DESENVOLVIMENTO E ANÁLISE EXPERIMENTAL DE HEURÍSTICAS GRASP PARA O PROBLEMA DO CAIXEIRO VIAJANTE COM GRUPAMENTOS

MÁRIO MESTRIA , LUIZ SATORU OCHI , SIMONE DE LIMA MARTINS

Instituto de Computação - Universidade Federal Fluminense
Rua Passo da Pátria 156 - Bloco E - 3º andar - São Domingos. CEP.: 24210-240
Niterói, RJ, Brasil
E-mails: mmestria@ic.uff.br, satoru@ic.uff.br, simone@ic.uff.br

Abstract— In this paper are proposed GRASP heuristics for obtaining approximate solutions for a generalization of the Traveling Salesman Problem (TSP) know in literature as the Clustered Traveling Salesman Problem (CTSP). In this problem, the vertices are partitioned into clusters and all vertices of each cluster have to be visited contiguously. Five versions of GRASP heuristic are proposed to solve the CTSP including a standard GRASP (G1), GRASP with adaptive memory procedures (AMP) (G2, G3 and G4) and a hybrid GRASP including AMP and VND metaheuristic (G5). Compare the performance of the proposed algorithms among then and with an exact algorithm using the CPLEX software. Computational results show that versions including adaptive memory procedure outperform the standard GRASP concerning quality of the solutions.

Keywords— Computational Intelligence, Metaheuristics, Adaptive Memory Algorithms.

Resumo— Neste artigo são propostas heurísticas GRASP para obter soluções aproximadas para uma generalização do Problema do Caixeiro Viajante (PCV) conhecido na literatura como Problema do Caixeiro Viajante com Grupamentos (PCVG). Neste problema, os vértices são particionados em grupos e todos os vértices de cada grupo têm que ser visitados de forma contígua. São propostas cinco heurísticas GRASP para resolver o PCVG incluindo um GRASP tradicional (G1), GRASP com módulos de memória adaptativa (MMA) (G2, G3 e G4) e uma versão híbrida GRASP incluindo MMA e a metaheurística VND. Compara-se o desempenho dos algoritmos propostos entre eles e com um algoritmo exato usando o *software* CPLEX. Resultados computacionais mostram que as versões com memória adaptativa superam o GRASP tradicional no tocante a qualidade das soluções geradas.

Palavras-chave— Inteligência Computacional, Metaheurísticas, Algoritmos com Memória Adaptativa.

1 Introdução

O Problema do Caixeiro Viajante (PCV) é um dos problemas mais estudados na área de algoritmos e otimização combinatória. Este problema possui um grande número de variações com aplicações em diversas áreas de conhecimento. Uma extensão do PCV, ainda pouco explorada embora aplicável em diferentes situações práticas, é o Problema do Caixeiro Viajante com Grupamentos (PCVG) (*The Clustered Traveling Salesman Problem*). Esta variante surgiu para resolver inicialmente um problema real envolvendo roteamento automático em sistemas de armazenagem, Chisman (1975).

O PCVG pode ser descrito da seguinte forma: seja $G = (V, E)$ um grafo completo com um conjunto de vértices $V=\{v_1,v_2,\ldots,v_n\}$ e um conjunto de arestas $E=\{(v_i,v_j) : v_i, v_j \in V, i \neq j\}$. O conjunto de vértices V é particionado em subconjuntos disjuntos $V_1, V_2,..., V_m$. Assumindo que um custo não-negativo c_{ij} é associado a cada aresta (v_i,v_j) pertencente a E, o PCVG consiste em determinar um ciclo Hamiltoniano de custo mínimo em G, tal que os vértices de cada grupo são visitados contiguamente. No caso particular onde só existe um subconjunto $V_i = V$, o PCVG se torna o PCV clássico. Desta forma o PCVG pertence à classe *NP-difícil* (Guttmann-Beck *et al*. (2000)).

Aplicações para o PCVG encontram-se, por exemplo, em roteamento automático em armazéns, despacho de veículos, desfragmentação de discos rígidos, sistemas de manufaturas (Laporte e Palekar (2002)) e em transações comerciais envolvendo supermercados, lojas e fornecedores de mercadorias.

Justificativas para abordar o PCVG incluem o fato de ainda existirem poucos trabalhos na literatura sobre ele e além disso, a maioria dos trabalhos partirem do princípio que já se conhece uma ordem de visita dos grupos, Gendreau *et al*. (1994), Potvin e Guertin (1995), Laporte *et al*. (1996) e Anily *et al*. (1999) limitando com isso seu uso em situações reais. Outro ponto se refere ao fato das heurísticas existentes para o PCVG serem bastante simples não utilizando módulos mais sofisticados que tornam metaheurísticas mais competitivas. Neste contexto, este trabalho trata do caso geral onde o PCVG não considera previamente nenhuma ordem de visitas aos grupos e para a sua solução propõem-se diferentes versões utilizando a metaheurística GRASP, que tem se mostrado como uma técnica muito competitiva para obter soluções aproximadas de boa qualidade para problemas de otimização combinatória (Resende e Ribeiro (2002)).

Neste trabalho coloca-se como um dos objetivos verificarem o impacto ao se incluir na metaheurística GRASP, módulos de memória adaptativa (MMA) e/ou uma busca local mais eficiente utilizando conceitos da metaheuristica *Variable Neighborhood Se-*

arch (VNS). Neste contexto foram desenvolvidas cinco versões utilizando a metaheurística GRASP. A primeira versão é um GRASP tradicional sem uso de memória entre suas iterações (G1). Nas demais versões incluem-se o uso de memória através da utilização e atualização dinâmica de um conjunto das k melhores soluções distintas geradas até o momento pelo GRASP denotado por *conjunto elite* (CE), cujas informações são aproveitadas por um módulo de reconexão de caminhos (*path relinking*) (RC) (Resende *et al*. 2008). A RC é utilizada em três versões: na primeira a busca local é mais intensiva onde a reconexão é feita entre cada par de soluções elite (RC1), a segunda (RC2) é uma versão mais leve onde a reconexão é feita apenas entre a solução da iteração atual do GRASP com uma solução elite e a terceira (RC3) é idêntica a RC1, mas ela é ativada durante as iterações do GRASP sempre que o CE for totalmente renovado.

Na segunda versão (G2) realiza-se a RC1 ao final da execução de G1. A terceira (G3) utiliza a RC2 durante as iterações do G1, a quarta (G4) adiciona a G3 uma segunda reconexão de caminhos (RC1) ao final de G3 e a última (G5) é uma versão híbrida GRASP que utiliza na sua etapa de busca local a metaheurística de Descida em Vizinhança Variável (*Variable Neighborhood Descent* - VND) e durante as iterações GRASP é usada a RC3.

A segunda seção descreve sucintamente os trabalhos relacionados ao PCVG. A terceira seção descreve as heurísticas propostas para o PCVG. A quarta seção apresenta os resultados computacionais e uma análise de desempenho e a última seção descreve as conclusões.

2 Trabalhos Relacionados

O PCVG foi proposto em Chisman (1975) para resolver um problema de roteamento automático em sistemas de armazenamento. O autor resolveu o problema transformando o PCVG em um PCV adicionando uma penalidade constante de valor M aos custos das arestas que ligam quaisquer dois vértices associados com grupos diferentes. Usando um algoritmo *branch-and-bound* resolveu o PCV resultante. Limites inferiores baseado em Relaxação Lagrangeana foram desenvolvidos em Jongens e Volgenant (1985) para o PCVG.

Em Laporte *et al*. (1996) uma Busca Tabu combinada com uma fase de diversificação usando um Algoritmo Genético foi proposta para o PCVG, mas a ordem de visita dos grupos já era definida *a priori*.

Um Algoritmo Genético (AG) básico foi proposto para o PCVG em Potvin e Guertin (1996). No trabalho de Ding *et al*. (2007) um AG também básico usa o critério de visitação dos vértices de cada grupo de forma aleatória sem estabelecer uma relação com as distâncias entre os vértices. A escolha da ordem de visitação de cada grupo também é aleatória.

Observa-se que a inclusão de algumas novas técnicas como, por exemplo, reconexão de caminhos (RC) (Silva *et al.* (2007); Bastos *et al*. (2005)) em algumas metaheurísticas como GRASP e ILS têm melhorado seu desempenho. Estas estratégias ainda não foram exploradas para a resolução do PCVG.

3 Heurísticas para o PCVG

Cinco heurísticas GRASP foram desenvolvidas para o PCVG. A primeira versão (G1) utiliza a estrutura do GRASP tradicional, onde a cada iteração executa-se uma fase de construção de uma solução e uma fase de busca local. Na etapa de construção, utiliza-se o 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-*optimal* desenvolvido originalmente para o PCV. As versões G2, G3 e G4, também utilizam na busca local o método 2-*optimal*. Em G5 a busca local utilizada foi desenvolvida utilizando-se a metaheurística VND (Hansen e Mladenović (2003)).

Apesar dos bons resultados obtidos pela metaheurística GRASP em diferentes problemas de otimização, um dos gargalos deste método na sua forma tradicional (G1) é a ausência 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 utiliza-se memória adaptativa no GRASP através do uso e atualização dinâmica de um conjunto elite (CE) que supre os dados de entrada de uma técnica de busca denominada reconexão de caminhos *(path relinking)* (RC). A RC, proposta originalmente por (Glover, 1996), consiste basicamente em explorar as soluções intermediárias entre duas soluções extremas de boa qualidade. São definidas duas soluções: origem s_0 e destino s_d. Para gerar cada solução intermediária, movimentos são selecionados de forma a introduzir na solução corrente (inicialmente s_0), atributos presentes na solução destino (s_d). (Resende *et al*. (2008)). A segunda versão (G2) introduz em G1 a RC1 ao final da execução de G1.

A terceira versão (G3) incorpora ao G1 a reconexão de caminhos (RC2) a cada iteração de G1. A RC2 é realizada entre cada solução produzida pela busca local com uma solução extraída do conjunto de soluções elites - CE. 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 apresente uma diferença mínima na sua estrutura para cada solução presente no CE. Na quarta heurística (G4) introduz a RC2 durante a execução das iterações GRASP e ao final do GRASP ativa-se a RC1. A quinta heurística (G5) utiliza todos os módulos do G3 substituindo a busca local 2-*optimal* pela metaheurística VND e a reconexão de caminhos RC2 é substituída pela RC3 definida previamente.

Para o VND utilizam-se quatro tipos de estruturas de vizinhanças. A primeira N_1 é obtida através de um movimento de deslocamento do vértice semente no ciclo de uma posição para outra, 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 a uma rota parcial, entre dois vértices adjacentes pertencentes à rota, cuja inserção permita um custo adicional mínimo a solução. A rota construída após a inserção do vértice deverá permanecer viável. Diante destas duas definições de procedimentos pode-se estabelecer a segunda estrutura de vizinhança N_2, 1-*DropCheap*. Uma seqüência de procedimentos *Drop* e de *Cheap* é realizada sobre uma solução até que nenhum melhoramento seja encontrado. A terceira N_3 é obtida através de uma permuta entre os vértices de um mesmo grupo (*cluster*), 2-*Swap*. A quarta N_4 utiliza à heurística 2-*Optimal*. 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* (ou seja, na primeira melhora muda de vizinhança) (Hansen *et al.* 2003), enquanto utilizam a N_2 e a N_4, adota o método *best improvement* (analisam-se todas as soluções vizinhas e seleciona a melhor).

O parâmetro α para criar a Lista Restrita de Candidatos (LRC) na etapa de construção do GRASP utiliza a versão dinâmica ou reativa, onde seus valores variam no intervalo [0,1]. A estratégia reativa permite um melhor aproveitamento sobre a versão do GRASP com valor fixo de α em termos de robustez e qualidade de solução (Resende e Ribeiro (2002)).

4 Resultados Computacionais

Apesar de existirem métodos exatos e heurísticos para o PCVG, não se tem conhecimento de nenhuma biblioteca com instâncias do problema disponibilizada na literatura. Desta forma, tornou-se necessário a criação de um conjunto de instâncias para o PCVG genérico (*sem pré-fixação da seqüência de grupos a serem visitados*) para avaliar os métodos aqui propostos. Foram geradas instâncias de seis tipos: (tipo 1) instâncias do PCV TSPLIB95 (2007) agrupadas usando o algoritmo de clusterização *k-means*, (tipo 2) onde os vértices são agrupados em torno de um centro geométrico e disponível na literatura para o PCV Johnson e McGeoch (2002), (tipo 3) geradas através da interface Concorde Applegate *et al.* (2007), (tipo 4) geradas usando *layout* conforme Laporte *et al*. (1996), (tipo 5) semelhantes ao tipo 2, mas geradas com parâmetros diferentes Johnson e McGeoch (2002) e (tipo 6) instâncias do PCV TSPLIB95 (2007) onde a área plana retangular que compõe a instância é dividida em diversos quadriláteros e cada quadrilátero corresponde a um grupo.

A Tabela 1 mostra as instâncias com os seus identificadores, números de vértices (nós), número de grupos e tipo. Para analisar o desempenho empírico das heurísticas propostas, os resultados de G1, G2, G3, G4 e G5 foram comparados, quando possível, com a solução ótima obtida pelo método exato usando a formulação matemática da literatura proposta em Chisman (1975) e o *software* CPLEX11.2. Devido ao elevado número de iterações ocorridas no CPLEX para a maioria das instâncias, determina-se um tempo limite de 25.200 segundos para sua execução. Para a instância I_3, a solução ótima foi encontrada pelo CPLEX em 442s. Todas as outras execuções utilizaram o tempo limite de 25.200s (CPLEX *não obteve solução ótima, ou ainda não se tem esta garantia).*

Os principais parâmetros utilizados nas heurísticas são descritos a seguir. O valor da penalização dos custos das arestas que ligam nós pertencentes a grupos distintos foi igual a 10*max$\{c_{ij}\}$. A expressão max$\{c_{ij}\}$ significa o maior custo entre todos os custos das arestas (v_i,v_j).

Tabela 1. Instâncias com seus identificadores, número de nós, número de grupos e tipo.

Instâncias	Id.	# nós	# V_i	Tipo
10-lin318	I_1	318	10	1
10-pcb442	I_2	442	10	1
25-eil101	I_3	101	25	6
144-rat783	I_4	783	144	6
300-20-111	I_5	300	20	5
500-25-308	I_6	500	25	5
300-6	I_7	300	6	3
700-20	I_8	700	20	3
C1k. 0	I_9	1000	10	2
C1k.1	I_{10}	1000	10	2
200-4-h	I_{11}	200	4	4
600-8-z	I_{12}	600	8	4

O número de iterações do GRASP foi fixado em 200 (critério de parada). A cardinalidade do conjunto de soluções elite (CE), foi igual a 10. Devido à natureza aleatória do GRASP, executam-se dez vezes cada heurística aqui proposta para cada instância. Para as instâncias (**I**.), na Tabela 2 são mostrados os tempos médios medidos em segundos com 200 iterações.

Observa-se que G1 e G3 são aquelas que consomem o menor tempo médio, fato que era esperado, pois estas não utilizam a RC (G1) ou utilizam a RC mais leve RC2 (G3). Apesar de G2, G4 e G5 apresentarem tempos médios maiores que G1 e G3, estes tempos ainda são bem menores que os tempos exigidos pelo *software* CPLEX, exceto as instâncias I_9 e I_{10} para a heurística G5.

Na avaliação entre as melhores soluções obtidas pelas heurísticas GRASP e o CPLEX, compara-se a melhor solução de cada método com a melhor solução conhecida (ótima ou não). Neste contexto o *gap* médio do CPLEX ficou em 1,11% (isso porque em

muitas instâncias o CPLEX não conseguiu convergir para um ótimo no tempo limite estabelecido), G1 em 1,01%, G2 em 0,74%, G3 em 0,95%, ***G4 em 0,72%*** e ***G5 em 0,72%***. O CPLEX obteve a melhor solução (solução ótima ou não) em três num total de 12 instâncias, G2 e G4 obtiveram duas melhores soluções cada e ***G5 cinco melhores soluções***. Na comparação entre a melhor solução conhecida com as soluções médias obtidas pelas heurísticas (10 execuções de cada instância), o *gap* de G1 ficou em 1,47%, de G2 em 1,13%, de G3 em 1,21%, de G4 em 1,04% e de ***G5 igual a 1,02%***.

Tabela 2. Tempos médios de execução das heurísticas com 200 iterações.

	Tempos Médios (segundos)				
I.	G1	G2	G3	G4	G5
I_1	57,1	158,9	87,5	184,8	496,4
I_2	151,0	451,1	227,7	495,3	1385,7
I_3	2,5	3,7	4,0	6,4	8,4
I_4	750,3	3718,3	1134,0	4132,3	17507,0
I_5	51,7	121,3	75,7	142,8	308,7
I_6	225,9	720,5	331,2	822,5	2303,8
I_7	49,4	99,5	76,0	119,7	283,5
I_8	593,8	1618,1	881,6	1837,8	4605,0
I_9	1762,7	7592,9	2608,7	8749,6	39790,0
I_{10}	1765,8	9421,5	2616,6	10297,1	31141,0
I_{11}	15,8	34,0	25,1	42,4	76,9
I_{12}	364,8	1533,1	555,0	1652,1	4765,0

De acordo com os resultados apresentados anteriormente, onde o critério de parada é um número máximo de iterações, ou seja, utiliza-se o mesmo número para todas as heurísticas, pode-se observar que G5 obteve os melhores resultados, mas exigiram tempos computacionais maiores que G1, G2, G3 e G4.

Tabela 3. *Gaps* entre os melhores valores obtidos pelas heurísticas e pelo CPLEX com tempo limitado.

		Melhores Valores				
I.	**CPLEX**	**G1**	**G2**	**G3**	**G4**	**G5**
I_1	1,54	0,97	0,80	0,93	**0,76**	**0,76**
I_2	1,95	1,12	0,73	0,99	**0,66**	0,70
I_3	**0,01**	0,04	0,04	0,05	0,03	**0,01**
I_4	0,27	0,21	0,19	0,21	0,20	**0,14**
I_5	0,86	0,55	0,45	0,51	**0,43**	**0,43**
I_6	0,95	0,69	0,61	0,66	0,58	**0,56**
I_7	0,60	0,67	0,51	0,60	**0,49**	0,53
I_8	0,87	0,67	0,63	0,65	0,64	**0,43**
I_9	1,99	1,63	1,61	1,64	1,60	**1,40**
I_{10}	1,69	1,28	1,27	1,31	1,27	**1,13**
I_{11}	1,87	1,68	1,28	1,60	**1,19**	1,36
I_{12}	3,76	2,23	**1,80**	2,14	1,96	1,86
gap médio	1,36	0,98	0,83	0,94	0,82	**0,78**

Neste contexto, realizam-se novos testes computacionais colocando como critério de parada um tempo máximo idêntico para as cinco heurísticas. Coloca-se um tempo limite de 2 horas para o CPLEX e 720 segundos para as cinco heurísticas GRASP (pois as heurísticas são executadas 10 vezes). Somente a execução da instância I_3 não foi limitada por duas horas, devido à solução ótima ter sido encontrada antes pelo CPLEX. A comparação entre os melhores valores obtidos pelas heurísticas e pelo CPLEX com este novo critério de parada é mostrada na Tabela 3. Nesta tabela observa-se que o *gap* médio do CPLEX ficou em 1,36%, de G1 em 0,98%, de G2 em 0,83%, de G3 em 0,94%%, de G4 em 0,82% e de ***G5 igual a 0,78%***.

Ou seja, G5 obteve novamente o melhor desempenho nesta nova bateria de testes com oito melhores soluções, enquanto G4 obteve cinco e G2 e CPLEX apenas uma. Observa-se desta forma a importância de se usar uma RC de forma intensiva ao final do GRASP (G2 e G4). Além disso, pelos resultados das tabelas 3 e 4 concluem-se mesmo de forma empírica que a melhor estratégia foi a de utilizar conjuntamente um método de descida baseado no VND com a RC (G5). Nesta segunda bateria de testes, observa-se também que apesar de G2, G4 e G5 exigirem mais tempo por iteração, estas necessitam de menos iterações para atingir bons limites superiores de um valor ótimo.

Tabela 4. *Gaps* entre os valores médios obtidos pelas heurísticas e pelo CPLEX com tempo limitado.

		Valores Médios				
I.	**CPLEX**	**G1**	**G2**	**G3**	**G4**	**G5**
I_1	1,54	1,78	1,18	1,21	1,18	**1,09**
I_2	1,95	1,93	1,24	1,36	1,22	**1,09**
I_3	0,01	0,35	0,22	0,14	0,18	**0,06**
I_4	0,27	0,28	0,23	0,25	0,24	**0,17**
I_5	0,86	0,85	0,64	**0,63**	**0,63**	0,64
I_6	0,95	0,92	**0,73**	0,78	**0,73**	0,75
I_7	0,60	1,09	0,82	**0,78**	**0,78**	0,83
I_8	0,87	0,82	0,71	0,75	0,72	**0,50**
I_9	1,99	1,88	1,76	1,79	1,76	**1,61**
I_{10}	1,69	1,54	1,44	1,47	1,42	**1,26**
I_{11}	1,87	3,30	2,37	**2,01**	2,30	2,52
I_{12}	3,76	3,32	2,43	2,70	2,54	**2,34**
gap médio	1,36	1,51	1,15	1,16	1,14	**1,07**

A Tabela 4 ilustra o desempenho médio das heurísticas e do CPLEX quando utiliza um tempo limite como critério de parada. O *gap* médio do G1 ficou em 1,51%, de G2 em 1,15%, de G3 em 1,16%, de G4 em 1,14% e de ***G5 igual a 1,07%***. Novamente os resultados mostram a superioridade da versão G5 em relação às outras. No entanto ressalta-se o bom comportamento médio das cinco heurísticas aqui propostas mostrando que todas elas possuem uma robustez necessária para a sua confiabilidade prática. Finalmente uma nova bateria de testes foi efetuada apenas entre G4, G5 e o método exato CPLEX, para 27 instâncias de pequeno porte, mas sem limitação de tempo para o CPLEX. O número de iterações de G4 e G5 foi de 200. Para todas as instâncias, o CPLEX encontrou soluções ótimas. G4 encontrou 15 soluções ótimas com tempo médio 3,3s, enquanto que G5 alcançou 18 soluções ótimas com tempo médio de 6,05s. O *gap* médio de G4 em relação ao ótimo foi de 0,25% e ***G5 igual a 0,21%***. Este resultado reforça o

potencial de G5 em encontrar soluções melhores em tempo computacional viável.

5 Conclusões

Este trabalho apresentou propostas de heurísticas GRASP com e sem memória para a solução do PCVG. Uma análise experimental foi realizada, cujos resultados computacionais mesmo que de forma empírica, mostram que a adição de módulos com memória adaptativa nas heurísticas GRASP (G2, G3, G4 e G5) melhoram significativamente o desempenho da versão tradicional (G1) e que o uso conjugado de memória com busca local mais efetiva (VND) no GRASP (G5) tende a produzir os algoritmos mais eficientes. Trabalhos futuros incluem a adaptação dos módulos aqui propostos (memória e VND) para outras metaheurísticas tais como os Algoritmos Evolutivos e o *Iterated Local Search* (ILS).

Agradecimentos

1. Os autores agradecem a Jean-Yves Potvin, *Département d'informatique et de recherche opérationnelle*, *Université de Montréal* pelo material bibliográfico e a David S. Johnson - *AT&T Labs - Research*, pelos códigos para gerar instâncias do tipo 2.

2. Este trabalho tem o apoio parcial dos seguintes órgãos de fomento: CNPq (CT-INFO, UNIVERSAL & Bolsa de Produtividade); CAPES (Pró-Engenharia & PROCAD-NF); FAPERJ (PENSA RIO, Prioridade Rio & Cientista do Estado).

Referências Bibliográficas

Anily, S., Bramel, J. e Hertz, A. (1999). A 5/3-approximation Algorithm for the Clustered Traveling Salesman Tour and Path Problems, *Operations Research Letters*. **24**(1-2): 29-35.

Applegate, D., Bixby, R., Chvátal, V. e Cook, W. (2007). Concorde TSP Solver, http://www.tsp.gatech.edu/concorde/index, acesso em 22/12/2007.

Bastos, L. O., Ochi, L. S., e Macambira, E. (2005). GRASP with Path Relinking for the SONET Ring Assignment Problem. *Proc. of the 5th Int. Conf. on Hybrid Intelligence System*, 239-244.

Chisman, J. A. (1975). The Clustered Traveling Salesman Problem, *Computers & Operations Research*. **2**(2): 115-119.

CPLEX (2009). ILOG CPLEX 11.2 User's Manual e Reference Manual, 2009, ILOG S.A.

Ding, C., Cheng, Y. e He, M. (2007). Two-Level Genetic Algorithm for Clustered Traveling Salesman Problem with Application in Large-Scale TSPs, *Tsinghua Sci. e Technology* **12**(4): 459-465.

Gendreau, M., Laporte, G. e Potvin, J. Y. (1994). Heuristics for the Clustered Traveling Salesman Problem, *Technical Report nº CRT-94-54*, Universidade de Montreal, Montreal, Canadá.

Glover, F. (1996). Tabu search and adaptive memory programing: advance, applications and challenges. R. S. Barr, R. V. Helgason and J. L. Kennington, pp.1-75.

Guttmann-Beck, N., Hassin, R., Khuller, S. e Raghavachari, B. (2000). Approximation Algorithms with Bounded Performance Guarantees for the Clustered Traveling Salesman Problem, *Algorithmica*. **28**: 422-437.

Hansen, P. e Mladenović, N. (2003). Variable Neighborhood Search, in Glover, F. W. e Kochenberger, G. A., (eds), *Handbook of Metaheuristics*, 145-184.

Johnson, D. S. e McGeoch, L. A. (2002). Experimental Analysis of Heuristics for the STSP,in G. Gutin e A. Punnen (eds), *The Traveling Salesman Problem e its Variations*, 369-443.

Jongens, K. e Volgenant, T. (1985). The Symmetric Clustered Traveling Salesman Problem, *European Journal of Operational Research*. **19**(1): 68-75.

Laporte, G. e Palekar, U. (2002). Some Applications of the Clustered Travelling Salesman Problem, *Journal of the Operational Research Society*. **53**(9): 972-976.

Laporte, G., Potvin, J.-Y. e Quilleret, F. (1996). A Tabu Search Heuristic using Genetic Diversification for the Clustered Traveling Salesman Problem, *J. of Heuristics*. **2**(3): 187-200.

Potvin, J.-Y. e Guertin, F. (1995). A Genetic Algorithm for the Clustered Traveling Salesman Problem with a Priori Order on the Clusters, *Technical Report nº CRT-95-06*, Canadá.

Potvin, J.-Y. e Guertin, F. (1996). The Clustered Traveling Salesman Problem: A Genetic Approach, in I. H. Osman e J. Kelly (eds), *Metaheuristics: Theory & Applications*, chap. 37,619-631.

Resende, M. G., Martí , R., Gallego, M. e Duarte, A. (2008). GRASP and Path Relinking for the Max-Min Diversity Problem, *Computers & Operations Research*.

Resende, M. e Ribeiro, C. (2002). Greedy Randomized Adaptive Search Procedures. *Handbook of Metaheuristics*, Kluwer,219-249.

Silva, G. C., de Andrade, M. R. Q., Ochi, L. S., Martins, S. L. e Plastino, A. (2007). New heuristics for the maximum diversity problem, *Journal of Heuristics*, **13**: 315-336.

TSPLIB95 (2007). Traveling Salesman Problem, http://www.iwr.uniheidelberg.de/groups/comopt/software/TSPLIB95/index.html, acesso em 17/09/2007.