utiliza uma lista constituída pelas arestas que foram removidas da AGM durante a fase de construção e uma lista de *clusters* penalizados, ou seja, que não satisfazem a restrição de capacidade. O procedimento verifica a possibilidade de realizar migrações de vértices entre os *clusters* utilizando essa lista.

Neste sentido, para cada aresta da lista, são visualizados os *clusters* aos quais os seus vértices pertencem. Em seguida, são verificados os requisitos comuns a todas as buscas locais para que seja possível realizar a migração de vértices entre os *clusters*.

A Figura 3.14 ilustra a aplicação do procedimento Busca Local 1 (um) e seus respectivos comentários:

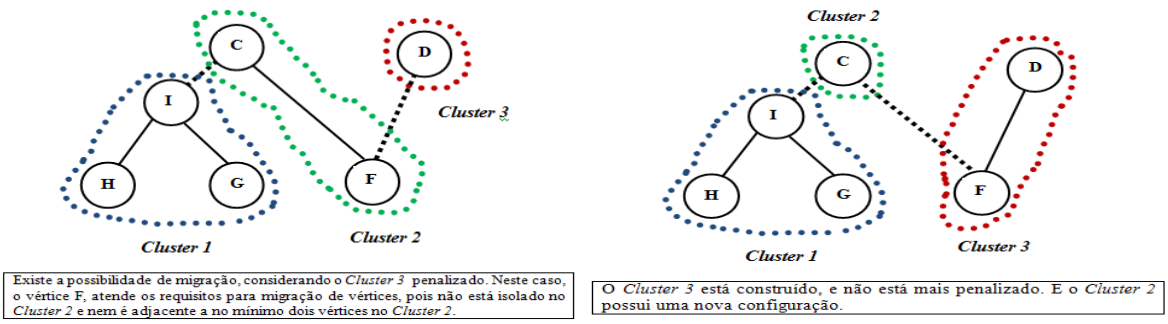


Figura 3.14 – Busca Local 1 – possibilidade de migração de vértices

No exemplo anterior, demonstrado pela Figura 3.14, há uma possibilidade de migração entre vértices.

A Figura 3.15 ilustra exemplos em que a Busca Local proíbe a migração de vértices e suas respectivas explicações:

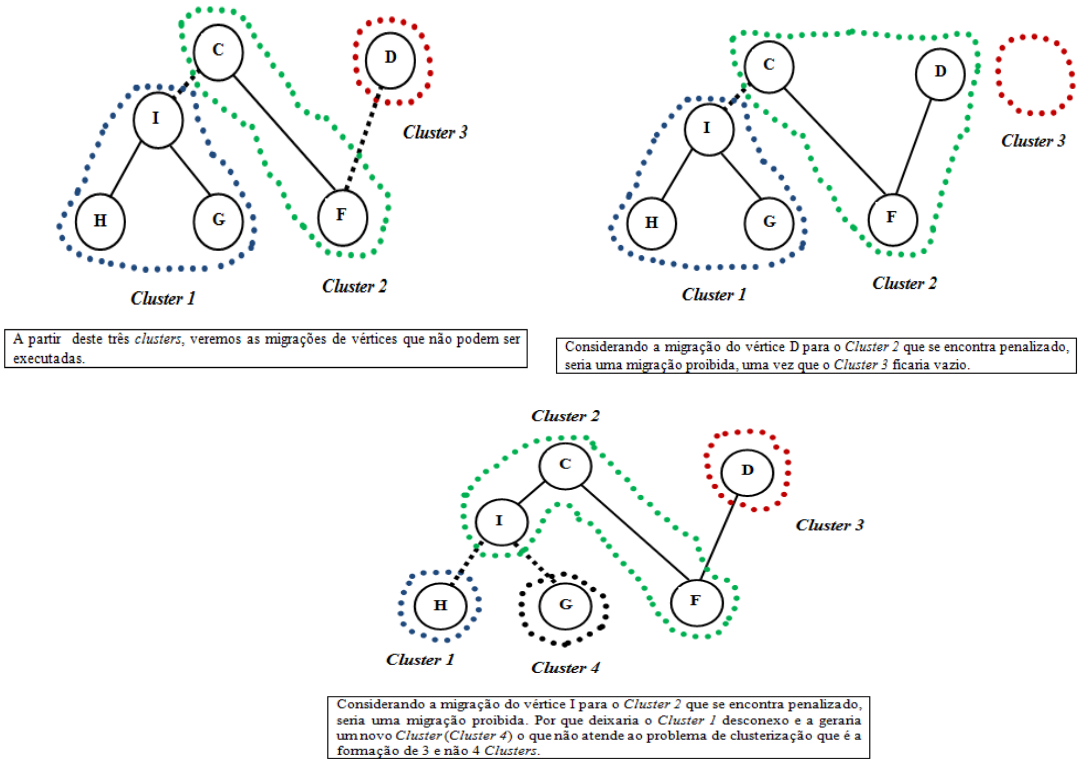


Figura 3.15 – Busca Local 1 – proibida a migração de vértices

Na Figura 3.16 tem-se um exemplo de Busca Local onde há a possibilidade de migração de vértices garantindo a restrição de capacidade mínima por *cluster*. Neste exemplo, demonstrado pela ilustração abaixo, a capacidade mínima pré-estabelecida foi de cinco unidades.

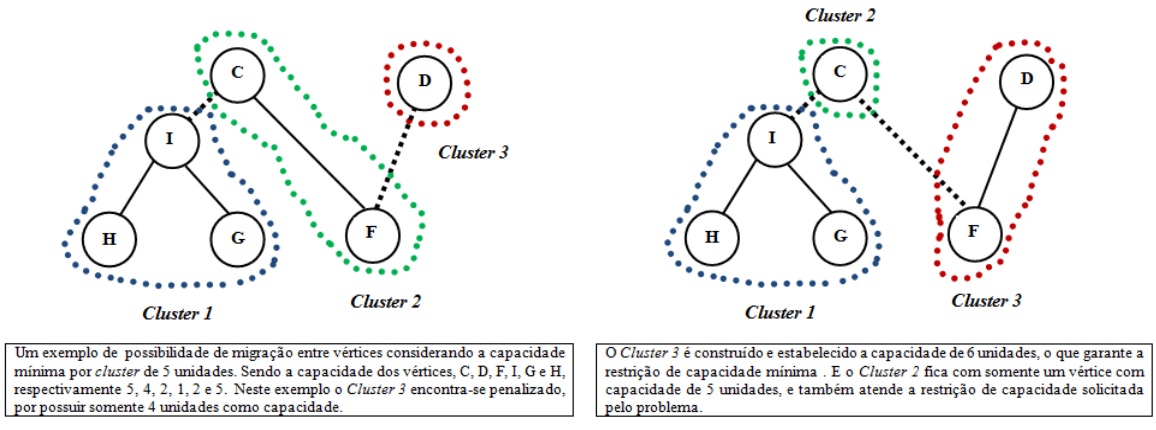


Figura 3.16 – Busca Local 1 – migração de vértices com *cluster* capacitados

3.2.3.2 BUSCA LOCAL 2

Assim como o procedimento Busca Local 1, o procedimento de Busca Local 2 também tem como objetivo eliminar as penalidades. Esse procedimento verifica a possibilidade de realizar migrações de vértices entre os *clusters*, considerando o Grafo Original, e não apenas a AGM. Tendo em vista que esse procedimento trabalha com o Grafo Original, há a necessidade de que seja verificado se as restrições de conexidade entre os vértices internos aos *clusters* estão sendo satisfeitas, ou seja, se os vértices internos aos *clusters* são conexos.

Inicialmente, esse procedimento seleciona, de forma aleatória, uma aresta $a = (i, j)$ do Grafo Original antes da construção, e em seguida, são efetuadas as seguintes verificações:

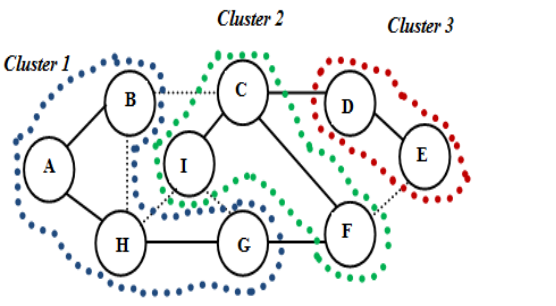
- A aresta é selecionada quando apenas um dos *clusters*, em que os vértices da aresta selecionada pertence está penalizado. Em caso negativo, o procedimento seleciona outra aresta, também aleatoriamente;

- Um *cluster* é candidato a receptor do vértice a ser migrado e outro *cluster* é candidato a doador do vértice, se apenas um destes *clusters* estiver penalizado;

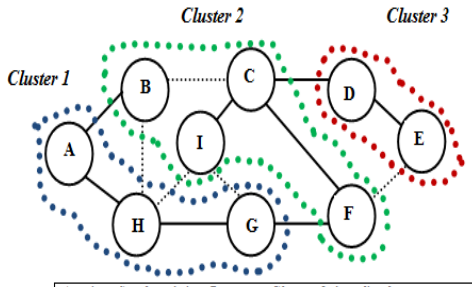
- A migração de um vértice para o *cluster* receptor deve ocorrer somente se os demais vértices do *cluster* doador forem conexos entre si, ou seja, se a restrição de conexidade for atendida.

Sendo estas verificações atendidas, é possível a migração do vértice do *cluster* doador para o *cluster* receptor.

A ilustração da Figura 3.17 apresenta a aplicação do procedimento Busca Local 2 (dois) e suas explicações. As arestas em negrito pertencem à AGM construída, e as demais arestas, em conjunto com as arestas da AGM, fazem parte do Grafo Original:



O *Cluster 2* está penalizado, e *Cluster 1* não está penalizado, a migração do vértice *B* para o *Cluster 2* não penalizaria este *cluster*. Pode-se observar que os demais vértices do *Cluster 1* são conexos entre si.



A migração do vértice *B* para o *Cluster 2* é realizada, uma vez que foram considerados todos os requisitos para realizar a migração. Observa-se que a migração é possível considerando o Grafo Original, pois não existe conexidade se fosse considerada a AGM.

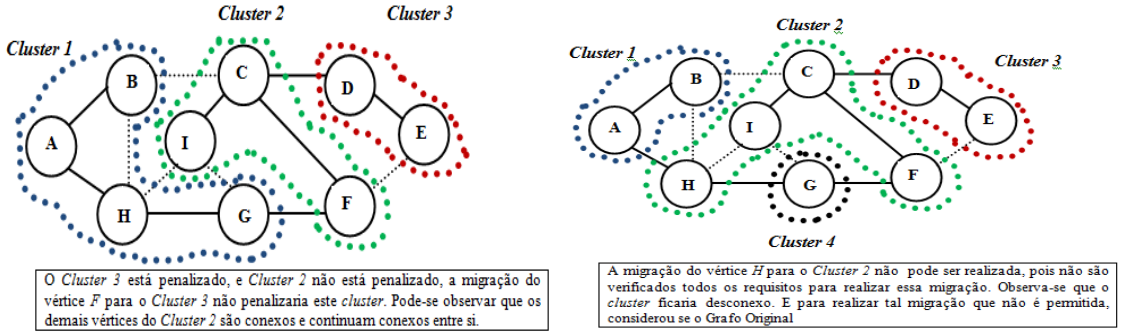


Figura 3.17 – Busca Local 2 – possibilidade de migração de vértices

Neste exemplo há uma possibilidade de migração dos vértices entre os *clusters*.

A Figura 3.18 ilustra exemplos em que a Busca Local 2 (dois) proíbe a migração de vértices:

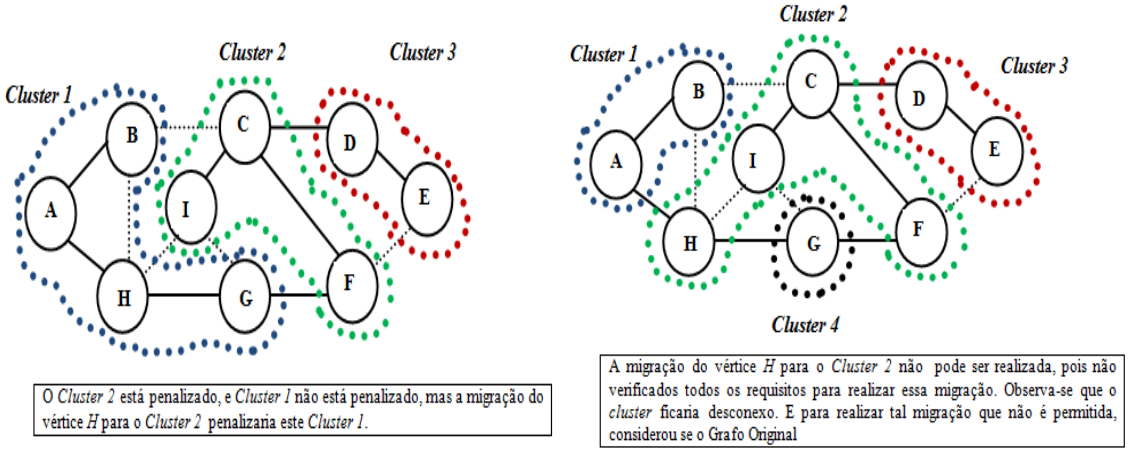


Figura 3.18 – Busca Local 2 – proibida a migração de vértices

3.2.3.3 BUSCA LOCAL 3

O procedimento de Busca Local 3 (três) também verifica a possibilidade de realizar migrações de vértices entre o *clusters* considerando o Grafo Original e a AGM construída. E assim como o Busca Local 2, verifica se as restrições de conexidade entre os vértices internos aos *clusters* estão sendo satisfeitas, ou seja, se os vértices internos aos *clusters* são conexos.

O Busca Local 3 (três) inicialmente seleciona, de forma aleatória, uma aresta $a = (i, j)$ de Grafo Original, antes da construção. A partir desta aresta, são efetuadas as verificações descritas abaixo, sendo que algumas são idênticas àquelas realizadas na Busca Local 2 (dois):

- Uma aresta é selecionada quando apenas um dos *clusters*, em que os vértices da aresta selecionada pertence, está penalizado. Em caso negativo, o algoritmo seleciona outra aresta também aleatoriamente;

- Os *clusters* podem atuar como receptores ou como doadores de vértices. Mas para isso, ambos os *clusters* não devem estar ou tornar-se penalizados. E as verificações são analisadas em ambos os sentidos.

- Mesmo com a migração de um vértice para o *cluster* receptor, os vértices do *cluster* doador devem continuar atendendo a restrição de conexidade;

- A migração de um vértice entre os *clusters* receptor e doador só é realizada quando é observada uma redução no valor da função objetivo.

3.2.3.4 BUSCA LOCAL 4

O procedimento de Busca Local 4 (quatro) utiliza uma lista de arestas removidas da AGM para geração de *clusters*. Dessa forma, esse procedimento busca melhorar a qualidade da solução, ou seja, minimizar a função objetivo através da união e da divisão de *clusters* já construídos.

São selecionados, aleatoriamente, os *clusters* aos quais os vértices das arestas removidas pertencem, podendo se unir em um *cluster* para, em seguida, este *cluster* ser dividido através da aplicação do procedimento construtor 1 (um).

A Figura 3.19 ilustra um exemplo da aplicação do Busca Local 4 (quatro) e suas respectivas explicações:

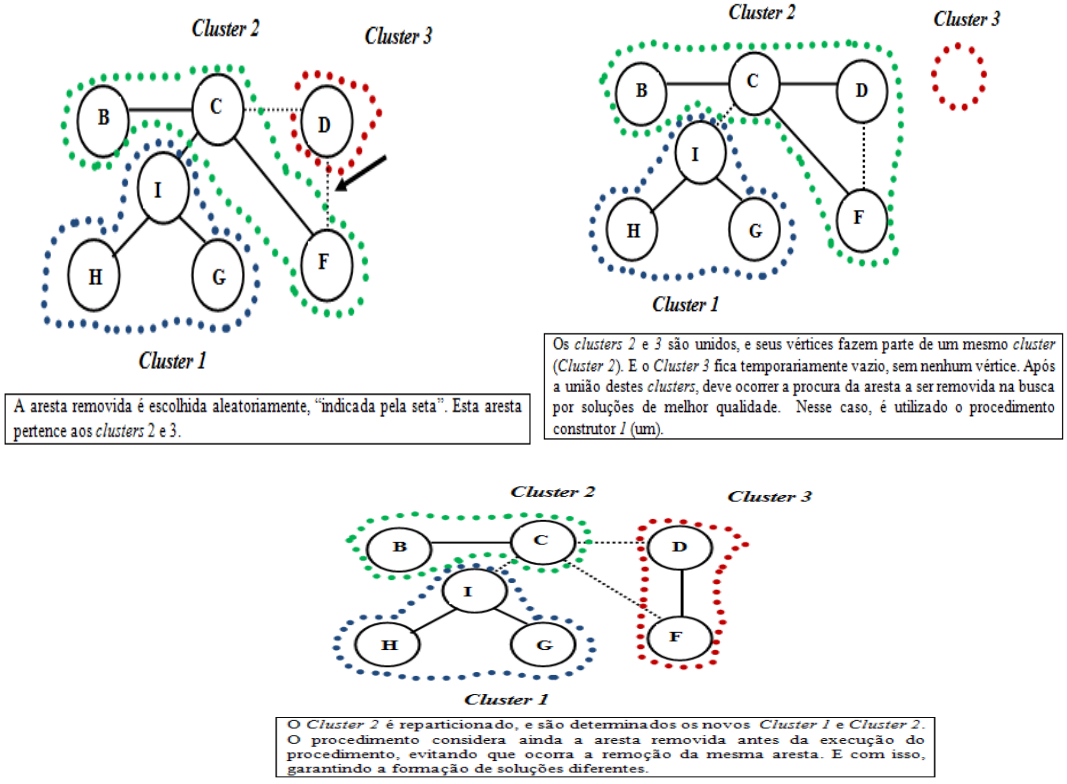


Figura 3.19 – Busca Local 4 - execução

3.2.3.5 BUSCA LOCAL 5

O procedimento de Busca Local 5 (cinco) é muito parecido com o procedimento de Busca Local 4 (quatro), e tem a finalidade de produzir novas soluções não penalizadas (que atendam a restrição de capacidade mínima), considerando a AGM.

Inicialmente, utiliza-se uma lista de arestas removidas da AGM para geração de clusters, escolhidas de forma aleatória, e depois realiza-se a união de dois *clusters*. Posteriormente, os *clusters* são divididos em dois novos *clusters* a partir de uma LRC composta pelas α arestas que, se removidas, definem novos *clusters*. Esses *clusters* devem atender a restrição de capacidade mínima e, simultaneamente, devem possuir a maior diferença, em valor absoluto, considerando o somatório das capacidades dos *clusters* obtidos, conforme Equação 16, a seguir:

$$Cap_{cl(A,B)} = \left|getTotal\left(get_{Cluster}(A)\right) - getTotal\left(get_{Cluster}(B)\right)\right| \quad (16)$$

sendo:

- $Cap_{cl(A,B)}$ somatório da capacidade dos *cluster* e tem como objetivo maximizar o seu valor;
- A vértice do *cluster X* da AGM;
- B vértice do *cluster Y* da AGM;
- $getTotal\ (X)$ retorna o somatório das capacidades do *cluster (X)*;
- $get_{Cluster}(A)$ retorna o *cluster* em que está contido o vértice A;
- $get_{Cluster}(B)$ retorna o *cluster* em que está contido o vértice B;

A Figura 3.20 apresenta um Grafo que é o resultado da união de dois *clusters*, o *Cluster 1* e do *Cluster 2*. Este grafo será usado para exemplificar o Procedimento de Busca Local 5 (cinco) com a restrição de capacidade mínima definida para este problema com o valor de 7 unidades.

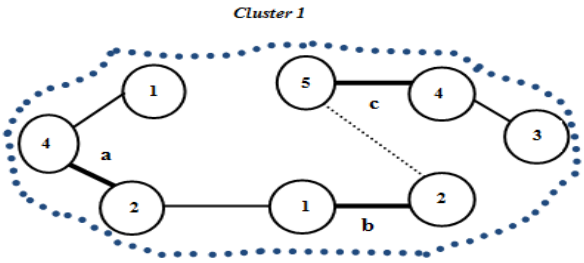
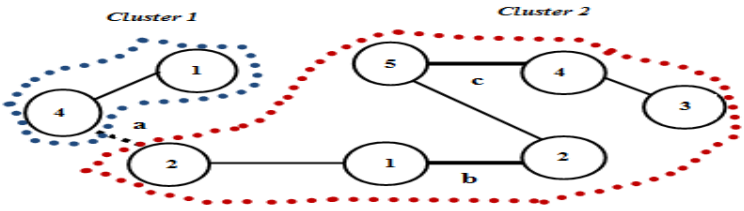


Figura 3.20 – Grafo da União do *Cluster 1* e *Cluster 2*

Na Figura anterior, cada vértice contém um número que corresponde a sua capacidade. As letras significam as arestas candidatas à remoção, e a linha pontilhada representa a aresta

removida da AGM que foi reinserida no Grafo para unir os *clusters 1* e *2*, gerando o único *Cluster 1*. Na figura 3.21 é ilustrada a remoção da aresta *a* (vide comentários).

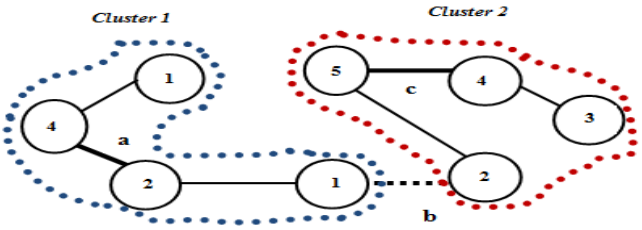


$$Cap_{cl(A,B)} = |getTotal(get_{Cluster}(A)) - getTotal(get_{Cluster}(B))|$$
$$Cap_{cl(A,B)} = |(17) - (5)|$$
$$Cap_{cl(A,B)} = |12| = 12$$

O reparticionamento do *Cluster 1* através da remoção da aresta ***a*** gerou o *Cluster 1* com 5 unidades e o *Cluster 2* com 17 unidades. Portanto, a diferença o somatório da capacidade dos *clusters* é igual a12. E tem se ainda que o *Cluster 1* não atende a restrição de capacidade mínima definida com o valor de 7 unidades.

Figura 3.21 – Busca Local 5 - Remoção da aresta *a*

Na figura 3.22 é ilustrada a remoção da aresta *b* (vide comentários).

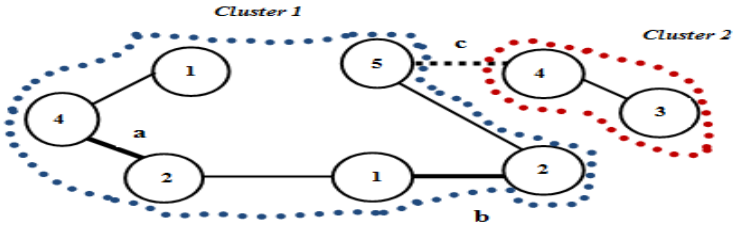


$$Cap_{cl(A,B)} = |getTotal(get_{Cluster}(A)) - getTotal(get_{Cluster}(B))|$$
$$Cap_{cl(A,B)} = |(14) - (8)|$$
$$Cap_{cl(A,B)} = |6| = 6$$

O reparticionamento do *Cluster 1* através da remoção da aresta ***b*** gerou o *Cluster 1* com 8 unidades e o *Cluster 2* com 14 unidades. Portanto, a diferença entre o somatório da capacidade dos *clusters* é igual a 6. Ambos os *clusters* atendem a restrição de capacidade mínima.

Figura 3.22 – Busca Local 5 - Remoção da aresta *b*

Na figura 3.23 é ilustrada a remoção da aresta *c* (vide comentários).



$$Cap_{cl(A,B)} = |getTotal(get_{Cluster}(A)) - getTotal(get_{Cluster}(B))|$$
$$Cap_{cl(A,B)} = |(15) - (7)|$$
$$Cap_{cl(A,B)} = |8| = 8$$

O reparticionamento do *Cluster 1* através da remoção da aresta ***c*** gerou o *Cluster 1* com 15 unidades e o *Cluster 2* com 7 unidades. Portanto, a diferença o somatório da capacidade dos *clusters* é igual a 8. Ambos os *clusters* atendem a restrição de capacidade mínima.

Figura 3.23 – Busca Local 5 - Remoção da aresta *c*

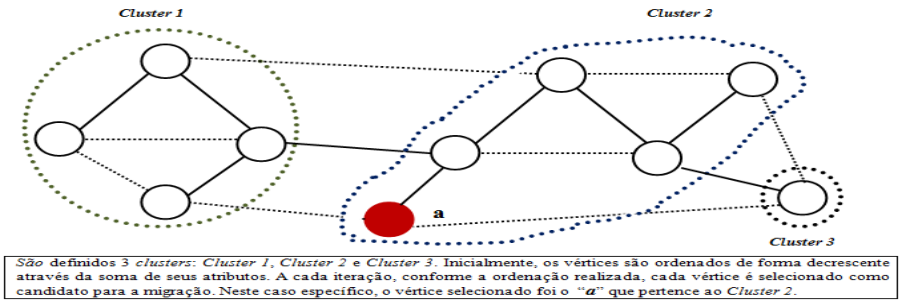
Analisando os exemplos das Figuras 3.21, 3.22 e 3.23, percebe-se que somente as arestas *b* e *c* podem fazer parte da LRC, uma vez que a remoção da aresta *a* irá gerar um *cluster* com penalidade. E, ainda objetivando maximizar o somatório das capacidades dos *clusters* obtidos, escolhe-se a aresta *c* por possuir valor igual a 8, enquanto a remoção da aresta *b* atingiu somente o valor de 6 unidades. Assim, são formandos dois novos *clusters.*

3.2.3.6 BUSCA LOCAL 6

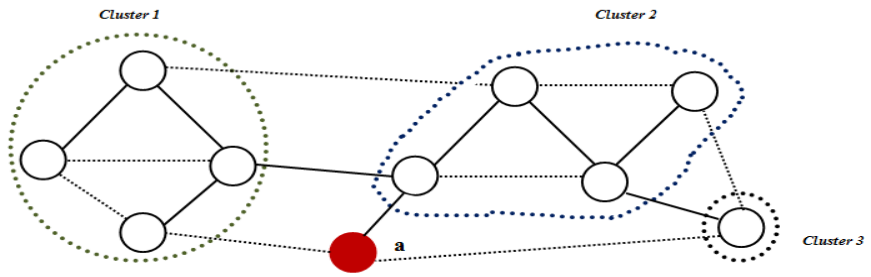
O Procedimento Busca Local 6 (seis) é aplicado essencialmente com o objetivo de minimizar a função objetivo da solução. Dessa forma, a migração dos vértices entre *clusters* é realizada tendo o Grafo Original como base somente se houver a melhoria da qualidade da solução.

Nesse Procedimento de Busca Local 6 (seis) o centroide de um cluster é utilizado e representa o ponto médio, ou seja, corresponde a média da soma de todos os atributos de um *cluster.*

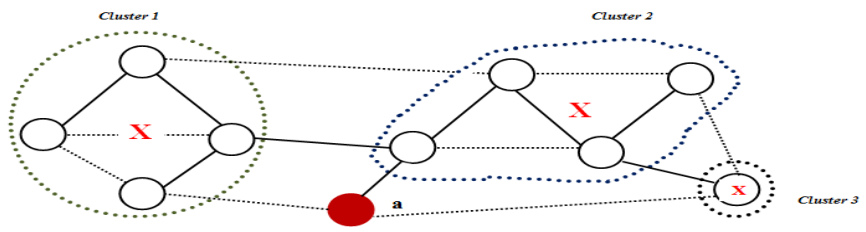
A Figura 3.24 traz um exemplo da aplicação da Busca Local 6 (seis):



São definidos 3 *clusters*: *Cluster 1*, *Cluster 2* e *Cluster 3*. Inicialmente, os vértices são ordenados de forma decrescente através da soma de seus atributos. A cada iteração, conforme a ordenação realizada, cada vértice é selecionado como candidato para a migração. Neste caso específico, o vértice selecionado foi o "***a***" que pertence ao *Cluster 2*.



Como se fosse uma fase transitória, o vértice "***a***" fica isolado, sem nenhuma associação a nenhum *cluster*. É necessário verificar se o vértice "***a***" pode ser migrado. Para isso, é analisado caso seja efetivada a migração, se a restrição de conexidade continua sendo satisfeita no *Cluster 2*. Neste caso, o vértice "***a***" pode ser migrado.



Nessa etapa, o vértice "***a***" que é candidato a migração está isolado e são calculados os centróides de cada *cluster*. Cada *cluster* possui a média da soma de seus atributos. Simbolicamente, a letra X, em cor vermelha, representa a posição correspondente aos valores calculados para cada *cluster*, o centróide.

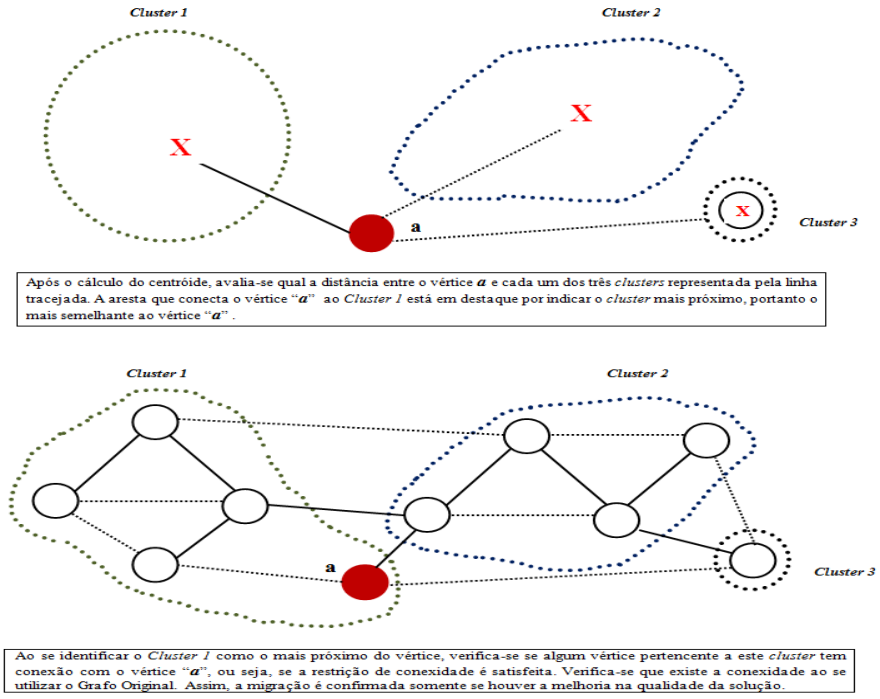


Figura 3.24 – Busca Local 6 – execução

3.3 ALGORITMOS HEURÍSTICOS PROPOSTOS

Os algoritmos heurísticos propostos neste trabalho de doutorado são o resultado de combinações da metaheurística GRASP com os procedimentos de construção das soluções iniciais (Construtor 1, Construtor 2, Construtor 3, Construtor 4 e Construtor 5), buscas locais (Busca Local 1, 2, 3, 4, 5 e 6) e com o procedimento de *Path-Relinking* (PR) (Reconexão por Caminhos (RC)) e um procedimento VND e RVND.

Também são descritas as combinações da metaheurística VNS e suas variações (GVNS, RVNS e *Skewed* VNS) com os procedimentos de construção das soluções iniciais e buscas locais as mesmas combinações já utilizadas na metaheurística GRASP.

3.3.1 PATH-RELINKING (PR)

O procedimento *Path-Relinking* (PR) (Reconexão por Caminhos (RC)) foi proposto inicialmente para compor a fase de diversificação da metaheurística *Scatter Search* [GLOVER AND LAGUNA, 1997]. Esse procedimento é baseado em uma estratégia de busca local que gera novas soluções, considerando a exploração de um "caminho que conecta" duas soluções extremas de boa qualidade. A RC tem início com a escolha de uma solução denominada Solução Base (SB) e, posteriormente, se constrói um caminho pelo espaço de soluções (vizinhanças) que a conecte até outra solução denominada Solução Guia (SG) gerando, assim, novos caminhos chamados de soluções (vizinhanças) intermediárias a cada movimento

executado. O fundamento básico da RC é a possível existência de soluções de boa qualidade no espaço que contém todas as soluções com elementos comuns das duas soluções SB e SG. Levando em consideração que o tamanho desse espaço é substancialmente grande, a RC explora as trajetórias entre as soluções SB e SG através de um procedimento guloso, onde a melhor solução encontrada no caminho é mantida. Em outras palavras, SB é modificada passo a passo até chegar a SG. Todas as soluções intermediárias obtidas em cada passo formam o caminho de SB até SG:

$$SB = S_0, S_1, \ldots, S_z = SG \tag{17}$$

sendo *z* correspondente ao número de soluções intermediárias entre a solução SB e SG.

Portanto, a cada movimento, produz-se uma nova solução intermediária (SI) formada pelos elementos pertencentes à SB, e com a inserção de um atributo proveniente da SG, que substitui um dos elementos da SB original.

Esta solução intermediária é utilizada como a nova solução inicial no próximo passo da execução. Esse processo se repete, inserindo-se mais elementos da SG, até o momento em que será gerada uma solução idêntica à solução guia. Neste momento, considera-se a SG atingida, e a execução da RC para as soluções SB e SG é concluída [GLOVER E LAGUNA, 1997].

Na verdade, a RC aqui proposta trabalha com uma solução guia que vem do conjunto elite (*pool* das *e* melhores soluções distintas geradas pela metaheurística até o momento) e uma solução base que corresponde à solução resultante da busca local da iteração corrente. A partir destas duas, são produzidas as soluções intermediárias com o potencial de contribuir, de forma mais significativa, que as demais no processo de exploração de novas soluções de alta qualidade.

E que através de uma sequência de soluções geradas (soluções intermediárias), que derivam delas, possa encontrar uma solução vizinha melhor que as já conhecidas, ou seja, que possam produzir um ótimo local de qualidade superior aos ótimos locais encontrados até o presente momento [HO AND GENDREAU, 2006]. O conjunto elite é uma estrutura de dados na qual se armazena uma quantidade limitada de soluções. Estas soluções podem contribuir de forma mais significativa que as demais no processo de exploração de novas soluções de alta qualidade [HO AND GENDREAU, 2006].

São diversas são as abordagens de Reconexão por Caminhos encontradas na literatura. São descritas algumas dessas abordagens [RESENDE E RIBEIRO, 2002]:

- reconexão periódica: a reconexão por caminhos não é aplicada sistematicamente, mas apenas periodicamente;

- reconexão *forward*: a solução escolhida como guia é a melhor entre duas selecionadas para a reconexão;

- reconexão *backward*: a solução escolhida como guia é a pior entre as duas selecionadas para a reconexão (*abordagem utilizada neste trabalho*);

- reconexão *back and forward*: são exploradas os dois caminhos separadamente, primeiro no sentido *forward* e depois no *backward*;

- reconexão mista: são explorados simultaneamente os dois caminhos (*forward* e *backward*), até que se encontre em uma solução intermediária equidistante das duas soluções iniciais.

Há, ainda, dois aspectos fundamentais nos quais se baseia a utilização do procedimento de reconexão por caminhos, quais sejam: a intensificação, que consiste na concentração da busca por outras soluções em regiões promissoras do espaço de possíveis soluções, e a diversificação, que caracteriza a exploração de áreas mais diversas no espaço de soluções [GREISTORFER, 2004].

Uma vez que a aplicação do procedimento de Reconexão de Caminhos pode demandar uma substancial quantidade de tempo computacional de execução, tal procedimento deve ser utilizado com parcimônia, quando inserido em uma metaheurística. No que concerne ao GRASP, este procedimento pode ser aplicado a cada *m* iterações consecutivas do GRASP. Para tanto, as *e* melhores soluções que foram obtidas após as *m* iterações do GRASP são armazenadas em um conjunto denominado conjunto elite. Ou seja, a cada *m* iterações toma-se como solução base (SB) a solução produzida após a busca local do GRASP, e toma-se como solução guia (SG) uma solução escolhida aleatoriamente do conjunto elite. Em seguida aplica-se o procedimento de RC nestas duas soluções e toma-se a Melhor Solução Intermediária (MSI). Se esta solução for melhor do que a melhor solução atual, atualiza-se esta solução. Além disso, verifica-se se a MSI é melhor do que alguma solução do conjunto elite, e novamente, se esta condição for satisfeita, atualiza-se o conjunto elite.

3.3.2 GRASP com RECONEXÃO DE CAMINHOS

Uma das formas de incorporar um mecanismo de memória adaptativa no GRASP é com o procedimento de Reconexão de Caminhos [GLOVER, 1996][GLOVER AND MARTI, 2006]. O objetivo do GRASP com RC é encontrar soluções intermediárias de melhor qualidade, que estejam entre as duas soluções, do GRASP e do conjunto elite [LAGUNA

AND MARTI, 1999]. O GRASP com RC foi aplicado com sucesso em diversos trabalhos relatados na literatura, dentre os quais os [FESTA ET. AL, 2007][MATEUS ET. AL, 2011][BASTOS ET. AL, 2005].

Neste trabalho, o GRASP com RC foi implementado utilizando as mesmas configurações da fase de construção e da fase de busca local utilizadas pelo GRASP na sua forma padrão, não sendo necessárias alterações nessas estruturas. O RC utilizou como técnica de otimização a abordagem *backward*, que é aplicada da melhor solução (Solução Base) para a pior solução (Solução Guia). A SB é obtida da iteração atual do GRASP, enquanto a SG é uma solução selecionada do conjunto elite.

No que diz respeito ao procedimento de RC implementado neste trabalho, a cardinalidade do conjunto elite foi definida como quatro e cada solução intermediária foi estabelecida a partir de uma migração dos vértices entre *clusters* da AGM ou do Grafo Original. Ao se construir uma solução intermediária, há o objetivo em minimizar o valor de uma função objetivo e**,** simultaneamente, produzir soluções válidas, ou seja, que satisfaçam as restrições de conexidade e de capacidade mínima do problema de clusterização em questão.

A estratégia para escolher as soluções que irão pertencer ao conjunto elite é a seguinte: inicialmente, este conjunto encontra-se vazio, mas a cada iteração do GRASP, uma solução é gerada e inserida neste conjunto. Quando o conjunto elite já está completo, a solução gerada na atual iteração do GRASP é inserida neste conjunto, caso o seu custo (função objetivo) seja menor que o custo da solução de pior qualidade do conjunto elite. Depois de verificada essa condição, necessita-se saber qual será a solução que deve ser retirada do conjunto elite. Caso seja observado que o custo da solução que deseja entrar no conjunto elite é inferior a pelo menos, a solução de maior custo já armazenada no repositório do conjunto elite, a nova solução é inserida e a solução de maior custo é retirada.

Neste trabalho, a atualização do conjunto elite, bem como a aplicação do RC, ocorrem quando a iteração do GRASP é múltipla de 10 ou quando 25% do conjunto elite for atualizado desde a última ativação (mas a primeira ativação se dá, obrigatoriamente, após a iteração 10). O processo termina quando são atingidas 100 iterações do GRASP.

3.3.3 VARIABLE NEIGHBORHOOD DESCENT (VND) OU DESCIDA EM VIZINHANÇA VARIÁVEL

Variable Neighborhood Descent ou Método de Descida em Vizinhança Variável (VND) é uma estratégia de refinamento de vizinhanças. O que difere a estratégia VND do método de busca local tradicional é que, ao invés de utilizar uma única estrutura de vizinhança,

considera-se mais de uma vizinhança. Ao atingir um ótimo local com relação à vizinhança atual, é iniciada uma nova busca local empregando outra vizinhança.

O algoritmo termina quando a solução corrente é um ótimo local de melhor qualidade em relação aos ótimos explorados em todas as vizinhanças empregadas. Portanto, as vizinhanças estendidas possibilitam a busca por soluções de melhor qualidade que estão "*mais distantes*" da solução atual. Tal procedimento permite ao método escapar de ótimos locais de baixa qualidade com respeito a uma vizinhança menor [HANSEN AND MLADENOVIC 2003].

A estrutura do VND é de fácil implementação e os seus componentes de vizinhanças podem ser alterados de forma a aumentar ou diminuir a sua complexidade para ter compromisso entre qualidade de solução e tempo computacional [CHAVES AND LORENA 2005].

O método VND [MLADENOVIC 1995] [MLADENOVIC AND HANSEN 1997] tem como característica a exploração do espaço de soluções através de trocas sistemáticas de estruturas de vizinhanças. Os autores se inspiraram em três princípios ao idealizar este método:

1º - Um ótimo local com relação a uma estrutura de vizinhança não é necessariamente um ótimo local relativo à outra estrutura de vizinhança;

2º - Um ótimo global é um ótimo local com relação a todas as possíveis estruturas de vizinhanças;

3º - Para muitos problemas, ótimos locais com relação a uma ou mais estruturas de vizinhanças são relativamente próximos.

O último princípio, segundo os autores, traz uma observação importante de natureza empírica. Mais especificamente, indica que um ótimo local frequentemente fornece algum tipo de informação sobre o ótimo global. Esse é o caso em que os ótimos local e global compartilham muitas variáveis com o mesmo valor, o que induz uma investigação sistemática da vizinhança de um ótimo local até a obtenção de uma nova solução de melhor valor.

São várias as aplicações de metaheurísticas híbridas utilizam como busca local o VND para solucionar problemas de otimização combinatória, sejam elas: Problema do Caixeiro Viajante com Coleta de Prêmios [CHAVES AND LORENA 2005], *Traveling Purchaser Problem* [SILVA ET AL 2000] e Problema do Caixeiro Viajante com Coleta e Entrega de uma Comodidade [HERNÁNDEZ-PÉREZ ET AL 2009].

A partir de uma solução inicial *s*, uma solução vizinha $s' \in N^v(s)$ é obtida pela aplicação de uma operação de movimento, sendo *v* corresponde ao número de vizinhanças

(vide figura 3.25). No caso do problema abordado neste trabalho, essa operação corresponde a mover alguns vértices de um *cluster* $C_i \in s$ para outro *cluster* $C_j \in s$. Um movimento corresponde à atribuição de todos os vértices de s para outro *cluster* C_j. Para cada atribuição, teremos uma nova solução que será uma solução vizinha de s.

Neste trabalho foram consideradas seis estruturas de vizinhanças, a saber: N^1, N^2, N^3, N^4, N^5 *e* N^6.

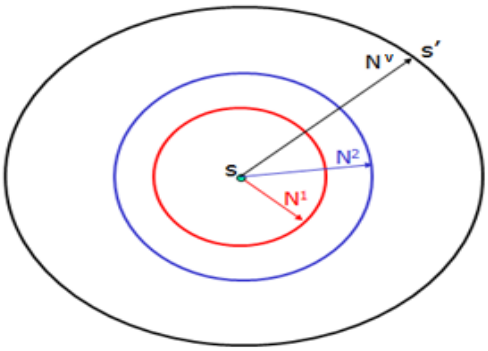


Figura 3.25 – VND com $N^v(s)$ vizinhança

- Busca na vizinhança N^1: nessa estrutura de vizinhança, um vizinho é gerado a partir de uma lista de arestas removidas da AGM e de uma lista de *clusters* penalizados. Através do VND verifica-se a possibilidade de realizar migrações de vértices entre os *clusters* utilizando estas listas. Neste caso, são respeitadas as restrições de conexidade e de capacidade mínima por *cluster*.

- Busca na vizinhança N^2: nessa estrutura de vizinhança, verifica-se a possibilidade de realizar migrações de vértices entre os *clusters* considerando o Grafo Original. Parte-se de um Grafo Original para verificar se as restrições de conexidade entre os vértices internos aos *clusters* são satisfeitas.

- Busca na vizinhança N^3: essa estrutura de vizinhança é muito similar à estrutura de vizinhança N^2, uma vez que também verifica a possibilidade de realizar migrações de vértices entre os *clusters* e considera o Grafo Original. E, assim como a estrutura da vizinhança N^2, é verificado se os vértices internos dos *clusters* estão conexos.

- Busca na vizinhança N^4: a estrutura de vizinhança N^4 utiliza uma lista de arestas removidas da AGM para geração de um *cluster*. O VND procura por vizinhos que melhorem a solução através da união e da divisão de *clusters* já construídos.

- Busca na vizinhança N^5: a estrutura de vizinhança N^5 considera a AGM e é similar a vizinhança N^4. Difere apenas por buscar novas soluções não penalizadas, que atendam a restrição de capacidade mínima.

- Busca na vizinhança N^6: o VND na estrutura de vizinhança N^6 procura minimizar a função objetivo da solução através da migração dos vértices entre *clusters* que é realizada utilizando o Grafo Original.

Neste trabalho, As vizinhanças N^1, N^2, N^3, N^4, N^5 *e* N^6 são integradas na busca local do GRASP usando o algoritmo VND, ou seja, N^1 corresponde a Busca Local 1, N^2 corresponde a Busca Local 2, N^3 corresponde a Busca Local 3, N^4 corresponde a Busca Local 4, N^5 corresponde a Busca Local 5 e N^6 corresponde a Busca Local 6.

Primeiramente, são investigados todos os possíveis movimentos da vizinhança N^1 que levam a uma solução que seja um mínimo local de melhor qualidade nesta vizinhança. Quando este mínimo local for alcançado, parte-se para a investigação de soluções na vizinhança N^2 que seja o mínimo local da vizinhança em N^2 . Posteriormente, retorna-se à investigação de soluções na vizinhança N^1 para garantir o seu mínimo local entre N^1 *e* N^2.

O mesmo procedimento é realizado incluindo-se as vizinhanças N^3, N^4, N^5 *e* N^6 até que não existam mais movimentos em nenhuma das vizinhanças que diminuam o custo da solução. Portanto, o VND assegura que o mínimo local obtido não corresponde apenas à melhor solução encontrada para uma vizinha, mas sim o a melhor solução dentre todas as 6 (seis) vizinhanças pré-estabelecidos pelo problema definido nesta tese.

3.3.4 GRASP com VND

A estratégia VND (*Variable Neighborhood Descent*) ou Método de Descida em Vizinhança Variável foi inicialmente introduzida no método GRASP para o problema de Steiner em grafos, com soluções de acordo com a estratégia gulosa randomizada do método. Nele, a hibridização foi feita substituindo-se a busca local por uma estratégia VND que explorava duas estruturas de vizinhanças [RIBEIRO E VIANA 2005]. A metaheurística GRASP também foi combinada a uma estratégia de VND para o problema da árvore geradora mínima com restrição de grau [RIBEIRO E SOUZA 2002]

Assim, propõe-se, neste trabalho, utilizar o VND com um módulo de busca local a ser incorporado à metaheurística GRASP para o problema de clusterização de grafos com restrição de capacidade e conexidade.

O objetivo do VND é identificar os clusters que garantam as restrições de conexidade e capacidade mínima através de uma intensificação da busca por melhor solução na primeira, na segunda, na terceira, na quarta, na quinta e na sexta vizinhança do *cluster*.

O algoritmo 3 mostra o pseudo-código do algoritmo GRASP com VND implementado neste trabalho:

Algoritmo 3: GRASP com VND

Entrada: Grafo *G = (A,V)*

1. melhorSolução ←0;

2. solução ←AlgoritmoConstrução;

3. melhorSolução ←VND (solução);

Saída: melhorSolução.

Figura 3.26 – Pseudo-código do algoritmo GRASP com VND

Inicialmente, tem-se como entrada para a execução da metaheurística GRASP com VND o Grafo com seus vértices e suas arestas.

Linha 1 inicializa-se a variável melhor Solução com o valor 0 (zero), variável está que irá receber o melhor resultado do GRASP.

Linha 2: é construída uma solução inicial para o problema;

Linha 3: realizada uma busca local - VND na solução inicial;

A saída do algoritmo GRASP com VND é a melhor solução encontrada.

3.3.5 RANDOM VARIABLE NEIGHBORHOOD DESCENT (RVND) OU DESCIDA EM VIZINHANÇA VARIÁVEL COM ORDEM ALEATÓRIA

Random Variable Neighborhood Descent (RVND) ou Método de Descida em Vizinhança Variável com Ordem Aleatória [SUBRAMANIAN ET AL 2010] é uma variação do método *Variable Neighborhood Descent* (VND) ou Método de Descida em Vizinhança Variável [MLADENOVIC AND HANSEN, 1997].

O método VND Clássico [HANSEN ET AL 2010] trabalha com um conjunto de vizinhanças previamente ordenadas, já o RVND utiliza uma ordem aleatória de vizinhanças a cada execução. Essa é uma vantagem, uma vez que não é necessário fazer testes para descobrir qual a melhor ordem [PENNA ET AL, 2011].

A estrutura do RVND é simples de implementar, ao invés de usar uma sequência determinística de vizinhanças para explorar o espaço de soluções, utiliza-se uma ordem aleatória a cada execução. Assim, sempre que em uma determinada vizinhança não for

possível melhorar a solução corrente, o RVND seleciona, aleatoriamente, outra vizinhança para continuar a busca pelo espaço de soluções [SUBRAMANIAN ET AL 2010].

Então, sendo $N = \{N^1, N^2, N^3, ..., N^v\}$ o total de vizinhanças, o RVND selecionará aleatoriamente a ordem em que estas estrutura N^vs serão executadas.

Neste trabalho foram consideradas como estruturas de vizinhanças as mesmas definidas no VND, ou seja, seis estruturas de vizinhanças, quais sejam: N^1, N^2, N^3, N^4, N^5 *e* N^6.

3.3.6 GRASP com RVND

A estratégia RVND (*Random Variable Neighborhood Descent*) ou Método de Descida em Vizinhança Variável com Ordem Aleatória foi utilizado com a metaheurística GRASP, inicialmente, em Souza et. al, 2010, para resolver o problema de planejamento operacional da mineração em céu aberto, com alocação dinâmica de caminhões.

Neste trabalho, o RVND foi incorporado ao GRASP como um módulo de busca local, sendo que a escolha da mesma acontece de forma aleatória. Mais especificamente, o RVND tem sua aplicação dentro do GRASP condicionada à busca por melhores soluções nas vizinhanças dos *clusters*, ou seja, age na intensificação das soluções do GRASP.

3.3.7 *VARIABLE NEIGHBORHOOD SEARCH* (**VNS**) OU BUSCA EM VIZINHANÇA VARIÁVEL

Em 1997, Hansen e Mladenović propuseram uma metaheurística que se baseava em uma troca sistemática de vizinhanças, associada a um algoritmo aleatório na determinação de pontos iniciais da busca local, chamada de Busca em Vizinhança Variável, conhecida na literatura em inglês como *Variable Neighborhood Search* (VNS). Contrariamente a outras metaheurísticas baseadas em métodos de busca local, o VNS não segue uma trajetória, mas explora incrementalmente vizinhanças mais ou menos distantes da solução corrente, partindo da solução atual para a nova se, e somente se, uma melhora ocorrer. Portanto, é baseada num princípio simples: explorar o espaço de soluções através de trocas sistemáticas de estruturas de vizinhanças durante o processo de busca [MLADENOVIC AND HANSEN, 1999] [MLADENOVIC AND HANSEN, 2001d].

Diferentemente de outras metaheurísticas, a estrutura do VNS básico e suas extensões são simples e requerem poucos, e às vezes nenhum parâmetro. Sua disseminação foi rápida, com inúmeras publicações [HANSEN ET AL, 2001] [HANSEN AND MLADENOVIC,

2001a] [HANSEN AND MLADENOVIC, 2001b] [HANSEN AND MLADENOVIC, 2002] [HANSEN AND MLADENOVIC, 2003a] [HANSEN AND MLADENOVIC, 2003b].

Uma característica importante do VNS é que este método aceita somente movimentos de melhora da solução e sempre que isto ocorre, ele retorna à primeira vizinhança. A ideia intrínseca a este método é explorar uma vizinhança ao máximo, enquanto resultados satisfatórios forem sendo obtidos.

O VNS possui um procedimento de perturbação que possui a finalidade de diversificação da solução a fim de evitar a ciclagem (retorno sistemático ao mínimo local anteriormente alcançado) do algoritmo. A cada iteração, uma solução vizinha da solução inicial é escolhida aleatoriamente e servirá de base para a execução da busca local.

O algoritmo VNS trabalha com várias vizinhanças. Portanto, seja N_v o conjunto finito de estruturas de vizinhanças pré-selecionadas, com ($v = 1, ..., v_{max}$) e $N_v(s)$ o conjunto de soluções na *v-é*sima vizinhança de *s*. Além do conjunto N_v de vizinhanças, usa-se a função de avaliação *f*, a ser minimizada. Uma solução ótima s (ótimo global) é uma solução viável de tal maneira que para cada solução viável $s' \in S$, tem-se que $f(s) < f(s')$.

Uma ilustração (Figura 3.27) do VNS pode ser vista no Algoritmo 4.

Algoritmo 4: VNS

Entrada: Grafo *G = (A,V)*

1. **Inicialização:** selecionar o conjunto de estruturas de vizinhanças Nv ($v=1, ..., v_{max}$) que serão usadas na busca;
   Encontrar uma solução inicial *s* e Escolher um critério de parada;
2. **Enquanto** (*critério de parada não satisfeito*) **faça**
3. v ← 1;
4. **Enquanto** ($v \leq v_{max}$) **faça**
5. **Perturbação (*shaking*):** gerar aleatoriamente uma solução *s'* na *v-ésima* vizinhança de *s* ($s' \in N_v(s)$);
6. **Busca Local:** aplicar um método de busca local em *s'*, obtendo um ótimo local *s'';*
7. **Mover ou não:**
   Se o ótimo local *s''* é melhor que o atual, ou seja, $f(s'') < f(s)$ **então**
8. Mover para lá (*s* ← ~~*s''*~~),
9. Ir para a próxima vizinhança (*v* ← ~~*1*~~);
   Senão
10. Passar para a próxima vizinhança (*v* ← ~~*k*~~ + *1*);
    fimSe
   fimEnquanto
  fimEnquanto

Saída: melhorSolução

Figura 3.27 – Pseudo-código do algoritmo VNS

A seguir, tem-se uma explicação do algoritmo VNS.

3.3.7.1 ALGORITMO VNS

O algoritmo VNS proposto nesta tese é composto por duas fases, a saber: Na primeira fase é construída uma Árvore Geradora Mínima (AGM) e, em seguida, realizado o seu particionamento em *n clusters* de forma que satisfaça às restrições de conexidade e capacidade tratadas neste trabalho. A segunda fase é caracterizada pela aplicação de um procedimento VNS na solução com menor valor de função objetivo (Equação 7) resultante da etapa anterior.

A partir de uma solução inicial *s*, uma solução vizinha $s' \in N^v(s)$ é obtida pela aplicação de uma operação de movimento, sendo *v* correspondente ao número de vizinhanças.

Assim com foi utilizado na metaheurística GRASP, define-se o Grafo Original (G) considerando as informações existentes na base dados utilizada (Censo Demográfico 2010) definindo dessa forma os seus vértices e arestas existentes. Aplica-se o algoritmo de *Kruskal* para se construir a AGM *T* com custo mínimo. A geração da AGM implica geração de um ótimo local para o problema, uma vez que o valor da função objetivo calculado em cada *cluster*, escolhendo-se seu valor em função da distância de cada um dos vértices aos centróides (média dos vértices em relação aos atributos).

Parte-se da premissa de que a AGM corresponde a um único *cluster*. Então, inicia-se o particionamento da AGM, onde cada aresta que for eliminada irá dividir a árvore em duas novas subárvores, ou seja, dois *clusters* (Construtor 1, Construtor 2, Construtor 3, Construtor 4 ou Construtor 5). Assim, deve-se, particionar a AGM obtendo como resultado *k* grafos conexos (*clusters*). E assim sucessivamente, até que não seja mais possível encontrar uma aresta que, ao ser removida, produza *clusters* com capacidade maior ou igual a um valor mínimo já pré-estabelecido.

Ao se concluir essa primeira etapa do algoritmo VNS, são produzidas *n + 1* soluções iniciais s, dentre estas, seleciona-se a solução com menor valor associado à função objetivo. Então, calcula-se a distância de cada um dos vértices *V* que compõe o *cluster* ao seu respectivo centroide. Coloca-se, em ordem crescente as distâncias obtidas nas vizinhanças Busca Local 1 (Vizinhança 1), Busca Local 2 (Vizinhança 2), Busca Local 3 (Vizinhança 3), Busca Local 4 (Vizinhança 4), Busca Local 5 (Vizinhança 5) e Busca Local 6 (Vizinhança 6). Sortear um grupo doador dentro os *k* grupos (subárvores) *clusters*, atribuindo maior chance de seleção aos grupos cujos soma das distâncias dos setores (vértices) são maiores. Em seguida, selecionar *t* objetos doares na Vizinhança N_1 e tentar realocá-los a um dos *(k – 1)* grupos

receptores, de forma a produzir a maior redução possível na função objetivo. Se houver redução na função objetivo, continua-se em N_1, senão alterna-se para N_2, e assim sucessivamente, enquanto não obtiver resultado melhor. O objetivo principal é possibilitar aos objetos mais distanciados do centroide uma maior chance de mudar de *cluster*.

Posteriormente, deve-se definir o número de vizinhanças do VNS, que no caso deste trabalho segue as mesmas estruturas vizinhanças definidas na Seção 3.3.3.

O procedimento de perturbação é semelhante ao Procedimento de Busca Local 1, porém com o objetivo de perturbar a solução, evitando que esta fique estagnada em ótimos locais de baixa qualidade e regenerando possíveis soluções inválidas que tenham sido eliminados pelos outros procedimentos de busca local. A Figura 3.28 ilustra um exemplo da aplicação do Procedimento de Perturbação e suas respectivas explicações:

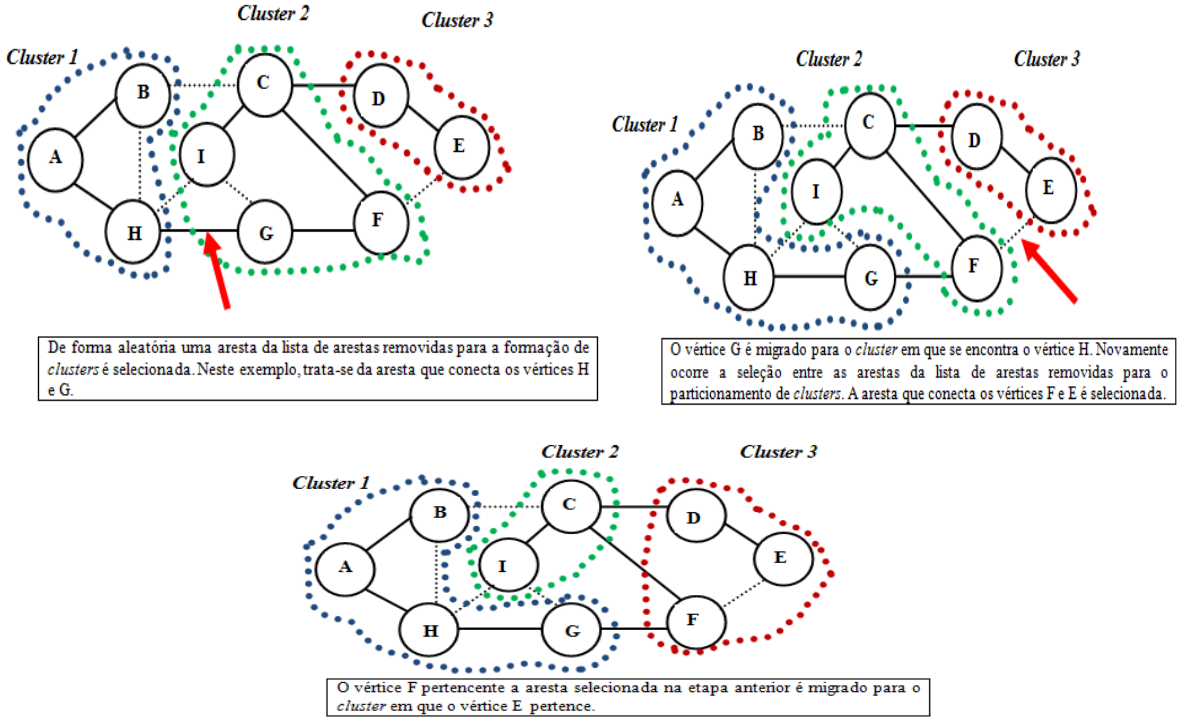


Figura 3.28 – Procedimento de Perturbação

O procedimento de Perturbação procura selecionar os *x* vértices do cluster doador na vizinhança N^1 e tentar realocá-los a um dos *(k – 1) clusters* receptores, de forma a tentar reduzir o valor da função objetivo, conforme Figura 3.28

A finalidade desta etapa é produzir soluções $s' \in N^v(s)$ de qualidade superior à solução s. Caso haja redução no valor da função objetivo, deve-se permanecer em N^1 a procura de melhores resultados, senão, troca-se de vizinhança, e assim sucessivamente, até

que o critério de parada seja satisfeito, sempre utilizando o procedimento de perturbação em cada vizinhança.

Em [HANSEN AND MLADENOVIC 2001a] foram propostas várias extensões da VNS e em [HANSEN AND MLADENOVIC 2003a] é descrito um tutorial sobre VNS, incluindo também várias extensões, exemplos e questões em relação à implementação. Na literatura aparecem diversas formas de estender a VNS, às quais geralmente são acrescentadas algumas características adicionais. A evolução da VNS pode ser vista nas Seções a seguir.

3.3.8 *GENERAL VARIABLE NEIGHBORHOOD SEARCH* (**GVNS**) OU BUSCA GERAL EM VIZINHANÇA VARIÁVEL

A estratégia GVNS (*General Variable Neighborhood Search*) ou Busca Geral em Vizinhança Variável é inspirada na ideia de buscar boas soluções por meio de estruturas de vizinhanças diferentes. Um algoritmo similar ao desenvolvido neste trabalho já existe na literatura com bons resultados [SOUZA ET AL 2010]. Ele foi utilizado para resolver o Problema de Planejamento Operacional de Lavra em Minas a Céu Aberto, produzindo soluções viáveis e competitivas quando comparadas com as soluções do modelo de programação matemática executado pelo *CPLEX*. A metaheurística GVNS, pode ser divida em duas fases, a saber: perturbação e busca local. Na fase de perturbação, o algoritmo gera uma solução aleatória em uma dada vizinhança (exploração estocástica das estruturas de vizinhança) e na fase de busca local, é aplicado o método VND.

O algoritmo GVNS baseia-se nos critérios de busca local em torno de uma solução corrente com intuito de explorar novas regiões, o que é análogo ao algoritmo VNS. No entanto, a diferença fundamental está na fase de melhoria da busca local usando o VND. O GVNS tem sido um dos métodos que mais obteve êxito recentemente, por exemplo, analisando os trabalhos de [ANDREATA E RIBEIRO 2002] [BRINBERG ET AL 2000] [CAPOROSSI ET AL 1999] [CAPOROSSI E HANSEN 2000] [RIBEIRO E SOUZA 2002].

O algoritmo GNVS aplicado ao problema tratado neste trabalho segue os mesmos passos implementados no algoritmo VNS, com mesmas estruturas de vizinhanças utilizadas neste, diferindo apenas na busca local onde foi aplicado um VND.

3.3.9 REDUCED VARIABLE NEIGHBORHOOD SEARCH (**RVNS**) OU BUSCA REDUZIDA EM VIZINHANÇA VARIÁVEL

Uma redução da busca em vizinhança variável original foi também proposta por Mladenovic 1997, gera uma solução inicial sem executar uma busca local nessa solução, já se

direcionada à exploração das estruturas de vizinhanças [MLADENOVIC AND HANSEN 1997]. Tal fato pode melhorar o tempo do algoritmo VNS em casos em que um método de busca local tem um custo computacional muito elevado. Essa redução é chamada *Reduced Variable Neighborhood Search* (RVNS).

Esse tipo de algoritmo VNS chamado de RVNS [MARTINS 2009] é inspirado em dois aspectos fundamentais no processo de busca relacionados com a intensificação e a diversificação.

No algoritmo RVNS um conjunto de estruturas de vizinhanças N_v ($v=1, ..., v_{max}$) será considerado ao redor da solução atual *s* (o qual pode ser ou não um ótimo local). Geralmente essas vizinhanças estão aninhadas, isto é, os elementos da vizinhança N_1 também são elementos na vizinhança N_2 e assim sucessivamente.

Deve-se observar que o algoritmo RVNS (Figura 3.29) produz uma escolha de vizinhos mais dinâmica a partir da seleção de vizinhos de todas as estruturas de vizinhança (diversificação) e priorizando a primeira estrutura de vizinhança (intensificação) nas fases iniciais da busca. Entretanto, uma componente importante da estrutura RVNS é a capacidade de encontrar novas regiões promissoras a partir de um ótimo local.

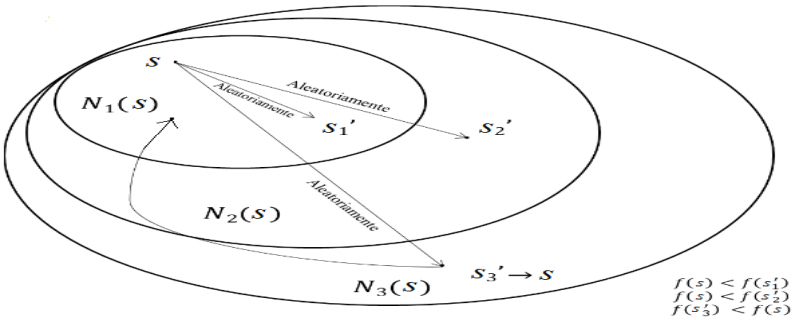


Figura 3.29 – Comportamento do RVNS com três vizinhanças
Adaptado de Souza, 2011.

Observa-se que o algoritmo RVNS aplicado ao problema tratado neste trabalho segue os mesmos passos do algoritmo VNS, utilizando até as mesmas estruturas de vizinhanças.

3.4.0 *SKEWED VARIABLE NEIGHBORHOOD SEARCH* (**SVNS**) OU BUSCA INCLINADA EM VIZINHANÇA VARIÁVEL

Ao assumir vizinhanças cada vez maiores, as soluções geradas aleatoriamente podem diferenciar-se substancialmente da solução atual, e por isso, o VNS se degenera em uma

heurística de partidas múltiplas (no sentido de que se realizam, interativamente, descidas com soluções geradas aleatoriamente). Além disso, se a solução corrente não está no vale mais profundo**,** esta informação é em parte irrelevante. [LIN AND KERNIGHAN 1973] [HANSEN AND MLADENOVIC 2000]. Seria, então, interessante modificar o esquema do VNS para explorar melhor aqueles vales que estão mais longe da solução incumbente. Isto pode ser feito aceitando realizar um movimento para uma solução de valor próximo da incumbente atual, mas não necessariamente melhor, desde que esta solução esteja longe da incumbente [VANSTEENWEGEN AND SOUFFRIAU 2009] [VANSTEENWEGEN AND SOUFFRIAU 2009] [VANSTEENWEGEN ET AL 2010]. Para lidar com esses casos, surgiu uma nova variantes do VNS, o algoritmo *Skewed Variable Neighborhood Serch* (SVNS) ou Busca Inclinada em Vizinhança Variável.

O SVNS introduz uma função que controla a distância entre as soluções produzidas (métrica dentro das estruturas de vizinhança).

3.4.0.1 ALGORITMO **SVNS**

O algoritmo SVNS que será apresentado a seguir é composto por duas fases, ou seja, segue o mesmo esquema do algoritmo VNS (Seção 3.3.7.1) com algumas peculiaridades a serem abordadas a seguir:

Na etapa da Agitação (*shaking*) deve-se sortear, assim como no método de seleção da roleta utilizado nos algoritmos genéticos, um *cluster* com probabilidade inversamente proporcional ao valor da função objetivo, ou seja, atribuindo maior chance de escolha aos *clusters* cujas somas das distâncias dos vértices aos respectivos centroides sejam maiores.

Uma vez definida o *cluster*, logo em seguida, definir a vizinhança, no caso deste trabalho igual a três. O *k = 3*, e como já se sabe que quais vértices pertencem a quais *clusters*, pode-se determinar a distância (em ordem crescente) de cada um dos objetos aos centroides:

- seja *k = 1*, seleciona-se os *p%* vértices que estejam, a no máximo 15% distante do centroide;

- seja *k = 2*, seleciona-se os *p%* vértices que estejam, a no máximo 10% distante do centroide;

- seja *k = 3*, seleciona-se os *p%* vértices que estejam, a no máximo 5% distante do centroide;

Uma vez selecionados os *p%* vértices, fica definido um novo *cluster p* a ser testado. A finalidade desta etapa é produzir soluções $s' \in N^v(s)$ de qualidade superior à solução *s*. Após as trocas, os centroides são recalculados e a busca prossegue. O objetivo dessa etapa é alocar

os demais vértices ao centroide mais próximo e efetuar as atualizações necessárias dos centroides considerando as restrições de conexidade e capacidade.

Nas etapas seguintes pode-se considerar como resultado da solução, a solução atual ou uma solução já armazenada (em caso de não haver melhoramento desta etapa parcial). A ideia é tornar-se mais tolerante na aceitação de uma solução retornada pela busca local, não necessariamente melhorando a solução candidata. Um *cluster* com custo um pouco pior pode ser aceito, porém tais limites de aceitação precisam ser definidos. Neste sentido, foram considerados neste trabalho os resultados que não apresentaram melhoria, desde que o valor não se afastasse muito do valor atual. Devido a isso, precisamos guardar a melhor solução encontrada até então. Já a função objetivo do ótimo local *f(s'')* foi substituída por *f(s'')* – $\alpha\rho$*(s, s'')*, onde ρ*(s, s'')*, é a diferença do valor da função objetivo *s* e *s''* e α é um parâmetro. Definiu-se $\alpha = 0.9$, após um conjunto de testes para ajuste deste parâmetro.

E finalizou-se o algoritmo SVN quando o critério de parada foi satisfeito.

CAPÍTULO 4 - EXPERIMENTOS COMPUTACIONAIS

O presente capítulo traz os resultados relativos a um conjunto de experimentos computacionais realizados com os algoritmos descritos no Capítulo 3 deste trabalho de tese.

Conforme apresentado no Capítulo 3, os algoritmos considerados nos experimentos do presente trabalho foram os algoritmos construtivos GRASP: Padrão, com Reconexão de Caminho, com Busca Local VND e com Busca Local Randômica RVND, além dos algoritmos VNS Padrão e suas variantes RVNS, GVNS e SVNS.

As instâncias utilizadas para avaliar os algoritmos supracitados foram obtidas a partir dos dados do Censo Demográfico 2010 do IBGE (dados de uso público disponíveis no site do IBGE).

4. 1 TECNOLOGIAS

As implementações dos algoritmos propostos foram feitas em Linguagem *C*, utilizando como ambiente de desenvolvimento o *NetBeans 7.2*. Todos os experimentos computacionais foram realizados em um computador dotado de processador AMD de 1.4Ghz e com 4GB de memória *RAM* e sistema operacional *Windows 7*.

4.2 INSTÂNCIAS UTILIZADAS

Este trabalho, em seu estudo de caso, utiliza 64 (sessenta e quatro) instâncias referentes aos problemas de Criação de Áreas de Ponderação Agregadas e de Agregados de Municípios [CENSO DEMOGRÁFICO 2010]. No caso do censo demográfico, os objetos podem estar associados, por exemplo, a domicílios, setores censitários, áreas de ponderação (APOND) e a municípios.

Utilizou-se um conjunto de instâncias obtidas a partir dos dados da amostra do Censo Demográfico 2010. Para cada instância foram utilizados dois arquivos do tipo texto. O primeiro arquivo contendo as informações de vizinhança entre os vértices/objetos (setores censitários) e o segundo contendo os atributos associados a estes vértices/objetos (setores censitários), sejam eles: total de domicílios particulares permanentes em cada setor censitário (variável de capacidade mínima), proporção de domicílios com banheiros e total de rendimento nominal mensal dos domicílios particulares permanentes.

Observa-se, ainda, que para todas as instâncias utilizadas nesse trabalho, a variável total de domicílios particulares permanente (disponível para cada setor censitário) do IBGE foi utilizada como variável de capacidade no processo de formação dos grupos. Assim,

estipulou-se previamente, que a soma dos valores total de domicílios associados a cada uma das APONDs não deveria ser inferior a um parâmetro *P* pré-estabelecido.

O arquivo referente à vizinhança é utilizado para determinar a relação entre os vértices do Grafo *G* a partir do qual será construída a AGM *T*. Assim, cada objeto corresponde a um vértice do grafo ao qual também terá associado seus atributos. Tais atributos devem ser normalizados e submetidos ao cálculo das distâncias d_{ij} nas arestas *(i, j)* de *G*.

Tabela 4.1: Relação das instâncias utilizadas

Código	Vértices Número de Objetos (Setores Censitários)	Arestas (Número de Vizinhanças)
1	88	438
2	157	720
3	20	224
4	102	586
5	34	298
6	28	322
7	22	326
8	15	106
9	53	604
10	75	470
11	42	288
12	67	528
13	35	354
14	30	318
15	70	305
16	32	254
17	44	362
18	25	438
19	259	1518
20	202	1058
21	323	2140
22	83	442
23	27	316
24	24	246
25	21	318
26	411	3150
27	46	384
28	12	105
29	97	514
30	101	499
31	59	735
32	382	2643
33	26	452
34	77	481
35	38	367
36	93	497
37	122	658
38	240	1283
39	221	1197
40	19	121
41	72	364
42	94	502
43	489	3881
44	29	468
45	152	707

46	90	514
47	48	397
48	189	913
49	64	514
50	100	561
51	523	4287
52	11	102
53	146	691
54	82	435
55	359	2577
56	307	1925
57	63	501
58	18	114
59	162	743
60	592	4997
62	274	1836
62	581	4728
63	60	471
64	404	3002

4.3 ALGORITMOS

Conforme descrito no Capítulo 3, os procedimentos e as estratégias implementadas nesta tese utilizam o algoritmo GRASP e incorporam o procedimento *Path-Relinking* (PR), uma estratégia de busca local baseada no VND e outra estratégia de Busca local baseada num RVND. Além de utilizar o algoritmo VNS Padrão e suas variantes GVNS, RVNS e SVNS. Todos esses algoritmos utilizam como dado de entrada, especificamente, a AGM construída com o procedimento *Kruskal* em sua forma padrão e a AGM construída considerando o procedimento *Kruskal* modificado através da aleatoriedade de seus pesos, além de trabalhar também com o Grafo Original. Estes três tipos de dados de entrada direcionam a definição três categorias de algoritmos para cada versão algoritmo implementado, sejam eles assim denominados:

- GRASP padrão considerando: (i) a AGM com *Kruskal padrão*, (ii) apenas a AGM com *Kruskal* modificado e (iii) apenas o Grafo Original;

- GRASP com o procedimento *Path-Relinking* considerando: (i) a AGM com *Kruskal padrão*, (ii) apenas a AGM com *Kruskal* modificado e (iii) apenas o Grafo Original;

- GRASP com a estratégia VND considerando: (i) a AGM com *Kruskal padrão*, (ii) AGM com *Kruskal* modificado e (iii) apenas o Grafo Original;

- GRASP com a estratégia RVND considerando: (i) a AGM com *Kruskal padrão*, (ii) apenas a AGM com *Kruskal* modificado e (iii) apenas o Grafo Original;

- VNS padrão considerando: (i) a AGM com *Kruskal padrão*, (ii) apenas a AGM com *Kruskal* modificado e (iii) apenas o Grafo Original;

- GVNS considerando: (i) a AGM com *Kruskal padrão*, (ii) apenas a AGM com *Kruskal* modificado e (iii) apenas o Grafo Original;

- RVNS considerando: (i) a AGM com *Kruskal padrão*, (ii) apenas a AGM com *Kruskal* modificado e (iii) apenas o Grafo Original;

- SVNS considerando: (i) a AGM com *Kruskal padrão*, (ii) apenas a AGM com *Kruskal* modificado e (iii) apenas o Grafo Original;

Além disso, observamos que cada grupo de algoritmos é definido a partir de uma combinação de procedimentos de construção e de busca local implementados neste trabalho, como será visto logo à frente nas Tabelas 4.2, 4.3, 4.4, 4.5, 4.6 e 4.7 perfazendo um total de 40 (quarenta) versões de algoritmos considerando a AGM construída com *Kruskal* padrão, 40 (quarenta) versões de algoritmos considerando a AGM construída com *Kruskal* modificado e 40 (quarenta) versões de algoritmos considerando somente o Grafo Original.

Além da combinação dos procedimentos de construção e de busca local implementados neste trabalho, devem ser informados aos algoritmos o critério de parada, quais versões dos procedimentos de construção e de busca serão executados e os valores de alguns parâmetros:

- **Critério de parada**: pode ser utilizada a quantidade de iterações ou o tempo de execução;

- **Procedimentos Construtores** (**C**): todos os algoritmos devem especificar qual algoritmo construtor deve ser utilizado. Nesse trabalho, foram implementadas cinco versões de procedimentos construtores, sendo que em todas as versões a restrição de conexidade foi atendida; Assim, C-1 representa o procedimento construtor versão 1 (um), e assim por diante.

- **Procedimentos de Busca Local** (**BL**): versões de procedimentos de busca local que serão utilizadas pelo algoritmo. Assim, BL-1 representa a busca local versão 1 (um), e assim sucessivamente;

- **Alfa** ($\boldsymbol{\alpha}$): este parâmetro é utilizado com o objetivo de atribuir aos procedimentos construtores um comportamento semi-guloso, mediante a manipulação do tamanho da LRC (Lista Restrita de Candidatos). Este parâmetro, no caso específico deste problema de clusterização com restrição de capacidade e conexidade representa o limite máximo do tamanho da LRC, considerando que pode existir uma quantidade inferior de elementos desta LRC;

- **Quantidade de *clusters*** (***k***): representa o número de *clusters* utilizados no problema;

- **Capacidade Mínima por *cluster*** ($\boldsymbol{Capacidade_{Min\_Cluster}}$): é o valor do fator de ajuste utilizado para cálculo da capacidade mínima por *cluster*, conforme Equação 14. A

Tabela 4.2 apresenta os algoritmos GRASP Padrão, com *Path-Relinking*, com VND e RVND considerando apenas o Grafo Original com as combinações dos procedimentos de construção e os de busca local, com suas respectivas siglas.

Tabela 4.2: Siglas dos algoritmos utilizando o Grafo Original

ALGORITMOS	SIGLA	CONSTRUTORES					BUSCA LOCAL		
		C-1	C-2	C-3	C-4	C-5	BL-2	BL-3	BL-6
GRASP Padrão	G1	X					X	X	X
	G2		X				X	X	X
	G3			X			X	X	X
	G4				X		X	X	X
	G5					X	X	X	X
GRASP com Path-Relinking	GPR1	X					X	X	X
	GPR2		X				X	X	X
	GPR3			X			X	X	X
	GPR4				X		X	X	X
	GPR5					X	X	X	X
GRASP com VND	GVND1	X					X	X	X
	GVND2		X				X	X	X
	GVND3			X			X	X	X
	GVND4				X		X	X	X
	GVND5					X	X	X	X
GRASP com RVND	GRVND1	X					X	X	X
	GRVND2		X				X	X	X
	GRVND3			X			X	X	X
	GRVND4				X		X	X	X
	GRVND5					X	X	X	X

A Tabela 4.3 apresenta os algoritmos GRASP Padrão, com *Path-Relinking* e com VND considerando somente a AGM construída utilizando o *Kruskal* modificado com as combinações dos procedimentos de construção e os de busca local, com suas respectivas siglas, implementados neste trabalho.

Tabela 4.3: Siglas dos algoritmos utilizando a AGM construída *Kruskal* Modificado

ALGORITMOS	SIGLA	CONSTRUTORES					BUSCA LOCAL		
		C-1	C-2	C-3	C-4	C-5	BL-1	BL-4	BL-5
GRASP Padrão	AGMM1	X					X	X	X
	AGMM2		X				X	X	X
	AGMM3			X			X	X	X
	AGMM4				X		X	X	X
	AGMM5					X	X	X	X
GRASP com Path-Relinking	PR_AGMM1	X					X	X	X
	PR_AGMM2		X				X	X	X
	PR_AGMM3			X			X	X	X
	PR_AGMM4				X		X	X	X
	PR_AGMM5					X	X	X	X
GRASP com VND	V_AGMM1	X					X	X	X
	V_AGMM2		X				X	X	X
	V_AGMM3			X			X	X	X
	V_AGMM4				X		X	X	X
	V_AGMM5					X	X	X	X
GRASP com RVND	RV_AGMM1	X					X	X	X
	RV_AGMM2		X				X	X	X

GRASP com RVND	**RV_AGMM3**			**X**			**X**	**X**	**X**
	RV_AGMM4				**X**		**X**	**X**	**X**
	RV_AGMM5					**X**	**X**	**X**	**X**

A Tabela 4.4 apresenta os algoritmos GRASP Padrão, com *Path-Relinking* e com VND considerando somente a AGM construída utilizando o *Kruskal* padrão com as combinações dos procedimentos de construção e os de busca local, com suas respectivas siglas, implementados neste trabalho.

Tabela 4.4 Siglas dos algoritmos utilizando a AGM construída *Kruskal* Padrão

ALGORITMOS	**SIGLA**	**CONSTRUTORES**					**BUSCA LOCAL**		
		C-1	**C-2**	**C-3**	**C-4**	**C-5**	**BL-1**	**BL-4**	**BL-5**
GRASP Padrão	**AGM1**	**X**					**X**	**X**	**X**
	AGM2		**X**				**X**	**X**	**X**
	AGM3			**X**			**X**	**X**	**X**
	AGM4				**X**		**X**	**X**	**X**
	AGM5					**X**	**X**	**X**	**X**
GRASP com Path-Relinking	**PR_AGM1**	**X**					**X**	**X**	**X**
	PR_AGM2		**X**				**X**	**X**	**X**
	PR_AGM3			**X**			**X**	**X**	**X**
	PR_AGM4				**X**		**X**	**X**	**X**
	PR_AGM5					**X**	**X**	**X**	**X**
GRASP com VND	**V_AGM1**	**X**					**X**	**X**	**X**
	V_AGM2		**X**				**X**	**X**	**X**
	V_AGM3			**X**			**X**	**X**	**X**
	V_AGM4				**X**		**X**	**X**	**X**
	V_AGM5					**X**	**X**	**X**	**X**
GRASP com RVND	**RV_AGM1**	**X**					**X**	**X**	**X**
	RV_AGM2		**X**				**X**	**X**	**X**
	RV_AGM3			**X**			**X**	**X**	**X**
	RV_AGM4				**X**		**X**	**X**	**X**
	RV_AGM5					**X**	**X**	**X**	**X**

A Tabela 4.5 apresenta os algoritmos VNS Padrão e suas variações (GVNS, RVNS E SVNS) e considerando apenas o Grafo Original com as combinações dos procedimentos de construção e os de busca local, com suas respectivas siglas.

Tabela 4.5 Siglas dos algoritmos utilizando também o Grafo Original

ALGORITMOS	**SIGLA**	**CONSTRUTORES**					**BUSCA LOCAL**		
		C-1	**C-2**	**C-3**	**C-4**	**C-5**	**BL-2**	**BL-3**	**BL-6**
VNS Padrão	**VNS1**	**X**					**X**	**X**	**X**
	VNS2		**X**				**X**	**X**	**X**
	VNS3			**X**			**X**	**X**	**X**
	VNS4				**X**		**X**	**X**	**X**
	VNS5					**X**	**X**	**X**	**X**
GVNS	**GVNS1**	**X**					**X**	**X**	**X**
	GVNS2		**X**				**X**	**X**	**X**
	GVNS3			**X**			**X**	**X**	**X**
	GVNS4				**X**		**X**	**X**	**X**
	GVNS5					**X**	**X**	**X**	**X**
RVNS	**RVNS1**	**X**					**X**	**X**	**X**
	RVNS2		**X**				**X**	**X**	**X**

	RVNS3			**X**			**X**	**X**	**X**
	RVNS4				**X**		**X**	**X**	**X**
	RVNS5					**X**	**X**	**X**	**X**
SVNS	**SVNS1**	**X**					**X**	**X**	**X**
	SVNS2		**X**				**X**	**X**	**X**
	SVNS3			**X**			**X**	**X**	**X**
	SVNS4				**X**		**X**	**X**	**X**
	SVNS5					**X**	**X**	**X**	**X**

A Tabela 4.6 apresenta os algoritmos VNS Padrão e suas variações (GVNS, RVNS E

Tabela 4.6 - Siglas dos algoritmos utilizando a AGM construída com *Kruskal* modificado

ALGORITMO	**SIGLA**	**CONSTRUTORES**					**BUSCA LOCAL**		
		C-1	**C-2**	**C-3**	**C-4**	**C-5**	**BL-1**	**BL-4**	**BL-5**
VNS Padrão	**VNS_AGMM1**	**X**					**X**	**X**	**X**
	VNS_AGMM2		**X**				**X**	**X**	**X**
	VNS_AGMM3			**X**			**X**	**X**	**X**
	VNS_AGMM4				**X**		**X**	**X**	**X**
	VNS_AGMM5					**X**	**X**	**X**	**X**
GVNS	**GVNS_AGMM1**	**X**					**X**	**X**	**X**
	GVNS_AGMM2		**X**				**X**	**X**	**X**
	GVNS_AGMM3			**X**			**X**	**X**	**X**
	GVNS_AGMM4				**X**		**X**	**X**	**X**
	GVNS_AGMM5					**X**	**X**	**X**	**X**
RVNS	**RVNS_AGMM1**	**X**					**X**	**X**	**X**
	RVNS_AGMM2		**X**				**X**	**X**	**X**
	RVNS_AGMM3			**X**			**X**	**X**	**X**
	RVNS_AGMM4				**X**		**X**	**X**	**X**
	RVNS_AGMM5					**X**	**X**	**X**	**X**
SVNS	**SVNS_AGMM1**	**X**					**X**	**X**	**X**
	SVNS_AGMM2		**X**				**X**	**X**	**X**
	SVNS_AGMM3			**X**			**X**	**X**	**X**
	SVNS_AGMM4				**X**		**X**	**X**	**X**
	SVNS_AGMM5					**X**	**X**	**X**	**X**

A Tabela 4.7 apresenta os algoritmos VNS Padrão e suas variações (GVNS, RVNS E SVNS considerando somente a AGM construída utilizando o *Kruskal* padrão com as combinações dos procedimentos de construção e os de busca local, com suas respectivas siglas, implementados neste trabalho.

Tabela 4.7 - Siglas dos algoritmos utilizando a AGM construída com *Kruskal* padrão

ALGORITMO	**SIGLA**	**CONSTRUTORES**					**BUSCA LOCAL**		
		C-1	**C-2**	**C-3**	**C-4**	**C-5**	**BL-1**	**BL-4**	**BL-5**
VNS Padrão	**VNS_AGM1**	**X**					**X**	**X**	**X**
	VNS_AGM2		**X**				**X**	**X**	**X**
	VNS_AGM3			**X**			**X**	**X**	**X**
	VNS_AGM4				**X**		**X**	**X**	**X**
	VNS_AGM5					**X**	**X**	**X**	**X**
GVNS	**GVNS_AGM1**	**X**					**X**	**X**	**X**
	GVNS_AGM2		**X**				**X**	**X**	**X**
	GVNS_AGM3			**X**			**X**	**X**	**X**
	GVNS_AGM4				**X**		**X**	**X**	**X**
	GVNS_AGM5					**X**	**X**	**X**	**X**
RVNS	**RVNS_AGM1**	**X**					**X**	**X**	**X**
	RVNS_AGM2		**X**				**X**	**X**	**X**
	RVNS_AGM3			**X**			**X**	**X**	**X**

	RVNS_AGM4				**X**		**X**	**X**	**X**
	RVNS_AGM5					**X**	**X**	**X**	**X**
SVNS	**SVNS_AGM1**	**X**					**X**	**X**	**X**
	SVNS_AGM2		**X**				**X**	**X**	**X**
	SVNS_AGM3			**X**			**X**	**X**	**X**
	SVNS_AGM4				**X**		**X**	**X**	**X**
	SVNS_AGM5					**X**	**X**	**X**	**X**

Após a realização de um conjunto de testes preliminares com 24 instâncias do Censo Demográfico 2010 com base nos parâmetros descritos para cada grupo de algoritmos, foram estabelecidas as seguintes configurações para o Algoritmo GRASP: determinou-se critério de parada igual a 100 iterações, o alfa ($\boldsymbol{\alpha}$) igual a 10 (dez) pois contém o número máximo de arestas com os maiores valores de custos gerados nas soluções iniciais (procedimentos construtores), mas durante todos os experimentos realizados neste trabalho o valor de alfa não passou de 3 (três), o $\boldsymbol{k}$ igual a 3 e a variável fator de ajuste igual a 25% (para o cálculo da capacidade mínima por *cluster*).

4.4 EXPERIMENTOS REALIZADOS

Os experimentos realizados neste trabalho têm como objetivo, através de execuções sistemáticas e análises probabilísticas, demonstrar mesmo que de forma empírica, a eficiência e a robustez dos algoritmos propostos no que se refere à qualidade das soluções obtidas e à obtenção de soluções válidas (aquelas cujas restrições do problema abordado são atendidas).

Um primeiro experimento considerou 100 execuções de cada algoritmo apresentado na Seção 4.4.1 em cada uma das instâncias apresentadas na Tabela 4.1, para obtenção de três *clusters*. A partir dos resultados obtidos, são apresentadas análises avaliando os algoritmos em relação à obtenção de soluções classificadas como soluções de excelente qualidade, soluções razoáveis e soluções de qualidade péssima. Além disso, foram avaliados os tempos de execução dos algoritmos que utilizam apenas o Grafo Original; a AGM construída com *Kruskal* modificado e AGM construída com *Kruskal* padrão, as versões de procedimentos construtores, bem como os algoritmos GRASP e VNS.

4.4.1 EXPERIMENTOS COM ALGORITMOS PROPOSTOS

Com base na Apêndice A, que apresenta os resultados computacionais obtidos na execução dos algoritmos propostos neste trabalho para cada uma das 64 instâncias utilizadas, a presente seção trata de mostrar os gráficos construídos com base naquelas tabelas a partir de cem execuções das melhores soluções obtidas, o tempo de execução (em segundos) e o *Gap*. O *Gap* (Equação 18) que representa os valores de desvio percentual em relação à melhor

solução (menor valor de função objetivo) encontrada para uma dada instância no experimento:

$$Gap = 100 * \frac{solução - solução_{best}}{solução_{best}} \quad (18)$$

sendo:

- $solução$: valor da função objetivo do algoritmo corrente de uma instância em determinado instante;
- $solução_{best}$: menor (melhor) valor conhecido da função objetivo para um dado experimento;

São consideradas soluções válidas, ou seja, aquelas cujas restrições de conexidade e de capacidade mínima por *cluster* forem atendidas.

Por tratar-se de um problema de minimização, a melhor solução (*Soluçãobest)* é justamente aquela que apresenta o menor valor da função objetivo dentre as 120 (Cento e Vinte) versões associadas ao GRASP padrão, ao GRASP com *Path-Relinking*, ao GRASP com VND e GRASP com RVND, além do VNS padrão e suas variações (GVNS, RVNS e SVNS) . Nas citadas tabelas, as soluções cujo valor do *Gap* for igual a 0% estão com a célula na cor cinza, pois correspondem aos melhores resultados. E as soluções cujo valor do *Gap* é superior a 10% estão assinaladas com a cor preta e com a formatação em negrito, pois se referem aos piores resultados.

As Figuras 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7 e 4.8 apresentam gráficos com os quantitativos das soluções obtidas por algoritmo, classificadas conforme a sua qualidade, sejam elas: Melhores (com *Gap* igual a 0%), Razoáveis (com *Gap* maior do que 0% e menor ou igual a 10%) e Péssimas (*Gap* acima de 10%). Uma vez que as Tabelas presentes no Apêndice A apresentaram os melhores resultados produzidos pelos algoritmos para cada instância, o quantitativo máximo apresentado no eixo *x* corresponde ao total de instâncias consideradas nos experimentos, ou seja, sessenta e quatro.

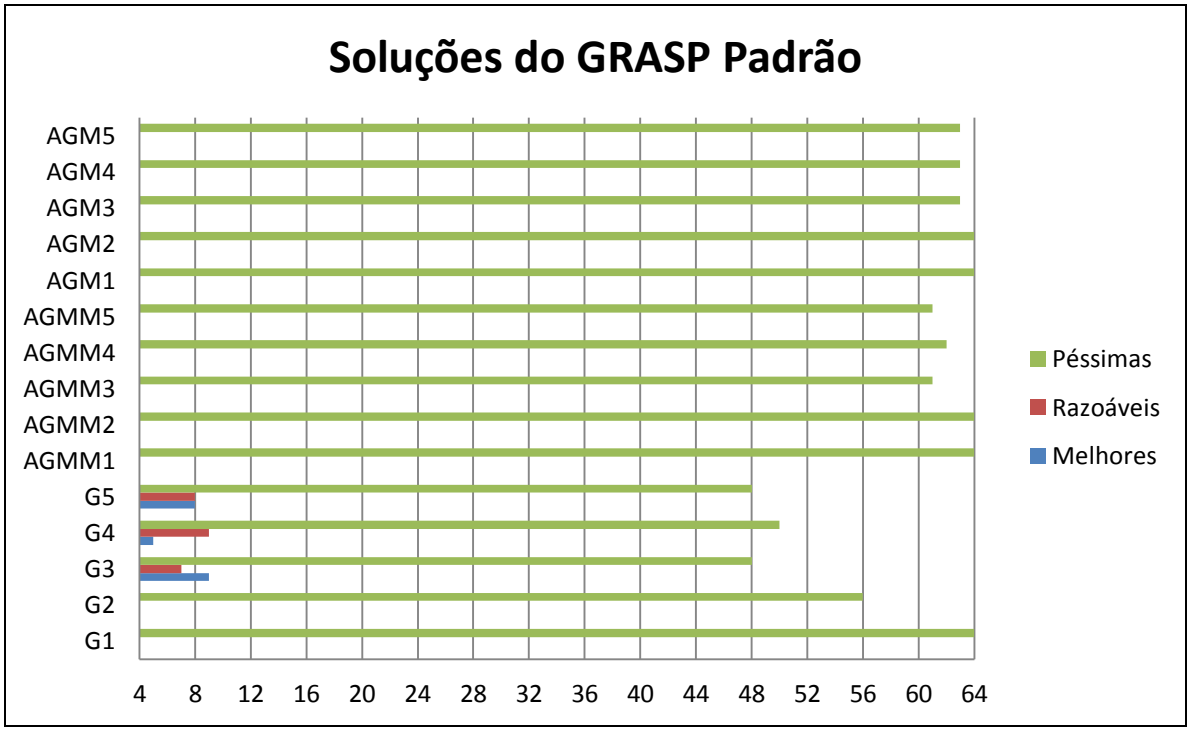


Figura 4.1 - Soluções GRASP Padrão

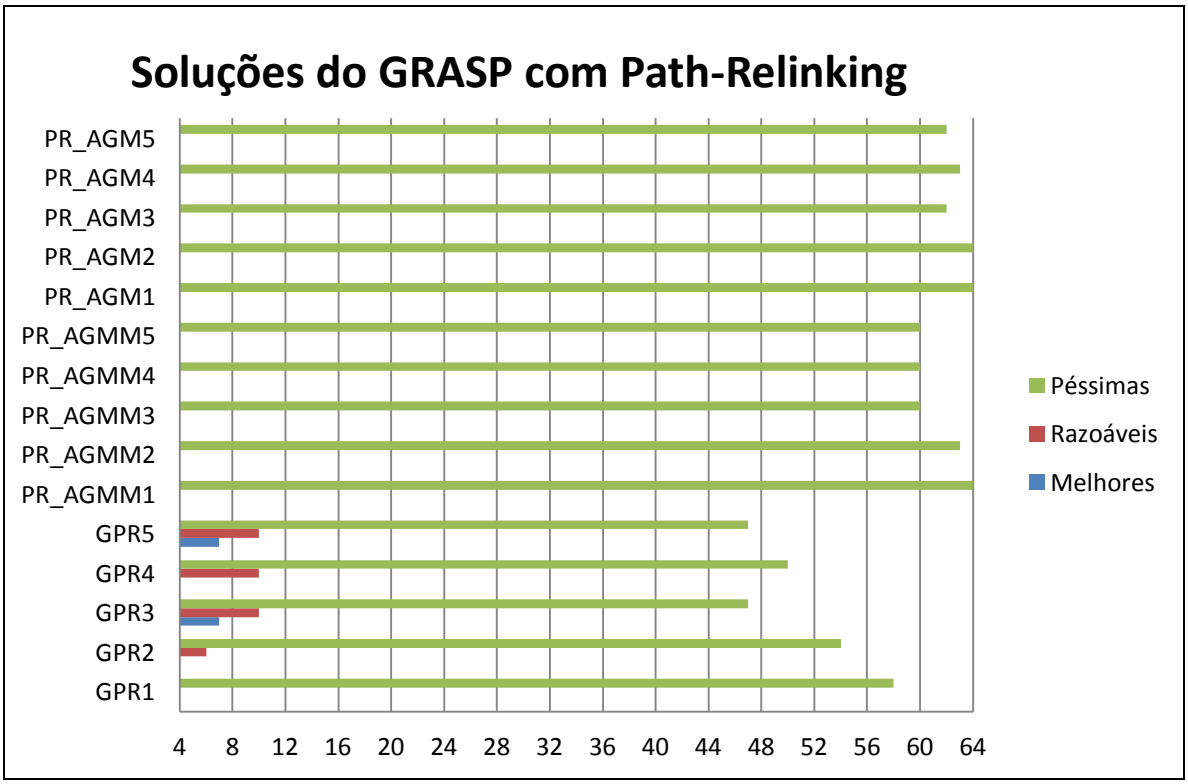


Figura 4.2 - Soluções GRASP com *Path-Relinking*

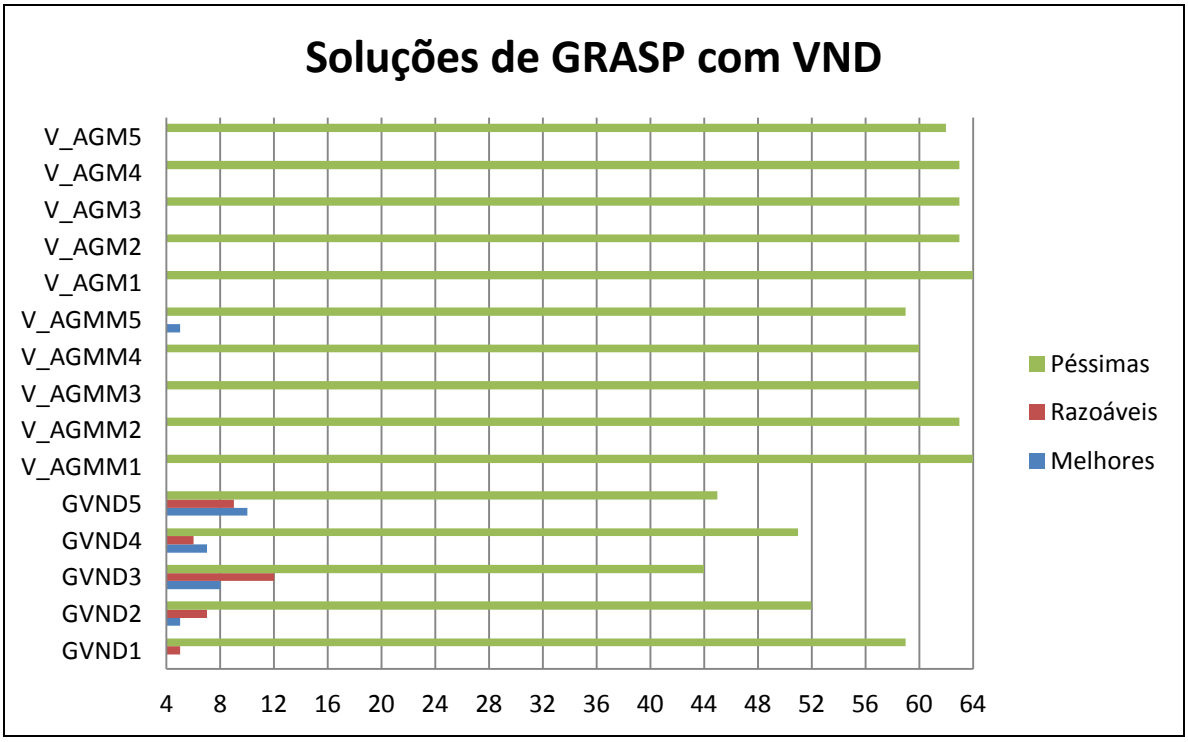


Figura 4.3 - Soluções GRASP com VND

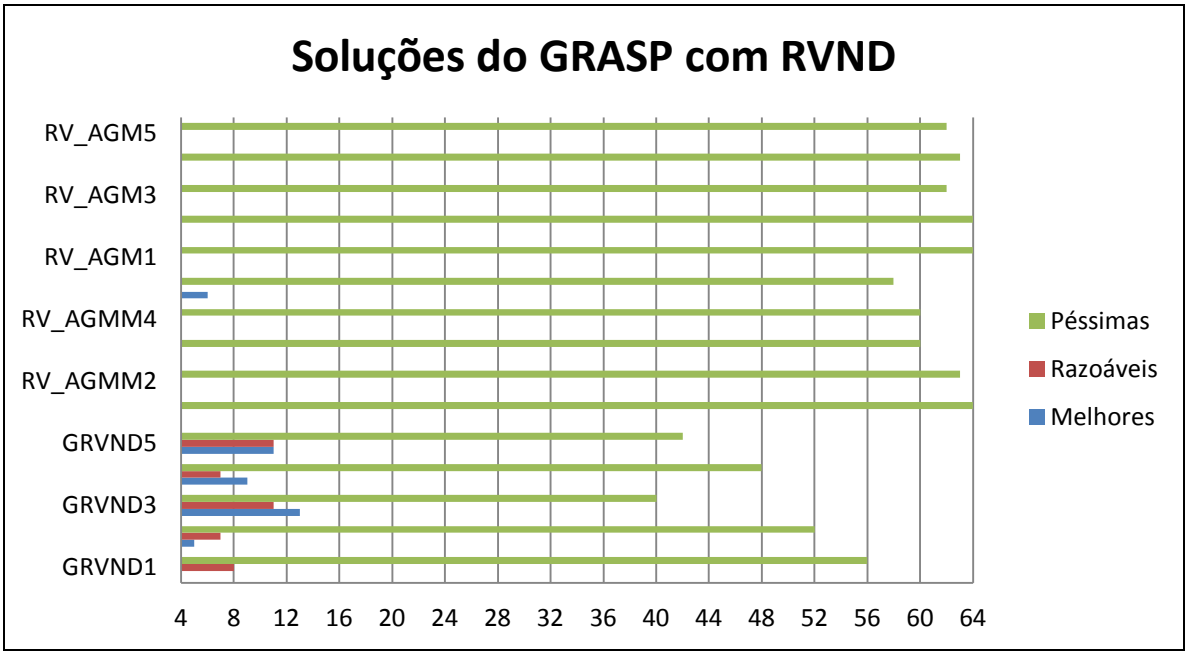


Figura 4.4 - Soluções GRASP com RVND

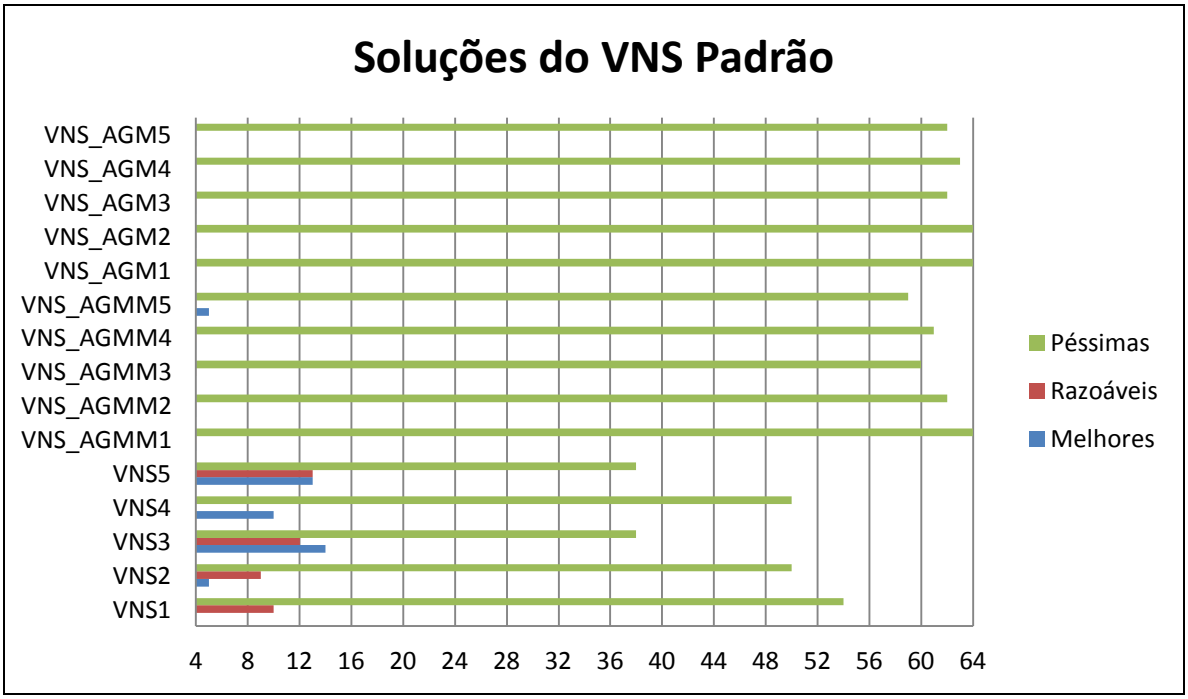


Figura 4.5 - Soluções VNS Padrão

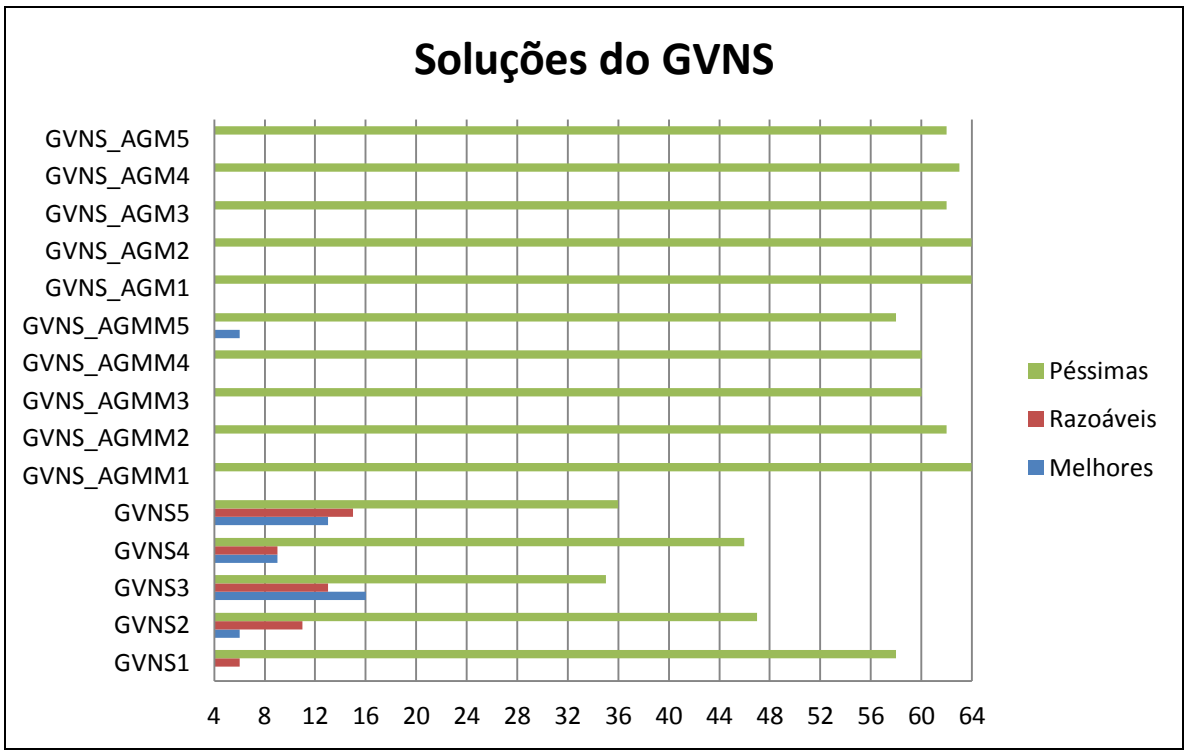


Figura 4.6 - Soluções GVNS

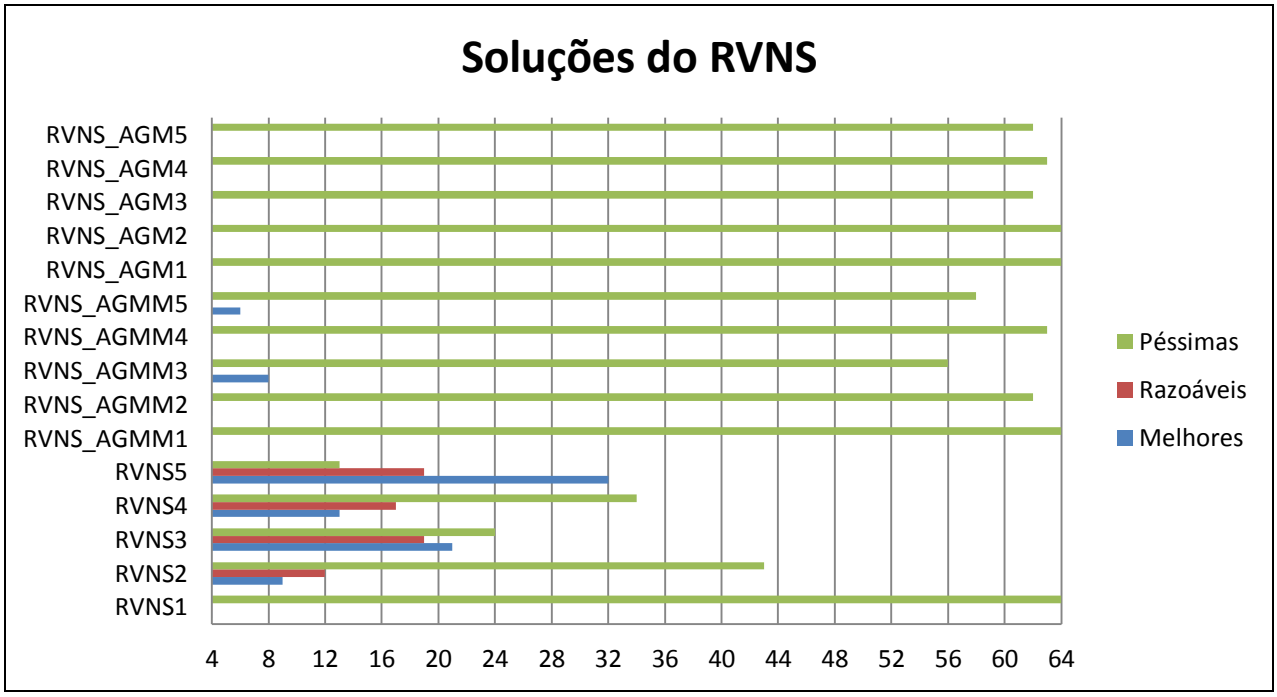


Figura 4.7 - Soluções RVNS

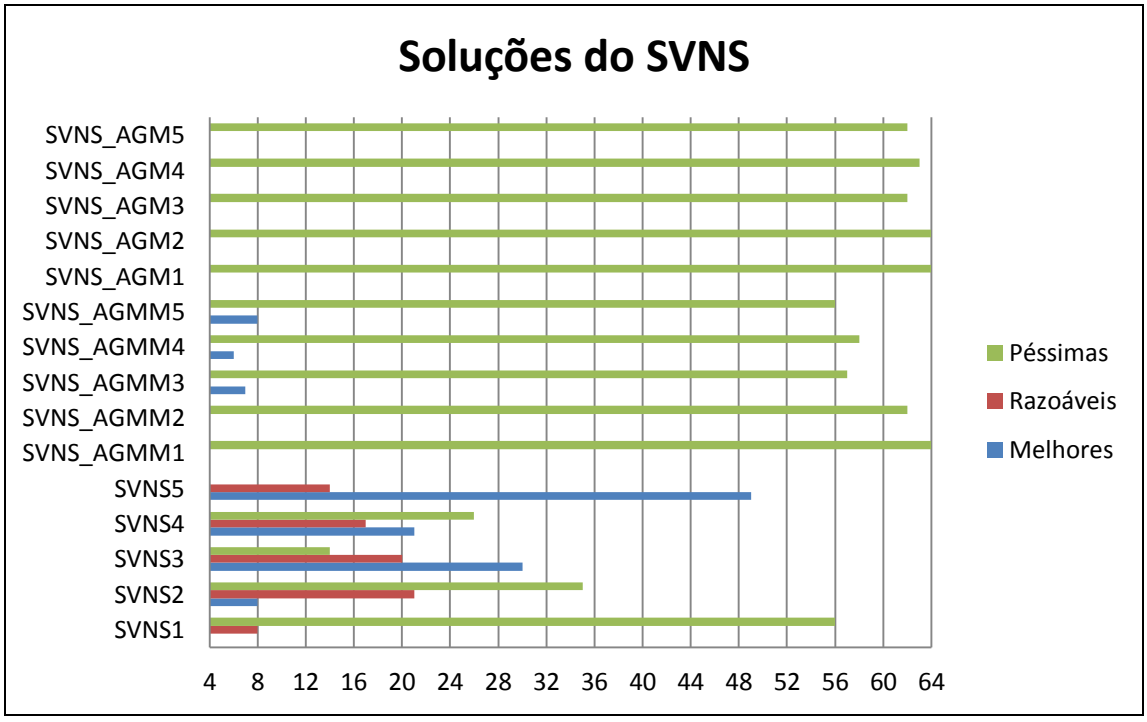


Figura 4.8 - Soluções SVNS

Agora, uma vez considerados os dados das tabelas e as informações sumarizadas nas figuras anteriores, faz-se necessária a realização de uma análise comparativa entre os melhores algoritmos GRASP Padrão, GRASP com *Path-Relinking*, GRASP com VND, GRASP com RVND, VNS Padrão, GVNS, RVNS e SVNS para cada uma das 64 (sessenta e quatro) instâncias utilizadas, o que pode ser visualizado na Figura 4.9:

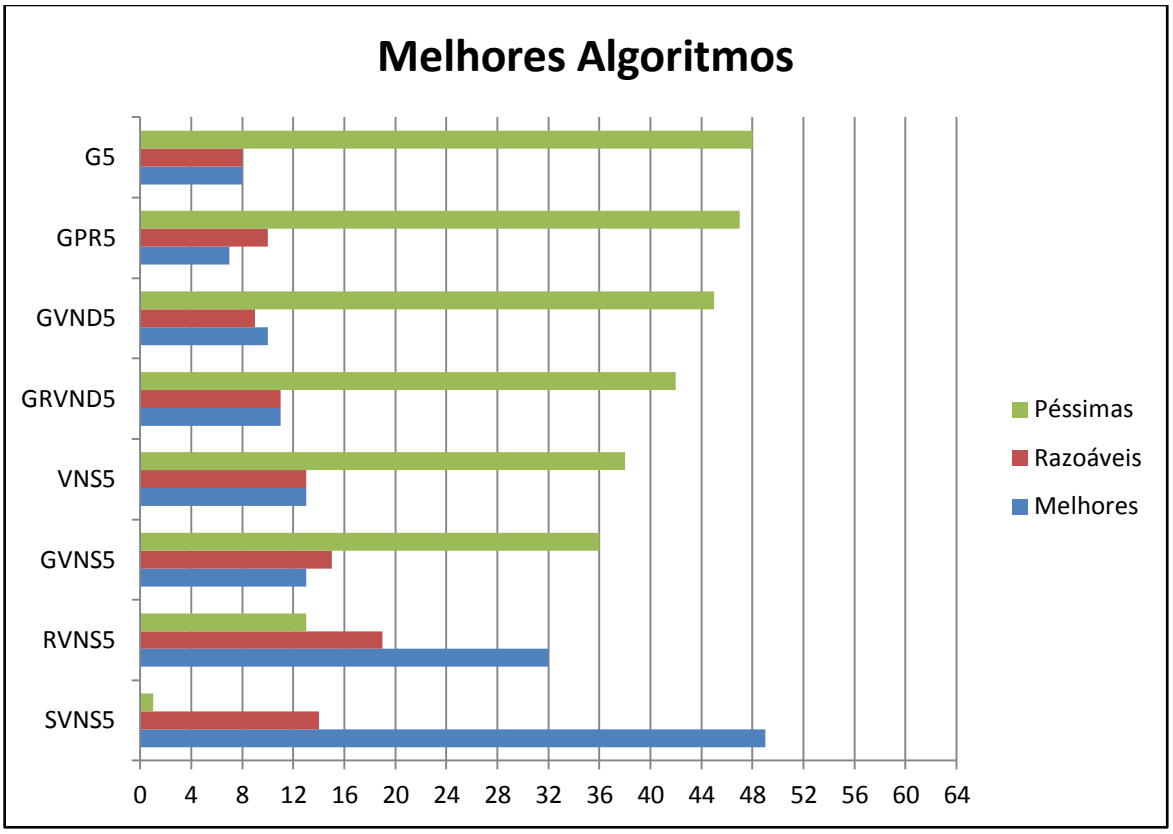


Figura 4.9 – Melhor Solução por Algoritmo

Uma primeira análise da figura 4.9 indica que, dentre todos os algoritmos, o SVNS5 apresentou um melhor desempenho no que concerne à qualidade das soluções produzidas. Mais especificamente, apresenta o maior número de soluções melhores e o menor número de soluções péssimas. E os piores resultados são apresentados pelo GRASP Padrão. Mas especificamente o GRASP Padrão apresenta a maior números soluções péssimas igual a 48, seguido do GRASP com *Path-Relinking* com 47, o GRASP com VND com 45, o GRASP com RVND com 42, o VNS Padrão com 38, o GVNS com 36, o RVNS com 13 e por último o SVNS com o menor número igual a 1

A Figura 4.10, apresenta a média das versões melhores, razoáveis e péssimas do algoritmo SVNS por categoria enumeradas como: Grafo Original, AGM Modificada (gerada com *Kruskal* modificado por pesos aleatórios) e AGM Padrão (gerada com *Kruskal* padrão).

Foram testadas 5 (cinco) versões de cada de cada categoria para cada uma das 64 instâncias tratada nesse trabalho, perfazendo um total de 320 (trezentos e vinte) resultados diferentes.

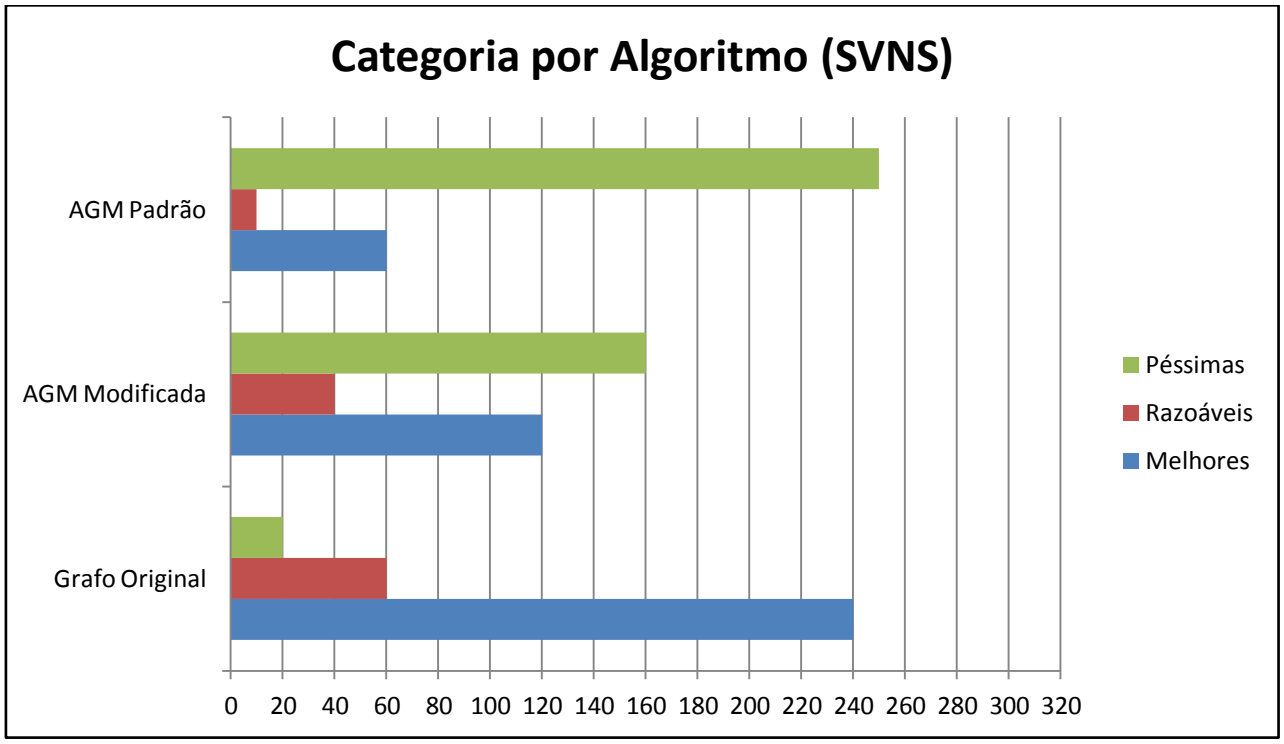


Figura 4.10 – Melhor Categoria por Algoritmo (SVNS)

Na análise da figura 4.10 percebe-se que, o Grafo Original apresente um resultado superior em relação ao número de soluções melhores são 240 (duzentos e quarenta) em comparação a AGM Modificada que alcança somente metade deste resultado, foram 120 (cento e vinte) melhores e a AGM Padrão que só consegue alcançar 60 (sessenta) destas. O número de soluções péssimas do Grafo Original também mostra que sua performance computacional é melhor em relação as outras categorias, pois só atingi 20 soluções péssimas, enquanto a AGM Modificada chega ao número de 160 (cento e sessenta) e a AGM Padrão se encontra com 250 (duzentos e cinquenta) soluções péssimas.

A Figura 4.11 ilustra um comparativo entre o algoritmo SVNS5 (Grafo Original), SVNS_AGMM5 (AGM com *Kruskal* Modificado) e SVNS__AGM5 (AGM com *Kruskal* Padrão) e o *software* SKATER:

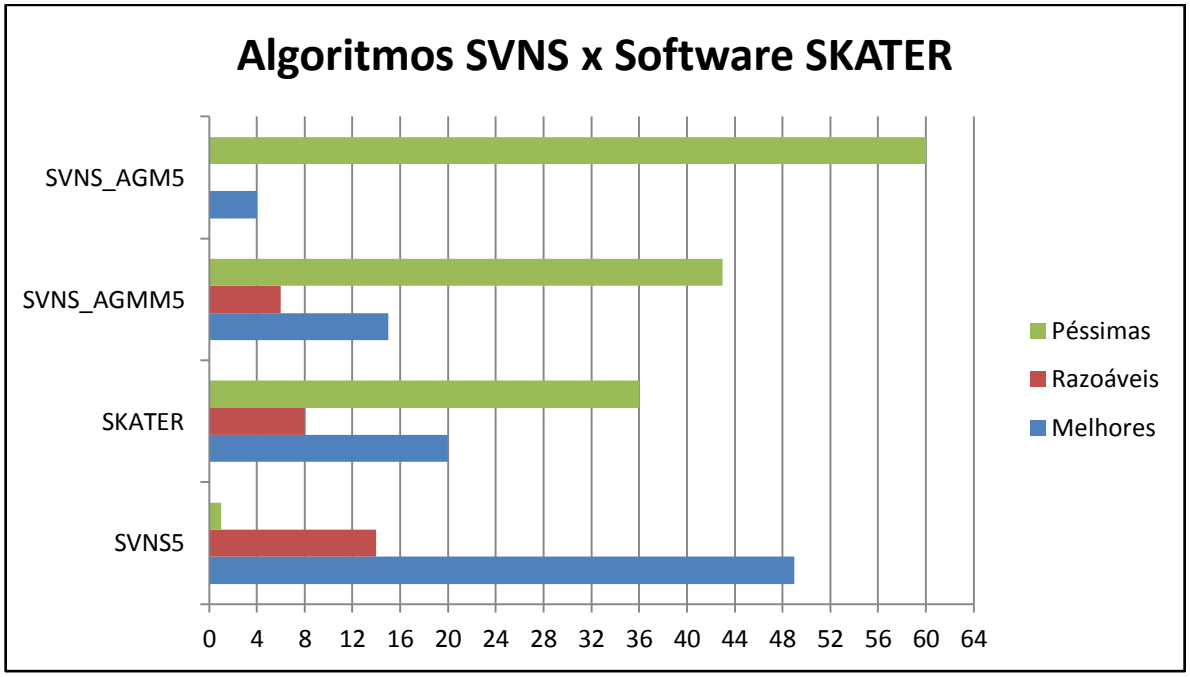


Figura 4.11 – Comparativo algoritmos SVNS e o software SKATER

Na Figura 4.11, foram considerando-se 100 execuções, para cada uma das 64 (sessenta e quatro) instâncias, para obtenção de 3 *clusters* para a obtenção de soluções válidas, desde que sejam garantidas as restrições de capacidade e conexidade. O algoritmo SVNS5 (Grafo Original) apresentou o melhor resultado, seguido do software *Skater*. O *Skater* consegui 20 (vinte) soluções melhores, 8 (oito) soluções razoáveis e 36 (trinta e seis) soluções péssimas. E com a pior performance aparece o algoritmo SVNS com AGM (construída com *Kruskal* padrão).

4.5 AVALIAÇÃO PROBABILÍSTICA DOS RESULTADOS

Para se verificar o desempenho dos algoritmos também foi realizada uma análise de probabilidade empírica [AIEX ET AL 2007]. Esse tipo de abordagem consiste em tomar os tempos de processamento que cada algoritmo requer para atingir um valor alvo no que se refere à função objetivo. Neste sentido, foram estabelecidos neste trabalho dois alvos de avaliação obtidos em testes anteriores, a saber: um alvo considerado fácil, dado pelo pior valor da função objetivo de uma solução válida (em relação ao algoritmo) e um alvo considerado difícil, dado pelo menor (melhor) valor da função objetivo. Dessa forma, no instante em que cada algoritmo encontra uma solução menor ou igual ao alvo fixado, o tempo é registrado e uma nova execução do algoritmo é iniciada. Em outras palavras, um novo critério de parada é inserido para cada algoritmo, que consiste na obtenção de uma solução

cujo valor atinja ou ultrapasse o alvo, dentro do tempo limite pré-estabelecido em 3600 segundos.

Cada versão de algoritmo foi executada 100 vezes para cada instância, usando os mesmos parâmetros dos experimentos anteriores. Os tempos tomados e dispostos em ordem crescente numa lista *L*. A cada tempo de CPU *t* obtido, foi associada a probabilidade empírica do tempo sendo $p_i = \left(i - \frac{1}{2}\right)/100$, sendo i a ordem que *t* aparece na lista ordenada *L*. A plotagem é feita então tomando cada ponto (t_i, p_i) para $i = 1, 2, 3, \ldots 100$, num gráfico, que estabelece uma distribuição de probabilidade empírica do tempo que cada algoritmo consome para alcançar o alvo pré-estabelecido [AIEX ET AL 2007].

Para esta análise foram utilizadas as instâncias vinte e oito, trinta e nove e instância sessenta, onde são apresentados os gráficos com a probabilidade empírica ao longo do tempo em segundos. Nota-se que cada curva do gráfico representa a convergência empírica de cada algoritmo, e que quanto mais à esquerda estiver a curva, melhor (mais rápido) será a convergência do algoritmo (mais eficiente), ou seja, o tempo que o algoritmo consome para atingir o alvo.

Nas Figuras 4.12, 4.13, 4.14, 4.15, 4.16 e 4.17 são apresentadas as distribuições de probabilidades empíricas ao longo do tempo de execução através dos gráficos *TTTPlots*. Com o objetivo de avaliar os algoritmos utilizou-se no experimentos instâncias de tamanhos diferentes tanto na quantidade de vértices (nº de objetos) quanto no número de arestas. Padronizou-se que as instâncias com número de até 100 (cem) vértices independente do número de arestas seria denominada, neste trabalho, como instância de pequeno porte, por exemplo instância 28. E que as instâncias com número acima de 100 (cem) e menor que 400 vértices independente do número de arestas seria denominado de instância de médio porte, como a instância 39. E que a instância 60, seria descrita como de grande porte, uma vez que tem o número de vértices acima de 400 sem padronizar o número de arestas.

No entanto, percebe-se que algumas versões de algoritmos não conseguiram atingir o alvo fácil (valor da maior função objetivo) e o alvo difícil (valor da menor função objetivo) em nenhuma das 100 execuções e nem em tempo requerido durante o experimento de (3600 segundos), por isso não aparecem nas respectivas Figuras.

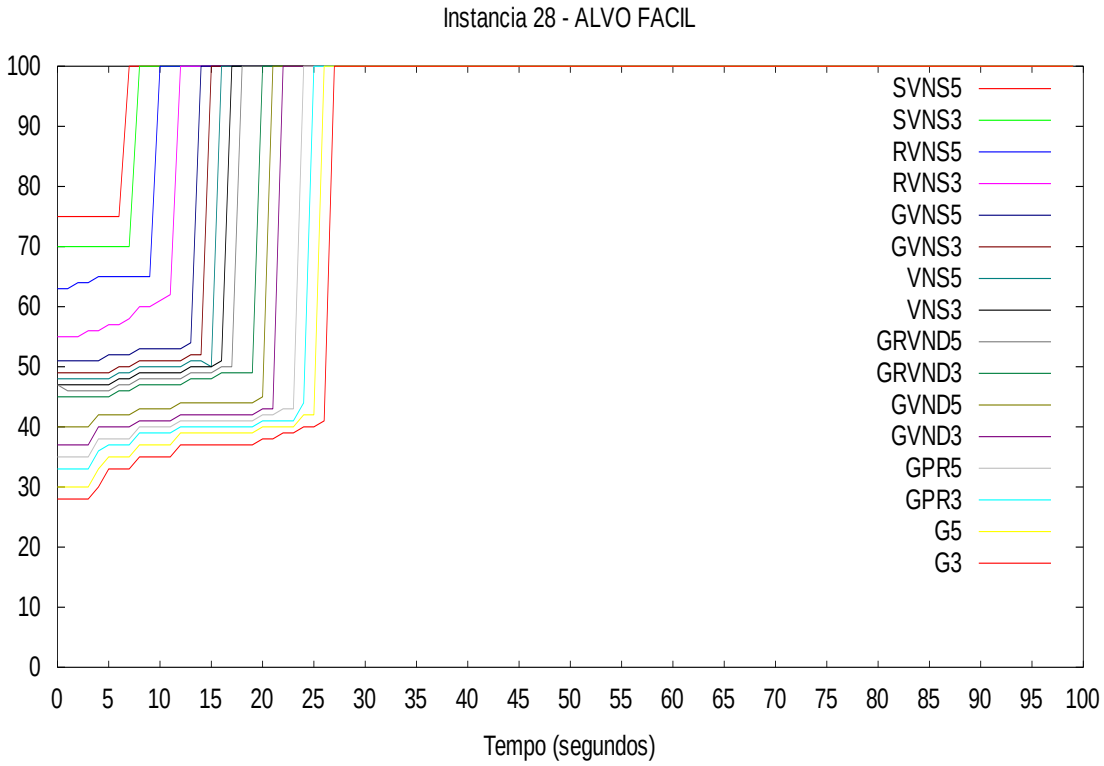


Figura 4.12 – Análise probabilística da instância 28 – alvo fácil

Pelos resultados ilustrados na Figura 4.12 verifica-se em termos de qualidade da solução gerada, que a versão o algoritmo SVNS5 (utiliza o procedimento construtor C-5 e as buscas locais BL-01, BL-04 e BL-05) apresentou o melhores resultado seguidos dos algoritmos SVNS3, RVNS5 e RVNS3 que obtiveram resultados muito próximos entre si. Para os experimentos que utilizam o Construtor 1 não foi possível alcançar o alvo para nenhuma das versões de algoritmos geradas. E as análises probabilísticas com as versões de Construtor 2 e 4 apresentaram resultados insatisfatórios, ou seja, a maioria das versões desses algoritmos não atingiu o alvo em nenhuma das 100 execuções ou dentro do limite de tempo estabelecido.

E o pior resultado foi constatado o algoritmo G3 (GRASP incluindo o Construtor 3 e buscas locais BL-01, BL-04 e BL-05) que só consegue atingir o alvo em 27 segundos.

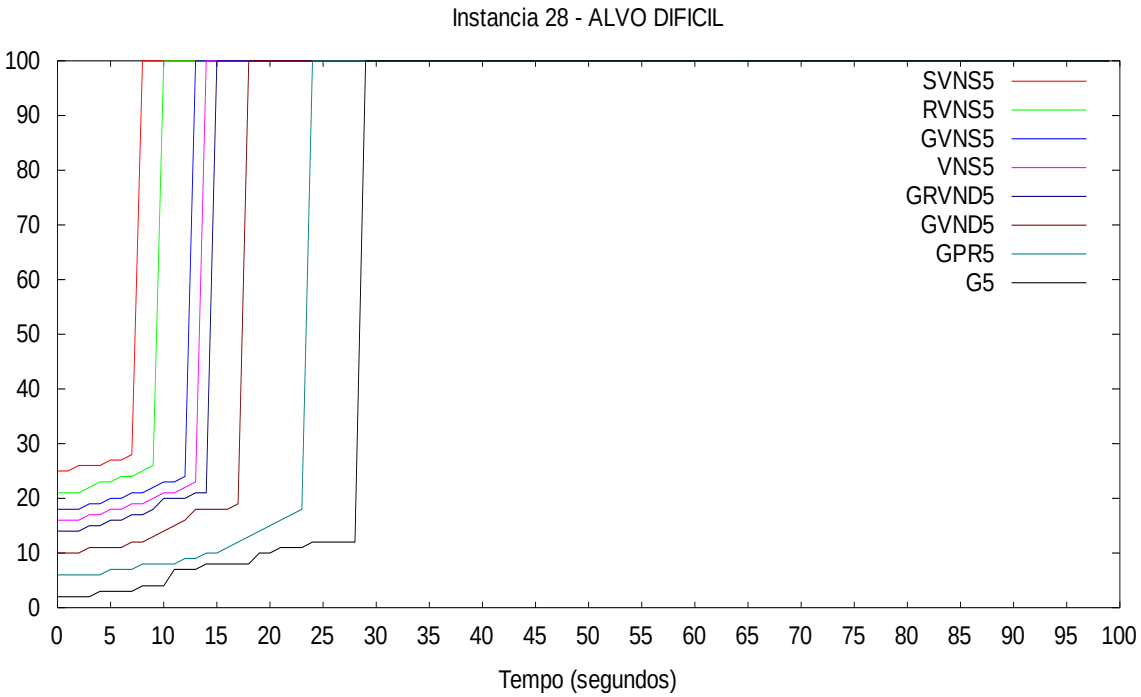


Figura 4.13 – Análise probabilística da instância 28 – alvo difícil

Com base na Figura 4.13, verificou-se que mais uma vez, em termos de qualidade da solução gerada, a versão do SVNS, o SVNS5 apresentou os melhores resultados. O desempenho das demais versões dos algoritmos propostos, por sua vez, mantiveram a mesma ordem dos testes com a instância 28 alvo fácil.

O SVNS5 alcança o alvo fácil em 7 segundos, já o pior resultado representado pelo algoritmo G3 atingi o alvo em 27 segundos.

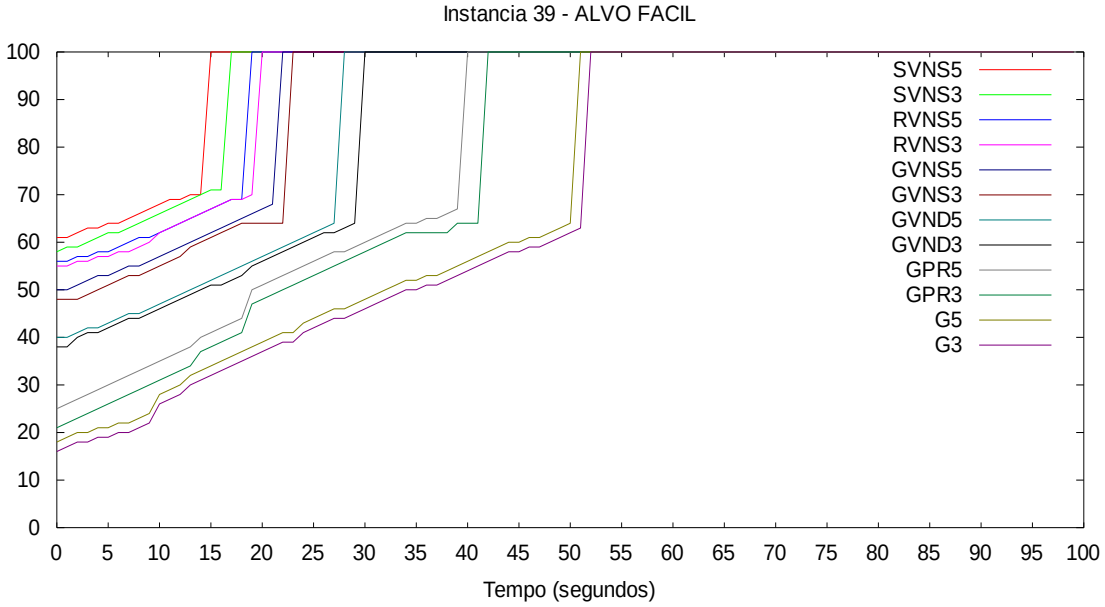


Figura 4.14 – Análise probabilística da instância 39 – alvo fácil

Neste caso, da Figura 4.14, houve uma competitividade maior entre as versões SVNS5 e SVNS3. Isso nos mostra que as metaheurísticas ainda são sensíveis aos parâmetros de entrada do problema. Mas de forma geral, percebe-se que não ocorre nenhuma diferença nesta bateria de testes quando comparada com a bateria de testes anteriores. Ou seja, o SVNS5 foi novamente o melhor algoritmo e os piores desempenhos ficaram com as versões do GRASP G5 e G3.

É constatado ainda que entre cada par de versões que utilizaram os mesmos procedimentos de busca local (BL-01, BL-04, BL-05) o construtivo que mais se destacou para o conjunto de testes foi o Construtor 05 (C-05) seguido do Construtor 03 (C-03). Pois entre as versões SVNS5, que utiliza o C-05 se mostrou mais eficiente que o SVNS3, que utilizar o C-03. O mesmo acontece com as versões RVNS5 e RVNS3, GVNS5 e GVNS3, GVND5 e GVND3, GPR5 e GPR3, e por último G5 e G3. Destaca-se que os algoritmo GVNS5, VNS5 e GRVND5 apresentaram resultados similares, por isso optou-se por visualizar no gráfico somente o resultado do GVND5. Os algoritmos GVNS3, VNS3 e GRVND3 também apresentaram resultados semelhantes, e também optou-se por mostrar no gráfico somente o resultado do algoritmo GVND3.

Mais uma vez, os algoritmos o Construtor 1 não atingi o alvo em nenhuma das versões dos algoritmos gerados. E as versões do construtor 2 e do 4 apresentaram baixo desempenho em que a maioria dos algoritmos não conseguiu atingir o alvo, em nenhuma das 100 execuções e nem dentro do limite estabelecido.

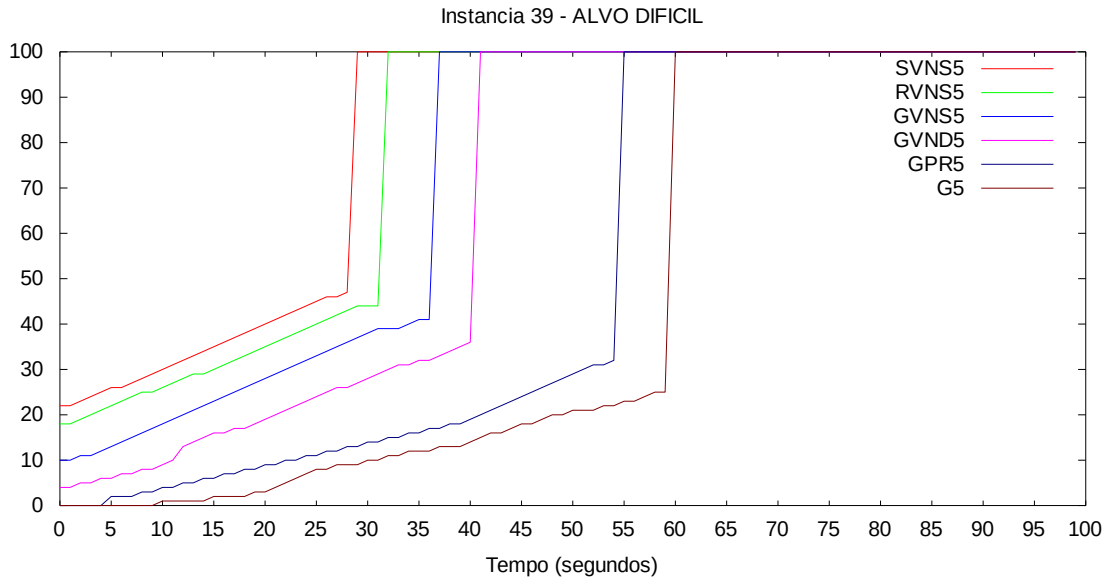


Figura 4.15 – Análise probabilística da instância 39 – alvo difícil

Pela Figura 4.15 é constado que a versão do SVNS o SVNS5 apresenta o melhor resultado, assim como, nos testes anteriores. O SVNS5 atingi o alvo em 28 segundos, tem-se na sequencia como os melhores resultados RVNS5, GVNS5, GVND5, GPR5 e G5. Informa-se que o GVND5 representa também os resultados das versões VNS5 e GRVND5 que apresentam resultados similares.

E o pior resultado continua com o GRASP utilizando o Construtor 05 e Buscas Locais BL-01, BL-04, BL-05, ou seja, o G5 que converge ao alvo com 59 segundos, sendo 30 segundos a mais que o melhor resultado.

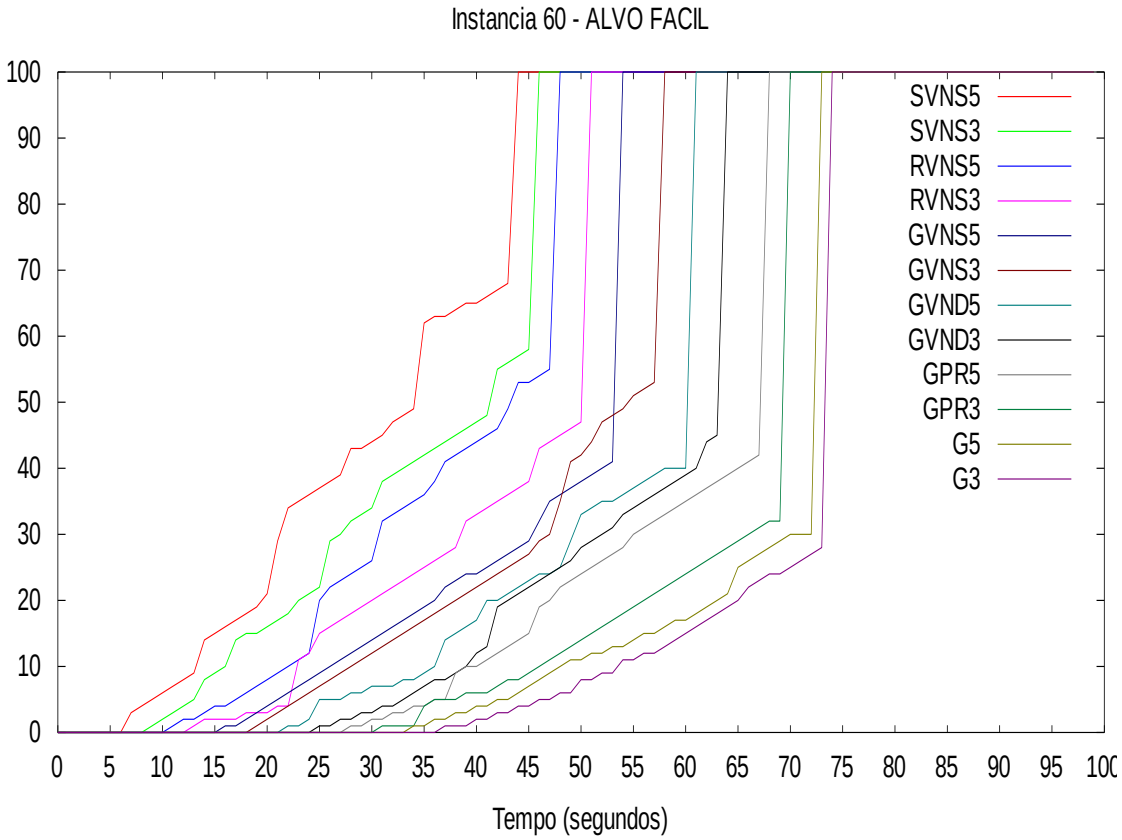


Figura 4.16 – Análise probabilística da instância 60 – alvo fácil

Com base na Figura 4.16 temos a análise probabilística da instância 60 (considerada de grande porte) para o alvo fácil, observa-se que os algoritmos apresentados apresentam mudanças significativas em relação aos tempos computacionais exigidos quando se compara com os resultados das instâncias de pequeno e médio porte. O algoritmo SVNS5 atingir o alvo em 42 segundos na instância 60, ou seja leva muito mais tempo do que o utilizado na instância 39 que levou 14 segundos e na instância 28 chega ao alvo em 7 segundos.

O Construtor 1 continua sem atingir o alvo em nenhuma das versões dos algoritmos gerados. E as versões do construtor 2 e do 4 continuaram apresentando baixo desempenho

pois a maioria de seus algoritmos não consegue atingir o alvo, em nenhuma das 100 execuções e nem dentro do limite estabelecido.

Já os piores resultados foi representado pelo GRASP através do algoritmo G5 e G3 e são também os que consomem mais tempo de execução para atingir o alvo, o primeiro leva 72 segundos e o segundo 74s.

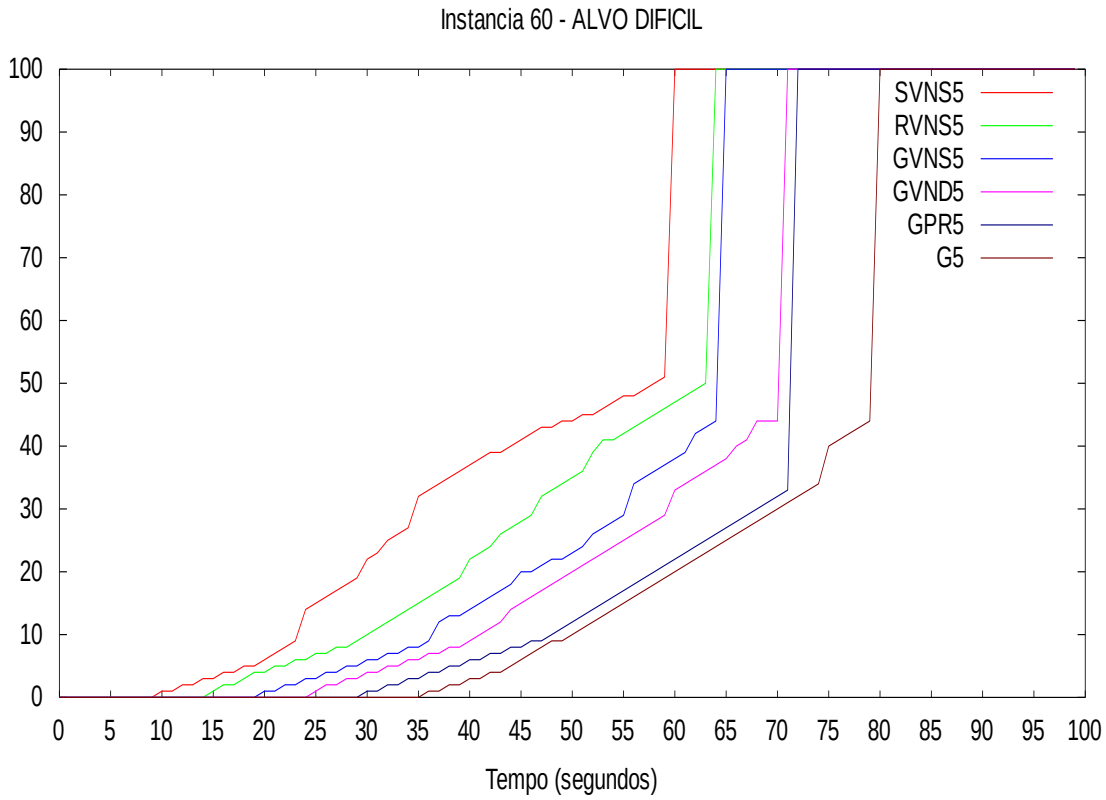


Figura 4.17 – Análise probabilística da instância 60 – alvo difícil

Com base na Figura 4.17 temos a análise probabilística da instância 60 (considerada de grande porte) para o alvo difícil. Optou-se por mostrar somente os melhores resultados obtidos. O pior algoritmo foi o G5 que levou aproximadamente 80 segundos, enquanto o melhor resultado o algoritmo SVNS5 atingi 100% de probabilidade em 59 segundos. O segundo melhor resultado fica com o algoritmo RVNS5 que consegui convergir em 63 segundos. As versões VNS5 e GRVND5 apresentam resultados iguais ao GVND5 por isso optou-se por demonstrar somente o resultado do GVND5 que leva 64 segundos para atingir o alvo.

Pode-se concluir que em todas as instâncias seja ele de pequeno porte (instância 28), de médio porte (instância 53) ou de grande porte (instância 60), os resultados para atingir o alvo difícil se repetem onde o SVNS5 apresenta o melhor resultado. O algoritmo G3 que representa GRASP Padrão com Construtor-03 e Busca Locais BL-01, BL-04 e BL-05 se

mostrou como o pior resultado em todos os experimentos no que se refere ao alcance do alvo fácil e o algoritmo G5 que representa o GRASP Padrão com Construtor-05 e Busca Locais BL-01, BL-04 e BL-05 se mostrou como o pior resultado em todos os experimentos no que se refere ao alcance do alvo difícil.

4.6 ANALISE DE ROBUSTEZ DOS RESULTADOS

De forma a avaliar os algoritmos no que concerne à robustez, foi aplicada uma análise estatística básica. Mais especificamente, os *k* melhores algoritmos foram executados 200 vezes, considerando cada uma das instâncias. A partir dos resultados obtidos com as 200 execuções calculou-se a solução média dos melhores algoritmos, o desvio padrão e os coeficientes de variação das instâncias 28 (pequeno porte), 39 (médio porte) e 60 (grande porte). A Tabela 4.8 a seguir trazem estas medidas e a melhor solução (solução *best*) obtida a partir das Tabelas presentes no Apêndice A, para uma instância de pequeno, médio e grande porte e a Tabela 4.9 traz o melhor tempo de execução de cada algoritmo e o tempo médio de cada execução de algoritmo em segundos.

Tabela 4.8 Resultados de Análise de Robustez para a instância 28 (pequeno porte), 39 (médio porte) e 60 grande (porte)

INSTÂNCIA	**28**		**39**		**60**	
ALGORITMO	**SVNS5**	**RVNS**	**SVNS5**	**RVNS**	**SVNS5**	**RVNS**
SOLUÇÃO BEST	**25,89**	**25,89**	**73,06**	**73,06**	**108,73**	**108,73**
Solução Média	25,89	25,89	70,85	72,19	110,02	114,21
Desvio-Padrão	0,00	0,00	0,94	1,28	3,63	5,09
Coeficiente-Variação	0,00	0,00	0,013	0,017	0,032	0,044

Para a instância 28 (pequeno porte) e a instância 39 (médio porte) os algoritmos SVNS5 e RVNS5 se mostraram robustos, visto que para a primeira instância não apresentou nenhuma variação nos valores de desvio-padrão e, consequentemente, nos valores de coeficiente de variação. Já a instância 39 apresenta valores de desvio-padrão e coeficiente de variação relativamente baixos. A instância 60 apresenta alterações nos valores de desvio-

padrão, mas mantém a robustez dos algoritmos SVNS5 e RVNS5 com baixa variação no seu coeficiente de variação.

Tabela 4.9 Medidas dos tempo de processamento dos algoritmos (segundos)

INSTÂNCIA	**28**		**39**		**60**	
ALGORITMO	**SVNS5**	**RVNS5**	**SVNS5**	**RVNS5**	**SVNS5**	**RVNS5**
TEMPO BEST	**30**	**32**	**2308**	**2317**	**2760**	**2741**
Tempo Médio	48	43	2365	2340	2828	2786

No que diz respeito aos tempos de processamento das instâncias 28 e 39 observa-se um equilíbrio entre os tempos dos dois algoritmos (SVNS5 e o RVNS5) tendo como base seus tempos médios que demonstram valores muitos próximos. Já a instância 60 percebe-se visivelmente um tempo de processamento maior algoritmo SVNS5 em comparação com o RVNS5.

CAPÍTULO 5 – CONCLUSÕES E TRABALHOS FUTUROS

O presente capítulo apresenta as conclusões dos estudos e experimentos realizados a respeito do problema de agrupamento com restrições de capacidade e conexidade para um caso real do Censo Demográfico 2010 do IBGE. Também são descritos os futuros desdobramentos deste trabalho de tese.

5.1 CONCLUSÕES

Neste trabalho foi proposto um estudo das metaheurísticas GRASP e VNS aplicadas ao problema de clusterização com restrições de capacidade e de conexidade. A partir desse estudo foram elaborados métodos construtivos, de buscas locais, de memória adaptativa utilizando a reconexão de caminhos e de busca local exaustiva usando um VND e RVND, aplicado ao Grafo Original e a AGM construída com *Kruskal* padrão e com *Kruskal* modificado.

Neste sentido, foram propostas diferentes versões de heurísticas, a saber: uma versão que difere da tradicional (GRASP da literatura) por utilizar o mecanismo de reconexão de caminhos, uma versão que utiliza uma busca local exaustiva (módulo VND), uma versão que utiliza uma busca local exaustiva randômica (módulo RVND), uma versão que utiliza o VNS padrão, uma versão que utiliza um GVNS (*General Variable Neighborhood Search* ou Busca Geral Com Vizinhança Variável), uma versão que utiliza um RVNS (*Reduced Variable Neighborhood Search* ou Busca Reduzida Em Vizinhança Variável) e uma versão que utiliza um SVNS (*Skewed Variable Neighborhood Search)* ou Busca Inclinada Com Vizinhança Variável.

Os resultados e as análises apresentados no Capítulo 4 indicam que os algoritmos que operam sobre o Grafo Original tiveram considerável ganho de desempenho frente à AGM, s7endo o Grafo Original e a AGM utilizados como "ponto de partida" para aplicação dos procedimento de particionamento, de construção e de busca local. O Capítulo 4 apresenta ainda a AGM Modificada pela utilização de pesos aleatórios para as arestas. Numa primeira análise que trata dos gráficos, ou seja, do quantitativo das soluções obtidas por algoritmo, todos os algoritmos que utilizaram o Grafo Original obtiveram maior número de soluções melhores (*Gap* igual a 0%). Na avaliação probabilística dos resultados somente os algoritmos que utilizaram o Grafo Original conseguem alcançar o Alvo Fácil e Alvo Difícil. Durante a análise de robustez percebe-se que os melhores resultados foram também representados por algoritmos gerados a partir do Grafo Original. Já os algoritmos que utilizam a AGM com

Kruskal padrão tive um rendimento razoável quando compara-se o desempenho da AGM com pesos aleatórios. E na análise probabilística e de robustez a AGM com pesos aleatórios é melhor do que a AGM padrão, porém essas variações da AGM ficaram atrás do resultado do Grafo Original que continua sendo o melhor resultado. A estrutura do Grafo Original tende a ampliar as possibilidades de operação dos procedimentos de construção e busca local no que diz respeito às migrações de vértices, e assim gerar soluções válidas, mas facilmente, o que não pode ser repetido para a AGM construída com o algoritmo de *Kruskal* modificado e nem com a AGM construída com o algoritmo de *Kruskal* padrão.

A maior contribuição deste trabalho foi no que diz respeito à utilização de heurísticas como o GRASP com reconexão de caminhos, com VND e com RVND, além da utilização de VNS padrão e variações (GVNS, RVNS e SVNS), utilizados para a solução do problema de clusterização com restrições de capacidade e de conexidade.

Os resultados apresentados demonstram que as heurísticas que são geradas pela união de procedimentos de construção e seus procedimentos de busca local tende a ser mais complexas e com a utilização de hibridismo melhoram o desempenho do algoritmo que é construído com heurísticas padrão (GRASP Clássico e VNS Puro) sem nenhum tipo de combinação de metaheurísticas. Assim, o pior resultado foi observado a partir do GRASP padrão, seguido pelo GRASP com reconexão de caminho, posterior ao GRASP com VND, seguido do GRASP com RVND, e por fim VNS padrão e suas variações GVNS, RVNS e SVNS com o melhor resultado.

Destaca-se neste trabalho, o algoritmo SVNS5, que se manteve em primeiro lugar nos experimentos realizados através de execuções sistemáticas, tendo como critério de parada o número de iterações. Assim, procurou-se demonstrar sua eficácia no que se refere à qualidade das soluções obtidas e da obtenção de soluções válidas (restrições do problema são atendidas). Esta versão do SVNS5 que utiliza o Construtor 5 que cria uma espécie de super-nó capaz de unir vértices que estejam distantes uns dos outros, mas que sejam conexos e que estejam de acordo com o critério de capacidade pré-estabelecido no problema. Além disso, o SVNS5 demonstra ser um algoritmo eficiente pois converge mais rápido do que todas as outras versões para boas soluções, especialmente no caso de soluções alvo difíceis.

Em uma segunda etapa foi realizada uma comparação entre todas as heurísticas no que ser refere a uma avaliação probabilística dos resultados e obteve-se a seguinte constatação: o SVNS5 obteve o melhor desempenho, permitindo melhor aproveitamento para os melhores valores (alvo difícil) e piores valores (alvo fácil) da função objetivo durante a análise probabilística empírica. Ao realizar os testes envolvendo soluções alvo, o RVNS5, RVNS3,

GVNS5 e GVNS3 conseguiram atingir os maiores números de alvos em tempos menores do que o algoritmo G5 (GRASP com Construtor 5) e G3 (GRASP com Construtor 3)

Percebe-se que os algoritmos que utilizaram o Construtor 1 apresentaram os piores resultados na obtenção de soluções válidas em todos os experimentos realizados. O segundo pior resultado fica com o construtor 2 que consegue melhor o desempenho dos algoritmos que utilizaram o Construtor 1, mas não consegue alcançar os resultados do construtor 4, que surge como um melhoramento do Construtor 2. Já o Construtor 3 consegue alcançar um grande números de soluções válidas, com um grande número de soluções interessantes e um menor número de soluções péssimas nos primeiros experimentos realizados. Mas nada comparado ao excelente resultado do Construtor 5 que pela própria essência deste método, busca soluções diversificadas e consegue alcançar o maior número de soluções melhores e um pequeno número de soluções péssimas nas primeiras análises realizadas.

E na análise probabilística a maioria das versões, principalmente algoritmos construídos com Grafo Original e que utilizaram o Construtor 5 conseguiram alcançar o alvo fácil e o alvo difícil. Percebe-se que a construção de soluções iniciais melhores (Construtor 3 e Construto 5) permite que a busca local consiga tratar de soluções diferentes e próximas (maior variabilidade de soluções) de ótimos locais, e por consequência, próximas de soluções de melhor qualidade para tentar encontrar um ótimo global. E outra constatação foi que os algoritmos convergiram para a solução alvo de acordo com a sequência: primeiro o SVNS, segundo o RVNS e posteriormente o GVNS, o VNS, o GRVND, o GVND, o GPR e por último o GRASP *G*. Os experimentos computacionais para análise de robustez dos algoritmos mostram que a abordagem proposta é competitiva, pois consegue obter bons resultados. Além disso, apresentar um valor baixo do coeficiente de variação, que mostra que o algoritmo é robusto, e gera soluções com baixa variabilidade. E que relação entre a solução média dos melhores algoritmos com a solução *best* é de proximidade, mostrando o quão perto da meta esses algoritmos conseguem chegar.

Assim, conclui-se que tenha-se atingido o objetivo principal deste trabalho, ou seja, desenvolver formas eficientes e robustas da metaheurística GRASP e VNS para o problema de clusterização com restrições de capacidade e de conexidade.

Como desdobramentos deste trabalho de tese foram desenvolvidos os seguintes artigos:

Santos, Nádia Mendes, Gustavo Silva Semaan, and Luiz Satoru Ochi. "*Metaheuristic GRASP with path-relinking to the solution of the graph partitioning problem with capacity*

and connexity constraints." Intelligent Data Engineering and Automated Learning-IDEAL 2012. Springer Berlin Heidelberg, 2012. 630-641.

Santos, Nádia Mendes, Gustavo Silva Semaan, and Luiz Satoru Ochi, Brito, José André Moura. Metaheurística híbrida para a solução de problema de particionamento de grafos com restrições de capacidade e de conexidade. X OPTIMA , Concepción, CHILE, 2013.

Santos, Nádia Mendes, Gustavo Silva Semaan, and Luiz Satoru Ochi, Brito, José André Moura. Metaheurística híbrida para a solução de problema de particionamento de grafos com restrições de capacidade e de conexidade. ELAVIO (Escola Latino-Iberoamericana de Verão em Pesquisa Operacional), Areia-PB, BRASIL, Fev-2014.

Santos, Nádia Mendes, Gustavo Silva Semaan, and Luiz Satoru Ochi, Brito, José André Moura. Metaheurística GRASP e GRASP-RVND aplicadas a problemas agrupamento com restrição de capacidade e de conexidade. CLAIO XVII (XVII Congresso Latino-Ibero-Americano em Pesquisa Operacional), Monterrey, MEXICO, Outubro-2014 (**Aprovado)**.

Santos, Nádia Mendes, Gustavo Silva Semaan, and Luiz Satoru Ochi, Brito, José André Moura. Metaheuristic GRASP and GRASP-RVND applied to clustering problems with capacity and connexity constraints. Pattern Recognition Letters. (**Submetido)**.

5.2 TRABALHOS FUTUROS

Durante a realização deste doutorado várias possibilidades de pesquisas mostraram-se interessantes. Não obstante, por mais que não se tenha tido tempo suficiente para implementá-las, as mesmas estão descritas a seguir como trabalhos futuros a serem executados e submetidos em revistas ou eventos científicos, são elas:

- Penalização das soluções: inserir nos algoritmos uma variável referente às eventuais penalidades existentes nos *clusters*. Para isto, deve-se agregar ao custo da solução um custo adicional conforme o grau de inviabilidade da solução, ou seja, deve-se agregar na função objetivo um fator de penalização que possui valor proporcional à inviabilidade da solução.

- Desenvolvimento de uma formulação de programação inteira para o problema de clusterização com restrições de capacidade e de conexidade;

- Implementar a formulação matemática e utilizar um bom *Upper Bound* gerado a partir de uma solução viável de boa qualidade obtida pela melhor resultado das metaheurísticas aplicadas ao problema de clusterização com restrições de capacidade e de conexidade. A expectativa é reduzir o tempo de processamento da formulação para produzir dessa forma soluções viáveis, ou tentar reduzir o tempo para atingir o ótimo global.

REFERÊNCIAS

[AGRAVAL ET AL 2005] Agraval, R.; Automatic Subspace Clustering of High Dimensional Data, Data Mining and Knowledge Discovery 11, pp. 5-33, 2005.

[AIEX ET AL. 2007] Aiex, R. M., Resende, M. G. C., and Ribeiro, C. C. (2007). TTT plots: a perl program to create time-to-target plots. Optimization Letters, 1:355_366.

[ANDRADE E CUNHA 2011] Andrade L, Cunha C. Algoritmo de colônia artificial de abelhas para um problema de clusterização capacitado. Escola Politécnica da USP, Brasil (2011).

[ANDRADE 2009] Andrade, A.V., Assis, L.P., Caires, L.F.V.: Aplicação do Método Path-Relinking na resolução do Problema Roteamento de Veículos com Coleta e Entrega Simultâneas. XLIISBPO (2009).

[AQUINO 2010] Aquino, A. A. Otimização em grafos: árvores geradoras com restrições. Monografia da Graduação de Computação. Universidade Federal de Pernambuco (UFPE). Centro de informática

[ASSUNÇÃO ET AL., 2006] Assunção, R. M.; Neves , M. C.; Câmara, G., Freitas , C. Efficient regionalization techniques for socio-economic geographical units using minimum spanning trees. International Journal of Geographical Information Science 20(7): 797-811, 2006.

[BASTOS ET AL., 2005] Bastos, L. O., Ochi, L. S.; Macambira, E.M. GRASP with Path Relinking for the SONET Ring Assignment Problem. Proc. of the 5th International Conference on Hybrid Intelligent Systems (HIS2005), pp. 239-244, 2005.

[BAUM, 1986] Baum, E.B. Iterated descent: A better algorithm for local search in combinatorial optimization problems. Technical report Caltech, Pasadena, CA. 1986.

[BAXTER, 1981] BaxteR, J. Local optima avoidance in depot location. Journal of the Operational Research Society 32, 1981.

[BERKHIN 2002] Berkhin, P. Survey of Clustering Data Mining Techniques. Accrue Software, 2002

[BINATO ET AL 2001] Binato, S.; Pereira, M. & Granville, S. A new Benders decomposition approach to solve power transmission network design problems. IEEE Transactions on Power Systems, 16(2), 235-240, May, 2001.

[BLUM AND ROLI 2003] Blum, C.; Roli, A. Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. ACM Computing Surveys, ACM, Nova York, NY, EUA, v. 35, n. 3, p. 268-308, 2003.

[BOLEY ET AL 1999] Boley, D.; Gini, M.; Gross, R.; Han, E.H.; Hastings, K.; Karypis, G.; Kumar, V.; Mobasher, B. & Moore, J. Partitioning based clustering for Web document categorization. Decision Support Systems, 27, 329-341, 1999.

[BONAMI 2012] Bonami, Pierre, et al. "On the solution of a graph partitioning problem under capacity constraints." Combinatorial Optimization. Springer Berlin Heidelberg, 2012. 285-296.

[BRITO E MONTENEGRO 2010] Brito, J. A. M. ; Montenegro, F. M. T.. Um Algoritmo VNS Aplicado ao Problema de Definição de Áreas de Ponderação. In: Simpósio de Pesquisa Operacional e Logística da Marinha, 2010, (Rio de Janeiro). Anais do SPOLM 2010, 2010.

[BRITO ET AL., 2010] Brito, J. A. M.; Antonio, J. R. D.; Bruno, F.C; Flávio, M. T. M. Um algoritmo de agrupamento aplicado à Definição das áreas de ponderação do Censo Demográfico 2010.

[BRITO ET AL., 2004] Brito, J. A. M.; Montenegro , F. M. T.; Brito L. R.; Passini , M. M.. Uma formulação de programação inteira para o problema de criação de áreas de ponderação agregadas, Anais do SOBRAPO, 2004.

[BROWN AND BOLKER 2004] Brown, D.H., AND Bolker B.M.. The effects of disease dispersal and host clustering on the epidemic threshold in plants. *Bulletin of Mathe matical Biology 66* (March 2004), 341-371.

[CENSO DEMOGRÁFICO 2010] Censo Demografico 2010- Resultados Gerais da Amostra – IBGE – ISSN 0104-3145.

[CESELLI ET AL 2013] Ceselli, Alberto, Fabio Colombo, Roberto Cordone, and Marco Trubian. "Employee workload balancing by graph partitioning." *Discrete Applied Mathematics* (2013). Volume 165, 2014, pp 112-129. Elsevier Science Publishers B. V. Amsterdam, The Netherlands, The Netherlands. ISSN 0166-218X

[CERVEIRA 2014] CERVEIRA A.; BAPTISTA J.; PIRES, E. J. S.; Advances in Intelligent Systems and Computing Volume 239, 2014, pp 109-119 ISBN 978-3-319-01853-9. Salamanca, Spain 2014.

[CHAVES E LORENA, 2005] Chaves, A. A.; Lorena, L. A. N. Hybrid Algorithms with Detection of Promising Areas for the Prize Collecting Travelling Salesman Problem. In: Proceedings of the Fifth International Conference on Hybrid Intelligent Systems (HIS '05). Washington, DC, EUA: IEEE Computer Society, 2005. p. 49-54. ISBN 0-7695-2457-5.

[CHIOU AND LAN, 2001] Chiou, Y.C. Lan, L.W. Genetic Clustering Algorithms. European Journal of operational Research 135, 2001.

[COLGANO 2012] Aquino, A. A. Uma proposta para a formalização do problema de Clusterização em grafos. Dissertação de Mestrado de Ciência em Computação. Universidade Federal do Paraná (UFPE). Centro de informática.

[CORTEZ ET AL 2012] Cortez, B.F; Brito, J.A.M. ; Dias, A.J.R; Montenegro, F.M.T.. Um algoritmo de agrupamento aplicado à definição das áreas de ponderação do Censo Demográfico de 2010. 20º Simpósio Nacional de Probabilidade e Estatística (JP). 2012.

[DAMODAR E PRASANTA, 2013] Edla, Damodar Reddy, and Prasanta K. Jana. "Minimum Spanning Tree Based Clustering Using Partitional Approach." Proceedings of the

International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA). Springer Berlin Heidelberg, 2013.

[DENG AND BARD 2011] Deng, Yumin, and Jonathan F. Bard. "A reactive GRASP with path relinking for capacitated clustering." *Journal of Heuristics* 17.2 (2011): 119-152.

[DIAS AND OCHI, 2003] Dias, C. R., and Ochi, L. S. Efficient evolutionary algorithms for the clustering problem in directed graphs. In *IEEE-CEC Congress on Evolutionary Computation em CD-ROM* (Canberra – Austrália, 2003), pp. 983-990

[DIAS, 2004] Dias, C.R. Algoritmos Evolutivos para o Problema de Clusterização de Grafos Orientados - Desenvolvimento e Analise Experimental. Dissertação de Mestrado, Universidade Federal Fluminense (UFF), 2004.

[DOVAL ET AL., 1999] Doval, D., Mancoridis, S. e Mitchell, B. S. Automatic Clustering of Software Systems using a Genetic Algorithm. In 1999 International Conference on Software Tools and Engineering Practice (STEP '99), 1999

[DREW, 2003] Drew, M. S., and Au, J. Clustering of compressed illumination-invariant chromaticity signatures for efficient video summarization. *Image and Vision Computing 21* (August 2003), 705-716.

[FANG ET AL, 2003] Fang, J. T., Shyu, W. L., Wu, C. S., and Chen, K. J. Clustering source output bits and equalizing bit error sensitivy to improve the quality and robustness of transmitted images over wireless channels. *Signal Processing: Image Communication 18* (November 2003), 947-956.

[FASULO, 1999] Fasulo, D. An Analysis of Recent Work on Clustering Algorithms. Technical Report,Dept. of Computer Science and Engineering, Univ. of Washington, 1999.

[FESTA 2007] Festa, P.; Pitsoulis, P.M; L. S and Resende, M.C.G. GRASP with path-relinking for the weighted MAXSAT problem. *Journal of Experimental Algorithms*, 11, 2007 Article 2-4: 1-16.

[FEO AND RESENDE, 1995] Feo, T.A.; Resende, M.G.C., Greedy randomized adaptive search procedures,Journal. of Global Optimization, 6, 109—133, 1995.

[FISHER 1987] FISHER, L. Knowledge acquisition via incremental conceptual clustering, Machine Learning 2, pp. 139-172. 1987

[GARAI AND CHAUDHURI 2004] Garai, G., Chaudhuri, B. B.; A novel genetic algorithm for automatic clustering *Pattern Recognition Letters*. pp. 173-187. 2004

[GLOVER ET AL, 2004] Glover, F.; Laguna, M.; Martí, R. Scatter Search and Path Relinking: Foundations and Advanced Designs. In: Onwubolu, G. C.; Babu, B. V. (Ed.). New Optimization Techniques in Engineering. Heidelberg: Springer-Verlag, 2004. p. 87-100.

[GLOVER AND KOCHENBERGER, 2002] Glover, F. ; Kochenberger, G. A. Handbook of Metaheuristics. Kluwer Academic Publishers, 2002.

[GLOVER ET AL, 2000] Glover, F.; Laguna, M.; Martí, R. Fundamentals of Scatter Search and Path Relinking. Control and Cybernetics, v. 29, n. 3, p. 653684, 2000.

[GLOVER 1999] Glover F. "Scatter Search and Path Relinking," In: D Corne, M Dorigo and F Glover (eds.) New Ideas in Optimisation, Wiley, 1999.

[GLOVER 1996] Glover, F. Tabu Search and Adaptive Memory Programing - Advances, Applications and Challenges. In: Barr, R. S.; Helgason, R. V.; KENNINGTON, J. L. (Ed.). Interfaces in Computer Science and Operations Research. Dordrecht, MA, EUA: Kluwer Academic Plublishres, 1996. p. 1-75.

[GLOVER AND LAGUNA, 1997] Glover, F., Laguna, M.: Tabu Search. Kluwer, Boston (1997) ISBN 0-7923-9965-X.

[GLOVER AND MARTI, 2006] Glover, F. And Marti, R. (2006) "Metaheuristics for Training Neural Networks", Springer Science, New York,Chapter 4.

[GUO ET AL, 2003] GUO, H., Renaut, R., Chen, K. and Reiman, E. Clustering huge data sets for parametric pet imaging. *Biosystems 71* (Septembr 2003), 81-92.

[GREISTORFER 2004] Greistorfer, P., 2004. Experimental pool design: Input, output and combination strategies for scatter search. In: Resende, M. G. C., de Sousa, J. P. (Eds.), Metaheuristics: Computer Decision-Making, Kluwer Academic Publishers,Boston, pp. 1-18.

[HAN AND KAMBER, 2001] Han, J., and Kamber, M. *Data Mining: Concepts and Techiniques.* Morgan Kaufmann, 2001.

[HANSEN AND MLADENOVIC 2003] Hansen, P.; Mladenovic, N. Variable Neighborhood Search. In: GLOVER, F. W.; KOCHENBERGER, G. A. (Ed.). Handbook of Metaheuristics. Boston, Dordrecht: London: Kluwer Academic Publisher, 2003. p. 145-184.

[HANSEN AND MLADENOVIC 2001] Hansen, P.; Mladenovic, J-Means: A new local search heuristic for minimum sum-of-squares clustering, *Pattern Recognition* 34: 405-413, 2001.

[HARTUV AND SHAMIR, 1999] Hartuv, E.; Shamir, R.. A Clustering Algorithm based on Graph Connectivity. Technical Report, Tel Aviv University, Dept. of Computer Science, 1999.

[HERNÁNDEZ-PÉREZ ET AL 2009] Hernández-pérez, H.; Rodríguez-martín, I.; Salazar-gonzález, J. J. A hybrid GRASP/VND heuristic for the one-commodity pickup-and-delivery traveling salesman problem. Computers & Operations Research, v. 36, n. 5, p. 1639 - 1645, 2009

[HO AND GENDREAU, 2006] Ho, S.C., Gendreau, M.: Path-relinking for the vehicle routing problem. Journal of Heuristics 12(1), 55–72 (2006).

[HRUSCHKA, E. R, 2001] Algoritmos genéticos de agrupamento para extração de regras de redes neurais. Tese de Mestrado, Universidade Federal do Rio de Janeiro, 2001.

[JAIN, 2010] Jain, A. K., "Data Clustering: 50 Years Beyond K-Means" , Pattern Recognition Letters, Vol. 31, No. 8, pp. 651-666, 2010.

[LAGUNA AND MARTI, 1999] Laguna, M.; MartÍ, R. Grasp and Path Relinking for 2-Layer Straight Line Crossing Minimization. INFORMS Journal on Computing, INFORMS, Institute for Operations Research and the Management Sciences (INFORMS), Linthicum, Maryland, USA, v. 11, n. 1, p. 44-52, 1999.

[LI ET AL 1994] Li, Y.; Pardalos, P.M.; Ramakrishnan, K.G. & Resende, M.G.C. (1994). Lower bounds for the quadratic assignment problem. Operations Research, 50, 387-410

[LOURENÇO ET AL 2002] Lourenço, H. R.; Martin, O. C.; Stützle, T. Iterated Local Search. In: Glover, F.; Kochenberger, G. (Ed.). Handbook of Metaheuristics. [S.l.]: Kluwer Academic Publishers, Boston, 2002. p. 321-353

[KAUFMAN 1990]KAUFMAN, Leonard; ROUSSEEUW, Peter J. Finding groups in data: an introduction to cluster analysis. New York: Wiley, 1990.

[MLADENOVIC AND HANSEN, 1997] Mladenovic, N.; Hansen, P. Variable Neighborhood Search. Computers Operations Research 24,1997.

[MLADENOVIC, 1995] Mladenovic, N. A Variable neighborhood algorithm: a new metaheuristic for combinatorial optimization. Abstracts of papers presented at Optimization Days, Montréal, 1995.

[NEIL AND RICO, 2013] Neil O., and Rico Z.. "Chain-Constrained Spanning Trees." Integer Programming and Combinatorial Optimization. Springer Berlin Heidelberg, 2013. 324-335.

[NEVES, 2003] Neves, M.C. Procedimentos Eficientes para Regionalização de Unidades Socioeconômicas em Bancos de Dados Geográficos. Tese de Doutorado, INPE, São José dos Campos, 2003.

[MATEUS ET AL 2011] Mateus, G.R, Resende, M.G.C. and Silva, R.M.A. GRASP with path-relinking for the generalized quadratic assignment problem. J. of Heuristics, 17:527–565, 2011.

[MANDAL ET AL 2012] Mandal, A., Dutta, J., & Pal, S. C. (2012). A New Efficient Technique to Construct a Minimum Spanning Tree. International Journal, 2(10).

[MONTEIRO 2009] Monteiro, J. B. Indicador de Criminalidade Geral Baseado em Métodos Multivariados e Estatística Espacial para Controle na Segurança Pública do Estado Monografia de Conclusão do Bacharelado em Estatísica. Instituto de Matemática, UFRGS.

[MURTY ET AL 2014] MURTY, M. R.; MURTHY, J. V.R.,; REDDY P.V.G.D Prasad.; NAIK, A.; SATAPATHY, S.C. Homogeneity Separateness: A new validity measure for clustering problems. CT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India- Vol I Advances in Intelligent Systems and Computing Volume 248, 2014, pp 1-10

[OPENSHAW 1977] Openshaw, S., A geographical solution to scale and aggregation problems in regionbuilding, partitioning and spatial modelling. Transactions of the Institute of British Geographers (New Series), 2, pp. 459–472, 1977.

[PENA ET AL 1999] PENA, J. M., Lazano, J. A., and Larranaga, P.An empirical comparison of four nitialisation methods for the K-means algorithm. Pattern Recognition Lett., 1999, 20, 1027–1040.

[PIAGET 1996] PIAGET. J Biologia e Conhecimento. 2ª Ed. Vozes : Petrópolis, 1996

[PITSOULIS AND RESENDE 2002] Pitsoulis, L. and Resende, M.G.C. (2002) Greedy randomized adaptive search procedures.. In P.M.Pardalos and M.G.C.Resende, editors, Handbook of Applied Optimization, pp. 168–181, Oxford University Press.

[RAIDL, 2006] Raidl, G.R. 2006. A unified view on hybrid metaheuristics. In Hybrid metaheuristics, Lecture notes in computer science 4030. Springer, Berlin, Heidelberg, 1-12.

[RESENDE AND FEO 1996] Resende, M. G. C. e Feo, T. A GRASP for satisfiability, em Johnson, D. S. e Trick M. A. (Eds) Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, 26, 499-520. DIMACS Series on Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, 1996

[RESENDE E RIBEIRO, 1997] Resende, M.G.C., Ribeiro, C.C.,1997 “A GRASP for Graph Planarization”, Networks, 29:173-189.

[RESENDE E RIBEIRO 2002] Resende, M.G.C., Ribeiro, C.C., 2003. Greedy randomized adaptive search procedures. In Glover, F., Kochenberger, G. (eds) Handbook of Metaheuristics. Kluwer, Dordrecht, pp. 219–249.

[RIBEIRO E SOUZA 2002] Ribeiro, C.C., Souza, M.C., 2002. Variable neighborhood search for the degree-constrained minimum spanning tree problem. Discrete Applied Mathematics, 118, 43–54.

[RIBEIRO E VIANA 2005] Ribeiro, C.C. Ribeiro and Vianna, D.S. A GRASP/VND heuristic for the phylogeny problem using a new neighborhood structure. International Transactions in Operational Research, 12:325–338, 2005.

[RODRIGUES 2001] Rodrigues, F. L. Metaheurística e sistema de suporte à decisão no gerenciamento de recursos florestais. 2001. 225 f. Tese (Doutorado em Manejo Florestal) – Universidade Federal de Viçosa, Viçosa, 2001.

[SANTOS ET AL 2012] Santos, Nádia Mendes, Gustavo Silva Semaan, and Luiz Satoru Ochi. "Metaheuristic GRASP with path-relinking to the solution of the graph partitioning problem with capacity and connexity constraints." Intelligent Data Engineering and Automated Learning-IDEAL 2012. Springer Berlin Heidelberg, 2012. 630-641.

[SANTOS ET AL 2013] Santos, Nádia Mendes, Gustavo Silva Semaan, and Luiz Satoru Ochi. Brito, José André Moura. Metaheurística híbrida para a solução de problema de particionamento de grafos com restrições de capacidade e de conexidade. X OPTIMA , Concepción, CHILE, 2013.

[SANTOS ET AL 2014] Santos, Nádia Mendes, Gustavo Silva Semaan, and Luiz Satoru Ochi. Brito, José André Moura. Metaheurística híbrida para a solução de problema de particionamento de grafos com restrições de capacidade e de conexidade. ELAVIO (Escola Latino-Iberoamericana de Verão em Pesquisa Operacional), Areia-PB, BRASIL, Fev-2014.

[SCHEUERER, 2006] Scheuerer, S. W. , R. A scatter search heuristic for the capacitated clustering problem. European Journal of Operational Research vol. 169, 2006.

[SEMAAN ET AL., 2008] Semaan, G. S., Ochi, L. S., Brito, J. A. M.; Montenegro, F. An Efficient Evolutionary Algorithm for the Aggregated Weighting Areas Problem. Proc. of the International Conference on Engineering Optimization, 2008.

[SEMAAN ET AL., 2009] Semaan, G.S., Brito, J. A. M.; Ochi, L. S. Um algoritmo evolutivo hibrido aplicado ao problema de clusterização em grafos com restrições de capacidade e contiguidade. Anais do IX Congresso Brasileiro de Redes Neurais e Inteligência Computacional (IX CBRN),Ouro Preto/MG, 2009.

[SEMAAN 2010] Semaan, G.S.: Algoritmos Heurísticos para o Problema de Particionamento de Grafos com Restrições de Capacidade e Conexidade. Dissertação de Mestrado, Universidade Federal Fluminense (UFF), Niterói – RJ (2010)

[SHIEH AND MAY, 2001] Shieh, H.M., May, M.D. Solving the Capacitated Clustering Problem with Genetic Algorithms. Journal of the Chinese Institute of Industrial Engineers, Vol. 18, 2001.

[SILVA ET AL 2000] Silva, M. B. da; Drummond, L. M. de A.; Ochi, L. S. Metaheurística GRASP/VNS para a solução de Problemas de Otimização Combinatória. In: Anais do XXXII Simpósio Brasileiro de Pesquisa Operacional (SBPO). [S.l.: s.n.], 2000. v. 1, p. 352-360.

[SOARES, 2004] Soares, S.S.R.F. Metaheurísticas para o Problema da Clusterização Automática, Dissertação de Mestrado, (UFF), Niterói-RJ, 2004.

[SOUZA 2011] Souza, R. F.F.: Planejamento da Expansão de Sistemas de Distribuição Usando a Metaheurística de Busca em Vizinhança Variável. Dissertação de Mestrado, Universidade Estadual Paulista “Julio de Mesquita Filho” (Campus de Ilha Solteira), Ilha Solteira – SP (2011)

[SPATH, 1980] Spath, H. Cluster Analysis Algorithms for Data Reduction and Classification of Objects. John Wiley & Sons, Ellis Horwood Ltd., 1980.

[STEINLEY AND BRUSCO 2007] Steinley, Douglas and Michael J. Brusco, “A comparison of heuristic procedures for minimum within-cluster Sums of squares partitioning” Psychometrika—vol. 72, no. 4, 583–600, 2007

[TRINDADE AND OCHI, 2006] Trindade, A.R.; Ochi, L.S. Um algoritmo evolutivo híbrido para a formação de células de manufatura em sistemas de produção. Abstracts in Operational Research / Statistical Theory and Methods Abstracts) vol. 26(2), 2006.

[TRINDADE AND OCHI, 2004] Trindade, A. R., and Ochi. Desenvolvimento e análise experimental de um algoritmo evolutivo para o problema de clusterização em sistemas de manufatura. In *Congresso Brasileiro de Computação (IV CBcomp)* – 2004.

[VANETIK, 2013] Vanetik, Natalia. "Mining Graphs of Prescribed Connectivity." Knowledge Discovery, Knowledge Engineering and Knowledge Management. Springer Berlin Heidelberg, 2011. 29-44.]

[WADSWORTH 1996] Wadsworth, B.J. Inteligência e afetividade da criança na teoria de Piaget. São Paulo: Pioneira, 1996.

[WANG, 2003] Wang, J. Formation of machine cells and families in cellular manufacturing systems using a linear assignment algorithm. *Automatica 39* (April 2003), 1607-1615.

[WELCH, 1983] Welch, J. W. Algorithmic complexity: three NP-hard problems in computational statistics. Journal of Statistical Computation and Simulation, 15, 17-25. DOI: 10.1080/00959658208810560.

[ZAGOURAS ET AL 2012] Zagouras, Athanassios, Athanassios A. Argiriou, Helena A. Flocas, George Economou, and Spiros Fotopoulos. "An advanced method for classifying atmospheric circulation types based on prototypes connectivity graph."Atmospheric Research (2012).

APÊNDICE A – APRESENTAÇÃO DAS TABELAS COM OS RESULTADOS COMPUTACIONAIS OBTIDOS NA EXECUÇÃO DOS ALGORITMOS PROPOSTOS NESTE TRABALHO PARA CADA UMA DAS 64 INSTÂNCIAS OBTIDAS DO CENSO DEMOGRÁFICO 2010.

Tabela A.1 – Resultados GRASP Padrão – Grafo Original para obtenção de 3 *clusters*

		GRASP PADRÃO									
		G1		G2		G3		G4		G5	
I	*Solução best*	Gap	T(s)	Gap	T(s)	Gap	T(s)	Gap	T(s)	Gap	T(s)
1	*22,54*	**35**	112	**23**	95	**21**	106	**28**	109	**19**	104
2	*39,08*	**42**	214	**34**	208	**24**	171	**33**	204	**22**	183
3	*5,72*	**45**	28	**38**	15	**29**	27	**26**	29	**20**	24
4	*26,14*	**47**	162	**43**	137	**32**	119	**35**	162	**28**	119
5	*7,35*	**27**	65	**21**	34	10	33	**18**	67	**12**	30
6	*6,41*	**21**	57	**17**	21	0	27	8	52	3	30
7	*6,17*	**14**	44	1	20	0	27	8	38	0	29
8	*4,96*	**17**	32	0	22	0	25	0	29	0	20
9	*14,35*	**36**	85	6	61	**12**	52	**21**	63	**18**	55
10	*19,78*	**39**	117	**32**	91	**19**	98	**27**	104	**22**	86
11	*10,27*	**47**	86	**41**	45	**28**	40	**33**	83	**25**	39
12	*17,33*	**20**	116	**19**	80	**15**	71	**17**	104	**13**	65
13	*7,42*	**27**	77	**24**	51	7	33	**18**	70	**11**	31
14	*5,94*	**45**	73	**44**	43	**19**	27	**36**	64	**29**	29
15	*18,53*	**41**	80	**31**	92	**13**	71	**25**	102	**17**	75
16	*7,09*	**38**	69	**33**	33	**19**	25	**28**	65	**21**	29
17	*10,81*	**59**	96	**47**	51	**32**	43	**43**	84	**29**	40
18	*6,33*	**30**	57	**18**	29	1	26	8	42	2	31
19	*57,14*	**47**	1479	**39**	1467	**27**	1381	**42**	1449	**37**	1368
20	*49,02*	**91**	1404	**83**	1374	**66**	1307	**74**	1340	**69**	1299
21	*70,51*	**23**	1603	**18**	1582	**11**	1412	**15**	1467	10	1398
22	*22,17*	**16**	117	**12**	82	7	91	0	113	2	100
23	*6,39*	**25**	68	0	25	0	27	7	48	0	31
24	*6,30*	**13**	57	**11**	28	0	23	9	42	2	30
25	*82,44*	**22**	51	6	19	2	26	0	41	2	23
26	*12,68*	**128**	476	**114**	388	**97**	451	**104**	476	**85**	417
27	*3,27*	**63**	114	**56**	53	**45**	32	**49**	98	**46**	41
28	*25,89*	**12**	33	**8**	28	0	20	5	24	0	17
29	*26,01*	**66**	151	**57**	127	**38**	143	**52**	139	**41**	114
30	*15,48*	**71**	184	**68**	149	**32**	177	**43**	165	**29**	123
31	*78,30*	**50**	101	**35**	76	**18**	65	**23**	79	**17**	64
32	*6,41*	**127**	1883	**101**	1860	**76**	1671	**80**	1764	**75**	1693
33	*20,08*	**32**	55	**19**	30	1	27	7	45	0	30
34	*8,15*	**43**	61	**28**	29	**17**	36	**28**	48	10	22
35	*25,44*	**34**	129	**33**	87	**12**	92	**29**	110	**15**	88
36	*37,83*	**63**	110	**57**	98	**14**	119	**31**	135	10	111
37	*56,91*	**59**	181	**46**	163	**28**	168	**39**	174	**25**	154
38	*53,49*	**98**	1460	**64**	1439	**55**	1335	**60**	1417	**52**	1352
39	*73,06*	**82**	1444	**67**	1402	**47**	1329	**53**	1391	**41**	1314
40	*5,37*	**20**	43	9	19	0	22	0	28	0	23
41	*19,24*	**31**	101	**22**	79	**11**	75	**18**	98	**13**	72
42	*25,67*	**33**	146	**24**	123	**19**	117	**24**	135	**16**	108
43	*93,18*	**187**	2002	**146**	1995	**131**	1811	**149**	1830	**138**	1797
44	*6,53*	**29**	70	**23**	49	2	34	8	51	4	30
45	*39,41*	**62**	217	**57**	194	**43**	228	**46**	215	**39**	183
46	*24,37*	**51**	140	**39**	134	**32**	129	**34**	125	**28**	112
47	*12,99*	**55**	136	**48**	70	**31**	69	**42**	117	**35**	44
48	*45,78*	**83**	302	**67**	244	**48**	237	**50**	259	**42**	215
49	*16,04*	**46**	90	**37**	81	**26**	72	**32**	93	**28**	67
50	*26,07*	**49**	171	**48**	155	**29**	130	**33**	148	**27**	129

51	***98,16***	**134**	2009	**126**	2001	**99**	1915	**109**	1997	**102**	1903
52	***3,08***	**10**	27	0	20	0	19	0	23	0	19
53	***38,50***	**58**	209	**47**	154	**29**	168	**36**	204	**29**	171
54	***22,01***	**39**	116	**35**	85	**14**	99	**23**	102	**18**	93
55	***74,19***	**82**	1692	**78**	1673	**60**	1495	**80**	1661	**61**	1510
56	***68,27***	**77**	1495	**72**	1481	**53**	1408	**69**	1450	**57**	1384
57	***15,91***	**45**	82	**36**	69	**29**	55	**32**	74	**24**	52
58	***5,33***	**21**	32	0	23	0	30	3	27	0	24
59	***40,21***	**62**	336	**57**	281	**41**	294	**48**	328	**33**	201
60	***108,73***	**153**	2038	**149**	2019	**122**	1971	**137**	2010	**114**	1908
61	***60,49***	**82**	370	**76**	292	**57**	354	**64**	312	**59**	333
62	***102,53***	**165**	2015	**158**	2011	**127**	1980	**144**	2013	**118**	1925
63	***15,67***	**69**	104	**57**	72	**28**	58	**39**	97	**20**	59
64	***80,24***	**117**	1898	**104**	1881	**89**	1722	**99**	1885	**96**	1702

Tabela A.2 – Resultados GRASP Padrão - AGM *Kruskal* modificado para obtenção de 3 *clusters*

		GRASP PADRÃO									
		AGMM1		**AGMM2**		**AGMM3**		**AGMM4**		**AGMM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**108**	106	**106**	104	**101**	96	**104**	105	**102**	94
2	***39,08***	**125**	113	**124**	110	**121**	117	**122**	111	**123**	115
3	***5,72***	**69**	38	**68**	35	**63**	33	**67**	36	**65**	32
4	***26,14***	**78**	132	**77**	129	**72**	124	**75**	130	**71**	121
5	***7,35***	**67**	49	**66**	44	**61**	47	**64**	48	**62**	49
6	***6,41***	**64**	40	**68**	37	**62**	33	**60**	39	**59**	36
7	***6,17***	**71**	48	**70**	42	**68**	41	0	44	**69**	39
8	***4,96***	**62**	47	**60**	45	**59**	44	**60**	46	**58**	42
9	***14,35***	**121**	102	**122**	97	**111**	93	**119**	98	**113**	95
10	***19,78***	**108**	107	**109**	102	**104**	101	**105**	108	**102**	99
11	***10,27***	**149**	116	**147**	111	**138**	97	**144**	115	**140**	110
12	***17,33***	**81**	48	**83**	41	**79**	40	**80**	46	**78**	37
13	***7,42***	**93**	69	**90**	65	**88**	66	**91**	67	**89**	64
14	***5,94***	**67**	93	**69**	90	**61**	87	**64**	92	**63**	88
15	***18,53***	**104**	77	**102**	74	**99**	73	**102**	75	**98**	70
16	***7,09***	**125**	101	**127**	93	0	96	**124**	99	**118**	95
17	***10,81***	**93**	69	**93**	66	**89**	64	**91**	67	**88**	65
18	***6,33***	**157**	35	**155**	32	**144**	30	**148**	34	**143**	33
19	***57,14***	**194**	1924	**194**	1915	**188**	1903	**193**	1918	**187**	1900
20	***49,02***	**187**	2173	**184**	2154	**173**	2148	**182**	2164	**179**	2140
21	***70,51***	**151**	2100	**153**	2084	**148**	2077	0	2093	**147**	2081
22	***22,17***	**94**	41	**92**	36	**89**	34	**93**	38	**88**	33
23	***6,39***	**102**	36	**104**	32	**96**	30	**100**	35	**97**	27
24	***6,30***	**75**	39	**75**	31	**72**	36	**73**	37	**71**	35
25	***82,44***	**187**	397	**186**	381	**181**	385	**183**	388	**179**	383
26	***12,68***	**214**	125	**215**	120	**206**	118	**212**	122	**208**	119
27	***3,27***	**67**	24	**66**	21	0	24	**62**	23	**59**	22
28	***25,89***	**101**	127	**104**	124	**98**	119	**99**	125	**97**	118
29	***26,01***	**138**	121	**135**	117	**129**	118	**132**	120	**128**	121
30	***15,48***	**142**	154	**141**	148	**135**	147	**140**	149	**137**	145
31	***78,30***	**164**	2095	**163**	2114	**159**	2066	**162**	2087	**158**	2065
32	***6,41***	**157**	98	**158**	93	**150**	94	**155**	95	**152**	92
33	***20,08***	**143**	42	**142**	47	**138**	43	**140**	45	**139**	41
34	***8,15***	**98**	99	**100**	102	**92**	100	**97**	104	**93**	98
35	***25,44***	**105**	87	**108**	88	**101**	83	**104**	86	**103**	85
36	***37,83***	**183**	121	**182**	116	**177**	117	**180**	119	**178**	116
37	***56,91***	**94**	206	**95**	201	**93**	191	**92**	204	**91**	195
38	***53,49***	**179**	2033	**173**	2015	**169**	2017	**170**	2021	**168**	2016
39	***73,06***	**228**	2084	**229**	2066	**226**	2063	**224**	2071	**221**	2058
40	***5,37***	**63**	38	**62**	31	**58**	36	**61**	37	0	35
41	***19,24***	**101**	55	**100**	53	0	50	**99**	52	**96**	48
42	***25,67***	**97**	104	**95**	97	**93**	100	**94**	103	**92**	98

43	***93,18***	**294**	2113	**293**	2081	**285**	2077	**291**	2108	**287**	2071
44	***6,53***	**81**	47	**81**	41	**75**	45	**80**	46	**77**	47
45	***39,41***	**143**	243	**142**	236	**139**	230	**140**	239	**138**	228
46	***24,37***	**218**	152	**217**	144	**214**	140	**215**	148	**212**	142
47	***12,99***	**137**	97	**136**	91	**125**	91	**128**	93	**124**	92
48	***45,78***	**141**	204	**142**	195	**134**	187	**139**	198	**137**	194
49	***16,04***	**95**	117	**96**	114	**89**	108	**93**	115	**89**	110
50	***26,07***	**102**	135	**104**	127	**98**	125	**100**	133	**96**	131
51	***98,16***	**293**	2102	**295**	2081	**289**	2074	**291**	2090	**287**	2083
52	***3,08***	**57**	22	**56**	18	**53**	15	**56**	19	0	16
53	***38,50***	**108**	217	**107**	213	**104**	207	**104**	215	**105**	210
54	***22,01***	**67**	154	**66**	142	**69**	140	**65**	144	**63**	143
55	***74,19***	**191**	2187	**190**	2164	**188**	2165	**190**	2172	**187**	2152
56	***68,27***	**218**	104	**216**	102	**211**	100	**215**	103	**212**	98
57	***15,91***	**92**	116	**93**	119	**89**	121	**90**	118	**88**	117
58	***5,33***	**53**	27	**51**	32	**49**	30	**50**	31	**48**	27
59	***40,21***	**74**	393	**74**	384	**71**	379	**73**	387	**72**	368
60	***108,73***	**427**	2415	**425**	2404	**422**	2401	**424**	2425	**420**	2411
61	***60,49***	**182**	318	**180**	320	**173**	318	**180**	319	0	319
62	***102,53***	**455**	2488	**453**	2472	**449**	2464	**453**	2477	**450**	2456
63	***15,67***	**184**	115	**182**	114	**173**	112	**181**	116	**177**	115
64	***80,24***	**293**	2101	**287**	2095	**280**	2076	**284**	2085	**281**	2079

Tabela A.3 – Resultados GRASP Padrão - AGM *Kruskal* padrão para obtenção de 3 *clusters*

		GRASP PADRÃO									
		AGM1		**AGM2**		**AGM3**		**AGM4**		**AGM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**129**	99	**128**	95	**126**	87	**129**	96	**124**	83
2	***39,08***	**148**	120	**144**	117	**135**	114	**141**	118	**137**	112
3	***5,72***	**73**	39	**60**	38	**61**	31	**69**	33	**66**	30
4	***26,14***	**88**	129	**85**	126	**76**	122	**79**	128	**75**	119
5	***7,35***	**74**	52	**73**	49	**65**	43	**68**	49	**67**	44
6	***6,41***	**68**	39	**66**	35	**63**	32	**64**	36	**61**	31
7	***6,17***	**87**	44	**83**	38	**76**	34	**81**	40	**78**	32
8	***4,96***	**71**	47	**67**	43	**60**	43	**63**	44	**59**	41
9	***14,35***	**129**	98	**126**	94	**115**	87	**124**	96	**118**	90
10	***19,78***	**122**	104	**117**	98	**112**	95	**115**	101	0	86
11	***10,27***	**198**	115	**177**	110	**163**	93	**179**	112	**159**	107
12	***17,33***	**104**	46	**99**	40	**86**	39	**88**	43	**83**	35
13	***7,42***	**118**	67	**112**	63	**99**	61	**103**	64	**95**	59
14	***5,94***	**73**	98	**74**	92	**65**	85	**69**	95	**67**	84
15	***18,53***	**120**	75	**116**	69	**104**	66	**111**	71	**101**	65
16	***7,09***	**141**	99	**128**	91	**118**	94	**127**	96	**123**	92
17	***10,81***	**117**	67	**109**	64	**101**	62	**102**	65	**95**	61
18	***6,33***	**163**	34	**156**	29	**148**	30	**155**	32	**147**	30
19	***57,14***	**214**	1919	**205**	1901	**196**	1893	**199**	1904	**194**	1891
20	***49,02***	**188**	2165	**188**	2128	**185**	2119	**187**	2137	**183**	2112
21	***70,51***	**163**	2093	**161**	2077	**157**	2057	0	2085	**154**	2069
22	***22,17***	**105**	38	**100**	34	**91**	33	**99**	35	**93**	32
23	***6,39***	**116**	39	**115**	35	**96**	28	**112**	37	**107**	25
24	***6,30***	**83**	40	**80**	36	**78**	34	**80**	38	**76**	34
25	***82,44***	**204**	383	**189**	375	**181**	371	**183**	376	**180**	369
26	***12,68***	**250**	119	**244**	116	**225**	102	**241**	117	**236**	103
27	***3,27***	**70**	24	**71**	19	**66**	23	**69**	22	**68**	20
28	***25,89***	**104**	125	**100**	120	**99**	112	**98**	119	**95**	109
29	***26,01***	**147**	120	**148**	117	**135**	114	**139**	118	**133**	116
30	***15,48***	**168**	143	**155**	128	**141**	140	**153**	140	**144**	129
31	***78,30***	**177**	2078	**173**	2069	**174**	2053	**173**	2071	**171**	2048
32	***6,41***	**166**	96	**164**	91	**153**	88	**160**	93	**152**	87
33	***20,08***	**148**	44	**144**	42	**141**	41	**142**	43	**140**	40
34	***8,15***	**104**	102	**102**	101	**97**	98	**100**	99	**99**	91

35	*25,44*	**118**	85	**116**	80	**102**	73	**115**	82	**108**	76
36	*37,83*	**197**	118	**194**	112	**181**	110	**190**	115	**183**	110
37	*56,91*	**109**	201	**108**	195	**99**	183	**102**	197	**99**	181
38	*53,49*	**201**	2023	**194**	2016	0	2009	**191**	2018	**182**	2003
39	*73,06*	**258**	2067	**251**	2051	**245**	2046	**247**	2055	**238**	2037
40	*5,37*	**66**	36	**48**	32	**61**	33	**45**	35	**35**	32
41	*19,24*	**117**	53	**110**	51	**112**	47	**113**	50	**106**	43
42	*25,67*	**99**	101	**99**	99	**97**	98	**98**	100	**94**	95
43	*93,18*	**348**	2110	**325**	2093	**289**	2064	**314**	2095	**291**	2053
44	*6,53*	**88**	49	**84**	45	**73**	40	**83**	46	**79**	43
45	*39,41*	**166**	234	**157**	227	**146**	222	**151**	229	**145**	217
46	*24,37*	**248**	148	**244**	143	**224**	141	**239**	144	**221**	140
47	*12,99*	**144**	94	**140**	89	**124**	90	**142**	91	**128**	90
48	*45,78*	**153**	199	**150**	191	**144**	184	**149**	193	**140**	187
49	*16,04*	**96**	111	**94**	106	**91**	101	**93**	109	**90**	104
50	*26,07*	**150**	129	**142**	127	**112**	124	**141**	128	**98**	123
51	*98,16*	**321**	2094	**311**	2082	**304**	2069	**299**	2084	**306**	2071
52	*3,08*	**74**	19	**62**	15	**60**	16	**21**	17	**20**	14
53	*38,50*	**128**	214	**125**	208	**112**	201	**122**	211	**118**	203
54	*22,01*	**76**	144	**73**	141	**63**	140	**69**	142	**66**	140
55	*74,19*	**215**	2128	**212**	2101	**206**	2081	**208**	2114	**201**	2098
56	*68,27*	**248**	102	**241**	97	**222**	99	**239**	100	**227**	96
57	*15,91*	**111**	124	**105**	117	**99**	118	**101**	115	**97**	114
58	*5,33*	**59**	32	**56**	33	**53**	29	**52**	30	**51**	26
59	*40,21*	**97**	381	**92**	369	**87**	367	**91**	373	**88**	358
60	*108,73*	**458**	2439	**448**	2402	**444**	2388	**447**	2419	**441**	2406
61	*60,49*	**341**	312	**336**	301	**321**	296	**338**	303	**323**	297
62	*102,53*	**490**	2475	**483**	2452	**477**	2448	**480**	2459	**474**	2440
63	*15,67*	**202**	120	**197**	116	**192**	110	**199**	118	**191**	114
64	*80,24*	**324**	2100	**315**	2079	**304**	2074	**316**	2080	**308**	2076

Tabela A.4 – Resultados GRASP com *Path-Relinking* – Grafo Original para obtenção de 3 *clusters*

		GRASP COM PATH-RELINKING									
		GPR1		**GPR2**		**GPR3**		**GPR4**		**GPR5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**28**	121	**19**	**98**	**17**	108	**20**	114	**16**	107
2	*39,08*	**35**	210	**32**	**205**	**21**	174	**25**	208	**19**	194
3	*5,72*	**30**	30	**28**	**18**	**15**	25	**19**	30	**18**	26
4	*26,14*	**41**	174	**33**	**148**	**29**	117	**32**	163	**24**	121
5	*7,35*	**19**	61	**16**	**35**	7	30	**11**	65	8	31
6	*6,41*	**18**	62	0	**23**	4	31	9	56	0	30
7	*6,17*	**12**	56	0	**21**	0	28	5	40	0	31
8	*4,96*	6	29	0	**20**	2	24	3	30	1	22
9	*14,35*	**33**	87	4	**60**	**14**	53	**19**	69	**15**	58
10	*19,78*	**31**	126	**27**	**92**	**12**	90	**23**	112	**18**	88
11	*10,27*	**39**	95	**34**	**47**	**25**	41	**31**	84	**24**	40
12	*17,33*	**18**	110	**16**	**82**	**12**	73	**15**	108	8	70
13	*7,42*	**25**	81	**21**	**54**	8	55	**17**	71	**12**	32
14	*5,94*	**49**	75	**43**	**42**	**17**	29	**38**	63	**26**	30
15	*18,53*	**38**	82	**27**	**95**	**15**	74	**20**	101	**18**	79
16	*7,09*	**42**	74	**36**	35	**21**	29	**26**	63	**22**	30
17	*10,81*	**51**	97	**45**	53	**28**	46	**38**	88	**20**	41
18	*6,33*	**27**	60	**19**	30	0	29	6	45	1	33
19	*57,14*	**42**	1574	**35**	1519	**26**	1433	**39**	1502	**36**	1423
20	*49,02*	**74**	1524	**70**	1504	**57**	1391	**68**	1475	**64**	1402
21	*70,51*	**119**	1699	**96**	1591	**69**	1508	**92**	1529	**62**	1496
22	*22,17*	**14**	124	**11**	87	5	93	9	117	1	102
23	*6,39*	**13**	74	9	27	0	28	0	49	2	34
24	*6,30*	10	61	8	29	1	28	6	41	0	32
25	*82,44*	**14**	54	5	20	2	27	4	42	1	25
26	*12,68*	**115**	483	**107**	381	**93**	459	**99**	480	**82**	420

27	*3,27*	**61**	111	**54**	52	**44**	30	**48**	99	**41**	42
28	*25,89*	4	36	2	29	0	21	0	28	0	19
29	*26,01*	**57**	155	**48**	139	**26**	154	**47**	146	**39**	117
30	*15,48*	**66**	189	**52**	141	**27**	183	**39**	174	**27**	128
31	*78,30*	**49**	104	**33**	73	9	69	**22**	81	**14**	62
32	*6,41*	**123**	1886	**99**	1875	**77**	1682	**85**	1776	**78**	1699
33	*20,08*	**15**	67	**14**	31	0	29	4	47	1	32
34	*8,15*	**38**	69	**27**	28	**16**	35	**23**	49	**11**	24
35	*25,44*	**37**	136	**36**	90	10	93	**28**	114	**14**	91
36	*37,83*	**56**	112	**52**	100	**15**	117	**29**	146	**11**	114
37	*56,91*	**61**	199	**44**	168	**29**	172	**36**	187	**23**	159
38	*53,49*	**94**	1514	**67**	1507	**57**	1409	**62**	1493	**54**	1415
39	*73,06*	**85**	1536	**68**	1518	**50**	1401	**59**	1499	**48**	1418
40	*5,37*	**14**	42	6	20	0	23	3	30	0	27
41	*19,24*	**32**	109	**23**	81	6	80	**14**	97	7	75
42	*25,67*	**30**	158	**19**	137	**17**	116	**21**	144	**15**	112
43	*93,18*	**176**	2012	**140**	2006	**128**	1813	**145**	1930	**130**	1827
44	*6,53*	**23**	73	**19**	44	3	29	6	57	4	32
45	*39,41*	**61**	225	**58**	183	**40**	207	**48**	212	**36**	198
46	*24,37*	**49**	144	**36**	122	**29**	119	**31**	138	**28**	115
47	*12,99*	**59**	148	**44**	71	**28**	74	**35**	126	**30**	48
48	*45,78*	**78**	299	**61**	241	**40**	248	**52**	273	**43**	235
49	*16,04*	**32**	93	**28**	82	**23**	69	**31**	97	**25**	66
50	*26,07*	**41**	165	**37**	148	**29**	132	**30**	159	**24**	128
51	*98,16*	**124**	2017	**118**	2010	**93**	1925	**104**	2008	**99**	1914
52	*3,08*	9	34	0	20	0	27	0	32	0	22
53	*38,50*	**51**	196	**45**	164	**28**	173	**33**	204	**20**	182
54	*22,01*	**40**	123	**28**	89	**17**	98	**24**	105	**15**	97
55	*74,19*	**78**	1752	**74**	1697	**59**	1525	**78**	1683	**63**	1514
56	*68,27*	**73**	1590	**68**	1534	**47**	1442	**67**	1517	**55**	1458
57	*15,91*	**33**	98	**34**	83	**21**	64	**29**	91	**22**	69
58	*5,33*	**12**	35	4	26	**4**	33	0	27	0	25
59	*40,21*	**58**	341	**53**	278	**30**	290	**40**	333	**28**	208
60	*108,73*	**137**	2045	**122**	2028	**119**	1973	**125**	2015	**109**	1914
61	*60,49*	**92**	364	**88**	281	**63**	349	**69**	311	**57**	336
62	*102,53*	**153**	2033	**148**	2015	**115**	1985	**134**	2029	**112**	1938
63	*15,67*	**64**	108	**51**	75	**24**	60	**37**	99	**16**	63
64	*80,24*	**102**	1904	**99**	1899	**90**	1735	**92**	1894	**81**	1719

Tabela A.5 – Resultados GRASP com *Path-Relinking* – AGM *Kruskal modificado* para obtenção de 3 *clusters*

		GRASP COM PATH-RELINKING									
		PR_AGMM1		**PR_AGMM2**		**PR_AGMM3**		**PR_AGMM4**		**PR_AGMM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**97**	108	**98**	105	**90**	104	**93**	107	**91**	106
2	*39,08*	**116**	114	**119**	111	**114**	110	**115**	112	**113**	115
3	*5,72*	**55**	41	**53**	39	**50**	37	**53**	40	**51**	33
4	*26,14*	**69**	137	**67**	127	**64**	128	0	132	**63**	124
5	*7,35*	**51**	48	**54**	47	**49**	46	**50**	49	**48**	48
6	*6,41*	**56**	37	**58**	38	0	36	**55**	38	**51**	33
7	*6,17*	**49**	49	**45**	43	**43**	41	**48**	45	**44**	40
8	*4,96*	**39**	44	**39**	41	**33**	42	**37**	43	**36**	44
9	*14,35*	**102**	97	**101**	93	**95**	92	**100**	94	**97**	90
10	*19,78*	**88**	112	**86**	109	**80**	111	**83**	110	**81**	110
11	*10,27*	**118**	118	**123**	114	**109**	115	**116**	116	0	117
12	*17,33*	**61**	51	**60**	50	**58**	54	**60**	52	**59**	53
13	*7,42*	**59**	69	**57**	67	**52**	66	0	68	**53**	64
14	*5,94*	**53**	100	**52**	83	**44**	91	**50**	94	**47**	90
15	*18,53*	**94**	83	**95**	80	**91**	83	**92**	85	**88**	84
16	*7,09*	**98**	104	0	101	**92**	100	**97**	102	**93**	101
17	*10,81*	**67**	75	**69**	71	**65**	72	**66**	73	**63**	71
18	*6,33*	**129**	37	**134**	34	**128**	35	**124**	36	**121**	36
19	*57,14*	**184**	1944	**181**	1937	**173**	1931	**179**	1942	**175**	1942

20	***49,02***	**153**	2241	**155**	2239	**147**	2125	**150**	2240	**148**	2118
21	***70,51***	**144**	2095	**146**	2075	**144**	2079	**141**	2083	**143**	2074
22	***22,17***	**78**	41	**77**	42	**75**	41	0	44	**73**	42
23	***6,39***	**89**	37	**88**	33	**80**	30	**86**	35	**81**	34
24	***6,30***	**45**	39	**46**	31	**39**	37	**44**	38	**42**	36
25	***82,44***	**177**	40	**179**	32	**174**	35	**176**	37	**171**	30
26	***12,68***	**208**	127	**207**	119	**197**	120	**206**	123	0	118
27	***3,27***	**62**	25	**63**	21	**59**	20	**61**	22	**59**	19
28	***25,89***	**88**	123	**86**	125	**84**	126	**85**	124	**82**	125
29	***26,01***	**119**	136	**122**	131	**116**	132	**117**	135	**114**	134
30	***15,48***	**98**	152	**99**	138	**91**	136	**97**	141	**92**	139
31	***78,30***	**141**	2157	**140**	2122	**135**	2124	**138**	2149	**133**	2133
32	***6,41***	**137**	104	**135**	95	**127**	96	**132**	102	**128**	98
33	***20,08***	**119**	41	**124**	37	**107**	38	**112**	39	**108**	36
34	***8,15***	**85**	102	**84**	93	**79**	94	**83**	97	**78**	92
35	***25,44***	**93**	88	**92**	81	0	85	**91**	86	**89**	83
36	***37,83***	**151**	127	**150**	112	**142**	127	**148**	125	**144**	124
37	***56,91***	**90**	209	**88**	183	**85**	193	**86**	198	**83**	188
38	***53,49***	**148**	2033	**144**	2014	**131**	2011	**140**	2025	**136**	2008
39	***73,06***	**209**	2082	**203**	2061	**184**	2044	**198**	2073	**183**	2021
40	***5,37***	**48**	39	**47**	37	**41**	35	**46**	38	**42**	37
41	***19,24***	**93**	53	**95**	54	**86**	53	**91**	56	**88**	51
42	***25,67***	**79**	104	**83**	102	**75**	104	**77**	108	**71**	105
43	***93,18***	**281**	2118	**280**	2103	**278**	2099	**279**	2113	**276**	2100
44	***6,53***	**65**	49	**64**	46	0	49	**63**	48	**61**	45
45	***39,41***	**118**	248	**116**	227	**111**	228	**112**	239	**110**	224
46	***24,37***	**201**	151	**208**	138	**191**	139	**199**	144	**193**	141
47	***12,99***	**99**	93	**100**	90	**84**	88	**97**	91	**86**	85
48	***45,78***	**104**	209	**103**	184	**95**	173	**100**	192	**98**	174
49	***16,04***	**77**	114	**77**	115	**72**	117	**76**	116	**73**	118
50	***26,07***	**75**	136	**78**	124	**69**	125	0	132	**68**	121
51	***98,16***	**283**	2101	**282**	2088	**274**	2083	**281**	2096	**277**	2087
52	***3,08***	**45**	23	**42**	21	**37**	20	**40**	22	**39**	20
53	***38,50***	**99**	218	**99**	219	**92**	215	**97**	220	**95**	223
54	***22,01***	**71**	157	**74**	148	**68**	149	**70**	154	0	152
55	***74,19***	**183**	2004	**181**	1993	**169**	1987	**180**	1998	**173**	1985
56	***68,27***	**209**	101	**210**	89	**201**	92	**204**	93	**202**	91
57	***15,91***	**77**	109	**76**	104	**70**	105	**75**	106	**71**	103
58	***5,33***	**42**	25	**44**	23	**39**	25	**40**	24	**39**	22
59	***40,21***	**45**	392	**43**	395	**38**	396	**42**	393	**37**	392
60	***108,73***	**394**	2428	**391**	2415	0	2408	**390**	2419	**385**	2410
61	***60,49***	**168**	319	**166**	313	**159**	315	**167**	316	0	317
62	***102,53***	**404**	2487	**408**	2468	**396**	2474	**402**	2478	**397**	2469
63	***15,67***	**142**	114	**144**	106	**128**	109	**137**	110	**132**	108
64	***80,24***	**269**	2076	**267**	2048	**261**	2031	**255**	2063	**258**	2032

Tabela A.6 – Resultados GRASP com *Path-Relinking* – AGM *Kruskal* padrão para obtenção de 3 *clusters*

		GRASP COM PATH-RELINKING									
		PR_AGM1		**PR_AGM2**		**PR_AGM3**		**PR_AGM4**		**PR_AGM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**104**	109	**101**	105	**93**	101	**98**	106	**95**	103
2	***39,08***	**119**	115	**120**	111	**117**	110	**118**	112	**116**	114
3	***5,72***	**65**	40	**62**	35	**52**	36	**59**	38	**54**	31
4	***26,14***	**77**	53	**74**	48	**67**	42	**72**	53	**66**	42
5	***7,35***	**59**	51	**57**	47	**52**	45	**54**	49	**51**	44
6	***6,41***	**62**	37	**59**	36	**51**	34	**58**	37	**53**	31
7	***6,17***	**55**	47	**53**	43	**44**	40	**51**	45	**49**	38
8	***4,96***	**54**	47	**48**	44	**35**	41	**46**	46	**38**	42
9	***14,35***	**107**	96	**106**	92	**101**	88	**105**	94	**99**	86
10	***19,78***	**89**	114	**88**	111	**82**	108	**86**	113	**83**	107
11	***10,27***	**123**	119	**122**	117	**116**	116	**119**	118	**117**	115
12	***17,33***	**70**	52	**68**	50	**67**	51	**65**	54	**64**	50

13	*7,42*	**66**	68	**62**	65	**53**	63	**61**	67	**55**	61
14	*5,94*	**58**	99	**55**	87	**52**	84	**53**	91	**49**	83
15	*18,53*	**99**	88	**93**	83	**90**	82	**94**	86	**91**	81
16	*7,09*	**106**	102	**104**	99	**98**	96	**101**	100	**95**	98
17	*10,81*	**79**	73	**77**	68	**71**	62	**75**	70	**68**	67
18	*6,33*	**134**	36	**132**	32	**127**	31	**130**	35	**125**	33
19	*57,14*	**191**	1931	**188**	1922	**185**	1907	**183**	1927	**181**	1915
20	*49,02*	**160**	2178	**158**	2110	**150**	2083	**156**	2138	**151**	2094
21	*70,51*	**157**	2075	**153**	2061	**149**	2062	**150**	2067	**148**	2055
22	*22,17*	**86**	44	**80**	40	**76**	41	**81**	42	**77**	40
23	*6,39*	**99**	36	**97**	31	0	27	**94**	33	**89**	29
24	*6,30*	**49**	37	**48**	32	**44**	33	**47**	35	**46**	31
25	*82,44*	**187**	39	**183**	30	**179**	28	**180**	32	**178**	26
26	*12,68*	**128**	125	**128**	118	**125**	117	**127**	121	**122**	109
27	*3,27*	**73**	22	**74**	17	**71**	15	**69**	19	**67**	13
28	*25,89*	**90**	124	**88**	121	**85**	120	**87**	122	**86**	118
29	*26,01*	**127**	135	**123**	130	**118**	132	**121**	131	**116**	129
30	*15,48*	**108**	142	**104**	131	**95**	128	**99**	135	**97**	130
31	*78,30*	**149**	2140	**144**	2116	**132**	2075	**139**	2128	0	2087
32	*6,41*	**145**	101	**142**	96	**136**	91	**141**	97	**133**	93
33	*20,08*	**120**	39	**117**	35	**112**	36	**115**	37	**110**	33
34	*8,15*	**94**	98	**91**	91	**87**	84	**83**	96	**81**	88
35	*25,44*	**101**	87	**98**	82	**96**	85	**99**	85	**95**	84
36	*37,83*	**174**	125	**163**	120	**151**	119	**158**	121	**153**	116
37	*56,91*	**97**	194	**95**	179	**89**	186	**91**	182	**88**	177
38	*53,49*	**151**	2021	**146**	2007	**137**	1999	**142**	2011	**140**	1995
39	*73,06*	**204**	2073	**196**	2054	**189**	2033	**193**	2068	**188**	2003
40	*5,37*	**55**	38	**53**	35	**51**	32	**48**	36	**46**	34
41	*19,24*	**104**	54	**99**	48	**92**	50	**96**	52	**93**	46
42	*25,67*	**85**	103	**84**	100	**81**	99	**83**	102	**76**	97
43	*93,18*	**307**	2116	**293**	2114	**289**	2085	**281**	2105	**283**	2089
44	*6,53*	**70**	48	**71**	45	**63**	48	**69**	47	**66**	41
45	*39,41*	**121**	236	**119**	218	**117**	219	**115**	229	**111**	216
46	*24,37*	**222**	144	**215**	136	**196**	138	**208**	140	**199**	137
47	*12,99*	**107**	89	**95**	83	**93**	77	**92**	86	**89**	74
48	*45,78*	**121**	190	**114**	181	**101**	155	**108**	184	**103**	161
49	*16,04*	**85**	123	**83**	114	**81**	110	**79**	115	**77**	112
50	*26,07*	**84**	122	**79**	115	**73**	108	**75**	119	**72**	104
51	*98,16*	**315**	2086	**293**	2071	**281**	2059	**308**	2075	**285**	2063
52	*3,08*	**52**	24	**49**	20	**41**	19	**47**	21	0	18
53	*38,50*	**114**	221	**112**	218	**111**	216	**112**	219	**110**	217
54	*22,01*	**88**	164	**85**	153	**79**	144	**81**	158	**78**	147
55	*74,19*	**221**	1993	**202**	1971	0	1968	**194**	1984	**183**	1960
56	*68,27*	**234**	96	**227**	90	**211**	84	**215**	91	**209**	88
57	*15,91*	**90**	108	**86**	103	**78**	102	**83**	104	**76**	101
58	*5,33*	**52**	29	**47**	21	**41**	26	0	22	**43**	19
59	*40,21*	**55**	402	**49**	390	**42**	389	**46**	391	**38**	387
60	*108,73*	**412**	2419	**404**	2406	**386**	2397	**399**	2407	**391**	2401
61	*60,49*	**186**	320	**183**	318	**179**	305	**181**	321	**173**	309
62	*102,53*	**451**	2473	**428**	2460	**400**	2451	**1416**	2466	**402**	2444
63	*15,67*	**157**	111	**153**	108	**138**	106	**150**	109	**143**	103
64	*80,24*	**296**	2056	**284**	2039	**269**	2023	**271**	2041	**265**	2028

Tabela A.7 – Resultados GRASP com VND – Grafo Original para obtenção de 3 *clusters*

		GRASP COM VND									
		GVND1		**GVND2**		**GVND3**		**GVND4**		**GVND5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**25**	116	**18**	93	**14**	111	**16**	127	**14**	110
2	*39,08*	**33**	214	**29**	208	**20**	189	**22**	219	**17**	203
3	*5,72*	**24**	34	**21**	19	**12**	29	**17**	33	**15**	28
4	*26,14*	**36**	179	**29**	151	**22**	127	**24**	174	**21**	140
5	*7,35*	**17**	83	**12**	39	6	34	7	68	7	35

6	***6,41***	**14**	74	0	28	0	30	**11**	54	3	31
7	***6,17***	10	61	0	23	1	27	4	49	0	27
8	***4,96***	5	28	2	22	0	24	0	31	0	26
9	***14,35***	**23**	84	0	61	**15**	50	**20**	75	**12**	57
10	***19,78***	**29**	132	**20**	97	**13**	93	**18**	114	**15**	89
11	***10,27***	**34**	97	**31**	45	**21**	40	**27**	81	**20**	44
12	***17,33***	**16**	112	**13**	88	5	70	**12**	106	6	71
13	***7,42***	**22**	87	**15**	51	0	33	**14**	79	9	35
14	***5,94***	**45**	77	**38**	43	**12**	30	**29**	66	**17**	32
15	***18,53***	**37**	81	**29**	94	**14**	79	**18**	102	**16**	80
16	***7,09***	**46**	78	**41**	37	**23**	31	**25**	67	**19**	34
17	***10,81***	**38**	109	**35**	48	**24**	44	**31**	89	**21**	45
18	***6,33***	**51**	72	**24**	25	0	27	8	51	0	27
19	***57,14***	**49**	1997	**37**	1980	**27**	1932	**33**	1942	**25**	1947
20	***49,02***	**72**	2103	**71**	2059	**55**	1985	**64**	1997	**59**	1945
21	***70,51***	**99**	1948	**76**	1934	**68**	1940	**45**	1919	**51**	1994
22	***22,17***	**15**	136	**14**	93	4	95	8	122	0	99
23	***6,39***	**12**	77	7	28	0	30	4	53	1	31
24	***6,30***	8	71	6	23	2	27	0	49	0	28
25	***82,44***	10	68	3	21	1	26	0	48	0	27
26	***12,68***	**103**	496	**92**	382	**80**	463	**88**	491	**83**	421
27	***3,27***	**64**	115	**51**	53	**39**	33	**43**	101	**34**	45
28	***25,89***	1	34	0	28	0	27	0	29	0	24
29	***26,01***	**55**	167	**48**	132	**27**	144	**45**	164	**32**	123
30	***15,48***	**72**	206	**69**	148	**28**	182	**36**	190	**24**	140
31	***78,30***	**42**	101	**33**	74	10	63	**18**	88	**11**	64
32	***6,41***	**114**	2492	**97**	2462	**74**	2457	**81**	2477	**75**	2409
33	***20,08***	**11**	75	10	25	3	31	3	53	2	27
34	***8,15***	**52**	147	**21**	91	**15**	95	**18**	117	10	93
35	***25,44***	**29**	93	**29**	50	8	43	**20**	84	**12**	39
36	***37,83***	**44**	124	**48**	101	**11**	121	**33**	152	**16**	118
37	***56,91***	**56**	203	**46**	172	**27**	184	**42**	199	**29**	167
38	***53,49***	**89**	2022	**65**	1991	**48**	1931	**59**	1927	**47**	1981
39	***73,06***	**73**	2019	**62**	1972	**40**	1987	**51**	1909	**44**	1963
40	***5,37***	**12**	31	7	21	0	27	0	33	0	28
41	***19,24***	**28**	117	**21**	86	3	84	**11**	99	8	84
42	***25,67***	**31**	164	**17**	109	**18**	119	**22**	157	**13**	118
43	***93,18***	**165**	2088	**143**	2064	**125**	2053	**138**	2078	**129**	2004
44	***6,53***	**18**	81	**16**	37	4	30	**2**	62	2	31
45	***39,41***	**59**	235	**59**	190	**41**	212	**45**	224	**37**	201
46	***24,37***	**40**	158	**35**	129	**21**	114	**29**	147	**25**	116
47	***12,99***	**42**	142	**38**	69	**27**	73	**32**	127	**27**	49
48	***45,78***	**53**	303	**54**	243	**42**	259	**48**	281	**39**	237
49	***16,04***	**34**	90	**27**	89	**26**	65	**29**	94	**22**	67
50	***26,07***	**38**	179	**34**	142	**20**	121	**25**	167	**23**	140
51	***98,16***	**112**	2598	**111**	2583	**91**	2549	**101**	2590	**96**	2557
52	***3,08***	4	49	0	21	0	28	0	35	0	24
53	***38,50***	**47**	201	**36**	176	**22**	171	**31**	204	**19**	183
54	***22,01***	**38**	129	**29**	91	**13**	102	**22**	119	**11**	101
55	***74,19***	**75**	2434	**71**	2403	**56**	2394	**60**	2399	**58**	2390
56	***68,27***	**69**	1980	**66**	1961	**48**	1947	**54**	1957	**51**	1936
57	***15,91***	**35**	99	**32**	89	**19**	68	**27**	92	**20**	67
58	***5,33***	8	29	1	21	2	39	0	29	0	28
59	***40,21***	**56**	331	**49**	281	**29**	293	**41**	339	**27**	211
60	***108,73***	**123**	2715	**114**	2683	**118**	2641	**112**	2670	**104**	2644
61	***60,49***	**85**	358	**86**	297	**59**	353	**67**	309	**56**	338
62	***102,53***	**137**	2710	**137**	2671	**111**	2657	**128**	2699	**113**	2612
63	***15,67***	**49**	114	**48**	81	**23**	63	**41**	97	**18**	65
64	***80,24***	**97**	2071	**91**	2052	**92**	2028	**88**	2051	**84**	2012

Tabela A.8 – Resultados GRASP com VND – AGM *Kruskal modificado* para obtenção de 3 *clusters*

	GRASP COM VND

		V_AGMM1		V_AGMM2		V_AGMM3		V_AGMM4		V_AGMM5	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**93**	115	**92**	109	**85**	104	**90**	112	**88**	102
2	***39,08***	**112**	127	**111**	114	**103**	112	**110**	124	**104**	107
3	***5,72***	**48**	40	**46**	37	**43**	35	**45**	44	0	33
4	***26,14***	**54**	149	**52**	133	**49**	127	**50**	141	0	124
5	***7,35***	**43**	50	**42**	46	**35**	39	**41**	48	**38**	44
6	***6,41***	**51**	42	**51**	38	**47**	36	**50**	39	**49**	37
7	***6,17***	**44**	51	**43**	52	**40**	49	**42**	50	**39**	48
8	***4,96***	**38**	46	**38**	49	**35**	47	**37**	48	**36**	47
9	***14,35***	**93**	104	**91**	108	0	106	**90**	107	0	108
10	***19,78***	**85**	118	**84**	113	**79**	114	**82**	115	**76**	112
11	***10,27***	**114**	125	**112**	118	**110**	115	**112**	122	**108**	119
12	***17,33***	**54**	53	**51**	52	**49**	51	0	54	**48**	50
13	***7,42***	**51**	71	**50**	70	**45**	67	**50**	70	**47**	66
14	***5,94***	**48**	101	**47**	97	**42**	93	**46**	99	**44**	94
15	***18,53***	**89**	86	**87**	82	**81**	81	**88**	85	**85**	79
16	***7,09***	**94**	103	**93**	100	**89**	99	**92**	102	**90**	101
17	***10,81***	**59**	71	**59**	64	**51**	67	**57**	68	**54**	69
18	***6,33***	**123**	40	**121**	43	**117**	38	**120**	44	**118**	39
19	***57,14***	**171**	1959	**170**	1938	**163**	1925	**168**	1947	**166**	1921
20	***49,02***	**149**	2307	**147**	2294	0	2283	**144**	2304	0	2279
21	***70,51***	**136**	2115	**135**	2082	**130**	2065	**133**	2093	**131**	2068
22	***22,17***	**75**	43	**74**	41	**72**	40	**74**	42	**71**	40
23	***6,39***	**87**	38	**86**	39	**83**	38	0	40	**85**	36
24	***6,30***	**44**	40	**43**	44	**40**	43	**42**	45	**40**	41
25	***82,44***	**162**	41	**161**	37	**155**	36	**159**	38	**157**	33
26	***12,68***	**191**	132	**190**	120	**181**	123	**189**	127	**183**	126
27	***3,27***	**58**	27	0	25	**51**	24	**54**	26	**52**	23
28	***25,89***	**87**	136	**85**	131	**80**	129	**83**	135	**81**	127
29	***26,01***	**117**	148	**115**	142	0	140	**114**	144	**110**	141
30	***15,48***	**93**	171	**90**	164	**88**	166	**92**	169	**88**	165
31	***78,30***	**129**	2164	**128**	2138	**119**	2147	**127**	2153	**124**	2150
32	***6,41***	**121**	108	**119**	92	**114**	93	**120**	94	**117**	91
33	***20,08***	**112**	45	**111**	40	**106**	41	**110**	42	**107**	40
34	***8,15***	**84**	109	**84**	87	**83**	85	**80**	89	**82**	86
35	***25,44***	**87**	92	**88**	84	**80**	83	**85**	87	**81**	85
36	***37,83***	**145**	133	**144**	116	**141**	137	0	126	**142**	129
37	***56,91***	**88**	219	**92**	204	**82**	200	**86**	213	**83**	197
38	***53,49***	**142**	2053	**139**	2039	**133**	2021	**137**	2041	**132**	2036
39	***73,06***	**203**	2108	**210**	2088	**193**	2064	**201**	2093	**197**	2073
40	***5,37***	**47**	41	**46**	39	**40**	44	**44**	40	**42**	42
41	***19,24***	**85**	57	**84**	58	0	57	**82**	59	**78**	55
42	***25,67***	**78**	106	**77**	114	**70**	102	**75**	108	**73**	100
43	***93,18***	**261**	2124	**264**	2139	**255**	2130	**258**	2142	0	2133
44	***6,53***	**59**	51	**63**	49	**56**	52	**56**	50	**53**	55
45	***39,41***	**111**	257	**115**	264	**109**	261	0	258	**108**	263
46	***24,37***	**194**	163	**192**	189	**188**	177	**189**	174	**187**	179
47	***12,99***	**93**	108	**111**	106	**104**	109	**106**	110	**103**	114
48	***45,78***	**98**	212	**97**	210	**92**	214	**95**	209	**93**	211
49	***16,04***	**63**	121	**64**	117	**58**	119	**62**	118	**59**	116
50	***26,07***	**74**	147	**73**	139	**70**	145	**71**	141	**70**	140
51	***98,16***	**266**	2112	**265**	2103	**260**	2091	**263**	2107	**261**	2087
52	***3,08***	**39**	24	**38**	22	**33**	24	**37**	23	**34**	22
53	***38,50***	**91**	227	**93**	220	**86**	218	**90**	224	**88**	217
54	***22,01***	**58**	169	**59**	152	**57**	197	**58**	157	**57**	144
55	***74,19***	**164**	2012	**161**	2001	**155**	1993	**162**	2008	**157**	1987
56	***68,27***	**203**	116	**208**	118	**197**	119	**202**	117	**199**	115
57	***15,91***	**69**	118	**73**	111	**66**	113	**65**	114	**66**	116
58	***5,33***	**38**	28	**37**	27	0	28	**35**	29	**34**	28
59	***40,21***	**44**	395	**43**	382	**42**	386	**41**	387	**40**	385
60	***108,73***	**391**	2468	**390**	2442	**385**	2440	**389**	2451	**386**	2444
61	***60,49***	**144**	327	**142**	318	**139**	321	**140**	324	**137**	319

62	*102,53*	**399**	2501	**401**	2481	**396**	2485	**397**	2493	**395**	2490
63	*15,67*	**138**	125	**139**	122	**128**	123	**135**	124	**129**	125
64	*80,24*	**251**	2122	**2109**	2101	**2076**	2081	**2100**	2115	**2087**	2084

Tabela A.9 – Resultados GRASP com VND – AGM *Kruskal* padrão para obtenção de 3 *clusters*

		GRASP COM VND									
		V_AGM1		V_AGM2		V_AGM3		V_AGM4		V_AGM5	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**115**	112	**113**	104	**91**	97	**102**	106	**94**	99
2	*39,08*	**114**	123	**112**	121	**104**	108	**109**	122	**103**	103
3	*5,72*	**59**	45	**57**	42	**51**	31	**54**	44	**48**	30
4	*26,14*	**62**	143	**59**	137	**52**	119	**53**	141	**51**	120
5	*7,35*	**73**	51	**64**	46	**59**	38	**55**	48	**56**	41
6	*6,41*	**58**	40	**55**	37	**48**	35	**54**	36	**51**	34
7	*6,17*	**50**	51	**49**	48	**43**	47	**46**	50	**44**	46
8	*4,96*	**44**	48	**41**	45	**38**	46	**39**	47	**38**	43
9	*14,35*	**119**	103	**118**	106	**107**	105	**110**	106	**108**	104
10	*19,78*	**90**	115	**88**	113	**81**	112	**84**	111	**80**	110
11	*10,27*	**122**	124	**119**	120	**114**	113	**117**	117	**110**	116
12	*17,33*	**63**	50	**61**	52	**54**	50	**53**	51	**52**	48
13	*7,42*	**70**	69	**64**	67	0	65	**59**	68	**53**	62
14	*5,94*	**51**	98	**49**	91	**39**	90	**48**	95	**41**	83
15	*18,53*	**103**	84	**94**	83	**83**	80	**88**	79	0	76
16	*7,09*	**99**	101	**99**	100	**96**	98	**97**	99	**95**	97
17	*10,81*	**68**	70	**66**	67	**51**	65	**61**	63	**57**	66
18	*6,33*	**129**	39	**128**	42	**119**	35	**126**	41	**122**	37
19	*57,14*	**173**	1944	**174**	1922	**171**	1904	**170**	1927	**168**	1903
20	*49,02*	**161**	2305	**155**	2295	**149**	2261	**153**	2301	**150**	2254
21	*70,51*	**150**	2091	**146**	2071	**138**	2059	**144**	2075	**140**	2055
22	*22,17*	**83**	40	**79**	73	**74**	39	**77**	65	**73**	38
23	*6,39*	**102**	41	**99**	38	**93**	32	**96**	39	**91**	33
24	*6,30*	**51**	46	**48**	44	**45**	41	**47**	45	**46**	40
25	*82,44*	**176**	39	**173**	37	**161**	33	**168**	37	**164**	32
26	*12,68*	**211**	130	**202**	124	**195**	122	**199**	119	**193**	121
27	*3,27*	**63**	26	**58**	25	**53**	22	**61**	24	**55**	23
28	*25,89*	**90**	135	**89**	131	**81**	119	**89**	133	**84**	118
29	*26,01*	**124**	141	**118**	137	**114**	127	0	139	**112**	129
30	*15,48*	**100**	168	**99**	165	**93**	164	**96**	166	**89**	162
31	*78,30*	**141**	2148	**137**	2133	**133**	2119	**129**	2135	0	2126
32	*6,41*	**127**	107	**122**	92	**115**	95	**121**	100	**116**	88
33	*20,08*	**115**	41	**112**	41	**110**	40	**113**	37	**109**	38
34	*8,15*	**96**	94	**94**	88	**86**	82	**87**	89	**84**	83
35	*25,44*	**89**	90	**89**	85	**82**	81	**88**	86	**85**	82
36	*37,83*	**157**	129	**153**	123	**141**	124	**151**	125	**144**	117
37	*56,91*	**98**	215	**96**	210	**87**	193	**94**	212	**89**	185
38	*53,49*	**146**	2041	**142**	2032	**138**	2016	**140**	2037	**135**	2019
39	*73,06*	**219**	2088	**213**	2080	**209**	2060	**210**	2074	**204**	2061
40	*5,37*	**47**	47	**47**	43	**43**	38	**46**	45	**44**	37
41	*19,24*	**89**	60	**88**	55	**77**	54	**84**	56	**79**	52
42	*25,67*	**77**	104	**76**	98	**72**	100	**75**	103	**74**	93
43	*93,18*	**304**	2138	**293**	2116	**275**	2112	**289**	2124	**271**	2104
44	*6,53*	**63**	64	0	60	**59**	50	**60**	62	**58**	51
45	*39,41*	**128**	260	**122**	259	**116**	255	**121**	261	**114**	257
46	*24,37*	**203**	184	**194**	171	**192**	173	**192**	180	**191**	176
47	*12,99*	**112**	114	**114**	111	**109**	106	**112**	112	**108**	108
48	*45,78*	**104**	215	**99**	210	**94**	212	**99**	213	**95**	209
49	*16,04*	**74**	120	**71**	119	**63**	117	**70**	115	**67**	117
50	*26,07*	**88**	144	**85**	140	**77**	142	**83**	137	**79**	135
51	*98,16*	**303**	2108	**291**	2093	**274**	2071	**289**	2095	**273**	2059
52	*3,08*	**44**	24	**41**	21	**37**	23	**40**	21	**36**	20
53	*38,50*	**98**	225	**96**	215	**91**	211	**88**	218	**95**	209
54	*22,01*	**69**	163	**67**	148	**63**	179	**60**	151	**64**	132

55	*74,19*	**180**	2009	**179**	1996	**165**	1977	**174**	2000	**163**	1965
56	*68,27*	**229**	122	**223**	118	**209**	114	**217**	119	**208**	112
57	*15,91*	**82**	119	**75**	110	**69**	107	**71**	114	**68**	108
58	*5,33*	**41**	28	**37**	27	**36**	26	**38**	28	**36**	26
59	*40,21*	**49**	391	**46**	384	**42**	383	**47**	386	**44**	379
60	*108,73*	**408**	2465	**403**	2448	**404**	2417	**402**	2453	**391**	2420
61	*60,49*	**153**	319	**147**	308	**145**	299	**144**	312	**142**	296
62	*102,53*	**422**	2490	**421**	2483	**410**	2464	**413**	2487	**409**	2475
63	*15,67*	**144**	137	**142**	129	**133**	122	**140**	131	**137**	124
64	*80,24*	**2173**	2119	**2158**	2100	**2142**	2059	**2150**	2114	**2141**	2061

Tabela A.10 – Resultados GRASP com RVND – Grafo Original para obtenção de 3 *clusters*

		GRASP COM RVND									
		GRVND1		**GRVND2**		**GRVND3**		**GRVND4**		**GRVND5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**28**	124	**19**	91	**13**	104	**15**	114	**12**	108
2	*39,08*	**31**	209	**27**	202	**12**	161	**20**	185	**13**	159
3	*5,72*	**20**	33	**16**	30	10	27	**14**	32	**11**	23
4	*26,14*	**34**	152	**28**	144	**17**	131	**22**	167	**16**	132
5	*7,35*	**18**	53	**14**	47	0	32	8	59	3	31
6	*6,41*	**13**	48	9	36	2	29	4	41	2	30
7	*6,17*	**11**	57	7	29	0	18	1	32	0	24
8	*4,96*	4	26	0	22	0	12	0	28	**20**	18
9	*14,35*	**19**	109	**16**	102	9	98	**11**	84	10	104
10	*19,78*	**32**	87	**21**	74	**16**	54	**18**	76	**14**	69
11	*10,27*	**39**	95	**33**	89	**15**	87	**24**	93	**18**	101
12	*17,33*	**18**	61	**14**	53	3	49	9	59	5	48
13	*7,42*	**19**	77	**12**	68	6	53	10	75	2	67
14	*5,94*	**42**	117	**39**	114	**18**	84	**35**	121	**14**	108
15	*18,53*	**31**	89	**27**	77	**13**	68	**19**	87	**12**	73
16	*7,09*	**44**	93	**30**	89	**14**	71	**21**	92	**14**	80
17	*10,81*	**39**	91	**28**	74	**12**	55	**26**	77	**15**	69
18	*6,33*	**53**	65	**39**	33	0	20	0	36	0	24
19	*57,14*	**38**	2338	**33**	2272	**21**	2264	**30**	2281	**22**	2267
20	*49,02*	**74**	2391	**69**	2375	**58**	2359	**68**	2379	**54**	2373
21	*70,51*	**123**	2126	**78**	109	**67**	2093	**42**	2112	**58**	2098
22	*22,17*	**15**	85	**12**	43	3	26	**11**	48	0	39
23	*6,39*	10	62	8	34	0	27	3	40	0	28
24	*6,30*	9	53	7	31	0	20	0	34	0	23
25	*82,44*	**12**	59	5	27	0	19	0	32	0	23
26	*12,68*	**106**	488	**98**	479	**77**	461	**82**	482	**75**	475
27	*3,27*	**51**	197	**44**	185	**18**	154	**37**	189	**28**	171
28	*25,89*	4	24	0	19	0	14	0	24	0	15
29	*26,01*	**52**	161	**45**	157	**33**	132	**41**	161	**30**	155
30	*15,48*	**73**	229	**54**	182	**18**	151	**37**	187	**19**	186
31	*78,30*	**35**	104	**22**	85	9	62	**16**	85	9	83
32	*6,41*	**104**	2495	**93**	2508	**71**	2443	**82**	2511	**73**	2497
33	*20,08*	8	68	7	29	0	30	0	38	0	26
34	*8,15*	**49**	134	**34**	123	6	117	**17**	122	9	125
35	*25,44*	**28**	87	**12**	94	3	81	**11**	97	6	93
36	*37,83*	**42**	118	**37**	192	**17**	165	**35**	196	**15**	182
37	*56,91*	**51**	243	**43**	277	**23**	259	**41**	283	**24**	271
38	*53,49*	**67**	2315	**74**	2283	**45**	2264	**67**	2292	**34**	2268
39	*73,06*	**68**	2296	**54**	2311	**39**	2298	**49**	2315	**37**	2306
40	*5,37*	9	29	5	28	0	19	0	31	0	20
41	*19,24*	**26**	74	**13**	57	0	55	7	63	6	54
42	*25,67*	**28**	156	0	122	8	112	**21**	128	10	117
43	*93,18*	**159**	2202	**141**	2189	**118**	2173	**142**	2194	**125**	2186
44	*6,53*	**18**	51	**12**	35	1	27	5	43	1	30

45	***39,41***	**56**	294	**44**	281	**31**	264	**36**	287	**32**	274
46	***24,37***	**37**	182	**31**	170	**23**	156	**27**	174	**21**	162
47	***12,99***	**40**	163	**35**	157	**23**	144	**30**	169	**24**	153
48	***45,78***	**55**	342	**49**	330	**40**	316	**41**	331	**37**	327
49	***16,04***	**31**	109	**23**	114	**11**	98	**19**	108	**15**	111
50	***26,07***	**37**	215	**32**	212	**17**	203	**25**	216	**19**	208
51	***98,16***	**114**	2694	**111**	2687	**94**	2675	**102**	2693	**93**	2681
52	***3,08***	2	22	0	18	0	13	0	24	0	14
53	***38,50***	**39**	248	**34**	234	**16**	209	**32**	238	**17**	224
54	***22,01***	**31**	153	**22**	141	**12**	128	**17**	149	9	139
55	***74,19***	**72**	2431	**67**	2429	**54**	2414	**61**	2437	**55**	2422
56	***68,27***	**68**	2435	**63**	2424	**47**	2405	**52**	2428	**48**	2418
57	***15,91***	**33**	148	**28**	130	**16**	116	**24**	124	**17**	119
58	***5,33***	6	39	0	27	0	17	0	31	0	20
59	***40,21***	**51**	371	**47**	368	**22**	363	**42**	370	**23**	361
60	***108,73***	**119**	2779	**110**	2774	**95**	2778	**109**	2789	**96**	2772
61	***60,49***	**84**	364	**73**	351	**50**	337	**63**	353	**52**	348
62	***102,53***	**132**	2783	**119**	2776	**99**	2764	**131**	2784	**102**	2775
63	***15,67***	**48**	151	**36**	147	**15**	142	**38**	152	**15**	136
64	***80,24***	**93**	2093	**88**	2086	**78**	2503	**79**	2518	**77**	2492

Tabela A.11 – Resultados GRASP com RVND – AGM *Kruskal* modificado para obtenção de 3 *clusters*

		GRASP COM RVND									
		RV_AGMM1		**RV_AGMM2**		**RV_AGMM3**		**RV_AGMM4**		**RV_AGMM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**84**	116	**83**	114	**77**	113	**82**	115	**78**	110
2	***39,08***	**102**	132	**100**	127	**93**	124	**98**	129	**95**	125
3	***5,72***	**45**	41	**44**	42	**37**	43	**43**	44	**39**	44
4	***26,14***	**49**	148	**48**	128	**44**	127	**46**	135	0	125
5	***7,35***	**41**	52	**38**	47	**36**	48	**39**	49	0	44
6	***6,41***	**52**	47	**50**	44	0	43	**48**	46	**46**	47
7	***6,17***	**39**	53	**39**	51	**34**	50	**38**	52	**36**	50
8	***4,96***	**37**	47	**36**	41	**30**	44	**35**	46	**32**	45
9	***14,35***	**91**	102	**89**	97	**87**	100	**88**	101	**85**	98
10	***19,78***	**77**	110	**75**	102	**70**	105	**75**	108	**72**	103
11	***10,27***	**102**	128	**101**	117	**95**	118	**100**	119	**97**	114
12	***17,33***	**49**	54	**48**	51	**44**	52	**47**	53	**42**	50
13	***7,42***	**53**	73	**52**	69	0	71	**51**	72	**50**	69
14	***5,94***	**44**	109	**44**	100	**39**	102	**42**	103	**38**	101
15	***18,53***	**85**	91	**83**	78	**79**	82	**81**	84	**79**	83
16	***7,09***	**89**	112	**89**	104	**86**	109	**88**	110	**85**	112
17	***10,81***	**57**	75	**56**	63	**54**	67	**55**	68	**52**	66
18	***6,33***	**121**	44	**117**	41	**113**	43	**120**	42	**116**	40
19	***57,14***	**163**	1967	**160**	1994	**149**	1966	**157**	1975	**151**	1958
20	***49,02***	**137**	2315	**134**	2303	**126**	2285	**128**	2309	**124**	2281
21	***70,51***	**125**	2132	**123**	2117	**121**	2107	**123**	2124	**119**	2095
22	***22,17***	**69**	49	**68**	48	0	46	**67**	47	**66**	44
23	***6,39***	**84**	40	**84**	42	**83**	39	**82**	41	**80**	38
24	***6,30***	**43**	44	**41**	37	**37**	38	**40**	40	0	39
25	***82,44***	**157**	39	**156**	41	**144**	42	**153**	43	**148**	40
26	***12,68***	**182**	141	**179**	127	**159**	123	**171**	139	**167**	124
27	***3,27***	**54**	28	**51**	26	**47**	27	**50**	27	**48**	28
28	***25,89***	**85**	140	**84**	133	**78**	126	**83**	137	**80**	125
29	***26,01***	**112**	153	**112**	145	**111**	147	**110**	148	**108**	142
30	***15,48***	**89**	188	**88**	171	**84**	173	**86**	175	**83**	174
31	***78,30***	**125**	2195	**123**	2132	**116**	2118	**122**	2188	**119**	2115
32	***6,41***	**117**	114	**116**	103	**112**	109	**115**	109	**110**	112
33	***20,08***	**104**	46	**102**	42	**98**	40	**100**	44	**96**	39
34	***8,15***	**77**	118	**75**	107	**72**	108	**74**	110	**72**	111
35	***25,44***	**86**	95	**85**	91	**0**	92	**83**	94	**81**	93
36	***37,83***	**143**	147	**141**	126	**130**	138	**138**	132	**133**	134
37	***56,91***	**87**	221	**87**	214	**78**	215	**86**	217	**79**	216

38	*53,49*	**138**	2076	**136**	2021	**132**	2008	**135**	2055	**130**	2001
39	*73,06*	**191**	2124	**189**	2118	**173**	2100	**187**	2122	**178**	2099
40	*5,37*	**44**	43	**40**	41	**39**	38	0	42	**36**	35
41	*19,24*	**83**	62	**82**	56	**79**	61	**81**	58	**77**	60
42	*25,67*	**77**	118	**75**	110	**72**	118	**75**	114	**71**	117
43	*93,18*	**258**	2137	**252**	2116	**239**	2103	**247**	2128	**244**	2096
44	*6,53*	**57**	55	**48**	50	**50**	51	**55**	53	0	50
45	*39,41*	**109**	263	**105**	246	**101**	244	**104**	251	**102**	248
46	*24,37*	**181**	168	**180**	159	**172**	153	**177**	164	**175**	165
47	*12,99*	**88**	122	**86**	120	**82**	117	**85**	124	0	119
48	*45,78*	**97**	217	**94**	211	**88**	204	**93**	215	**90**	202
49	*16,04*	**56**	123	**53**	118	**52**	120	**55**	119	0	119
50	*26,07*	**73**	154	**71**	151	**69**	153	**70**	152	**68**	151
51	*98,16*	**257**	2125	**255**	2117	**247**	2108	**254**	2120	**250**	2115
52	*3,08*	**38**	27	**35**	24	**31**	24	0	26	**32**	25
53	*38,50*	**83**	236	**81**	228	**78**	219	**80**	234	**78**	217
54	*22,01*	**57**	171	**55**	169	**52**	165	0	172	**53**	167
55	*74,19*	**162**	2021	**161**	2015	**157**	2013	**160**	2018	**158**	2012
56	*68,27*	**194**	118	**193**	107	**189**	101	**191**	109	**188**	100
57	*15,91*	**68**	119	**68**	103	**62**	104	**66**	115	**64**	111
58	*5,33*	**32**	28	0	26	**27**	26	0	27	**29**	28
59	*40,21*	**43**	397	**41**	385	0	377	**40**	392	**38**	373
60	*108,73*	**388**	2483	**385**	2468	**379**	2441	**381**	2471	**380**	2450
61	*60,49*	**142**	330	**142**	315	**139**	323	**141**	322	**138**	325
62	*102,53*	**391**	2512	**388**	2497	**381**	2481	**383**	2503	**380**	2473
63	*15,67*	**124**	133	**123**	115	**115**	118	**122**	128	**118**	122
64	*80,24*	**222**	2140	**220**	2112	**211**	2110	**217**	2136	**213**	2106

Tabela A.12 – Resultados GRASP com RVND – AGM *Kruskal* padrão para obtenção de 3 *clusters*

		GRASP COM RVND									
		RV_AGM1		**RV_AGM2**		**RV_AGM3**		**RV_AGM4**		**RV_AGM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**94**	117	**89**	112	**80**	111	**87**	114	**81**	109
2	*39,08*	**116**	130	**108**	121	**100**	122	**104**	125	**102**	118
3	*5,72*	**49**	46	**47**	45	**40**	41	**46**	41	**43**	42
4	*26,14*	**51**	191	**48**	132	**43**	125	**49**	136	**45**	123
5	*7,35*	**57**	49	**54**	48	**46**	46	**51**	45	**42**	41
6	*6,41*	**59**	46	**55**	43	**51**	42	**53**	44	**48**	45
7	*6,17*	**47**	52	**44**	51	**38**	49	**41**	50	**39**	48
8	*4,96*	**40**	46	**39**	45	**35**	41	**37**	39	**34**	43
9	*14,35*	**111**	101	**102**	100	**86**	93	**94**	98	**88**	96
10	*19,78*	**85**	108	**81**	106	**83**	104	**79**	101	**75**	100
11	*10,27*	**133**	121	**128**	117	**123**	115	0	118	**108**	111
12	*17,33*	**59**	53	**56**	52	**48**	48	**51**	54	**49**	49
13	*7,42*	**75**	72	**72**	71	**55**	68	**52**	73	**53**	67
14	*5,94*	**46**	107	**45**	101	**43**	99	**44**	104	**41**	98
15	*18,53*	**91**	59	**89**	82	**84**	77	**85**	84	**81**	79
16	*7,09*	**118**	110	**113**	109	**99**	106	**104**	102	**98**	108
17	*10,81*	**86**	72	**74**	67	**58**	65	**71**	68	**57**	63
18	*6,33*	**130**	45	**128**	44	**111**	38	**127**	45	**114**	36
19	*57,14*	**177**	1990	**170**	1963	**152**	1949	**163**	1981	**154**	1947
20	*49,02*	**142**	2310	**137**	2295	**129**	2234	**134**	2300	**128**	2221
21	*70,51*	**135**	2124	**129**	2108	0	2095	**127**	2115	**121**	2092
22	*22,17*	**77**	51	**75**	48	**72**	42	**71**	49	**69**	46
23	*6,39*	**96**	39	**93**	37	**85**	38	**89**	40	0	35
24	*6,30*	**51**	40	**49**	38	**41**	37	**44**	36	**42**	36
25	*82,44*	**63**	37	**57**	41	**48**	41	**51**	40	**49**	38
26	*12,68*	**192**	140	**185**	136	**171**	120	**182**	123	**174**	122
27	*3,27*	**67**	27	**62**	25	**57**	26	**58**	27	**54**	27
28	*25,89*	**96**	138	**94**	131	**86**	121	**92**	132	**85**	122
29	*26,01*	**126**	150	**122**	144	**115**	145	**119**	141	**114**	140
30	*15,48*	**99**	184	**97**	171	**91**	170	**94**	173	**89**	169

31	***78,30***	**139**	2142	**137**	2118	**124**	2099	**134**	2125	**127**	2093
32	***6,41***	**127**	115	**124**	113	**118**	111	**123**	114	**115**	110
33	***20,08***	**116**	45	**114**	42	**105**	38	**111**	40	**104**	36
34	***8,15***	**91**	118	**88**	109	**82**	109	**85**	113	**78**	107
35	***25,44***	**95**	104	**91**	96	**89**	98	**90**	102	0	94
36	***37,83***	**148**	141	**145**	137	**140**	132	**142**	139	**141**	131
37	***56,91***	**99**	218	**92**	209	**85**	208	**90**	211	**83**	208
38	***53,49***	**147**	2015	**144**	1988	**141**	1991	**142**	2004	**137**	1983
39	***73,06***	**195**	2120	**193**	2114	**179**	2093	**189**	2116	**186**	2087
40	***5,37***	**51**	41	**47**	39	0	38	**45**	40	**39**	37
41	***19,24***	**92**	65	**90**	62	**88**	59	**86**	63	**83**	56
42	***25,67***	**88**	117	**82**	115	**77**	117	**79**	116	**75**	112
43	***93,18***	**283**	2126	**276**	2104	**263**	2098	**271**	2110	**267**	2083
44	***6,53***	**70**	54	**68**	51	**64**	49	**62**	53	**59**	47
45	***39,41***	**119**	251	**117**	243	**110**	238	**113**	247	**107**	235
46	***24,37***	**201**	169	**194**	167	**181**	160	**188**	168	**183**	162
47	***12,99***	**99**	123	**96**	121	**91**	115	**90**	118	**85**	114
48	***45,78***	**123**	210	**118**	204	**101**	199	**112**	207	**97**	197
49	***16,04***	**69**	121	**66**	118	**54**	112	**61**	119	**58**	113
50	***26,07***	**92**	155	**87**	150	**77**	149	**84**	152	**75**	147
51	***98,16***	**289**	2119	**273**	2103	**259**	2109	**267**	2112	**264**	2094
52	***3,08***	**44**	28	**42**	25	**35**	24	**40**	27	**38**	23
53	***38,50***	**85**	235	**82**	232	**80**	215	**83**	233	**79**	214
54	***22,01***	**61**	170	**59**	163	**51**	155	**57**	167	**55**	152
55	***74,19***	**180**	2015	**177**	2004	**162**	1997	**173**	2010	**163**	1991
56	***68,27***	**199**	116	**196**	103	**194**	98	**199**	105	**191**	97
57	***15,91***	**81**	115	**77**	112	**71**	107	**75**	113	**72**	110
58	***5,33***	**38**	30	**35**	33	**29**	28	**33**	32	**31**	29
59	***40,21***	**55**	392	**49**	384	**42**	359	**44**	387	**40**	355
60	***108,73***	**418**	2475	**399**	2463	**385**	2432	**396**	2469	**394**	2424
61	***60,49***	**163**	328	**155**	321	**145**	320	**148**	322	**142**	319
62	***102,53***	**441**	2501	**428**	2497	**402**	2463	**417**	2499	**397**	2451
63	***15,67***	**148**	129	**154**	124	**147**	119	**151**	127	**142**	118
64	***80,24***	**239**	2128	**237**	2123	**221**	2091	**230**	2115	**224**	2083

Tabela A.13 – Resultados VNS Padrão – Grafo Original para obtenção de 3 *clusters*

		VNS PADRÃO									
		VNS1		**VNS2**		**VNS3**		**VNS4**		**VNS5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**29**	129	**20**	108	**12**	115	**16**	119	**11**	114
2	***39,08***	**28**	197	**25**	184	**11**	171	**18**	193	10	169
3	***5,72***	**17**	45	**14**	31	8	24	**11**	35	9	25
4	***26,14***	**33**	186	**25**	160	**12**	159	**23**	168	**15**	147
5	***7,35***	**15**	73	**12**	42	0	40	9	57	1	31
6	***6,41***	**11**	69	8	24	0	35	2	58	0	30
7	***6,17***	8	72	4	37	0	32	0	44	0	26
8	***4,96***	6	35	2	29	0	27	0	32	0	24
9	***14,35***	**17**	79	**14**	64	7	64	**12**	67	8	58
10	***19,78***	**25**	140	**19**	108	**13**	102	**18**	115	**12**	94
11	***10,27***	**34**	83	**28**	46	**14**	58	**22**	68	**16**	41
12	***17,33***	**19**	117	**17**	101	7	94	**15**	117	6	75
13	***7,42***	**20**	78	**13**	62	5	57	**11**	68	2	38
14	***5,94***	**46**	85	**41**	70	**22**	61	**37**	74	**16**	34
15	***18,53***	**28**	139	**24**	108	10	105	**16**	110	8	82
16	***7,09***	**31**	67	0	41	**12**	48	**19**	53	**15**	37
17	***10,81***	**42**	114	**35**	86	**17**	69	**23**	102	**11**	39
18	***6,33***	9	81	4	50	0	48	0	54	0	31
19	***57,14***	**50**	2362	**45**	2335	**18**	2327	**28**	2348	**20**	2330
20	***49,02***	**69**	2289	**63**	2273	**55**	2264	**67**	2279	**56**	2255
21	***70,51***	**88**	2437	**80**	2358	**47**	2346	**14**	2361	**36**	2342
22	***22,17***	**13**	144	**11**	108	2	93	9	104	1	115
23	***6,39***	**10**	58	8	33	0	47	2	44	0	28

24	*6,30*	9	55	5	30	0	32	0	51	0	20
25	*82,44*	8	50	3	31	0	25	0	40	0	30
26	*12,68*	**99**	514	**96**	472	**74**	469	**89**	479	**77**	514
27	*3,27*	**42**	161	**39**	149	**22**	143	**34**	156	**19**	64
28	*25,89*	5	42	0	27	0	24	0	33	0	20
29	*26,01*	**48**	191	**41**	178	**29**	168	**45**	184	**27**	170
30	*15,48*	**65**	218	**44**	184	**18**	171	**32**	193	**16**	159
31	*78,30*	**21**	134	**16**	102	7	83	**15**	98	8	94
32	*6,41*	**84**	2515	**79**	2498	**75**	2485	**78**	2506	**70**	2490
33	*20,08*	9	82	6	53	0	41	0	63	0	52
34	*8,15*	**35**	139	**29**	114	10	97	**18**	125	7	119
35	*25,44*	**22**	126	**18**	107	5	74	**13**	116	4	99
36	*37,83*	**37**	204	**33**	188	**12**	138	**31**	193	**16**	184
37	*56,91*	**41**	281	**39**	246	**25**	241	**38**	252	**22**	218
38	*53,49*	**80**	2393	**72**	2270	**41**	2253	**64**	2275	**42**	2230
39	*73,06*	**53**	2336	**48**	2321	**39**	2285	**42**	2326	**34**	2314
40	*5,37*	6	50	3	39	0	37	0	41	0	35
41	*19,24*	**24**	127	**15**	108	1	88	7	115	3	101
42	*25,67*	**27**	209	0	181	7	160	**14**	197	5	178
43	*93,18*	**151**	2645	**146**	2627	**103**	2573	**90**	2633	**98**	2605
44	*6,53*	10	71	8	34	0	62	**4**	64	0	58
45	*39,41*	**42**	270	**39**	261	**30**	257	**34**	266	**27**	233
46	*24,37*	**33**	167	**30**	148	**19**	131	**23**	159	**14**	115
47	*12,99*	**39**	157	**33**	123	**25**	123	**28**	131	**22**	119
48	*45,78*	**52**	361	**48**	335	**39**	315	**42**	349	**34**	287
49	*16,04*	**30**	129	**22**	98	**12**	101	**19**	115	**13**	85
50	*26,07*	**33**	227	**29**	191	**18**	186	**22**	203	**15**	173
51	*98,16*	**110**	2656	**107**	2632	**91**	2629	**99**	2641	**87**	2604
52	*3,08*	3	40	0	30	0	28	0	38	0	33
53	*38,50*	**36**	231	**29**	207	**13**	212	**31**	229	**15**	201
54	*22,01*	**24**	172	**17**	154	5	137	**14**	141	8	115
55	*74,19*	**69**	2455	**61**	2416	**53**	2393	**62**	2421	**54**	2381
56	*68,27*	**63**	2439	**62**	2378	**44**	2407	**53**	409	**46**	2384
57	*15,91*	**30**	127	**29**	109	**12**	93	**22**	115	**14**	100
58	*5,33*	7	42	0	31	0	25	0	33	0	29
59	*40,21*	**49**	401	**42**	372	**20**	367	**44**	386	**21**	331
60	*108,73*	**112**	2793	**107**	740	**94**	2731	**101**	2748	**92**	2694
61	*60,49*	**77**	367	**65**	322	**49**	317	**59**	329	**46**	298
62	*102,53*	**129**	2814	**114**	2241	**98**	2711	**122**	2757	**100**	2703
63	*15,67*	**26**	132	**23**	102	**14**	95	**20**	116	**12**	89
64	*80,24*	**99**	2509	**94**	2464	**76**	2451	**84**	2484	**75**	2446

Tabela A.14 – Resultados VNS Padrão – AGM *Kruskal modificado* para obtenção de 3 *clusters*

		VNS PADRÃO									
		VNS_AGMM1		**VNS_AGMM2**		**VNS_AGMM3**		**VNS_AGMM4**		**VNS_AGMM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**72**	119	**71**	107	**69**	105	**70**	112	**68**	108
2	*39,08*	**93**	146	**92**	138	**87**	134	**92**	141	**89**	132
3	*5,72*	**41**	42	**41**	37	**38**	36	**40**	38	**40**	36
4	*26,14*	**47**	157	**47**	144	**43**	143	**46**	146	**44**	145
5	*7,35*	**38**	53	**36**	51	**34**	49	**35**	52	0	47
6	*6,41*	**51**	49	**50**	37	**42**	38	**49**	39	**44**	36
7	*6,17*	**37**	57	**35**	58	**29**	55	**33**	59	**31**	56
8	*4,96*	**33**	50	**32**	45	**28**	48	**31**	44	**29**	47
9	*14,35*	**89**	104	**87**	101	**84**	97	**85**	108	0	104
10	*19,78*	**75**	113	**75**	107	**70**	110	**72**	113	**71**	109
11	*10,27*	**101**	137	**99**	129	**93**	127	**98**	132	**95**	125
12	*17,33*	**48**	55	**47**	52	**40**	51	**45**	54	**41**	53
13	*7,42*	**46**	77	**44**	68	**39**	64	**42**	69	**40**	66
14	*5,94*	**39**	112	**38**	117	0	111	**37**	115	**34**	108
15	*18,53*	**84**	93	**81**	86	**78**	84	**80**	88	**77**	82
16	*7,09*	**80**	117	**79**	110	0	107	**76**	115	**73**	112

17	***10,81***	**46**	79	**46**	69	**44**	78	**45**	74	**42**	69
18	***6,33***	**118**	45	**117**	42	**109**	41	**115**	44	**110**	43
19	***57,14***	**149**	1973	**145**	1952	**140**	1947	**144**	1966	**142**	1961
20	***49,02***	**125**	2321	**124**	2304	**119**	2285	**123**	2312	**118**	2296
21	***70,51***	**116**	2154	**115**	2118	**112**	2099	**114**	2148	**110**	2104
22	***22,17***	**58**	52	**56**	47	**45**	46	**56**	49	**51**	45
23	***6,39***	**78**	41	**77**	38	**72**	39	**75**	40	**73**	41
24	***6,30***	**37**	45	**35**	33	0	30	**34**	38	**32**	34
25	***82,44***	**149**	43	**147**	39	**138**	43	**146**	42	**144**	41
26	***12,68***	**178**	147	**175**	142	**172**	138	**173**	144	**169**	135
27	***3,27***	**53**	27	**51**	40	0	39	**49**	34	**50**	37
28	***25,89***	**78**	141	**78**	138	**74**	133	**77**	140	**76**	136
29	***26,01***	**104**	158	**103**	151	**101**	154	**102**	155	**100**	152
30	***15,48***	**85**	193	**84**	186	**80**	185	**82**	190	0	187
31	***78,30***	**117**	2200	**116**	2187	**110**	2179	**115**	2193	**108**	2185
32	***6,41***	**108**	117	**106**	112	**97**	110	**104**	114	**99**	113
33	***20,08***	**99**	48	**98**	49	**92**	47	**97**	51	**91**	48
34	***8,15***	**68**	129	**66**	131	**62**	123	**65**	130	**63**	127
35	***25,44***	**81**	97	**81**	93	**71**	96	**80**	95	**74**	94
36	***37,83***	**137**	154	**129**	149	**120**	144	**125**	152	**122**	142
37	***56,91***	**82**	235	**81**	221	**74**	227	**80**	233	**76**	231
38	***53,49***	**124**	2081	**121**	2028	**115**	2014	**120**	2069	**117**	2016
39	***73,06***	**178**	2136	**173**	2100	**166**	2093	**167**	2124	**164**	2087
40	***5,37***	**41**	45	**38**	42	**36**	40	**40**	41	0	39
41	***19,24***	**79**	61	**78**	58	**71**	55	**75**	60	**70**	59
42	***25,67***	**64**	114	**63**	106	**59**	107	**62**	109	**59**	109
43	***93,18***	**244**	2128	**241**	2094	**234**	2083	0	2115	**237**	2080
44	***6,53***	**48**	57	0	52	**42**	51	**46**	53	**40**	52
45	***39,41***	**101**	266	**99**	248	**93**	237	**98**	255	**95**	242
46	***24,37***	**173**	171	0	167	**165**	168	**167**	169	**164**	166
47	***12,99***	**86**	127	**85**	122	**79**	119	**83**	124	**78**	116
48	***45,78***	**94**	221	**93**	218	**91**	204	**90**	220	**92**	202
49	***16,04***	**55**	128	**55**	125	**48**	121	**53**	127	**50**	119
50	***26,07***	**69**	167	**68**	163	**65**	161	**67**	164	**65**	166
51	***98,16***	**254**	2141	**250**	2112	**239**	2107	**248**	2138	**242**	2099
52	***3,08***	**37**	30	**34**	33	**32**	34	0	34	**33**	36
53	***38,50***	**79**	241	**76**	219	**72**	204	**75**	227	**71**	221
54	***22,01***	**53**	175	**52**	172	**44**	170	**51**	174	**47**	173
55	***74,19***	**157**	2039	**155**	2006	**151**	1983	**152**	2014	**149**	1982
56	***68,27***	**182**	120	**181**	119	**180**	121	**179**	122	**181**	118
57	***15,91***	**64**	128	**63**	133	**57**	128	**62**	131	**59**	126
58	***5,33***	**33**	30	**32**	37	**28**	36	**31**	35	0	38
59	***40,21***	**49**	395	**49**	376	**44**	381	**48**	388	**47**	373
60	***108,73***	**375**	2517	**372**	2429	**367**	2414	**369**	2504	**363**	2400
61	***60,49***	**137**	336	**134**	315	**125**	323	**133**	328	**128**	319
62	***102,53***	**382**	2518	**379**	2467	**371**	2455	**375**	2493	**368**	2472
63	***15,67***	**118**	139	**115**	128	**107**	124	**114**	131	**109**	129
64	***80,24***	**215**	2152	**214**	2114	**210**	2100	0	2145	**212**	2094

Tabela A.15 – Resultados VNS Padrão – AGM *Kruskal* padrão para obtenção de 3 *clusters*

		VNS PADRÃO									
		VNS_AGM1		**VNS_AGM2**		**VNS_AGM3**		**VNS_AGM4**		**VNS_AGM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**93**	114	**87**	109	**79**	103	**82**	105	**77**	104
2	***39,08***	**120**	144	**112**	138	**92**	131	**101**	136	**94**	129
3	***5,72***	**53**	41	**49**	36	**43**	35	**48**	35	**42**	33
4	***26,14***	**69**	149	**63**	144	**51**	143	**57**	145	**53**	142
5	***7,35***	**62**	54	**59**	53	**50**	48	**55**	52	**51**	45
6	***6,41***	**73**	42	**65**	38	**49**	35	**53**	37	**49**	33
7	***6,17***	**45**	58	**42**	57	**36**	54	**39**	56	**38**	52
8	***4,96***	**44**	49	**41**	43	**38**	46	**39**	45	**35**	44
9	***14,35***	**102**	106	**96**	107	**89**	100	**94**	105	**88**	102

10	*19,78*	**95**	112	**92**	111	**80**	109	**87**	106	**81**	107
11	*10,27*	**116**	134	**111**	128	**100**	122	**104**	129	**99**	119
12	*17,33*	**58**	54	**53**	53	**48**	50	**50**	54	**47**	52
13	*7,42*	**61**	73	**57**	65	**44**	62	**49**	67	**42**	63
14	*5,94*	**42**	118	**41**	112	**38**	110	0	115	**38**	106
15	*18,53*	**98**	90	**94**	86	**85**	83	**89**	88	**82**	81
16	*7,09*	**89**	116	**85**	114	**77**	104	**83**	112	**78**	107
17	*10,81*	**74**	75	**66**	71	**54**	73	**55**	73	**52**	66
18	*6,33*	**130**	44	**126**	42	**117**	40	**123**	44	**112**	41
19	*57,14*	**174**	1966	**164**	1960	**156**	1953	**158**	1959	**153**	1958
20	*49,02*	**128**	2314	**129**	2291	**125**	2189	**127**	2299	**122**	2187
21	*70,51*	**125**	2123	**122**	2115	**117**	2096	**119**	2113	**118**	2093
22	*22,17*	**63**	50	**62**	46	**54**	42	**57**	46	**56**	41
23	*6,39*	**89**	40	**81**	41	**77**	38	**80**	36	**79**	39
24	*6,30*	**42**	44	**40**	36	**38**	32	**39**	31	**35**	33
25	*82,44*	**153**	45	**152**	41	**149**	42	**151**	44	**148**	40
26	*12,68*	**195**	154	**194**	138	**183**	125	**186**	143	**177**	122
27	*3,27*	**64**	41	**63**	35	**60**	36	**58**	39	**61**	33
28	*25,89*	**91**	140	**84**	139	**81**	128	**80**	139	**83**	127
29	*26,01*	**118**	155	**117**	153	**107**	151	**111**	149	**109**	148
30	*15,48*	**99**	189	**96**	183	**89**	180	**93**	184	**88**	182
31	*78,30*	**124**	2186	**121**	2165	**112**	2158	**116**	2170	**109**	2143
32	*6,41*	**122**	115	**118**	112	**108**	109	**112**	113	**104**	110
33	*20,08*	**57**	50	**55**	50	**49**	43	**53**	51	**49**	42
34	*8,15*	**78**	127	**74**	123	**67**	115	**69**	125	**68**	118
35	*25,44*	**94**	98	**81**	94	0	91	**84**	97	**78**	92
36	*37,83*	**149**	151	**135**	145	**131**	136	**136**	146	**129**	137
37	*56,91*	**90**	242	**86**	232	**82**	228	**81**	219	**84**	230
38	*53,49*	**138**	2075	**132**	2041	**123**	1998	**127**	2042	**120**	2002
39	*73,06*	**192**	2128	**189**	2103	**176**	2084	**188**	2117	**173**	2071
40	*5,37*	**61**	44	**55**	40	**42**	37	**49**	41	**44**	36
41	*19,24*	**99**	63	**94**	57	**81**	54	**89**	62	**85**	53
42	*25,67*	**75**	112	**69**	108	**62**	104	**60**	105	**61**	108
43	*93,18*	**266**	2120	**251**	2100	**244**	2071	**248**	2112	**245**	2063
44	*6,53*	**63**	56	**55**	53	0	50	**53**	49	**49**	51
45	*39,41*	**119**	253	**116**	107	**98**	239	**107**	244	0	236
46	*24,37*	**199**	174	**189**	184	**175**	167	**184**	172	**179**	165
47	*12,99*	**104**	123	**93**	120	**83**	114	**89**	119	**85**	113
48	*45,78*	**116**	218	**114**	215	**105**	199	**112**	216	**103**	197
49	*16,04*	**68**	124	**67**	123	**64**	120	**68**	123	**63**	115
50	*26,07*	**80**	165	**75**	163	**72**	159	**74**	162	**71**	160
51	*98,16*	**283**	2133	**279**	2114	**266**	2085	**271**	2124	**265**	2071
52	*3,08*	**51**	35	**42**	32	**37**	32	**41**	33	**39**	33
53	*38,50*	**97**	237	**93**	275	**81**	199	**86**	226	**80**	219
54	*22,01*	**56**	174	**56**	171	**49**	168	**54**	173	**48**	165
55	*74,19*	**170**	2028	**167**	2012	**159**	1981	**168**	2015	**165**	1977
56	*68,27*	**211**	126	**209**	121	**193**	118	**204**	124	**194**	116
57	*15,91*	**73**	137	**71**	130	**65**	127	**69**	132	**67**	123
58	*5,33*	**44**	36	**42**	33	**37**	35	**41**	32	**38**	36
59	*40,21*	**60**	392	**56**	385	**51**	374	**50**	387	**49**	369
60	*108,73*	**391**	2504	**386**	2480	**377**	2397	**381**	2485	0	2381
61	*60,49*	**142**	328	**137**	314	**127**	299	**135**	316	**129**	293
62	*102,53*	**346**	2441	**390**	2471	**386**	2408	**394**	2455	**381**	2412
63	*15,67*	**124**	135	**120**	130	**117**	122	**118**	132	**112**	121
64	*80,24*	**231**	2104	**224**	2098	**210**	2057	**221**	2083	**208**	2064

Tabela A.16 – Resultados GVNS – Grafo Original para obtenção de 3 *clusters*

		GVNS									
		GVNS1		**GVNS2**		**GVNS3**		**GVNS4**		**GVNS5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**22**	117	**18**	99	10	106	**15**	114	9	108

2	***39,08***	**27**	203	**21**	172	**11**	154	**19**	187	**12**	163
3	***5,72***	**15**	31	**13**	19	8	21	**12**	29	6	26
4	***26,14***	**32**	166	**27**	147	**14**	129	**21**	163	**11**	151
5	***7,35***	10	83	0	40	0	33	6	51	0	34
6	***6,41***	**11**	81	8	27	0	29	3	49	0	32
7	***6,17***	**12**	75	4	32	0	25	0	42	0	27
8	***4,96***	5	29	0	27	0	23	0	37	0	25
9	***14,35***	**14**	80	10	53	6	51	8	69	3	54
10	***19,78***	**25**	156	**17**	99	10	102	**15**	118	**11**	91
11	***10,27***	**36**	108	**29**	42	**17**	40	**25**	76	**12**	44
12	***17,33***	**19**	129	**18**	81	9	67	**13**	98	4	70
13	***7,42***	**16**	74	**12**	55	3	32	7	67	1	34
14	***5,94***	**48**	92	**43**	61	**21**	30	**38**	72	**13**	38
15	***18,53***	**29**	117	**25**	102	**12**	81	**23**	114	5	78
16	***7,09***	**36**	74	**33**	35	**18**	28	**31**	59	**14**	30
17	***10,81***	**39**	126	**34**	46	**15**	40	**29**	87	**12**	44
18	***6,33***	8	63	3	47	0	29	0	55	0	27
19	***57,14***	**41**	2351	**37**	2334	0	2367	**33**	2351	0	2338
20	***49,02***	**62**	2291	**38**	2341	0	2279	**29**	2282	**58**	2229
21	***70,51***	**94**	2448	**81**	2375	**36**	2428	**45**	2403	**37**	2391
22	***22,17***	**11**	151	9	104	1	94	8	117	2	96
23	***6,39***	**11**	75	7	21	0	25	2	51	0	28
24	***6,30***	10	70	4	22	0	29	0	48	0	29
25	***82,44***	**11**	62	3	24	0	26	0	46	6	28
26	***12,68***	**101**	503	**86**	468	**78**	504	**97**	483	**75**	450
27	***3,27***	**50**	125	**44**	116	**18**	93	**39**	137	**20**	41
28	***25,89***	5	44	0	27	0	15	0	25	0	24
29	***26,01***	**49**	207	**37**	181	**28**	172	**42**	193	**26**	127
30	***15,48***	**54**	198	**40**	166	**17**	164	**39**	187	**15**	115
31	***78,30***	**26**	128	**19**	101	9	91	**17**	104	8	72
32	***6,41***	**97**	2516	**85**	2485	**72**	2489	**79**	2491	**69**	2424
33	***20,08***	9	78	5	52	0	49	1	69	0	39
34	***8,15***	**28**	152	**23**	111	5	108	**19**	116	8	91
35	***25,44***	**19**	135	**16**	83	2	97	**12**	100	6	47
36	***37,83***	**33**	197	**29**	161	**13**	164	**27**	171	**15**	128
37	***56,91***	**39**	2234	**32**	2205	**24**	2209	**34**	2227	**21**	2173
38	***53,49***	**73**	2315	**58**	2259	**40**	2244	**51**	2261	**44**	2299
39	***73,06***	**59**	2327	**42**	2291	**38**	2289	**39**	2293	**35**	2254
40	***5,37***	10	43	0	28	0	28	0	31	0	28
41	***19,24***	**23**	121	**17**	91	1	94	8	96	2	88
42	***25,67***	**22**	198	**15**	171	4	167	**11**	174	3	127
43	***93,18***	**148**	2633	**91**	2612	**77**	2609	**87**	2615	**72**	2512
44	***6,53***	**12**	85	10	44	0	69	3	63	1	31
45	***39,41***	**41**	282	**38**	251	**28**	247	**35**	268	**25**	214
46	***24,37***	**35**	173	**27**	143	**15**	102	**22**	122	**12**	118
47	***12,99***	**38**	140	**32**	118	**23**	122	**29**	134	**19**	51
48	***45,78***	**51**	315	**49**	271	**38**	284	**45**	280	**33**	243
49	***16,04***	**32**	99	**24**	74	**12**	87	**19**	90	**11**	65
50	***26,07***	**29**	206	**28**	189	**14**	181	**23**	198	**12**	144
51	***98,16***	**104**	2649	**102**	2612	**89**	2599	**97**	2617	**85**	2556
52	***3,08***	4	45	0	21	0	31	0	33	0	25
53	***38,50***	**43**	220	3	173	**12**	198	**28**	201	**17**	192
54	***22,01***	**25**	137	**18**	115	5	112	**15**	122	4	100
55	***74,19***	**67**	2474	**56**	2392	**51**	2439	**63**	2408	**56**	2405
56	***68,27***	**62**	2421	**59**	2331	**43**	2376	**52**	2389	**47**	2354
57	***15,91***	**29**	94	**27**	72	**15**	108	**21**	91	**12**	69
58	***5,33***	**12**	37	0	18	0	32	0	30	0	24
59	***40,21***	**43**	361	**38**	294	**22**	327	**41**	348	**20**	203
60	***108,73***	**119**	2763	**114**	2694	**89**	2715	**102**	2742	**94**	2658
61	***60,49***	**75**	341	**72**	302	**48**	305	**63**	308	**42**	329
62	***102,53***	**123**	2728	**119**	2683	**99**	2717	**115**	2719	**101**	2621
63	***15,67***	**28**	109	**22**	73	**11**	84	**20**	104	**14**	66
64	***80,24***	**102**	2484	**101**	2421	**74**	2440	**92**	2466	**73**	2402

Tabela A.17 – Resultados GVNS – AGM *Kruskal modificado* para obtenção de 3 *clusters*

		GVNS									
		GVNS_AGMM1		GVNS_AGMM2		GVNS_AGMM3		GVNS_AGMM4		GVNS_AGMM5	
I	*Solução best*	Gap	T(s)	Gap	T(s)	Gap	T(s)	Gap	T(s)	Gap	T(s)
1	*22,54*	**67**	123	**65**	118	**59**	120	**63**	121	**58**	115
2	*39,08*	**88**	159	**84**	166	**77**	162	**79**	163	**75**	157
3	*5,72*	**39**	44	**37**	40	**34**	43	0	42	**35**	41
4	*26,14*	**48**	161	**46**	153	**41**	159	**45**	155	**43**	158
5	*7,35*	**37**	55	**37**	52	**32**	48	**36**	54	**30**	50
6	*6,41*	**52**	48	**51**	40	**48**	41	**50**	43	**45**	37
7	*6,17*	**38**	60	**36**	51	0	55	**35**	57	**32**	53
8	*4,96*	**29**	51	**27**	47	0	44	**26**	48	**23**	46
9	*14,35*	**88**	109	**87**	102	**83**	99	**85**	107	**83**	103
10	*19,78*	**69**	116	**68**	114	**61**	112	**67**	115	**63**	110
11	*10,27*	**94**	142	**89**	138	**84**	136	0	140	**82**	134
12	*17,33*	**43**	57	**42**	51	**39**	49	**41**	53	**38**	45
13	*7,42*	**45**	76	**43**	55	**38**	66	**40**	67	**39**	64
14	*5,94*	**32**	114	**32**	108	**25**	97	**30**	110	**27**	94
15	*18,53*	**79**	92	**76**	90	**73**	83	**75**	91	**71**	88
16	*7,09*	**72**	116	0	111	**61**	109	**68**	113	**63**	110
17	*10,81*	**44**	83	**42**	77	**38**	73	**41**	79	0	72
18	*6,33*	**115**	47	**112**	43	**101**	41	**109**	45	**103**	43
19	*57,14*	**139**	1980	**138**	1949	**132**	1953	**135**	1967	**129**	1962
20	*49,02*	**124**	2374	**123**	2359	**115**	2355	**122**	2361	**118**	2358
21	*70,51*	**118**	2191	**115**	2125	**108**	2149	**109**	2154	**103**	2144
22	*22,17*	**59**	58	**57**	51	0	48	**55**	53	**49**	45
23	*6,39*	**70**	63	**69**	65	**62**	57	**68**	58	**64**	59
24	*6,30*	**36**	48	**35**	35	**34**	41	**34**	42	**31**	37
25	*82,44*	**157**	46	**152**	49	**131**	46	**148**	51	**134**	42
26	*12,68*	**158**	154	**155**	155	**144**	147	**152**	156	**147**	151
27	*3,27*	**49**	38	**47**	44	**42**	35	**45**	47	**42**	37
28	*25,89*	**74**	163	**72**	152	**68**	144	**70**	155	**67**	149
29	*26,01*	**98**	174	**95**	166	**89**	161	**93**	169	**88**	163
30	*15,48*	**77**	202	**76**	195	**68**	199	**75**	200	**69**	193
31	*78,30*	**115**	2227	**111**	2201	**108**	2183	**109**	2215	**104**	2197
32	*6,41*	**99**	130	**93**	125	**89**	122	0	128	**87**	124
33	*20,08*	**96**	51	**95**	54	**85**	48	**94**	57	**89**	48
34	*8,15*	**57**	132	**54**	128	**47**	117	**51**	129	**48**	122
35	*25,44*	**63**	104	**62**	91	**58**	93	**61**	102	**58**	91
36	*37,83*	**115**	176	**110**	161	**105**	157	**108**	167	**103**	152
37	*56,91*	**74**	237	**73**	219	**70**	215	**71**	225	**69**	218
38	*53,49*	**116**	2095	**115**	2040	**106**	2014	**112**	2083	**108**	2009
39	*73,06*	**163**	2142	**158**	2113	**148**	2111	**153**	2138	0	2115
40	*5,37*	**34**	47	**33**	48	**29**	46	**31**	49	**28**	44
41	*19,24*	**77**	63	**76**	60	**68**	62	**76**	61	0	66
42	*25,67*	**53**	121	**49**	117	**41**	118	**47**	119	**42**	120
43	*93,18*	**246**	2139	**241**	2102	0	2097	**238**	2124	**233**	2095
44	*6,53*	**42**	67	0	62	**38**	63	**40**	65	**35**	61
45	*39,41*	**103**	271	**100**	268	**93**	262	**97**	269	**95**	266
46	*24,37*	**184**	189	**181**	173	**175**	163	**179**	177	**172**	169
47	*12,99*	**75**	139	**74**	118	**69**	114	**71**	122	**67**	112
48	*45,78*	**83**	244	**81**	227	**75**	225	**80**	239	**78**	228
49	*16,04*	**54**	131	**53**	125	**44**	124	**52**	129	**46**	121
50	*26,07*	**66**	181	**64**	172	**57**	167	**60**	175	0	164
51	*98,16*	**241**	2168	**240**	2127	**230**	2118	**237**	2144	**233**	2122
52	*3,08*	**33**	32	**31**	30	**28**	29	**29**	31	**26**	28
53	*38,50*	**78**	240	**77**	225	**70**	228	**74**	233	**71**	226
54	*22,01*	**49**	174	**48**	161	**39**	157	**44**	165	0	159
55	*74,19*	**142**	2057	**139**	2011	**135**	2004	**138**	2038	**133**	1998
56	*68,27*	**163**	141	**159**	128	**156**	125	**157**	134	**154**	121
57	*15,91*	**58**	133	**57**	119	**50**	120	**55**	125	**51**	123

58	***5,33***	**24**	32	**23**	28	**21**	34	0	29	0	32
59	***40,21***	**41**	402	**38**	393	**35**	388	**38**	397	**36**	394
60	***108,73***	**368**	2569	**364**	2431	**357**	2418	**360**	2495	**352**	2401
61	***60,49***	**129**	358	**127**	339	**121**	327	**123**	344	**118**	324
62	***102,53***	**371**	2553	**371**	2451	**355**	2412	**369**	2512	**358**	2410
63	***15,67***	**114**	148	**112**	144	**109**	145	**111**	147	**108**	144
64	***80,24***	**203**	2187	**193**	2139	**187**	2124	**188**	2160	**184**	2128

Tabela A.18 – Resultados GVNS – AGM *Kruskal* padrão para obtenção de 3 *clusters*

		GVNS									
		GVNS_AGM1		**GVNS_AGM2**		**GVNS_AGM3**		**GVNS_AGM4**		**GVNS_AGM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**83**	124	**79**	116	**69**	121	**74**	119	**66**	112
2	***39,08***	**99**	159	**92**	169	**88**	163	**91**	165	**84**	153
3	***5,72***	**55**	46	**53**	41	**45**	44	**48**	43	**42**	42
4	***26,14***	**66**	166	**61**	149	**53**	157	**59**	156	**55**	159
5	***7,35***	**49**	59	**44**	54	**40**	49	**43**	55	**39**	53
6	***6,41***	**68**	46	**61**	41	**49**	38	**56**	42	**53**	36
7	***6,17***	**51**	57	**48**	50	**37**	53	**42**	55	**39**	51
8	***4,96***	**40**	50	**41**	45	**34**	42	**39**	46	**32**	44
9	***14,35***	**93**	107	**90**	101	**86**	97	**88**	105	**85**	100
10	***19,78***	**89**	115	**84**	111	**71**	110	**76**	112	0	108
11	***10,27***	**122**	174	**115**	159	**99**	142	**104**	153	**91**	149
12	***17,33***	**57**	53	**55**	50	**48**	42	**52**	49	**45**	41
13	***7,42***	**51**	72	**48**	59	**39**	64	**44**	66	**41**	62
14	***5,94***	**40**	110	**37**	104	**31**	93	**35**	108	**32**	91
15	***18,53***	**91**	93	**88**	89	**79**	81	**84**	90	**77**	83
16	***7,09***	**79**	118	**76**	112	**62**	108	**73**	111	**69**	107
17	***10,81***	**48**	81	**45**	79	**40**	74	**44**	77	**40**	73
18	***6,33***	**124**	45	**121**	42	**112**	42	**118**	44	**115**	40
19	***57,14***	**161**	1974	**155**	1953	**148**	1941	**153**	1956	**140**	1932
20	***49,02***	**142**	2359	**134**	2351	**127**	2314	**132**	2341	**122**	2308
21	***70,51***	**144**	2148	**135**	2119	**118**	2095	**128**	2112	**112**	2083
22	***22,17***	**73**	56	**64**	50	**52**	47	**62**	51	**53**	42
23	***6,39***	**88**	64	**81**	63	**71**	56	**77**	55	**70**	53
24	***6,30***	**47**	42	**44**	33	**38**	33	0	39	**39**	35
25	***82,44***	**163**	43	**155**	45	**142**	44	**148**	52	**140**	41
26	***12,68***	**172**	151	**169**	148	**150**	137	**161**	142	**153**	139
27	***3,27***	**55**	42	**53**	41	**47**	39	**50**	43	**48**	34
28	***25,89***	**88**	161	**82**	150	**72**	135	**79**	150	**74**	137
29	***26,01***	**109**	172	**95**	165	**91**	160	**93**	167	**90**	162
30	***15,48***	**81**	200	**78**	193	**72**	195	**75**	198	**73**	191
31	***78,30***	**132**	2215	**128**	2195	**118**	2149	**122**	2203	**113**	2142
32	***6,41***	**116**	129	**104**	124	**97**	120	**99**	126	**95**	122
33	***20,08***	**98**	50	**95**	52	**91**	40	**93**	51	**91**	42
34	***8,15***	**66**	130	**54**	124	**48**	114	**54**	123	**49**	117
35	***25,44***	**74**	102	**70**	92	**61**	92	**69**	100	**64**	90
36	***37,83***	**129**	174	**122**	160	**114**	149	**119**	159	**112**	148
37	***56,91***	**83**	221	**84**	213	**72**	211	**81**	217	**73**	209
38	***53,49***	**145**	2073	**128**	2021	**117**	1985	**124**	2013	0	1993
39	***73,06***	**174**	2122	**169**	2115	**158**	2099	**163**	2116	**155**	2100
40	***5,37***	**48**	46	**44**	43	0	44	**39**	45	**36**	41
41	***19,24***	**102**	69	**99**	66	**82**	61	**93**	64	**84**	63
42	***25,67***	**57**	120	**53**	119	**45**	111	**51**	118	**48**	112
43	***93,18***	**316**	2115	**291**	2094	**269**	2073	**280**	2093	**264**	2075
44	***6,53***	**59**	65	**55**	63	**38**	62	**41**	63	**37**	60
45	***39,41***	**126**	270	**117**	266	**101**	251	**113**	264	**104**	253
46	***24,37***	**204**	183	**199**	175	**180**	160	**194**	173	**182**	162
47	***12,99***	**89**	135	**84**	122	**70**	114	**83**	121	**71**	110
48	***45,78***	**98**	240	**95**	224	**88**	222	**91**	230	**85**	221
49	***16,04***	**70**	130	**69**	123	**51**	120	**67**	121	**52**	117
50	***26,07***	**74**	179	**68**	170	**58**	164	**60**	169	**59**	162

51	***98,16***	**285**	2144	**277**	2116	**259**	2109	**273**	2138	**251**	2107
52	***3,08***	**49**	33	**42**	32	**38**	28	**40**	30	**37**	26
53	***38,50***	**91**	238	**83**	227	**75**	224	**79**	230	**78**	218
54	***22,01***	**66**	172	**59**	159	**46**	156	**54**	159	**45**	147
55	***74,19***	**163**	2051	**155**	2036	**147**	1993	**151**	2021	**142**	1983
56	***68,27***	**184**	139	**172**	124	**163**	128	**167**	132	**162**	120
57	***15,91***	**68**	128	**64**	125	**51**	114	**58**	124	**55**	112
58	***5,33***	**42**	35	**40**	34	0	31	**41**	32	**39**	30
59	***40,21***	**50**	397	**48**	391	**42**	386	**47**	389	**44**	386
60	***108,73***	**381**	2499	**377**	2415	**362**	2387	**374**	2396	**363**	2385
61	***60,49***	**139**	396	**133**	338	**124**	325	**130**	337	**127**	321
62	***102,53***	**388**	2512	**381**	2444	**362**	2393	**374**	2499	**365**	2385
63	***15,67***	**130**	149	**127**	148	**119**	141	**123**	145	**117**	139
64	***80,24***	**228**	2174	**226**	2135	**205**	2115	**214**	2154	**203**	2116

Tabela A.19 – Resultados RVNS – Grafo Original para obtenção de 3 *clusters*

		RVNS									
		RVNS1		**RVNS2**		**RVNS3**		**RVNS4**		**RVNS5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**17**	143	**14**	129	7	125	8	132	1	123
2	***39,08***	**29**	205	**23**	192	**15**	174	**19**	197	8	184
3	***5,72***	**22**	48	**17**	37	7	36	**11**	38	4	30
4	***26,14***	**20**	196	**19**	175	8	171	**16**	184	7	167
5	***7,35***	**19**	62	3	51	0	49	1	55	0	48
6	***6,41***	**12**	65	2	57	0	44	0	59	0	45
7	***6,17***	8	73	0	42	0	35	0	44	0	38
8	***4,96***	7	44	0	36	0	29	0	38	0	29
9	***14,35***	**24**	124	7	118	0	117	3	121	2	112
10	***19,78***	**33**	82	4	66	0	65	1	69	0	64
11	***10,27***	**32**	151	**21**	114	9	109	**12**	124	2	107
12	***17,33***	**40**	84	**18**	72	9	61	**13**	78	1	65
13	***7,42***	**15**	99	5	80	1	73	3	86	0	71
14	***5,94***	**27**	147	0	115	0	111	0	129	0	109
15	***18,53***	**26**	86	**19**	69	**11**	59	**14**	74	1	58
16	***7,09***	**18**	132	8	104	2	90	5	118	0	93
17	***10,81***	**33**	117	**23**	81	10	69	**18**	94	0	68
18	***6,33***	**19**	64	0	46	0	48	0	50	0	47
19	***57,14***	**59**	2331	**38**	2292	0	2284	0	2316	**11**	2272
20	***49,02***	**53**	2396	**24**	2374	**17**	2361	**21**	2382	0	2354
21	***70,51***	**86**	2151	**79**	2128	**42**	2104	0	2134	0	2099
22	***22,17***	**35**	63	**13**	47	5	38	9	59	**28**	41
23	***6,39***	**14**	68	0	49	0	45	0	56	0	44
24	***6,30***	**11**	66	0	47	1	46	0	48	0	42
25	***82,44***	10	69	0	41	0	37	0	44	0	39
26	***12,68***	**133**	543	**78**	522	**71**	481	**75**	526	0	473
27	***3,27***	**38**	194	**24**	167	**18**	158	**21**	181	6	152
28	***25,89***	9	58	0	49	0	31	0	55	0	32
29	***26,01***	**38**	215	**35**	183	**22**	170	**29**	198	**11**	171
30	***15,48***	**29**	228	**27**	192	**13**	181	**18**	203	6	179
31	***78,30***	**41**	137	**18**	104	6	94	**12**	118	10	91
32	***6,41***	**117**	2581	**77**	2515	0	2497	0	2526	**32**	2502
33	***20,08***	**14**	66	8	47	2	47	5	53	0	43
34	***8,15***	**42**	149	6	134	0	122	4	147	2	117
35	***25,44***	**29**	118	**14**	107	2	99	9	113	0	94
36	***37,83***	**37**	190	**19**	173	**14**	162	**17**	188	10	168
37	***56,91***	**48**	305	**24**	257	**18**	253	**23**	292	**15**	249
38	***53,49***	**66**	2322	**49**	2291	0	2267	**47**	2303	0	2261
39	***73,06***	**63**	2331	**45**	2371	**34**	2315	**39**	2319	0	2317
40	***5,37***	**10**	71	4	40	0	36	3	42	0	39
41	***19,24***	**21**	64	**11**	53	0	48	9	57	2	46

42	*25,67*	**35**	138	**28**	113	**19**	104	**24**	126	7	97
43	*93,18*	**162**	2216	**102**	2184	**64**	2178	**31**	2192	0	2173
44	*6,53*	**14**	65	**27**	60	0	45	2	63	0	48
45	*39,41*	**18**	301	**33**	286	**17**	270	**29**	299	**14**	261
46	*24,37*	**23**	193	**16**	168	10	149	**14**	171	6	144
47	*12,99*	**38**	175	**22**	142	**18**	133	2	151	**12**	128
48	*45,78*	**42**	368	**34**	339	**27**	319	**35**	348	**19**	324
49	*16,04*	**20**	147	**12**	126	9	111	**12**	137	5	109
50	*26,07*	**21**	230	**15**	181	8	196	**14**	221	6	193
51	*98,16*	**173**	2689	**104**	2655	**83**	2642	**88**	2674	0	2636
52	*3,08*	**5**	42	0	30	0	28	0	33	0	27
53	*38,50*	**20**	257	**19**	236	**14**	224	**18**	246	10	218
54	*22,01*	**22**	183	5	157	1	145	6	179	0	139
55	*74,19*	**86**	2471	**60**	2434	0	2418	**32**	2460	0	2415
56	*68,27*	**73**	2454	**49**	2426	**44**	2414	**2481**	2437	0	2405
57	*15,91*	**37**	135	8	128	7	114	**17**	129	**12**	109
58	*5,33*	**11**	69	4	40	0	39	0	41	0	38
59	*40,21*	**35**	432	**25**	395	**18**	380	**24**	418	**11**	375
60	*108,73*	**226**	2918	**99**	2861	**93**	2776	**97**	2894	0	2741
61	*60,49*	**63**	389	**42**	353	**39**	341	**43**	369	**41**	326
62	*102,53*	**151**	2791	**99**	2774	**98**	2769	0	2787	**92**	2738
63	*15,67*	**129**	147	**14**	129	9	123	**11**	136	6	119
64	*80,24*	**83**	2502	**75**	2497	**70**	2485	**51**	2503	0	2470

Tabela A.20 – Resultados RVNS – AGM *Kruskal modificado* para obtenção de 3 *clusters*

		RVNS									
		RVNS_AGMM1		**RVNS_AGMM2**		**RVNS_AGMM3**		**RVNS_AGMM4**		**RVNS_AGMM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**63**	127	**61**	121	**48**	121	**57**	124	**43**	119
2	*39,08*	**84**	164	**83**	153	**72**	144	**79**	159	**67**	147
3	*5,72*	**38**	46	**34**	42	**26**	38	**27**	44	**24**	41
4	*26,14*	**46**	163	**42**	155	**35**	147	**39**	159	**33**	156
5	*7,35*	**38**	59	**37**	53	**26**	50	**35**	57	**29**	58
6	*6,41*	**49**	50	**44**	43	**34**	48	**38**	52	**32**	41
7	*6,17*	**35**	63	**32**	61	0	57	**29**	64	**27**	53
8	*4,96*	**28**	54	**27**	52	0	49	**25**	53	**23**	42
9	*14,35*	**85**	118	**83**	109	**64**	113	**79**	112	**68**	110
10	*19,78*	**67**	115	**62**	112	**57**	110	**60**	114	**55**	107
11	*10,27*	**86**	136	**84**	128	**75**	122	**81**	132	**77**	126
12	*17,33*	**39**	58	**38**	51	0	57	**36**	56	**29**	49
13	*7,42*	**44**	80	**42**	74	**36**	68	**41**	77	**38**	63
14	*5,94*	**28**	117	**27**	114	**25**	109	**26**	116	**24**	110
15	*18,53*	**75**	98	**71**	93	**53**	92	**68**	94	**57**	96
16	*7,09*	**69**	118	**67**	116	**59**	117	**65**	119	0	115
17	*10,81*	**41**	86	**38**	82	0	84	**36**	85	0	82
18	*6,33*	**104**	49	**102**	48	**94**	44	**101**	46	**97**	43
19	*57,14*	**124**	1981	**118**	1952	**110**	1971	**115**	1974	**108**	1963
20	*49,02*	**113**	2397	**110**	2371	**93**	2381	**109**	2385	**95**	2379
21	*70,51*	**106**	2274	**104**	2093	**101**	2117	**103**	2138	**102**	2124
22	*22,17*	**51**	61	**50**	58	**37**	59	**48**	60	**39**	56
23	*6,39*	**68**	52	**63**	47	**44**	48	**57**	49	**46**	43
24	*6,30*	**34**	43	**32**	40	**28**	39	**30**	42	**29**	41
25	*82,44*	**159**	49	**154**	46	**141**	45	**143**	47	**139**	46
26	*12,68*	**153**	482	**151**	471	0	467	**149**	473	**142**	462
27	*3,27*	**47**	171	**46**	167	**41**	158	**45**	169	**42**	155
28	*25,89*	**69**	40	**63**	48	**49**	54	**58**	52	**51**	52
29	*26,01*	**94**	187	**90**	192	**76**	189	**88**	196	0	194
30	*15,48*	**76**	204	**72**	182	**65**	178	**71**	199	**69**	180
31	*78,30*	**105**	2241	**0**	2207	0	2219	**98**	2233	**87**	2203
32	*6,41*	**98**	133	**94**	129	**86**	127	**90**	135	**83**	124
33	*20,08*	**88**	52	**85**	48	**79**	51	**83**	49	**76**	46
34	*8,15*	**54**	137	**50**	124	**41**	121	**48**	128	**43**	127

35	*25,44*	**57**	108	**55**	98	**51**	99	**54**	104	**49**	101
36	*37,83*	**108**	185	**107**	173	**93**	175	**106**	182	**97**	167
37	*56,91*	**69**	252	**64**	241	**52**	236	**60**	249	**54**	232
38	*53,49*	**111**	2104	**109**	2082	0	2071	**103**	2093	**97**	2068
39	*73,06*	**152**	2183	**148**	2124	**135**	2113	**142**	2167	**137**	2104
40	*5,37*	**28**	52	**27**	49	**21**	47	**23**	51	0	50
41	*19,24*	**75**	67	**72**	65	**69**	64	**71**	74	**68**	67
42	*25,67*	**49**	138	0	144	**41**	135	**44**	142	**40**	137
43	*93,18*	**231**	2153	**230**	2114	**216**	2102	**228**	2139	**218**	2093
44	*6,53*	**37**	72	**37**	69	**32**	63	**36**	71	0	61
45	*39,41*	**96**	284	**92**	291	**81**	282	**89**	286	**83**	273
46	*24,37*	**179**	201	**170**	212	**152**	203	**164**	208	**155**	207
47	*12,99*	**68**	146	**65**	127	**52**	116	**63**	137	**51**	122
48	*45,78*	**76**	257	**76**	241	**67**	248	**74**	251	**69**	245
49	*16,04*	**51**	135	**48**	127	**41**	123	**45**	128	**38**	126
50	*26,07*	**67**	182	**65**	163	**46**	169	**53**	174	**47**	173
51	*98,16*	**239**	2173	**233**	2132	**214**	2125	**228**	2150	**217**	2131
52	*3,08*	**28**	34	**24**	33	**18**	25	0	27	**19**	26
53	*38,50*	**77**	242	**69**	239	**63**	237	**65**	240	**61**	239
54	*22,01*	**45**	186	**43**	175	**39**	178	**43**	177	**38**	174
55	*74,19*	**139**	2064	**134**	2017	**126**	2034	**125**	2042	**127**	2037
56	*68,27*	**154**	2038	**151**	1992	**142**	2023	**146**	2025	**139**	2008
57	*15,91*	**53**	142	**52**	133	**49**	129	**51**	137	**47**	128
58	*5,33*	**21**	33	**20**	30	0	31	**18**	32	0	31
59	*40,21*	**42**	409	**40**	399	**38**	397	**39**	402	**35**	395
60	*108,73*	**346**	2585	**337**	2493	**322**	2528	**328**	2541	**324**	2533
61	*60,49*	**115**	364	**113**	351	**107**	352	**112**	357	**108**	354
62	*102,53*	**344**	2599	**342**	2470	**324**	2481	**340**	2493	**325**	2477
63	*15,67*	**101**	153	**99**	159	**85**	160	**97**	164	**86**	157
64	*80,24*	**192**	2191	**187**	2164	**173**	2175	**183**	2177	**177**	2169

Tabela A.21 – Resultados RVNS – AGM *Kruskal* padrão para obtenção de 3 *clusters*

		RVNS									
		RVNS_AGM1		**RVNS_AGM2**		**RVNS_AGM3**		**RVNS_AGM4**		**RVNS_AGM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**93**	123	**84**	119	**61**	117	**79**	120	**55**	115
2	*39,08*	**99**	151	**91**	148	**79**	135	**84**	142	**73**	137
3	*5,72*	**51**	45	**49**	43	**38**	39	**42**	42	**37**	40
4	*26,14*	**73**	159	**66**	154	**44**	144	**57**	151	0	148
5	*7,35*	**49**	64	**43**	57	**35**	49	**41**	54	**34**	51
6	*6,41*	**57**	52	**55**	44	**36**	43	**48**	50	**37**	38
7	*6,17*	**48**	61	**42**	56	**28**	51	**39**	59	**31**	47
8	*4,96*	**42**	53	**40**	50	**29**	47	**41**	51	**30**	40
9	*14,35*	**91**	117	**88**	112	**73**	110	**82**	114	**76**	108
10	*19,78*	**89**	113	**75**	111	**60**	109	**74**	112	**63**	105
11	*10,27*	**102**	135	**94**	127	**85**	120	**89**	128	**81**	119
12	*17,33*	**48**	57	**43**	52	**37**	55	**41**	53	**35**	46
13	*7,42*	**69**	79	**61**	71	**44**	64	**56**	74	**42**	61
14	*5,94*	**42**	114	**39**	112	**33**	105	**38**	111	**33**	104
15	*18,53*	**83**	101	**77**	92	**67**	91	**72**	98	**68**	93
16	*7,09*	**81**	119	**78**	116	**65**	113	**74**	117	**61**	112
17	*10,81*	**56**	84	**49**	81	**38**	83	**42**	84	**35**	80
18	*6,33*	**123**	47	**118**	46	**101**	42	**115**	43	**104**	41
19	*57,14*	**146**	1978	**133**	1943	**118**	1964	**126**	1955	**113**	1923
20	*49,02*	**119**	2356	**115**	2348	**97**	2327	**100**	2332	**99**	2314
21	*70,51*	**128**	2147	**124**	2081	**118**	2095	**121**	2125	**115**	2104
22	*22,17*	**64**	60	**58**	59	**47**	55	**53**	58	**46**	53
23	*6,39*	**77**	51	**69**	50	**54**	42	**62**	48	**51**	41
24	*6,30*	**46**	44	**42**	42	**35**	36	**39**	40	**37**	38
25	*82,44*	**168**	50	**162**	47	**149**	44	**153**	48	**144**	45
26	*12,68*	**189**	475	**171**	469	**155**	465	**168**	470	**153**	461
27	*3,27*	**74**	169	**66**	163	**56**	151	**61**	159	**55**	148

28	***25,89***	**82**	56	**74**	47	**62**	52	**68**	53	**63**	49
29	***26,01***	**110**	193	**99**	189	**81**	185	**94**	191	**82**	187
30	***15,48***	**85**	199	**79**	180	**74**	174	**77**	183	**73**	175
31	***78,30***	**126**	2206	**119**	2184	**96**	2154	**113**	2195	0	2168
32	***6,41***	**111**	130	**108**	128	**90**	118	**104**	126	**92**	117
33	***20,08***	**99**	50	**93**	46	**85**	49	**89**	43	**82**	42
34	***8,15***	**61**	135	**59**	124	**48**	116	**53**	127	**49**	121
35	***25,44***	**84**	109	**75**	100	**63**	95	**72**	102	**67**	97
36	***37,83***	**127**	181	**124**	175	**105**	163	**119**	179	**108**	154
37	***56,91***	**83**	239	**79**	233	**66**	219	**68**	228	**63**	214
38	***53,49***	**118**	2099	**116**	2085	**104**	2014	**109**	2093	**102**	2008
39	***73,06***	**159**	2140	**155**	2120	**141**	2038	**147**	2119	**139**	2027
40	***5,37***	**48**	50	**43**	48	**32**	46	**39**	49	**33**	45
41	***19,24***	**89**	65	**85**	62	**73**	69	**78**	66	**75**	63
42	***25,67***	**72**	148	**61**	141	0	132	**52**	140	**49**	129
43	***93,18***	**289**	2006	**272**	1990	**245**	1985	**267**	1993	**238**	1987
44	***6,53***	**64**	74	**53**	63	**46**	58	**51**	69	**45**	55
45	***39,41***	**118**	291	**115**	285	**93**	267	**104**	279	**97**	261
46	***24,37***	**199**	194	**191**	177	**165**	186	**165**	188	**163**	183
47	***12,99***	**72**	138	**68**	122	**59**	110	**64**	129	**60**	115
48	***45,78***	**89**	255	**81**	244	**74**	236	**78**	247	**71**	228
49	***16,04***	**69**	131	**57**	123	**49**	121	**55**	126	**48**	123
50	***26,07***	**75**	175	**69**	169	**55**	165	**67**	172	**53**	162
51	***98,16***	**258**	2117	**241**	2099	**220**	2093	**236**	2108	**222**	2085
52	***3,08***	**30**	33	**28**	32	**19**	24	0	28	**21**	23
53	***38,50***	**89**	240	**83**	241	**74**	235	**79**	242	**69**	231
54	***22,01***	**55**	181	**52**	173	**41**	176	**48**	175	**42**	171
55	***74,19***	**154**	2045	**148**	2026	**135**	2017	**141**	2028	**133**	2010
56	***68,27***	**178**	2013	**170**	1985	**150**	1993	**156**	1999	**148**	1997
57	***15,91***	**73**	134	**68**	129	**57**	118	**63**	127	**56**	116
58	***5,33***	**28**	32	**22**	31	0	29	**19**	30	**17**	29
59	***40,21***	**57**	404	**53**	397	**42**	386	**48**	395	**44**	383
60	***108,73***	**369**	2549	**364**	2528	**350**	2514	**359**	2536	**351**	2508
61	***60,49***	**138**	359	**134**	350	**123**	336	**131**	348	**127**	337
62	***102,53***	**373**	2601	**362**	2495	**348**	2473	**359**	2492	**354**	2451
63	***15,67***	**114**	165	**109**	152	**92**	159	**101**	163	**93**	152
64	***80,24***	**216**	480	**208**	2159	**194**	2134	**199**	2156	**196**	2123

Tabela A.22 – Resultados SVNS – Grafo Original para obtenção de 3 *clusters*

		SVNS									
		SVNS1		**SVNS2**		**SVNS3**		**SVNS4**		**SVNS5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	***22,54***	**15**	141	7	135	0	128	4	139	0	124
2	***39,08***	**18**	193	9	184	2	167	**11**	172	0	163
3	***5,72***	**17**	49	10	41	1	37	6	44	0	38
4	***26,14***	**23**	198	**14**	185	4	168	9	191	0	176
5	***7,35***	**18**	68	2	59	0	50	0	62	0	53
6	***6,41***	6	62	0	47	0	44	0	51	0	42
7	***6,17***	4	57	0	45	0	38	0	48	0	36
8	***4,96***	5	49	1	36	0	30	2	38	0	28
9	***14,35***	**22**	131	4	122	0	115	0	129	0	119
10	***19,78***	**31**	108	3	94	0	71	0	83	0	67
11	***10,27***	**27**	149	**11**	123	2	101	5	118	0	104
12	***17,33***	**38**	77	**17**	68	4	63	10	74	0	59
13	***7,42***	7	91	0	82	0	79	0	88	0	75
14	***5,94***	**23**	139	4	128	0	115	0	135	3	108
15	***18,53***	**21**	92	**16**	79	1	73	8	87	0	62
16	***7,09***	**12**	119	1	98	0	86	1	102	0	91
17	***10,81***	**28**	122	**13**	86	0	72	**11**	99	0	63
18	***6,33***	**16**	56	2	47	0	47	0	48	0	45
19	***57,14***	**65**	2289	**43**	2269	**15**	2265	0	2271	0	2284

20	*49,02*	**50**	2406	**39**	2381	10	2371	**18**	2395	0	2379
21	*70,51*	**64**	2139	**50**	417	**30**	2104	**14**	2122	0	2101
22	*22,17*	**33**	61	8	54	0	43	1	58	0	39
23	*6,39*	10	57	0	50	0	41	0	52	0	41
24	*6,30*	**13**	55	1	49	0	42	0	50	0	40
25	*82,44*	**18**	58	**11**	42	0	39	0	44	0	37
26	*12,68*	**127**	541	**81**	527	**21**	470	**68**	541	4	498
27	*3,27*	**28**	193	**19**	182	8	178	**15**	183	0	164
28	*25,89*	8	50	0	33	0	30	0	36	0	30
29	*26,01*	**33**	214	**29**	192	10	181	**18**	204	0	173
30	*15,48*	**20**	233	**17**	203	4	194	10	217	0	189
31	*78,30*	**37**	127	9	101	8	82	**15**	103	2	90
32	*6,41*	**94**	2579	**71**	2512	0	2519	0	2521	**15**	2501
33	*20,08*	**11**	55	0	44	0	43	**16**	48	10	41
34	*8,15*	**38**	166	3	148	0	137	0	152	0	125
35	*25,44*	**25**	141	7	119	0	102	0	127	0	98
36	*37,83*	**30**	202	**22**	183	8	169	**16**	194	6	175
37	*56,91*	**41**	314	**37**	291	10	284	**16**	300	8	263
38	*53,49*	**59**	2337	**41**	2282	0	2263	**31**	2291	0	2269
39	*73,06*	**57**	2344	**42**	2329	**20**	2321	**22**	2337	0	2308
40	*5,37*	**15**	54	3	38	0	32	0	41	0	31
41	*19,24*	**23**	67	7	54	0	53	1	56	0	56
42	*25,67*	**31**	150	**28**	137	8	124	**12**	141	4	112
43	*93,18*	**145**	2226	**94**	2192	**44**	2185	**23**	2218	0	2171
44	*6,53*	**12**	59	0	49	0	44	1	53	0	46
45	*39,41*	**47**	296	**33**	281	**12**	276	**18**	288	1	269
46	*24,37*	**22**	185	**17**	157	5	152	8	169	0	147
47	*12,99*	**39**	162	**29**	138	**14**	141	**21**	149	0	135
48	*45,78*	**44**	353	**35**	340	**18**	338	**22**	344	1	321
49	*16,04*	**35**	151	**12**	136	6	110	8	141	0	114
50	*26,07*	**17**	234	10	203	9	209	**11**	218	2	199
51	*98,16*	**182**	2709	**101**	2672	**69**	2664	**75**	2693	0	2658
52	*3,08*	10	48	0	30	0	30	0	34	0	26
53	*38,50*	**33**	261	**21**	245	10	218	**15**	252	4	223
54	*22,01*	**22**	195	10	166	0	139	1	173	0	152
55	*74,19*	**78**	2462	**59**	2448	0	2421	0	2457	1	2427
56	*68,27*	**76**	2456	**48**	2433	0	2415	**40**	2441	1	2419
57	*15,91*	**29**	144	8	127	9	111	5	133	0	118
58	*5,33*	10	51	2	39	0	31	0	40	0	30
59	*40,21*	**32**	429	**27**	416	**11**	374	**21**	428	0	377
60	*108,73*	**201**	2814	**90**	2783	**88**	2773	**95**	2796	0	2760
61	*60,49*	**65**	371	**45**	352	**39**	347	**40**	354	0	339
62	*102,53*	**184**	2804	**101**	2780	**94**	2745	0	2798	10	2765
63	*15,67*	**33**	152	**12**	146	4	138	8	151	0	128
64	*80,24*	**72**	2511	**67**	504	**55**	2491	**45**	2500	0	2482

Tabela A.23 – Resultados SVNS – AGM *kruskal* modificado para obtenção de 3 *clusters*

		SVNS									
		SVNS_AGMM1		**SVNS_AGMM2**		**SVNS_AGMM3**		**SVNS_AGMM4**		**SVNS_AGMM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**57**	131	**43**	127	**26**	119	**38**	128	**24**	120
2	*39,08*	**79**	165	**74**	156	**61**	148	**66**	163	**58**	143
3	*5,72*	**30**	51	**29**	48	0	44	**28**	49	**24**	45
4	*26,14*	**44**	169	**41**	162	**34**	155	**37**	165	**31**	157
5	*7,35*	**37**	58	**32**	51	**16**	42	**29**	53	**18**	44
6	*6,41*	**46**	52	**43**	48	**38**	43	**44**	51	0	47
7	*6,17*	**33**	62	**30**	60	**24**	54	**29**	61	**21**	53
8	*4,96*	**25**	57	**23**	53	**22**	51	0	55	**20**	49
9	*14,35*	**81**	123	**78**	117	**61**	115	**77**	121	**65**	113
10	*19,78*	**64**	110	0	103	**43**	96	**56**	108	**47**	100
11	*10,27*	**85**	149	**84**	137	**78**	133	**82**	144	**77**	131
12	*17,33*	**35**	61	**33**	55	**22**	54	**28**	58	**21**	51

13	*7,42*	**38**	85	**37**	81	**29**	75	**34**	83	**30**	79
14	*5,94*	**24**	122	**21**	119	**20**	108	0	120	0	112
15	*18,53*	**71**	101	**69**	97	0	98	**63**	99	**56**	95
16	*7,09*	**67**	119	**64**	114	**51**	108	**55**	116	**48**	110
17	*10,81*	**42**	85	**41**	80	**39**	80	**0**	83	**37**	79
18	*6,33*	**99**	48	**93**	44	**78**	45	**85**	46	**81**	43
19	*57,14*	**123**	1996	**119**	1983	**112**	1973	**118**	1987	**104**	1981
20	*49,02*	**114**	2400	**112**	2391	**97**	2364	**109**	2396	0	2373
21	*70,51*	**108**	2285	**105**	2107	**98**	2098	**104**	2215	**97**	2085
22	*22,17*	**57**	66	**49**	49	**41**	41	**43**	53	**38**	45
23	*6,39*	**61**	53	**60**	55	0	54	**58**	56	**43**	52
24	*6,30*	**32**	44	**31**	41	**21**	39	**29**	42	**22**	38
25	*82,44*	**151**	49	**144**	42	**129**	40	**138**	47	**133**	40
26	*12,68*	**144**	490	**139**	486	**121**	475	**127**	489	**119**	473
27	*3,27*	**43**	183	**34**	172	**33**	164	**36**	177	**28**	159
28	*25,89*	**68**	41	**59**	30	**51**	33	**55**	38	**49**	35
29	*26,01*	**90**	193	0	181	**76**	183	**81**	187	**73**	180
30	*15,48*	**71**	202	**68**	197	**57**	198	**63**	199	**54**	195
31	*78,30*	**107**	2263	**101**	2251	0	2225	**97**	2257	**92**	2235
32	*6,41*	**94**	125	**84**	108	**71**	116	**75**	114	**62**	110
33	*20,08*	**85**	57	**82**	50	**73**	48	**76**	54	**69**	45
34	*8,15*	**49**	142	**44**	132	**29**	127	**35**	139	**31**	135
35	*25,44*	**56**	109	**53**	100	**38**	109	**48**	104	0	108
36	*37,83*	**98**	193	**92**	185	**72**	178	**85**	187	**73**	183
37	*56,91*	**67**	264	**64**	255	**52**	241	0	258	**55**	244
38	*53,49*	**101**	2113	**99**	2104	**81**	2097	**95**	2109	**83**	2085
39	*73,06*	**143**	2196	**141**	2182	**132**	2185	**136**	2187	**128**	2183
40	*5,37*	**22**	55	**19**	49	**13**	50	**19**	52	0	47
41	*19,24*	**68**	69	**62**	60	**51**	62	**53**	64	**47**	58
42	*25,67*	**47**	141	**41**	132	**27**	135	**38**	139	**23**	133
43	*93,18*	**225**	2160	**220**	2144	**203**	2149	**218**	2155	**204**	2153
44	*6,53*	**31**	71	**28**	67	0	65	**27**	69	0	67
45	*39,41*	**89**	297	**84**	284	**62**	271	**73**	285	**67**	252
46	*24,37*	**178**	203	**173**	189	**151**	187	**164**	192	**152**	183
47	*12,99*	**67**	154	**62**	131	**38**	128	**58**	144	**42**	125
48	*45,78*	**73**	261	**70**	249	**63**	242	**67**	253	**59**	251
49	*16,04*	**48**	142	**41**	133	**24**	137	**37**	137	**28**	139
50	*26,07*	**66**	197	**59**	174	**41**	163	**54**	182	**45**	159
51	*98,16*	**218**	2190	**214**	2152	**193**	2169	**208**	2173	**197**	2170
52	*3,08*	**22**	36	**20**	32	0	34	**19**	35	0	33
53	*38,50*	**75**	258	**69**	243	**52**	239	**65**	244	**56**	235
54	*22,01*	**44**	194	**38**	182	**29**	174	**36**	185	**30**	171
55	*74,19*	**133**	2096	**127**	2041	**120**	2069	0	2074	**121**	2072
56	*68,27*	**151**	2042	**149**	2008	**129**	2031	**146**	2035	**133**	2028
57	*15,91*	**46**	143	**45**	131	**42**	135	**44**	138	**41**	133
58	*5,33*	**18**	36	**17**	35	0	36	**15**	37	0	39
59	*40,21*	**46**	417	**42**	404	**32**	408	**38**	412	**33**	411
60	*108,73*	**337**	2609	**325**	2503	**313**	2571	**318**	2582	**309**	2568
61	*60,49*	**104**	376	**101**	354	**87**	360	**99**	364	**83**	357
62	*102,53*	**339**	2613	**328**	2597	**315**	2593	0	2610	**317**	2599
63	*15,67*	**98**	168	**92**	163	**72**	157	**86**	165	**75**	153
64	*80,24*	**194**	2297	**152**	2271	**155**	2264	**175**	2283	**169**	2269

Tabela A.24 – Resultados SVNS – AGM *kruskal* padrão para obtenção de 3 *clusters*

		SVNS									
		SVNS_AGM1		**SVNS_AGM2**		**SVNS_AGM3**		**SVNS_AGM4**		**SVNS_AGM5**	
I	***Solução best***	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**	**Gap**	**T(s)**
1	*22,54*	**65**	122	**61**	117	**29**	106	**41**	119	**28**	108
2	*39,08*	**83**	145	**85**	142	**66**	131	**74**	142	**63**	134
3	*5,72*	**59**	47	**54**	48	**25**	41	**35**	39	**27**	42
4	*26,14*	**48**	173	**52**	153	**39**	151	**40**	160	**38**	149
5	*7,35*	**44**	55	**36**	51	**19**	41	**38**	49	**21**	38

6	***6,41***	**71**	54	**68**	49	**52**	40	**69**	42	**55**	43
7	***6,17***	**69**	69	**50**	58	**39**	51	**40**	63	**33**	49
8	***4,96***	**70**	59	**65**	53	0	49	**43**	58	**34**	44
9	***14,35***	**157**	121	**96**	119	**81**	114	**154**	115	**78**	111
10	***19,78***	**113**	107	**99**	104	**55**	93	**63**	101	**51**	97
11	***10,27***	**145**	139	**108**	140	**98**	127	**107**	135	**93**	124
12	***17,33***	**102**	67	**68**	64	**35**	55	**51**	63	**38**	52
13	***7,42***	**53**	84	**37**	85	**27**	79	**40**	80	**29**	72
14	***5,94***	**96**	118	**81**	117	**68**	106	**74**	113	**65**	110
15	***18,53***	**82**	110	**77**	95	**66**	97	**81**	99	**68**	90
16	***7,09***	**89**	116	**75**	114	**41**	105	**69**	111	**44**	107
17	***10,81***	**151**	93	**118**	87	**101**	82	**123**	81	**106**	84
18	***6,33***	**128**	64	**112**	47	**85**	44	**107**	52	**99**	45
19	***57,14***	**164**	1985	**151**	1974	**142**	1960	**155**	1977	0	1963
20	***49,02***	**156**	2301	**140**	2285	**133**	2235	**141**	2296	**125**	2251
21	***70,51***	**145**	2205	**139**	2115	**110**	2008	**136**	2173	**114**	1998
22	***22,17***	**73**	57	**66**	48	**49**	37	**58**	44	**45**	40
23	***6,39***	**81**	56	**79**	53	**65**	52	**74**	53	**68**	51
24	***6,30***	**63**	42	**58**	39	**37**	33	**50**	40	**39**	35
25	***82,44***	**205**	48	**177**	45	**161**	37	**184**	44	**167**	32
26	***12,68***	**169**	483	**156**	479	**141**	462	**148**	481	**135**	451
27	***3,27***	**62**	174	**45**	167	**42**	159	**46**	162	**39**	153
28	***25,89***	**85**	39	**79**	36	**74**	30	0	33	**67**	32
29	***26,01***	**101**	190	**96**	174	**88**	167	**104**	185	**90**	163
30	***15,48***	**90**	198	**78**	189	**69**	185	**72**	194	**63**	182
31	***78,30***	**156**	2185	**121**	2159	**117**	2138	**122**	2170	**115**	2145
32	***6,41***	**143**	115	**97**	112	**88**	109	**96**	98	**84**	104
33	***20,08***	**72**	61	**99**	52	**86**	46	**91**	54	**84**	43
34	***8,15***	**57**	139	**60**	129	**44**	120	**38**	133	**42**	124
35	***25,44***	**74**	112	**69**	99	**48**	95	**63**	108	**40**	93
36	***37,83***	**130**	185	**113**	172	**88**	157	**96**	174	**85**	161
37	***56,91***	**82**	251	**85**	243	**71**	236	**83**	249	**70**	233
38	***53,49***	**149**	2104	**112**	2057	**96**	2012	**108**	2097	**91**	2001
39	***73,06***	**158**	2167	**165**	2145	**142**	2124	**152**	2159	**135**	2114
40	***5,37***	**44**	63	**31**	44	**24**	42	**33**	54	**25**	39
41	***19,24***	**89**	65	**85**	61	0	58	**71**	59	**53**	55
42	***25,67***	**55**	139	**54**	128	**39**	125	**46**	133	**37**	122
43	***93,18***	**251**	2144	**238**	2112	**242**	2083	**233**	2128	**241**	2098
44	***6,53***	**40**	70	**37**	74	**18**	65	**31**	68	**15**	63
45	***39,41***	**105**	277	**93**	268	**77**	259	**79**	275	**74**	241
46	***24,37***	**246**	196	**214**	189	**171**	185	**188**	192	**173**	181
47	***12,99***	**81**	142	**74**	139	**59**	121	**63**	137	**57**	118
48	***45,78***	**89**	251	**84**	252	**72**	238	**79**	244	**65**	247
49	***16,04***	**64**	141	**48**	138	**31**	130	**44**	129	**33**	134
50	***26,07***	**82**	184	**69**	175	**50**	161	**65**	169	**57**	163
51	***98,16***	**257**	2135	**245**	2107	**212**	2023	**236**	2128	**215**	2014
52	***3,08***	**44**	34	**41**	34	**25**	31	**39**	30	**24**	32
53	***38,50***	**82**	255	**77**	239	**61**	229	**73**	241	**68**	228
54	***22,01***	**49**	184	**45**	171	**39**	169	**41**	173	**37**	163
55	***74,19***	**155**	2088	**153**	2065	**136**	2054	**149**	2071	**138**	2059
56	***68,27***	**179**	2053	**162**	2023	**141**	2017	**157**	2039	**144**	2014
57	***15,91***	**74**	146	**69**	129	**53**	127	**66**	135	0	128
58	***5,33***	**33**	42	**27**	38	**14**	33	**24**	39	**12**	35
59	***40,21***	**68**	410	**58**	401	**46**	395	**55**	402	**45**	397
60	***108,73***	**362**	2599	**349**	2524	**323**	2508	**341**	2537	**315**	2503
61	***60,49***	**128**	368	**117**	355	**93**	352	**104**	366	**95**	349
62	***102,53***	**356**	2627	**342**	2593	**321**	2564	**339**	2615	**329**	2571
63	***15,67***	**112**	172	**99**	163	**84**	155	**93**	164	**81**	152
64	***80,24***	**217**	2246	**193**	2201	**168**	2218	**188**	2214	**179**	2185