PRODUÇÃO ONLINE
REVISTA CIENTÍFICA ELETRÔNICA DE ENGENHARIA DE PRODUÇÃO
ISSN 1676-1901

MÉTODOS HEURÍSTICOS USANDO GRASP, RECONEXÃO DE CAMINHOS E BUSCA EM VIZINHANÇA VARIÁVEL PARA O PROBLEMA DO CAIXEIRO VIAJANTE COM GRUPAMENTOS

HEURISTIC METHODS USING GRASP, PATH RELINKING AND VARIABLE NEIGHBORHOOD SEARCH FOR THE CLUSTERED TRAVELING SALESMAN PROBLEM

Mário Mestria * E-mail: mmestria@ifes.edu.br
*Instituto Federal do Espírito Santo (IFES), Vitória, ES

Resumo: O Problema do Caixeiro Viajante com Grupamentos (PCVG) é uma generalização do Problema do Caixeiro Viajante (PCV) em que o conjunto de vértices é particionado em grupos disjuntos e o objetivo é encontrar um ciclo Hamiltoniano de custo mínimo tal que os vértices em cada grupo são visitados na forma contígua. O PCVG é *NP*-difícil e nesse contexto são propostos métodos heurísticos para o PCVG usando GRASP, Reconexão de Caminhos e Método de Descida em Vizinhança Variável (MDVV). Os métodos heurísticos foram testados usando instancias Euclidianas com até 2000 vértices e grupos variando entre 4 a 15 vértices. Os testes computacionais foram executados para comparar o desempenho dos métodos heurísticos com um algoritmo exato usando o software CPLEX Paralelo. Os resultados computacionais mostraram que o método heurístico híbrido usando MDVV sobressaiu em relação aos outros métodos.

Palavras-chave: Pesquisa Operacional. Inteligência Computacional. Heurísticas. Reconexão de Caminhos. Método de Descida em Vizinhança Variável.

Abstract: The Clustered Traveling Salesman Problem (CTSP) is a generalization of the Traveling Salesman Problem (TSP) in which the set of vertices is partitioned into disjoint clusters and objective is to find a minimum cost Hamiltonian cycle such that the vertices of each cluster are visited contiguously. The CTSP is NP-hard and, in this context, we are proposed heuristic methods for the CTSP using GRASP, Path Relinking and Variable Neighborhood Descent (VND). The heuristic methods were tested using Euclidean instances with up to 2000 vertices and clusters varying between 4 to 150 vertices. The computational tests were performed to compare the performance of the heuristic methods with an exact algorithm using the Parallel CPLEX software. The computational results showed that the hybrid heuristic method using VND outperforms other heuristic methods.

Keywords: Operations Research. Computational Intelligence. Heuristics. Path Relinking. Variable Neighborhood Descent.

1 INTRODUÇÃO

O Problema do Caixeiro Viajante com Grupamentos (PCVG) é uma extensão do Problema do Caixeiro Viajante (PCV) (REINELT, 1994), cujo objetivo é encontrar um ciclo Hamiltoniano de custo mínimo que passe por todos os vértices do grafo e

visite os vértices de cada grupo de forma contígua. No caso do número de vértices de cada grupo seja igual a um, o PCVG se reduz ao PCV. Assim, podemos afirmar que o PCVG pertence à classe *NP*-difícil e com isso limita o uso exclusivo de métodos exatos. Nesse contexto, a metodologia adotada para solucionar o PCVG utilizará métodos heurísticos.

Existem duas versões para o PCVG, a primeira mais explorada, supõe-se uma prefixação da sequência de visita dos grupos. Na segunda, a sequência de visitas não é pré-fixada e a escolha da sequência é deixada para o algoritmo. A segunda versão é mais complexa, pois na busca da melhor sequência de visitas aos vértices, devemos também analisar a melhor sequência de visitas aos grupos. Neste trabalho, são propostos métodos heurísticos para solucionar o PCVG considerando a segunda versão, que de fato encontramos situações práticas.

O PCVG foi proposto em Chisman (1975) para resolver um problema real de roteamento automático em sistemas de armazenamento. O problema foi resolvido, para instâncias de pequeno porte, transformando o PCVG num PCV, adicionando uma penalidade de valor *M* aos custos das arestas intergrupos, através de um algoritmo *branch-and-bound*. Aplicações do PCVG podem ser encontradas em planejamento da produção (LOKIN, 1979), despacho de veículos de emergência, (WEINTRAUB *et al*., 1999), transações comerciais evolvendo supermercados, lojas e fornecedores de mercadorias (GHAZIRI; OSMAN, 2003), desfragmentação de discos rígidos e sistemas de manufaturas (LAPORTE; PALEKAR, 2002).

Como observamos, em vários ambientes de produção existem diversas aplicações que podem ser modeladas através do PCVG. Neste sentido a solução para o PCVG contribuirá para reduzir os custos no transporte e produção, pois estaremos minimizando os custos envolvidos entre os caminhos dos vértices.

Limites inferiores utilizando Relaxação *Lagrangeana* foram desenvolvidos em Jongens e Volgenant (1985) para o PCVG. O método para obtenção do limite inferior baseou-se na árvore geradora mínima desenvolvida primeiramente para o PCV. Neste trabalho foram criados vários conjuntos de instâncias testes variando o número de vértices de 80 a 150 com diferentes números de grupos. O algoritmo de Jongens e Volgenant (1985) obteve para instâncias testes, *gap* médio para os limites inferiores com valores iguais a 0,35%.

Em Laporte, Potvin e Quilleret (1996) uma Busca Tabu combinada com uma fase de diversificação usando um Algoritmo Genético (AG) foi proposta para o PCVG com prefixação da sequência de visita dos grupos. Os resultados mostram que a Busca Tabu foi competitiva comparada com o (AG) de Potvin e Guertin (1995) e se mostrou superior a um procedimento de inserção com pós-otimização de Gendreau, Laporte e Potvin (1994), mas em contrapartida requereu um esforço computacional maior.

Os autores Potvin e Guertin (1996) desenvolveram um AG para o PCVG e seu desempenho foi comparado com os resultados obtidos pela heurística GENIUS (GENDREAU; LAPORTE; POTVIN, 1994) e com os limites inferiores obtidos em (JONGENS; VOLGENANT, 1985). Esse AG resolveu problemas com até 500 vértices e grupos com quatro ou dez vértices. Os resultados ficaram em até 5,5% do limite inferior. O tempo computacional gasto pelo AG foi maior do que pela heurística GENIUS.

No trabalho de Ding, Cheng e He (2007), outro AG foi desenvolvido utilizando-se o critério de visita dos vértices de cada grupo de forma aleatória sem estabelecer uma relação com as distâncias entre os vértices. A escolha da sequência de visita de cada grupo também foi realizada de forma aleatória. O algoritmo foi implementado em dois níveis chamado de nível baixo e nível alto. No nível mais baixo, o AG acha um ciclo Hamiltoniano para cada grupo. No nível mais alto, o algoritmo escolhe aleatoriamente uma aresta a ser excluída no ciclo de cada grupo e simultaneamente determina a sequência de visita dos grupos de forma aleatória.

Foi observado na literatura que para a resolução do PCVG, os trabalhos partem do princípio de uma prefixação da sequência de visita dos grupos ou são algoritmos α-aproximados (GENDREAU; LAPORTE; POTVIN, 1994), (GENDREAU; LAPORTE; HERTZ, 1997), (ANILY; BRAMEL; HERTZ, 1999) e (GUTTMANN-BECK et al., 2000) que requerem escolhas dos vértices extremos dos grupos.

Dentre as metaheurísticas existentes na literatura, uma que se destaca é: *Greedy Randomized Adaptive Search Procedures* (GRASP), um algoritmo iterativo no qual cada iteração consiste de duas fases. Existe a fase de construção e a fase de busca local (RESENDE et al., 2010) e (FESTA; RESENDE, 2011). Na fase de construção uma solução viável é construída e na fase de busca local sua vizinhança

é investigada até um mínimo local ser encontrado. A melhor solução obtida ao final de um determinado número de iterações é retornada como a solução do algoritmo. Observamos que o GRASP obteve melhor desempenho com a introdução de Reconexão de Caminhos (ALOISE; RIBEIRO, 2011), (DENG; BARD, 2011) e (MATEUS; RESENDE; SILVA, 2011).

Em Hernández-Pérez, Rodríguez-Martín e Salazar-González (2009) foi apresentado um GRASP combinado com *Variable Neighborhood Descent* (VND) (HANSEN E MLADENOVIĆ, 2003) e (DUARTE et al., 2012) para um problema de roteamento. A estrutura do VND não é complexa, de fácil implementação e suas estruturas de vizinhanças (LAMB, 2012) podem ser alteradas aumentando ou diminuindo sua complexidade para se atingir um compromisso entre qualidade da solução e tempo computacional empregado.

Em Nagano e Mesquita (2012) foram utilizados métodos heurísticos para solucionar um problema de programação da produção, problema classificado como *NP*-difícil. Os autores Santos, Souza Jr. e Bouzada (2012) mostraram a dificuldade em ser resolver um problema modelado como programação inteira. Trata-se do problema de logística que envolve alocação, transporte e distribuição de suprimentos. No trabalho de Moraes e Nagano (2011) foi utilizado uma metodologia baseada em algoritmos genéticos para otimizar recursos financeiros de maneira a obter o melhor resultado para as organizações. Especificamente, o objetivo do trabalho foi desenvolver uma metodologia de otimização do saldo disponível de caixa com base nas premissas de minimização do custo. Em Anzanello e Fogliatto (2011), os autores desenvolveram e testaram quatro heurísticas para programação de tarefas em equipes de trabalhadores paralelos não relacionados. O objetivo do trabalho foi de minimizar o tempo demandado para término de tarefas caracterizadas pela necessidade de aprendizado dos trabalhadores em ambientes de Customização em Massa.

Como podemos observar em várias áreas, que existe a necessidade de ser resolver problemas de otimização cuja característica contém um espaço de solução enorme. Nestes problemas, nem sempre os métodos exatos alcançam soluções ótimas. Em alguns casos, as soluções exatas envolveriam tempos computacionais altos, da ordem de dias para solucioná-los, prejudicando nas tomadas de decisões,

por exemplo, numa linha de produção. Neste sentido, as heurísticas contribuem para soluções de problemas de otimização buscando qualidade nas soluções geradas com tempo computacional baixo.

Neste artigo foram desenvolvidos seis métodos heurísticos baseadas no GRASP para resolver o PCVG. A primeira versão é um GRASP tradicional (G). Nas demais versões, um módulo de busca intensiva que utiliza conceitos de memória adaptativa através de Reconexão de Caminhos é incorporado ao GRASP. Na segunda versão (GRC1) é adicionado um módulo de Reconexão de Caminhos (RC1) ao (G) executado depois das iterações de (G). Na terceira (GRC2) utiliza-se uma Reconexão de Caminhos menos intensa (RC2) durante as iterações do (G) e a na quarta (GRC1RC2) adiciona-se a Reconexão de Caminhos RC1 ao final de GRC2. Na quinta versão (GRC3) é utilizada a Reconexão de Caminhos (RC3) semelhante ao RC1, mas ativada durante as iterações do (G). Na última versão (GRC3-VND) é utilizada a RC3 e na busca local foi utilizado o VND, formando um método heurístico híbrido.

O artigo é estruturado conforme as seguintes seções. A segunda seção descreve a definição do PCVG. A terceira seção descreve a metodologia baseada em heurísticas para resolução do PCVG. A quarta seção são mostrados os resultados computacionais, o desempenho entre os métodos heurísticos e um método exato e as análises das distribuições de probabilidade. Na última seção, as considerações finais são apresentadas.

2 DEFINIÇÃO DO PROBLEMA

Uma definição para o PCVG, baseado em grafos pode ser dado a seguir: seja $G = (V, E)$ um grafo completo com conjunto de vértices $V=\{v_1,v_2,\ldots,v_n\}$ e um conjunto de arestas $E=\{(v_i,v_j):v_i,\ v_j\ \varepsilon\ V,\ i \neq j\ \}$. O conjunto de vértices V é particionado dentro de grupos disjuntos $V_1,\ V_2,...,\ V_m$. Assumindo que um custo não negativo c_{ij} é associado com a aresta $E=(v_i,\ v_j)$, o PCVG consiste em determinar um ciclo Hamiltoniano de custo mínimo em G, tal que os vértices de cada grupo são visitados contiguamente.

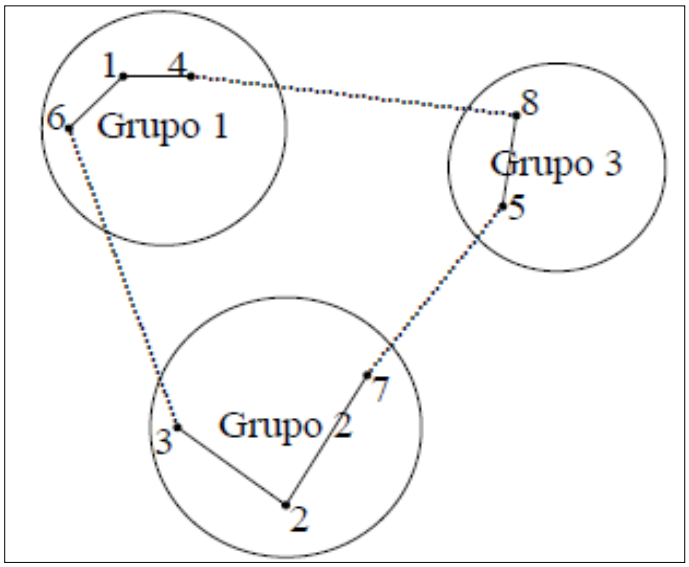
Teorema 1. *O PCVG, formando um ciclo Hamiltoniano, iniciando num vértice inicial v_i, passando por todos os vértices e por todos os grupos numa sequência contígua, inclusive pelo vértice final v_f e retornando ao vértice inicial v_i, é NP-difícil.*

Prova: Podemos reduzir o PCVG ao PVC e sabemos por hipótese que o PCV é *NP*-difícil (REINELT, 1994), então poderemos concluir que o PCVG será *NP*-difícil. Dado uma instância ***I*** do PCVG, representada por $PCVG_I$, com cada grupo contendo somente um vértice. Transformamos o $PCVG_I$ em um PCV_I. A solução ótima para o PCV_I é solução ótima para o $PCVG_I$ correspondente. Como o PCV_I é *NP*-difícil, consequentemente o $PCVG_I$, também é *NP*-difícil. A tese é válida para o $PCVG_I$, um caso reduzido, portanto podemos concluir que para o caso geral, PCVG é *NP*-difícil.

Para exemplificar o PCVG, temos o Grupo 1 representado por G_1, o Grupo 2 por G_2 e Grupo 3 por G_3, contendo seus respectivos vértices: G_1={1,4,6}, G_2={2,3,7} e G_3={5,8}. Uma solução para este exemplo é mostrado pela Figura 1 formando um ciclo Hamiltoniano que passa por todos os vértices, mas visitando primeiramente todos os vértices de cada grupo contiguamente.

As linhas tracejadas mostram as ligações entre os grupos através das arestas intergrupos e as linhas cheias, as ligações dentro do grupo, arestas intragrupos.

Figura 1 – Uma solução para o PCVG



3 METODOLOGIAS HEURÍSTICAS PARA SOLUCIONAR O PCVG

Para solucionar o PCVG utilizamos uma metodologia através de heurísticas que é constituída de forma básica de três fases: na primeira fase é construída uma solução, na segunda fase uma busca local é realizada e na terceira fase um mecanismo de memória é utilizado. Seis heurísticas foram desenvolvidas para o PCVG genérico onde a sequência de visitas aos grupos de vértices não é pré-fixada. A primeira versão (G) utiliza a estrutura do GRASP tradicional. Na etapa de construção, utilizamos uma adaptação do método de Inserção mais Próxima com penalização nas arestas cujos nós extremos pertençam a subconjuntos V_i distintos e na busca local, é utilizado o método 2-*Opt*.

Apesar dos bons resultados obtidos pelo GRASP em diferentes problemas de otimização, um dos limitantes deste método na sua forma clássica é a ausência de utilização de memória entre suas iterações, ou seja, a cada nova iteração, uma nova solução é obtida sem utilizar informação das soluções obtidas em iterações anteriores. Neste trabalho propomos a utilização de uma memória adaptativa no GRASP através da Reconexão de Caminhos (RC) (GLOVER, 1996). A RC consiste em explorar soluções intermediárias entre duas soluções extremas de boa qualidade. Para tanto, definidas as soluções origem s_o e destino s_d, para gerar cada solução intermediária, movimentos são selecionados de forma a introduzir na solução corrente (inicialmente s_o), atributos presentes na solução destino (s_d) (RESENDE et al., 2010).

A segunda versão (GRC1) introduz em (G) a Reconexão de Caminhos ao final de sua execução. Durante a execução do GRASP é gerado e atualizado um conjunto elite (CE) que armazena as *num_eli* melhores soluções distintas encontradas pelo GRASP. Tal procedimento equivale a usar uma memória que armazene em CE informações relevantes encontradas em iterações passadas. Como este conjunto sofre atualizações sempre que uma nova solução de boa qualidade é encontrada entre as iterações do GRASP, esta memória é também adaptativa. Após a finalização das iterações do GRASP, realizamos a RC entre todos os pares de solução do conjunto elite. Denotamos esta RC como RC1. Foi observado em trabalhos mais recentes (ALOISE; RIBEIRO, 2011), (DENG; BARD,

2011) e (MATEUS; RESENDE; SILVA, 2011) que a utilização de memória aplicada sobre um CE permite melhorar a qualidade das soluções produzidas pelos algoritmos.

A terceira versão (GRC2) incorpora à (G) um módulo de RC mais suave que a RC1 a cada iteração GRASP. A RC neste caso é realizada a cada iteração GRASP entre cada solução produzida pela busca local com uma solução extraída do conjunto de soluções elites (CE), conjunto das *num_eli* melhores soluções distintas geradas até o momento pelo GRASP, e denotamos esta RC por RC2. Na quarta heurística (GRC1RC2) é introduzido a RC2 durante a execução das iterações do GRASP e a RC1 ao final do GRASP. Esta heurística proporcionará avaliar a associação de ambas reconexões de caminhos (RC1 e RC2) em comparação a heurística tradicional (G) que não incorpora memória. Na quinta versão (GRC3) é introduzida a RC3 semelhante a RC1, mas esta é ativada durante as iterações do GRASP sempre que o CE for totalmente renovado. A RC3 é realizada em cada iteração do GRASP somente quando mudam todas as soluções do CE. Na sexta versão (GRC3-VND) utiliza a Reconexão de Caminhos RC3 e VND na fase de busca local substituindo o método 2-*Opt*.

Foram utilizadas quatro estruturas de vizinhanças no VND. A primeira (N_1) é obtida através de um movimento de deslocamento do vértice pertencente ao ciclo de uma posição para outra, denominada de 1-*Shift*. Para definir a segunda estrutura (N_2) precisam-se definir dois tipos de procedimentos: *Drop* e *Cheap*. O procedimento *Drop* varre o ciclo de uma solução e retira deste, o vértice que proporcionará a maior economia com sua retirada. O procedimento *Cheap* insere um vértice não pertencente ao ciclo parcial, entre dois vértices adjacentes pertencentes ao ciclo, cuja inserção permita um custo adicional mínimo a solução. O ciclo construído após a inserção do vértice deverá permanecer viável. Diante destes dois procedimentos pode-se estabelecer a segunda estrutura de vizinhança (N_2), 1-*DropCheap,* cuja definição é a seguinte: uma sequência de procedimentos *Drop* e *Cheap* é realizada sobre uma solução até que nenhum melhoramento seja encontrado. A terceira (N_3) é obtida através de permutas entre dois vértices de um mesmo grupo, 2-*Swap*. A quarta (N_4) utiliza o método 2-*Opt*. Neste trabalho adotam-se as seguintes estratégias para reduzir o esforço computacional decorrente do uso das estruturas

de vizinhanças. Quando se utilizam as vizinhanças N_1 e N_3, adota-se a estratégia *first improvement*, enquanto que N_2 e N_4, usa o método *best improvement*.

O CE utilizado na RC contém um tamanho pré-definido e a estratégia para escolher as soluções que irão pertencer ao conjunto elite é a seguinte. Uma solução gerada em uma iteração GRASP é inserida no conjunto elite caso seu custo da função objetivo seja menor que o custo da solução de pior qualidade do conjunto elite e esta solução apresente uma diferença mínima *dif_min* na sua estrutura para cada solução presente no CE.

O parâmetro α utilizado para gerenciar a cardinalidade da Lista Restrita de Candidatos (LRC) na etapa de construção do GRASP utiliza a versão reativa, onde os valores de α variam no intervalo [0,1]. A estratégia reativa tende a permitir uma melhor adequação do número de elementos contidos na LRC a cada etapa de uma execução do GRASP (RESENDE; RIBEIRO, 2002).

Figura 2 – Algoritmo GRASP proposto com todos componentes

```
Passo 1: S' ← Construcao_Aleatoria_Gulosa;
Passo 2: S* ← S';
Passo 3: para i=1 até num_iter faça
Passo 4:     S' ← Construcao_Aleatoria_Gulosa;
Passo 5:     S' ← Busca_Local (S');
Passo 6:       se  num_iter> num_eli então
Passo 7:          S ← Sel_Sol_Elites(U^elites,S');
Passo 8:          S' ← Rec_Cam(S, S');
Passo 9:       fim se
Passo 10:     se S' for melhor que S* então
Passo 11:          S* ← S';
Passo 12:     fim se
Passo 13: U^elites ← Max_Sol_Elites(S');
Passo 14: fim para
Passo 15: S^pos ← Pos_Otim(U^elites);
Passo 16: se S^pos for melhor que S* então
Passo 17:     S* ← S^pos;
Passo 18: fim se
```

O algoritmo baseado no GRASP, para o PCVG, está mostrado na Figura 2 com todos os componentes importantes aos seis métodos heurísticos propostos: construção da solução (passos 1 e 4), busca local (passo 5) e tipos de Reconexão de Caminhos. Para o algoritmo considera-se ainda: S^* a melhor solução, S^{pos} a

melhor solução na fase de pós-otimização, *num_iter* o número máximo de iterações e *num_eli* o número máximo da cardinalidade do conjunto elite. O componente no passo 7, *Sel_Sol_Elites*(U^{elites},S'), escolhe a solução elite (*S*) para realizar a Reconexão de Caminhos, após a inserção da solução (*S*'), que representa a busca local. No passo 8, *Rec_Cam*(S, S'), é a Reconexão de Caminhos aplicada durante as iterações entre (*S*) e (S'). O procedimento *Max_Sol_Elites*(S') (passo 13) representa a construção do conjunto de soluções elites. A fase de pós-otimização é representada pelo procedimento *Pos_Otim*(U^{elites}) no passo 15. Por último, a melhor solução S^* é atualizada no passo 11 ou 17 dependendo do tipo do método heurístico.

Cabe ressaltar que o algoritmo GRASP tradicional (G) não contém todos os componentes da Figura 2, que são: *Max_Sol_Elites*(S'), *Sel_Sol_Elites*(U^{elites},S'), *Rec_Cam*(S, S') e *Pos_Otim*(U^{elites}). O GRC1 não contém os procedimentos *Sel_Sol_Elites*(U^{elites},S') e *Rec_Cam*(S, S'), mas contém a fase de pós-otimização, *Pos_Otim*(U^{elites}). O GRC2, GRC3 e GRC3-VND não possui a fase de pós-otimização e o GRC1RC2 contém todos os componentes descritos na Figura 2.

4 RESULTADOS E DISCUSSÕES

Não foi encontrada nenhuma biblioteca pública com instâncias do PCVG na sua versão genérica sem prefixação da sequência de visita aos grupos. Assim, tornou-se necessária a criação de um conjunto de instâncias para o PCVG genérico para avaliar os métodos aqui propostos. Seis tipos diferentes de instâncias foram gerados agrupando os vértices em diversas configurações: (*tipo 1*): Adaptação de instâncias do PCV (REINELT, 2007) agrupadas usando o algoritmo de clusterização *k-means*; (*tipo 2*): Adaptação de instâncias disponíveis na literatura para o PCV geradas por Johnson e Mcgeoch (2002), agrupando-se os vértices em torno de um centro geométrico; (*tipo 3*): As instâncias são geradas através da interface Concorde de Applegate *et al.* (2007); (*tipo 4*): As instâncias são geradas usando *layout* proposto por Laporte, Potvin e Quilleret (1996); (*tipo 5*): Instâncias semelhantes aos do *tipo 2*, mas geradas com parâmetros diferentes de Johnson e McGeoch (2002); (*tipo 6*): Instâncias do PCV (REINELT, 2007) onde a área plana retangular que compõe a instância é dividida em diversos quadriláteros e cada quadrilátero

corresponde a um grupo. Todas estas instâncias são Euclidianas e podem ser acessadas em http://labic.ic.uff.br/Instance/index.php, onde constam os melhores valores alcançados pelas heurísticas ou através da formulação exata.

4.1 Resultados do CPLEX Paralelo

Para analisar o desempenho empírico das heurísticas propostas, os resultados de G, GRC1, GRC2, GRC1RC2 e GRC3 foram comparados com a solução ótima ou limites inferiores obtidos pelo método exato usando a formulação matemática da literatura proposta em Chisman (1975) e o *softwar*e CPLEX Paralelo, versão 11.2 (CPLEX, 2009). O CPLEX foi executado num computador com processador Intel Core 2 Quad e quatro núcleos, 2.83 GHz com 8 GB de RAM. O sistema operacional utilizado foi Linux Ubuntu versão 4.3.2-1. Todas as heurísticas utilizando GRASP apresentadas neste trabalho foram codificadas em linguagem de programação C e executadas no mesmo computador descrito anteriormente, mas utilizando somente um núcleo.

A Tabela 1 mostra as instâncias (coluna 1) com os seus identificadores (coluna 2), seguido do número de vértices (nós), número de grupos (V_i) e tipo. A Tabela 1 mostra também os custos das soluções e os limites inferiores gerados pelo *software* CPLEX. Na última coluna é mostrado o *gap* (%) entre custo da solução e limite inferior. Devido às longas iterações ocorridas no CPLEX para a maioria das instâncias, determinamos um tempo limite de 25200 segundos para sua execução. Para a instância I_3, a solução ótima foi encontrada pelo CPLEX em 442 segundos.

Tabela 1 – Instâncias com seus identificadores, número de nós, número de grupos, tipo e valores do CPLEX

Instâncias	Id.	# nós	# V_i	Tipo	CPLEX		
					Custo	Lim. Inf.	(%)
10-lin318	I_1	318	10	1	531931	526559,70	1,01
10-pcb442	I_2	442	10	1	546157	536478,37	1,77
25-eil101	I_3	101	25	6	**23671**	23668,63	**0,01**
144-rat783	I_4	783	144	6	916103	913715,52	0,26
300-20-111	I_5	300	20	5	310590	308627,83	0,63
500-25-308	I_6	500	25	5	367509	364114,62	0,92
300-6	I_7	300	6	3	**8956**	8916,41	**0,44**
700-20	I_8	700	20	3	41615	41274,00	0,82
C1k.0	I_9	1000	10	2	133638625	131360206,30	1,7
C1k.1	I_{10}	1000	10	2	130563491	128540131,50	1,55
200-4-h	I_{11}	200	4	4	**62835**	62391,08	**0,71**
600-8-z	I_{12}	600	8	4	132767	128083,87	3,53
						gap médio =	1,11

4.2 Resultados das Heurísticas usando GRASP e Reconexão de Caminhos

Os principais parâmetros utilizados em todas heurísticas são descritos a seguir. O valor de penalização dos custos das arestas *M* foi de 10*max{c_{ij}}. A expressão max{c_{ij}} significa o maior custo entre todos os custos das arestas (v_i, v_j). Os valores de alfa (α) na etapa de construção do GRASP estão no intervalo de [0,1]. O número máximo de iterações do GRASP *num_iter* foi fixado em 200. A cardinalidade do conjunto de soluções elites *num_eli* foi igual a 10 estabelecido após experimentos preliminares e a diferença mínima *dif_min* na estrutura entre as duas soluções (a solução candidata e a solução presente no conjunto elite) foi igual a cinco. Devido à natureza aleatória do GRASP, executamos dez vezes cada heurística aqui proposta para cada instância.

Na Tabela 2 são mostrados os *gaps* dos melhores valores para as heurísticas, assim como os *gaps* dos valores médios, ambos calculados em relação aos limites inferiores dado pela Tabela 1. Valores em negrito representam o melhor desempenho. Na avaliação entre as melhores soluções obtidas pelas heurísticas GRASP e o CPLEX, o *gap* médio do CPLEX ficou em 1,11%, G em 1,01%, GRC1

em 0,74%, GRC2 em 0,95%, ***GRC1RC2 em 0,72%*** e GRC3 em 0,82%.

Tabela 2 – Os melhores valores e valores médios obtidos pelas heurísticas

Id.	Melhores Valores					Valores Médios				
	G	**GRC1**	**GRC2**	**GRC1RC2**	**GRC3**	**G**	**GRC1**	**GRC2**	**GRC1RC2**	**GRC3**
I_1	1,14	0,80	1,02	**0,79**	0,85	1,75	1,26	1,37	**1,13**	1,32
I_2	1,22	0,71	1,10	**0,70**	0,86	1,94	1,32	1,49	**1,19**	1,47
I_3	0,09	0,05	0,06	0,06	0,05	0,35	0,27	0,21	**0,18**	0,29
I_4	0,21	**0,14**	0,21	**0,14**	**0,14**	0,28	0,20	0,25	**0,19**	0,21
I_5	0,58	**0,45**	0,54	0,48	0,50	0,84	0,66	0,71	**0,62**	0,71
I_6	0,71	0,57	0,70	**0,56**	0,63	0,92	0,73	0,81	**0,70**	0,79
I_7	0,75	0,53	0,62	0,52	0,57	1,08	0,89	0,89	**0,79**	0,93
I_8	0,67	0,59	0,64	0,59	**0,56**	0,83	0,71	0,75	**0,68**	0,74
I_9	1,61	**1,34**	1,63	1,42	1,51	1,88	1,60	1,74	**1,59**	1,71
I_{10}	1,31	1,05	1,29	**1,03**	1,10	1,54	1,26	1,41	**1,24**	1,35
I_{11}	1,74	1,09	1,59	1,05	1,34	3,07	2,42	2,33	**2,03**	2,62
I_{12}	2,07	1,54	2,00	**1,38**	1,71	3,16	2,24	2,61	**2,15**	2,53
gap méd.	1,01	0,74	0,95	**0,72**	0,82	1,47	1,13	1,21	**1,04**	1,22

O CPLEX obteve as melhores soluções em três num total de 12 instâncias, GRC1 obteve três melhores soluções, ***GRC1RC2 seis*** e GRC3 duas. Em uma única instância, a melhor solução foi alcançada simultaneamente por GRC1, GRC1RC2 e GRC3. Na avaliação das soluções médias obtidas pelas heurísticas, o *gap* de G ficou em 1,47%, de GRC1 em 1,13%, de GRC2 em 1,21%, de ***GRC1RC2 igual a 1,04%*** e de GRC3 em 1,22%.

Na Tabela 3 são mostrados os tempos médios medidos em segundos para as heurísticas. Observamos que G e GRC2 são aquelas que consomem o menor tempo médio, fato que era esperado, pois estas não utilizam a RC1 ao final do GRASP. Apesar de GRC1, GRC1RC2 e GRC3 apresentarem tempos médios maiores que G e GRC2, estes ainda são bem menores que o tempo limite de 25200 segundos utilizado para a execução do *software* CPLEX.

Tabela 3 – Os tempos médios das heurísticas GRASP (em segundos)

Id.	Tempos Médios (segundos)				
	G	**GRC1**	**GRC2**	**GRC1RC2**	**GRC3**
I_1	**57,05**	158,87	87,48	184,75	136,80
I_2	**151,01**	451,11	227,67	495,25	386,30
I_3	**2,48**	3,72	3,95	6,43	3,00
I_4	**750,30**	3718,32	1133,96	4132,28	3335,10
I_5	**51,68**	121,30	75,65	142,81	106,00
I_6	**225,93**	720,47	331,15	822,49	631,60
I_7	**49,44**	99,47	75,97	119,72	86,30
I_8	**593,80**	1618,11	881,57	1837,78	1470,90
I_9	**1762,68**	7592,92	2608,65	8749,56	7235,00
I_{10}	**1765,84**	9421,46	2616,59	10297,08	8190,90
I_{11}	**15,75**	33,95	25,07	42,37	29,90
I_{12}	**364,78**	1533,07	554,93	1652,09	1325,00

Os resultados apresentados até o momento, onde o critério de parada é o número de iterações e utilizando o mesmo número para todas as heurísticas, podemos observar que GRC1, GRC1RC2 e GRC3 obtiveram os melhores resultados, mas exigiram tempos computacionais maiores que G e GRC2. Realizamos novos testes colocando como critério de parada um tempo máximo idêntico para as cinco heurísticas. Foi utilizado um tempo limite de duas horas para o CPLEX e 720 segundos para as heurísticas GRASP, porque, para cada instância, as heurísticas são executadas 10 vezes e a melhor solução encontrada nestas execuções é retornada. Somente a execução da instância I_3 não foi limitada por duas horas, devido à solução ótima ser encontrada antes. Os valores obtidos pelo CPLEX com este novo critério de parada é mostrada na Tabela 4.

A primeira coluna da Tabela 4 é mostrada, os identificadores das instâncias da Tabela 1, seguida pelos valores de custo, limite inferior e gap (%) do CPLEX. A comparação entre os melhores valores obtidos pelas heurísticas com este novo critério é apresentada na Tabela 5.

Tabela 4 – Valores da nova execução do CPLEX com limite de tempo

Id.	CPLEX – Nova Execução		
	Custo	**Lim. Inf.**	**(%)**
I_1	534640	526412,07	1,54
I_2	547152	536478,33	1,95
I_3	**23671**	23668,63	**0,01**
I_4	916174	913715,52	0,27
I_5	311286	308595,45	0,86
I_6	367586	364108,13	0,95
I_7	8969	8915,18	0,60
I_8	41638	41274,00	0,87
I_9	134025123	131354923,50	1,99
I_{10}	130750874	128540131,50	1,69
I_{11}	63429	62244,84	1,87
I_{12}	132897	127901,75	3,76
		gap médio =	1,36

A descrição da Tabela 5 é semelhante a da Tabela 2 e mostra o *gap* médio de G em 0,98%, de GRC1 em 0,83%, de GRC2 em 0,94%, de ***GR1RC2 (GRC12) igual a 0,82%*** e GRC3 em 0,86%, contra o *gap* do CPLEX que ficou em 1,36% (Tabela 4). GRC1RC2 obteve novamente o melhor desempenho obtendo as melhores soluções em oito de um total de 12 instâncias, GRC1 com três melhores soluções, GRC3 com uma e o CPLEX obteve também uma melhor solução. Observamos desta forma a importância de utilizar Reconexão de Caminhos intensiva como é o caso da RC1 ao final do GRASP (GRC1 e GRC1RC2) e que a melhor opção foi usar a RC tanto durante as iterações (RC2) como após (RC1).

Nesta segunda bateria de testes observamos que apesar de GRC1, GRC1RC2 e GRC3 exigirem mais tempo por iteração, estas necessitam de menos iterações para atingir bons limites superiores de um valor ótimo.

Tabela 5 – *Gaps* das heurísticas GRASP com limite de tempo

Id.	Melhores Valores					Valores Médios				
	G	**GRC1**	**GRC2**	**GRC12**	**GRC3**	**G**	**GRC1**	**GRC2**	**GRC12**	**GRC3**
I_1	0,97	0,80	0,93	**0,76**	0,90	1,78	**1,18**	1,21	**1,18**	1,72
I_2	1,12	0,73	0,99	**0,66**	0,84	1,93	1,24	1,36	**1,22**	1,67
I_3	0,04	0,04	0,05	0,03	0,03	0,35	0,22	**0,14**	0,18	0,35
I_4	0,21	**0,19**	0,21	0,20	**0,19**	0,28	**0,23**	0,25	0,24	0,24
I_5	0,55	0,45	0,51	**0,43**	0,50	0,85	0,64	**0,63**	**0,63**	0,84
I_6	0,69	0,61	0,66	**0,58**	0,60	0,92	**0,73**	0,78	**0,73**	0,80
I_7	0,67	0,51	0,60	**0,49**	0,50	1,09	0,82	**0,78**	**0,78**	1,08
I_8	0,67	**0,63**	0,65	0,64	0,62	0,82	**0,71**	0,75	0,72	0,73
I_9	1,63	1,61	1,64	**1,60**	1,63	1,88	**1,76**	1,79	**1,76**	1,80
I_{10}	1,28	1,27	1,31	**1,27**	1,30	1,54	1,44	1,47	**1,42**	1,45
I_{11}	1,68	1,28	1,60	**1,19**	1,23	3,30	2,37	**2,01**	2,30	3,38
I_{12}	2,23	**1,80**	2,14	1,96	2,01	3,32	**2,43**	2,70	2,54	2,66
	0,98	0,83	0,94	**0,82**	0,86	1,51	1,15	1,16	**1,14**	1,39

A Tabela 5 ilustra também os valores médios. O *gap* médio do G ficou em 1,51%, de GRC1 em 1,15%, de GRC2 com 1,16%, de ***GRC1RC2 igual a 1,14%*** e de GRC3 igual a 1,39%. Novamente os resultados mesmo que empíricos mostram a superioridade das versões GRC1 e GRC1RC2 em relação a G, GRC2 e GRC3.

Tabela 6 – Comparação do CPLEX com GRC1RC2, instâncias de pequeno porte

Instâncias	GRC1RC2	$t_{GRC1RC2}$(s)	CPLEX	t_{CPLEX} (s)	GRC1RC2 (%)
5-eil51	437	1,00	437	12,31	**0,00**
10-eil51	440	1,00	440	74,38	**0,00**
15-eil51	437	1,00	437	2,04	**0,00**
5-berlin52	7991	1,20	7991	201,80	**0,00**
10-berlin52	7896	1,10	7896	89,17	**0,00**
15-berlin52	8049	1,10	8049	75,93	**0,00**
5-st70	695	2,30	695	13790,11	**0,00**
10-st70	691	2,00	691	4581,00	**0,00**
15-st70	692	2,00	692	883,50	**0,00**
5-eil76	561	2,70	559	83,70	0,36
10-eil76	564	2,40	561	254,30	0,53
15-eil76	567	2,50	565	49,66	0,35
5-pr76	108590	2,70	108590	99,29	**0,00**
10-pr76	109538	2,20	109538	238,13	**0,00**
15-pr76	110843	2,30	110678	261,94	0,15
10-rat99	1238	4,90	1238	650,67	**0,00**
25-rat99	1277	4,70	1269	351,15	0,63
50-rat99	1258	4,90	1249	2797,58	0,72
25-kroA100	21917	4,70	21917	3513,57	**0,00**
50-kroA100	21453	5,20	21453	947,55	**0,00**
10-kroB100	22475	4,80	22440	4991,44	0,16
50-kroB100	22653	5,20	22355	2579,22	1,33
25-eil101	672	4,60	663	709,45	1,36
50-eil101	651	5,40	644	275,33	1,09
25-lin105	14438	5,10	14438	6224,55	**0,00**
50-lin105	14534	5,70	14379	1577,21	1,08
75-lin105	14607	6,40	14521	15886,77	0,59
				gap médio =	0,25

No entanto novamente ressaltamos o bom comportamento médio das cinco heurísticas aqui propostas, mostrando que todas elas possuem uma robustez necessária para a sua confiabilidade prática. Novos testes foram efetuados somente com GRC1RC2 e o método exato CPLEX restritos as instâncias menores, mas sem limitação de tempo para o CPLEX. A heurística GRC1RC2 foi executada com número de iterações fixo igual a 200. A primeira coluna da Tabela 6 mostra as instâncias, na segunda encontra-se o melhor valor obtido por GRC1RC2, na terceira o tempo médio de execução $t_{GRC1RC2(s)}$ de GRC1RC2 medido em segundos e na quarta o valor do CPLEX. Na quinta e sexta colunas são mostrados o tempo demandado pelo CPLEX em segundos $t_{CPLEX(s)}$ e o *gap* do melhor valor obtido por GRC1RC2(%) em relação ao valor ótimo obtido pelo CPLEX.

Para todas as instâncias de pequeno porte, o CPLEX encontrou soluções ótimas. GRC1RC2 encontrou soluções ótimas em 15 de um total de 27 instâncias e o *gap* médio dos valores obtidos pelo GRC1RC2 em relação ao ótimo foi de 0,25%. Este último resultado reforça o potencial de GRC1RC2 em encontrar uma solução de boa qualidade em tempo computacional viável.

Para comparar a eficiência de inserir Reconexão de Caminhos durante as iterações e o uso de Reconexão de Caminhos no final do GRASP, realizamos mais uma bateria de testes em instâncias de grande porte. A comparação foi realizada entre o GRASP tradicional (G) e GRC1RC2. Como as instâncias são de portes maiores, estabelece um limite de 1080 segundos para cada execução de G e GRC1RC2. As execuções das heurísticas GRASP foram realizadas 10 vezes, obtendo um total de três horas para cada instância.

A Tabela 7 mostra as instâncias escolhidas para comparar G com GRC1RC2 com seus identificadores, número de nós, número de grupos, tipo e *gaps* encontrados. O cálculo do *gap* foi estabelecido em relação ao melhor valor encontrado entre G e GRC1RC2 ou o melhor valor médio. Nestes testes com instâncias e tempos maiores observamos novamente que a introdução dos módulos de Reconexão de Caminhos em GRC1RC2 é de fundamental importância na busca de soluções de boa qualidade. Na Tabela 7, foi observado que o *gap* médio dos melhores valores encontrados por G é igual a 0,35% e por ***GRC1RC2 igual a zero***. Nos valores médios o *gap* médio de G foi igual a 1,57% e de ***GRC1RC2 permanece***

igual à zero.

Tabela 7 – Comparação entre G e GRC1RC2 para instâncias de grande porte

Instâncias	Id.	# nós	# Vi	Tipo	Melhores Valores		Valores Médios	
					G	GRC1RC2	G	GRC1RC2
49-pcb1173	I_{13}	1173	49	6	0,40	**0,00**	2,94	**0,00**
100-pcb1173	I_{14}	1173	100	6	0,54	**0,00**	2,87	**0,00**
144-pcb1173	I_{15}	1173	144	6	0,08	**0,00**	2,68	**0,00**
10-nrw1379	I_{16}	1379	10	6	0,66	**0,00**	2,33	**0,00**
12-nrw1379	I_{17}	1379	12	6	0,26	**0,00**	2,96	**0,00**
1500-10-503	I_{18}	1500	10	5	0,32	**0,00**	1,03	**0,00**
1500-20-504	I_{19}	1500	20	5	0,06	**0,00**	1,36	**0,00**
1500-50-505	I_{20}	1500	50	5	0,03	**0,00**	0,65	**0,00**
1500-100-506	I_{21}	1500	100	5	0,52	**0,00**	1,19	**0,00**
1500-150-507	I_{22}	1500	150	5	0,01	**0,00**	0,43	**0,00**
2000-10-a	I_{23}	2000	10	4	0,86	**0,00**	0,53	**0,00**
2000-10-h	I_{24}	2000	10	4	0,25	**0,00**	1,11	**0,00**
2000-10-z	I_{25}	2000	10	4	0,17	**0,00**	1,29	**0,00**
2000-10-x1	I_{26}	2000	10	4	0,91	**0,00**	0,49	**0,00**
2000-10-x2	I_{27}	2000	10	4	0,19	**0,00**	1,64	**0,00**
gap médio =					0,35	**0,00**	1,57	**0,00**

4.3 Distribuições de Probabilidade do GRASP Tradicional e do GRASP com Reconexão de Caminhos

Análises gráficas das distribuições de probabilidade acumulativa (AIEX; RESENDE; RIBEIRO, 2007) foram obtidas para verificar o desempenho da introdução de módulos de Reconexão de Caminhos. Analisamos o desempenho do G com GRC1RC2. Três instâncias foram consideradas com número de vértices e grupos diferentes, dadas a seguir: a instância I_{11} (200-4-h) onde a melhor solução foi encontrada num tempo de 25200s usando o método exato CPLEX, cujo valor foi de 62835, a instância I_4 (144-rat783) onde a melhor solução conhecida (914965) foi alcançada por GRC1RC2 em um tempo médio de 4132,28s e a instância I_{10} (C1k.1) onde o melhor valor de 129872457 foi alcançado por GRC1RC2 num tempo médio de 10297,09s.

Para cada instância foram estabelecidos dois alvos, um mais difícil denominado de alvo 1 e um mais fácil (alvo 2). Para I_{11} o alvo 1 igual a 63600 (1,21% acima do melhor valor) e alvo 2 com 63800 (1,54% do melhor valor). Para I_4 foi estabelecido alvo 1 igual a 916400 (0,15%) e alvo 2 igual a 916700 (0,19%). Para I_{10}, alvo 1 com valor de 130280000 (0,31%) e alvo 2 igual a 130400000 (0,41%). O limite de tempo para atingirem os alvos foi definido como 85s para a instância I_{11} e

3600s para a I_4 e I_{10}, tempo suficiente para os alvos mais difíceis executarem a fase de pós-otimização do GRC1RC2. Para cada instância/alvo as heurísticas foram executadas 100 vezes com *sementes* diferentes, o que tornam estas execuções independentes.

Tabela 8 – Valores dos alvos das instâncias

	Alvo 1		Alvo 2		Melhores Valores		Valores Médios	
Id.	**Valor**	**(%)**	**Valor**	**(%)**	**G**	**GRC1RC2**	**G**	**GRC1RC2**
I_{11}	63600	1,21	63800	1,54	63493	63050	64368,21	63685,84
I_4	916400	0,15	916700	0,19	915651	914965	916324,00	915468,18
I_{10}	130280000	0,31	130400000	0,41	130250305	129872457	130545650	130160400

A Tabela 8 mostra os alvos e a porcentagem em relação a melhor solução encontrada, junto aos valores médios e melhores valores do G e GRC1RC2.

No gráfico da Figura 3 observamos o desempenho de G e GRC1RC2 para I_{11}. A heurística G atinge o alvo 2 em 21s, enquanto GRC1RC2 tem a probabilidade de atingir o alvo em 7s.

Na Figura 4 a heurística GRC1RC2 atinge o alvo 1 em 35s, enquanto que G em 35s só consegue atingir 75% do alvo. A heurística G atingiu o tempo limite estabelecido (85s) em duas execuções.

Figura 3 – Distribuição de Probabilidade Acumulativa para I_{11} (alvo 2)

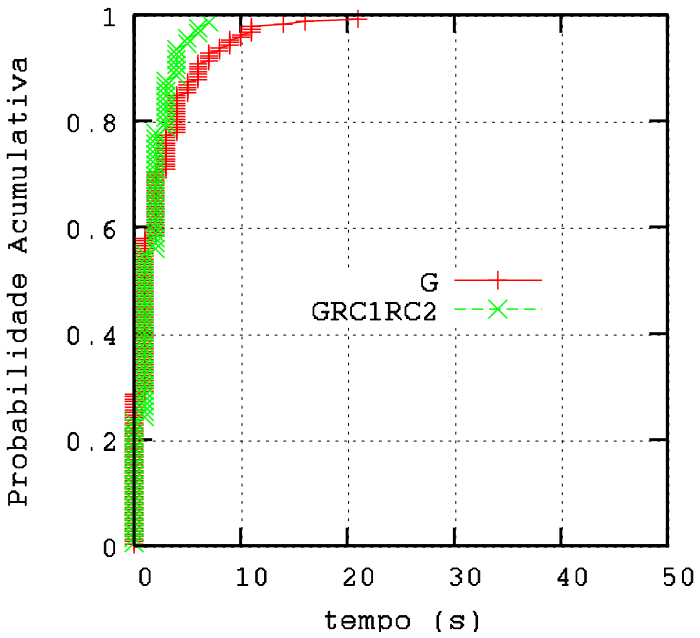
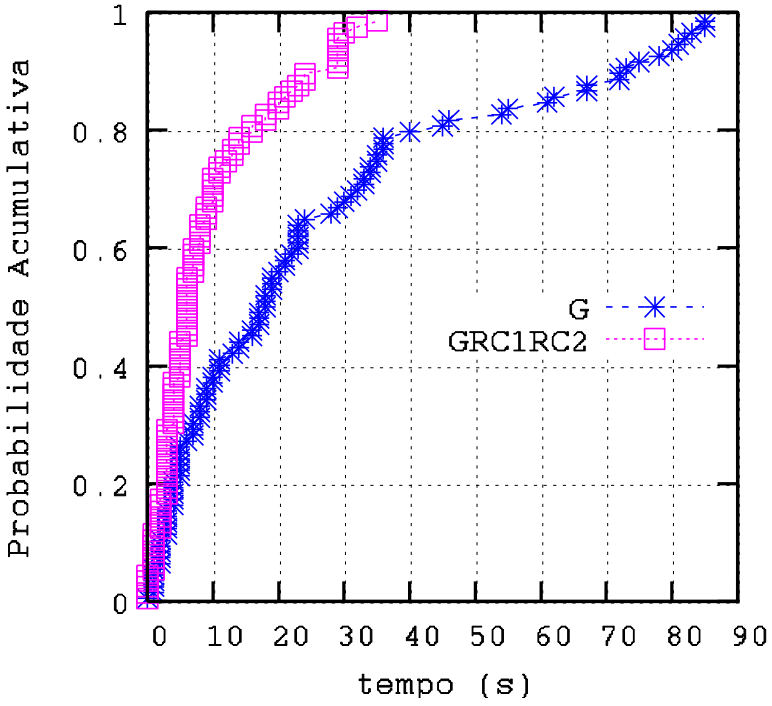


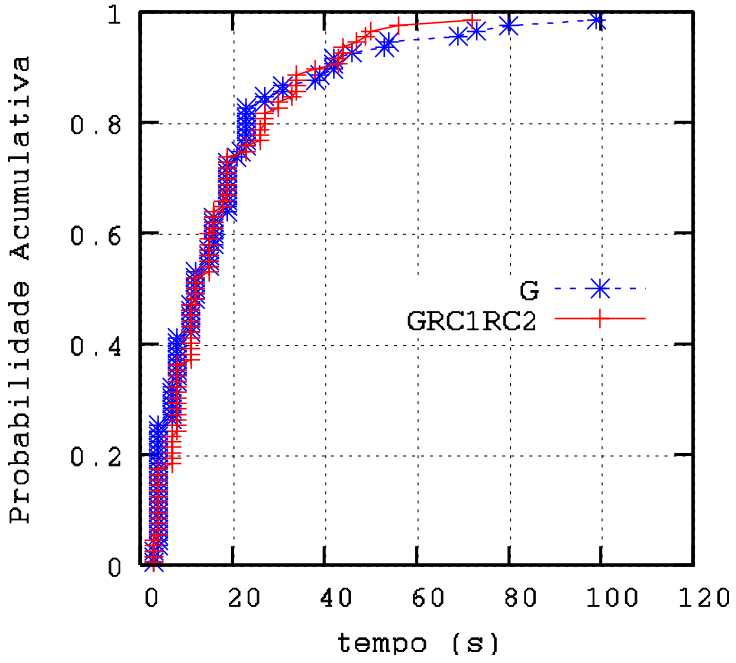
Figura 4 – Distribuição de Probabilidade Acumulativa para I_{11} para (alvo 1)



No gráfico da Figura 5 é mostrado o desempenho de G e GRC1RC2 para a instância I_4 (alvo 2), G atinge o alvo em 99s, enquanto GRC1RC2 necessita de 72s.

Na Figura 6, mostra o desempenho de G e GRC1RC2 para a instância I_4 (alvo 1). GRC1RC2 atinge este alvo em 1250s, enquanto que G o alcança em 3001s.

Figura 5 – Distribuição de Probabilidade Acumulativa para I_4 (alvo 2)



Na Figura 7 é mostrado o desempenho de G e GRC1RC2 para a instância I_{10} (alvo 2). GRC1RC2 atinge o alvo em 433s, enquanto que G o atinge em 360s.

A Figura 8 mostra o desempenho de G e GRC1RC2 para a instância I_{10} (alvo 1). O GRC1RC2 com 3478s conseguem atingir 0,95% do alvo, enquanto que G com 3488s só conseguem 0,46%, sendo que GRC1RC2 atingiu o tempo limite estabelecido (3600s) somente em três execuções e G atingiu o tempo limite em 53 execuções.

Figura 6 – Distribuição de Probabilidade Acumulativa para I_4 (alvo 1)

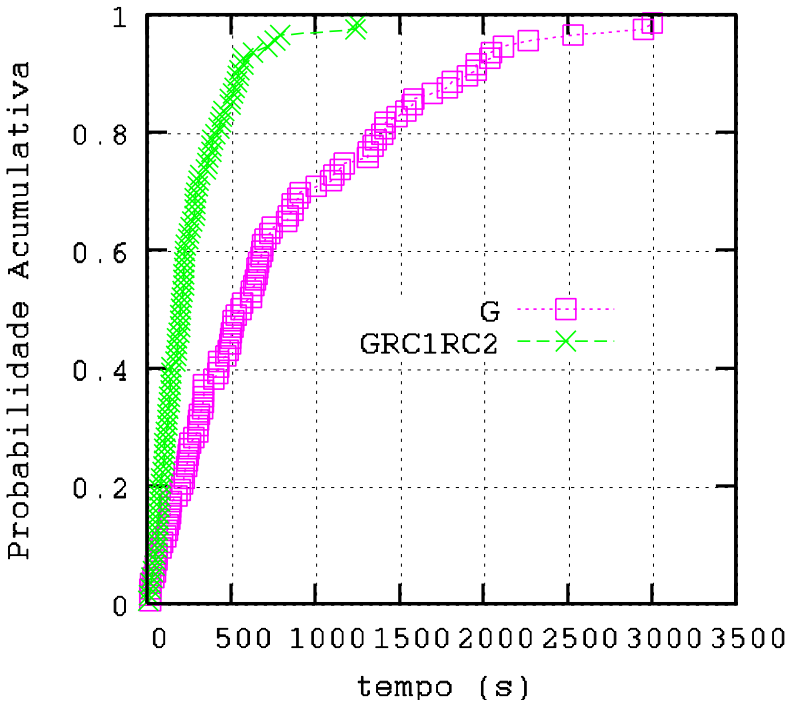


Figura 7 – Distribuição de Probabilidade Acumulativa para I_{10} (alvo 2)

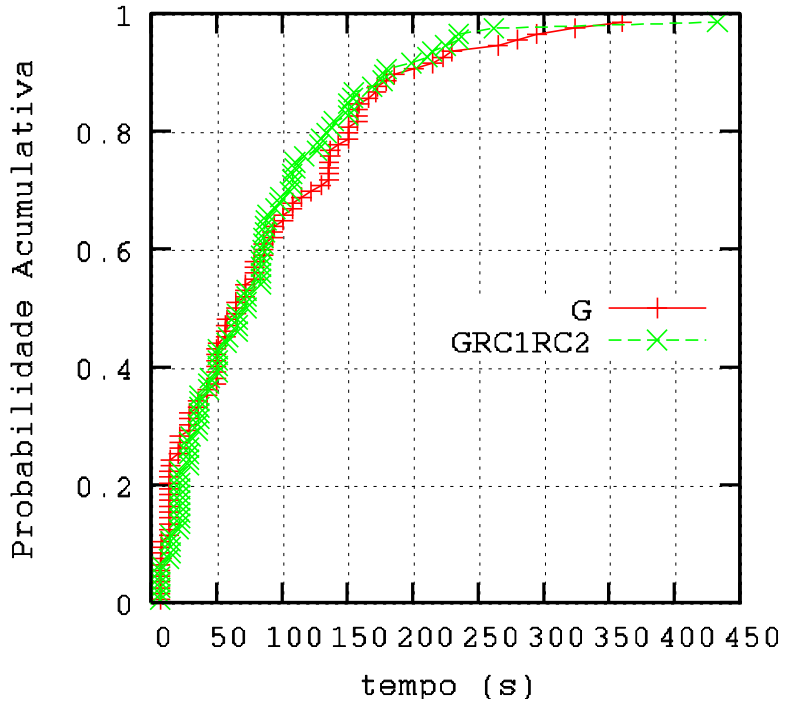
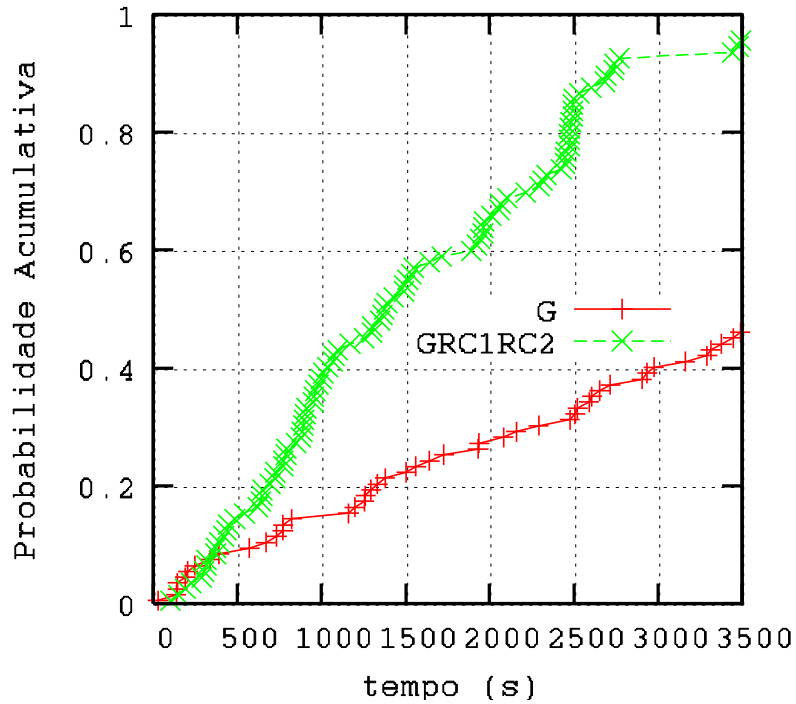


Figura 8 – Distribuição de Probabilidade Acumulativa para I_{10} (alvo 1)



4.4 Novos Resultados do CPLEX Paralelo

Para os testes computacionais com a heurística híbrida GRC3-VND foi ampliado o conjunto de instâncias da Tabela 1 e são mostrados na Tabela 9. A descrição da Tabela 9 é idêntica a da Tabela 1.

Foi determinado um tempo limite de 7200s para a execução das instâncias da Tabela 9, porque foram realizados testes preliminares e constataram-se longas iterações para atingir o valor ótimo no CPLEX, na maioria das instâncias, apesar do CPLEX ser executado na forma paralela. Para a instância K_{18}, a solução ótima foi encontrada pelo CPLEX em 442s.

Tabela 9 - Instâncias utilizadas pelo CPLEX

Instâncias	Id.	# nós	# V_i	Tipo	CPLEX		
					Valor	Lim. Inf.	(%)
50-gil262	K_1	262	50	1	135529	135374,68	0,11
10-lin318	K_2	318	10	1	534640	526412,07	1,54
10-pcb442	K_3	442	10	1	547152	536478,33	1,95
C1k.0	K_4	1000	10	2	134025123	131354923,50	1,99
C1k.1	K_5	1000	10	2	130750874	128540131,50	1,69
C1k.2	K_6	1000	10	2	144341485	141501445	1,97
300-6	K_7	300	6	3	8969	8915,18	0,60
400-6	K_8	400	6	3	9117	9021,51	1,05
700-20	K_9	700	20	3	41638	41274,00	0,87
200-4-h	K_{10}	200	4	4	63429	62244,84	1,87
200-4-x1	K_{11}	200	4	4	60797	60242,96	**0,91**
600-8-z	K_{12}	600	8	4	132897	127901,75	3,76
600-8-x2	K_{13}	600	8	4	132228	127901,75	3,27
300-5-108	K_{14}	300	5	5	68361	67128,93	1,80
300-20-111	K_{15}	300	20	5	311286	308595,45	0,86
500-15-306	K_{16}	500	15	5	196001	193522,8	1,26
500-25-308	K_{17}	500	25	5	367586	364108,13	0,95
25-eil101	K_{18}	101	25	6	23671	23668,63	**0,01**
42-a280	K_{19}	280	42	6	130043	129560,53	0,37
144-rat783	K_{20}	783	144	6	916174	913715,52	0,27
						gap médio	1,36

4.5 Resultados da Heurística usando GRASP, Reconexão de Caminhos e VND

Para GRC3-VND os resultados foram estabelecidos com o critério de parada num tempo máximo idêntico. Foi utilizado um tempo limite de 720 segundos a cada execução, porque para cada instância, a heurística é executada 10 vezes e a melhor solução encontrada nestas execuções por duas horas é retornada, assim como o

valor médio. Somente a execução da instância K_{18} não foi limitada por duas horas, devido à solução ótima ser encontrada antes.

Tabela 10 - Valores obtidos pela heurística híbrida GRC3-VND

Id.	Melhores Valores GRC3-VND	Valores Médios GRC3-VND	CPLEX
K_1	**0,07**	0,28	0,11
K_2	**0,76**	1,08	1,54
K_3	**0,70**	1,08	1,95
K_4	**1,38**	1,58	1,99
K_5	**1,12**	1,24	1,69
K_6	**1,48**	1,82	1,97
K_7	**0,52**	0,83	0,60
K_8	**0,82**	1,39	1,05
K_9	**0,42**	0,50	0,87
K_{10}	**1,35**	2,46	1,87
K_{11}	1,04	2,97	**0,91**
K_{12}	**1,83**	2,29	3,76
K_{13}	**1,83**	2,81	3,27
K_{14}	**1,47**	2,18	1,80
K_{15}	**0,43**	0,64	0,86
K_{16}	**0,95**	1,33	1,26
K_{17}	**0,56**	0,74	0,95
K_{18}	**0,01**	0,06	**0,01**
K_{19}	**0,17**	0,45	0,37
K_{20}	**0,14**	0,17	0,27
gap médio	***0,85***	*1,30*	1,36

A comparação entre os melhores valores de GRC3-VND e CPLEX com este critério de parada é mostrada na Tabela 10, assim como os valores médios. Compara-se a melhor solução de cada método com os limites inferiores ou com a solução ótima conhecida. Neste contexto o *gap* médio do CPLEX ficou em 1,36% e ***GRC3-VND em 0,85%***. Os valores em negrito na Tabela 10 mostram onde os métodos alcançaram o melhor valor. O CPLEX obteve uma melhor solução no total de 20 instâncias e GRC3-VND obteve 18 melhores soluções. Uma solução ótima foi encontrada em ambos os métodos. Na comparação entre as soluções médias

obtidas por GRC3-VND e os valores do CPLEX, o *gap* médio de *GRC3-VND ficou em 1,30%* contra 1,36% do CPLEX.

Novos resultados foram efetuados, mostrados na Tabela 11, com GRC3-VND e o CPLEX, restritos às instâncias menores, mas sem limitação de tempo para o CPLEX. Para este conjunto de instâncias, o CPLEX encontrou valores ótimos para todas as instâncias. A heurística GRC3-VND foi executada com número de iterações fixo igual a 200.

Na primeira coluna da Tabela 11 encontra-se a instância, na segunda o valor ótimo obtido pelo CPLEX e na terceira o tempo demandado pelo CPLEX em segundos t_{CPLEX}(s). Na quarta coluna o *gap* de GRC3-VND (%) em relação ao valor ótimo e na quinta o tempo médio de execução $t_{GRC3\text{-}VND}$(s) de GRC3-VND medido em segundos.

Os resultados apresentados na Tabela 11 mostram que GRC3-VND encontrou soluções ótimas em 18 de um total de 27 com *gap* de 0,21% e ***tempo médio de 6,05s***, bem menores do que 2266,73s do CPLEX.

Ao analisarmos a comparação de GRC1RC2 com GRC3-VND, para as instâncias de pequeno porte, o GRC1RC2 alcançou o *gap* médio de 0,25% com ***tempo médio de 3,3 segundos*** (Tabela 6), contra o gap médio do ***GRC3-VND em 0,21%*** e tempo médio de 6,05 (Tabela 11).

Pelos resultados apresentados constata-se que as heurísticas GRC1, GRC2, GRC1RC2, GRC3 e GRC3-VND, permitem alcançar resultados de boa qualidade para o PCVG em tempo computacional razoável. Todas as heurísticas utilizaram o método construtivo que penaliza as arestas intergrupos. Neste sentido, tratar o PCVG transformando-o em PCV é uma alternativa eficaz.

Tabela 11 – Resultados do CPLEX e GRC3-VND para instâncias de pequeno porte

Instâncias	CPLEX	t_{CPLEX} (s)	GRC3-VND (%)	$t_{GRC3\text{-}VND}$(s)
5-eil51	437	12,31	**0,00**	1,50
10-eil51	440	74,38	**0,00**	1,30
15-eil51	437	2,04	**0,00**	1,40
5-berlin52	7991	201,80	**0,00**	1,80
10-berlin52	7896	89,17	**0,00**	1,60
15-berlin52	8049	75,93	**0,00**	1,60
5-st70	695	13790,11	**0,00**	3,40
10-st70	691	4581,00	**0,00**	2,50
15-st70	692	883,50	**0,00**	2,90
5-eil76	559	83,70	0,54	3,80
10-eil76	561	254,30	0,71	3,20
15-eil76	565	49,66	0,53	3,60
5-pr76	108590	99,29	**0,00**	5,50
10-pr76	109538	238,13	**0,00**	4,00
15-pr76	110678	261,94	0,15	4,40
10-rat99	1238	650,67	**0,00**	9,70
25-rat99	1269	351,15	**0,00**	9,80
50-rat99	1249	2797,58	0,80	10,50
25-kroA100	21917	3513,57	**0,00**	8,60
50-kroA100	21453	947,55	**0,00**	9,60
10-kroB100	22440	4991,44	**0,00**	9,30
50-kroB100	22355	2579,22	0,54	9,50
25-eil101	663	709,45	1,21	7,80
50-eil101	644	275,33	1,09	9,10
25-lin105	14438	6224,55	**0,00**	10,50
50-lin105	14379	1577,21	**0,00**	12,40
75-lin105	14521	15886,77	0,23	14,10
valores médios	-	2266,73	0,21	***6,05***

4.6 Comparações entre GRC1RC2, GRC3-VND e CPLEX

Realizamos uma comparação entre GRC1RC2, GRC3-VND e CPLEX através de novos testes computacionais utilizando as instâncias da Tabela 1. Foi utilizado

um tempo limite de duas horas para o CPLEX e 720 segundos para cada execução das heurísticas. O *gap* da melhor solução e da solução média encontrada nestas execuções é retornado. Os *gaps* das heurísticas e do CPLEX são mostrados na Tabela 12 obtidos através dos limites inferiores dados pelo CPLEX.

Tabela 12 – *Gaps* do CPLEX e das heurísticas GRC1RC2 e GRC3-VND

Id.	Melhores Valores		Valores Médios		CPLEX
	GRC1RC2	GRC3-VND	GRC1RC2	GRC3-VND	
I_1	**0,76**	**0,76**	1,18	***1,08***	1,54
I_2	**0,66**	0,70	1,22	***1,08***	1,95
I_3	0,03	**0,01**	0,18	***0,06***	**0,01**
I_4	0,20	**0,14**	0,24	***0,17***	0,27
I_5	**0,43**	**0,43**	***0,63***	0,64	0,86
I_6	0,58	**0,56**	***0,73***	0,74	0,95
I_7	**0,49**	0,52	***0,78***	0,83	0,60
I_8	0,64	**0,42**	0,72	***0,50***	0,87
I_9	1,60	**1,38**	1,76	***1,58***	1,99
I_{10}	1,27	**1,12**	1,42	***1,24***	1,69
I_{11}	**1,19**	1,35	***2,30***	2,46	1,87
I_{12}	1,96	**1,83**	2,54	***2,29***	3,76
gap médio	0,82	**0,77**	1,14	***1,06***	1,36

A descrição da Tabela 12 é semelhante a da Tabela 2. Os valores em negrito na Tabela 12 mostra o *gap* dos melhores valores obtidos comparando-os entre os métodos (GRC1RC2, GRC3-VND e CPLEX). Os *gaps* dos melhores valores médios comparando as heurísticas GRC1RC2 e GRC3-VND são mostrados em negrito e itálico. O *gap* médio dos melhores valores de GRC1RC2 ficou em 0,82%, de **GRC3-VND igual a 0,77%** e CPLEX com 1,36%. Para os valores médios GRC1RC2 foi igual a 1,14% e ***GRC3-VND igual a 1,06%***. Uma solução ótima foi encontrada por ambos os métodos (CPLEX e GRC3-VND). GRC3-VND obteve seis melhores soluções no total de 12 instâncias e GRC1RC2 obteve 3 melhores soluções. Duas melhores soluções foram encontradas em ambas heurísticas. Os resultados empíricos mostram a superioridade de GRC3-VND sobre GRC1RC2. Ressaltamos o bom comportamento médio de GRC1RC2, mostrando que esta produz soluções de boa qualidade com tempo computacional viável.

5 CONSIDERAÇÕES FINAIS

Neste trabalho, a resolução do PCVG através de GRASP, VND e RC conseguiu alcançar as metas de solucionar o PCVG utilizando métodos heurísticos.

Além disto, estes métodos heurísticos solucionam o PCVG na versão genérica, onde a sequência de grupos não é determinada *a priori*. Para avaliar e validar estes métodos foram testadas diversas instâncias Euclidianas com características diferentes em grafos simétricos. Foram observados nestes experimentos que os seis métodos heurísticos baseados no GRASP produziram soluções de boa qualidade para o PCVG em tempo computacional viável.

Os métodos heurísticos que utilizaram RC conseguiram melhor desempenho do que o GRASP tradicional e foram competitivos com um método exato. Nos testes empíricos mostra que o uso de RC durante e após o GRASP, heurística GRC1RC2, é a melhor alternativa viável. As distribuições de probabilidade mostram que a heurística GRC1RC2 permite alcançar soluções alvos em tempo menor do que a versão tradicional (G).

O método híbrido GRC3-VND obteve melhor desempenho do que todos outros métodos. Observa-se que o uso de uma busca local intensiva foi fundamental para o desempenho do GRC3-VND. Constata-se que os métodos heurísticos híbridos alcançam melhor desempenho do que a forma clássica constituída de um único módulo heurístico tradicional.

Trabalhos futuros sugerem utilizar VNS, ILS e novas formas híbridas reunindo RC, VND e GRASP para solucionar o PCVG. Outras fontes de pesquisas futuras é verificar a construção de soluções iniciais não penalizando as arestas intergrupos e ainda estudar o PCVG Assimétrico.

AGRADECIMENTOS

Esta pesquisa tem apoio parcial do seguinte órgão de fomento: CAPES/PIQDTec.

REFERÊNCIAS

AIEX, R.; RESENDE, M; RIBEIRO, C. C. TTT plots: a perl program to create time-to-target plots. **Optimization Letters**. v.1, n.4, p. 355-366, 2007.

ALOISE, D.; RIBEIRO, C. C. Adaptive memory in multistart heuristics for multicommodity network design. **Journal of Heuristics**, v.17, n.2, p. 153-179, 2011.

ANILY, S.; BRAMEL, J.; HERTZ, A. A 5/3-approximation algorithm for the clustered traveling salesman tour and path problems. **Operations Research Letters**, v. 24

n.1-2, p. 29-35, 1999.

ANZANELLO, M. J.; FOGLIATTO, F. S. Programação de tarefas baseada em curvas de aprendizado para linhas de produção customizadas. **Revista Produção Online**, v.11, n. 3, p. 851-870, 2011.

APPLEGATE, D.; BIXBY, R.; CHVÁTAL, V.; COOK, W. **Concorde TSP Solver**. school of industrial and systems engineering, georgia tech. Disponível em: <http://www.tsp.gatech.edu/concorde/index.html>. Acesso em 22/12/2007.

CHISMAN, J. A. The clustered traveling salesman problem. **Computers & Operations Research**. v.2, n.2, p. 115-119, 1975.

CPLEX. ILOG CPLEX 11.2 **User's manual and reference manual**. ILOG S. A, 2009.

DENG, Y.; BARD, J. F. A reactive GRASP with path relinking for capacitated clustering. **Journal of Heuristics**, v.17, n.2, p. 119-152, 2011.

DING, C.; CHENG, Y.; HE, M. Two-level genetic algorithm for clustered traveling salesman problem with application in large-scale TSPs. **Tsinghua Science and Technology**. v.12, n.4, p. 459-465, 2007.

DUARTE, A. et al. Variable neighborhood search for the vertex separation problem. **Computers & Operations Research**, v.39, n.12, p. 3247-3255, 2012.

FESTA, P.; RESENDE, M. GRASP: basic components and enhancements. **Telecommunication Systems**, v.46, n.3, p. 253-271, 2011.

GENDREAU, M.; LAPORTE, G.; POTVIN, J. Y. **Heuristics for the clustered traveling salesman problem**. Relatório Técnico n°CRT-94-54, Centre de recherche sur les transports, Universidade de Montreal, Montreal, Canadá, 1994.

GENDREAU, M.; LAPORTE, G.; HERTZ, A. An Approximation algorithm for the traveling salesman problem with backhauls. **Operational Research.** v.45, n.4, p.639-641, 1997

GHAZIRI, H.; OSMAN, I. H. A Neural network for the traveling salesman problem with backhauls. **Computers & Industrial Engineering**. v.44, n.2, p. 267-281, 2003.

GLOVER, F. Tabu Search and Adaptive Memory Programing - Advances, Applications and Challenges. In Barr, R.S.; HELGASON, R.V.; KENNINGTON, J.L., (Eds.) **Interfaces in computer science and operations research**. Dordrecht, MA, EUA: Kluwer Academic Plublishres, 1996. p.1-75

GUTTMANN-BECK, N.et al. Approximation algorithms with bounded performance guarantees for the clustered traveling salesman problem. **Algorithmica**. v.28, n.4, p. 422-437, 2000.

HANSEN, P.; MLADENOVIĆ, N. Variable neighborhood search. In: GLOVER, F. W.; KOCHENBERGER, G. A. (Eds.). **Handbook of metaheuristics**, 2003. p. 145-184.

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.

JOHNSON, D. S.; MCGEOCH, L. A. Experimental analysis of heuristics for the STSP. In: GUTIN, G. ; PUNNEN, A. (Eds.). **The traveling salesman problem and its variations.** Dordrecht, Holanda: Kluwer Academic Plublishres, 2002. p. 369-443.

JONGENS, K.; VOLGENANT, T. The symmetric clustered traveling salesman problem. **European Journal of Operational Research**. v.19, n.1, p. 68-75, 1985.

LAMB, J. D. Variable neighbourhood structures for cycle location problems. **European Journal of Operational Research**, v. 223, n. 1, p. 15-26, 2012.

LAPORTE, G.; PALEKAR, U. Some applications of the clustered travelling salesman problem. **Journal of the Operational Research Society**. v.53, n.9, p.972-976, 2002.

LAPORTE, G.; POTVIN, J.-Y.; QUILLERET, F. A tabu search heuristic using genetic diversification for the clustered traveling salesman problem. **Journal of Heuristics**. v.2, n.3, p. 187-200, 1996.

LOKIN, F. C. J. Procedures for travelling salesman problems with additional constraints. **European Journal of Operational Research**, v.3, n.2, p. 135-141, 1979.

MATEUS, G. R.; RESENDE, M. G. C.; SILVA, R. M. A. Grasp with path-relinking for the generalized quadratic assignment problem. **Journal of Heuristics**, Springer Netherlands, v.17, n.5, p. 527-565, 2011.

MORAES, M. B. C.; NAGANO, M. S. Otimização do saldo de caixa com algoritmos genéticos: um estudo relacionando cruzamento e mutação no modelo de miller e orr. **Revista Produção Online**, v.11, n. 2, p. 399-417, 2011.

NAGANO, M. S.; MESQUITA, M. S. Métodos heurísticos para o problema de programação *flow shop* com tempos de *setup* separados. **Revista Produção Online**, v.12, n. 2, p. 499-521, 2012.

POTVIN, J.-Y.; GUERTIN, F. **A Genetic algorithm for the clustered traveling salesman problem with a priori order on the clusters**. Relatório Técnico nº CRT-95-06. Centre de recherche sur les transports, Universidade de Montreal, Montreal, Canadá, 1995.

POTVIN, J.-Y.; GUERTIN, F. The clustered traveling salesman problem: a genetic approach. In OSMAN, I. H.; KELLY, J. (Eds.). **Metaheuristics: theory & applications.** Norwell, MA, EUA: Kluwer Academic Plublishres, 1996. cap. 37, p.

619-631,

REINELT, G. The traveling salesman: computational solutions for tsp Applications. **Lecture Notes in Computer Science**, v.840, 213, 1994.

REINELT, G. **TSPLIB:** symmetric traveling salesman problem. Universität Heidelberg, Institut für Informatik, Heidelberg, Alemanha. Disponível em: <http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95>. Acesso em: 17 set. 2007.

RESENDE, M.; RIBEIRO, C. Greedy randomized adaptive search procedures. In: GLOVER, F.; KOCHENBERGER, G. (Eds). **Handbook of metaheuristics**. Norwell, MA, EUA: Kluwer Academic Publishers, 2002. p. 219-249,

RESENDE, M. G. et al. GRASP and path relinking for the max-min diversity problem. **Computers & Operations Research**. v.37, n.3, p. 498-508, 2010.

SANTOS, R. F.; SOUZA Jr., E. C.; BOUZADA, M. A. C. A aplicação da programação inteira na solução logística do transporte de carga: o solver e suas limitações na busca pela solução ótima. **Revista Produção Online**, v.12, n.1, p. 185-204, 2012.

WEINTRAUB, A. et al. An emergency vehicle dispatching system for an electric utility in Chile. **Journal of the Operational Research Society**. v.50, p. 690-696, 1999.

Artigo recebido em 20/06/2012 e aceito para publicação em 27/09/2012.