Aplicação de Algoritmos Genéticos e *Simulated Annealing* para o Problema do Caixeiro Viajante em uma Situação Real de Distribuição de Produtos

Paula Francis Benevides (UTFPR) paulabenevides@utfpr.edu.br
Flávia Konowalenko (UFPR) flavia.konowalendo@hotmail.com
Deise Maria Bertholdi Costa (UFPR) deise@ufpr.br
Luiz Fernando Nunes (UTFPR) nunes@utfpr.edu.br
AngelaOlandoski Barboza (UTFPR) aobarboza@yahoo.com

Resumo:
O transporte, em geral, absorve em média a porcentagem mais elevada de custos do que qualquer outra atividade logística. Por isso, muitas empresas estão repensando seus processos para redução dos mesmos. A otimização da distribuição de produtos é um problema estudado há muito tempo por pesquisadores de diversas áreas. Este tipo de problema é classificado como de otimização combinatória. Dentre as modelagens pode ser citado o Problema do Caixeiro Viajante (PCV) que tem por objetivo encontrar o menor caminho conectando-se *N* lugares de destino. O presente trabalho visa analisar e comparar, em termos de desempenho computacional e qualidade das soluções obtidas usando Algoritmos Genéticos e *Simulated Annealing* para construção das rotas para o PCV. Utilizou-se dados reais de uma distribuidora de produtos em uma determinada região da cidade de Curitiba (PR), Brasil. As coordenadas geográficas dos pontos de visitação foram extraídas do aplicativo *online Google Earth*, as quais foram convertidas em coordenadas cartesianas, para posterior aplicação dos algoritmos utilizados. Os resultados obtidos foram comparados com as rotas reais que estão sendo utilizadas por um determinado representante da referida distribuidora.
Palavras chave: Menor Custo, Problema do Caixeiro Viajante, *Simulated Annealing*.

Genetic algorithms and Simulated Annealing Applied to the Travelling Salesman Problem in a Real Situation of Distribution of Products

Abstract
The transport usually absorbs on average the highest percentage of costs than any other logistic activity. Therefore, many companies are rethinking their processes to reduce them. The optimization of product distribution is a problem studied long time ago by researchers in several areas. This type of problem is classified as combinatorial optimization. Of the modeling can be cited the Traveling Salesman Problem (TSP) which aims to find the shortest path connecting N places of destination. This paper to analyze and compare in terms of computational performance and quality of solutions obtained using genetic algorithms and simulated annealing for the construction of routes for the PCV. We used real data from a distributor of products in a specific region of Curitiba (PR), Brazil. The geographical coordinates of places to visit are from the online application Google Earth, which was converted into Cartesian coordinates, for subsequent application of the algorithms used. The results were compared with actual routes that are being used by a particular representative of that distributor.
Key-words: Low Cost, Traveling Salesman Problem, Simulated Annealing.

1. Introdução

O Problema do Caixeiro Viajante (PCV) consiste em estabelecer uma única rota que passe por cada nó de um grafo, uma e apenas uma vez, retornando ao nó inicial no final do percurso. Este roteiro Hamiltoniano deve ser feito de modo que a distância total percorrida seja mínima.

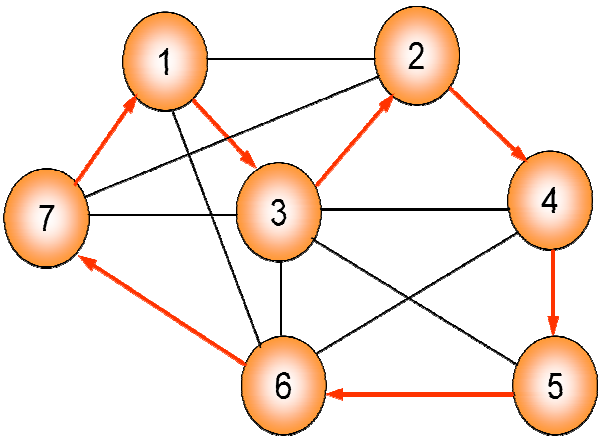

Figura 1 – Exemplo do PCV

O conjunto de rotas possíveis para o PCV Simétrico, isto é, se a distância do ponto "a" ao ponto "b" é igual ao do ponto "b" ao ponto "a",é o resultado de todas as combinações possíveis e pode ser calculado por $(n-1)!$, sendo "n" o número de nós. Este problema pertence a classe de problemas conhecida por NP-Hard, isto é, não existem algoritmos com limitação polinomial capazes de resolvê-lo. Assim a quantidade de passos de um algoritmo que possa solucioná-lo otimamente não pode ser dada por uma função polinomial do tamanho de sua entrada. Logo, apenas os problemas de pequeno porte podem ser solucionados de forma ótima. Problemas maiores tornam-se inviáveis através dos métodos exatos, haja vista o esforço computacional que seria exigido para resolvê-los. Muitas abordagens de algoritmos heurísticos, que fornecem soluções factíveis próximas da ótima, têm sido desenvolvidas para resolver os problemas NP-Hard, apresentando soluções aproximadas e as algumas vezes ótimas para o problema.

1.1Formulação Matemática para o PCV

Seja o grafo G(N,A) onde N representa o conjunto de vértices $(|N| = n)$ e A o conjunto de arcos. Se for admitido que os custos ou distâncias mínimas entre os nós da rede são dados pela matriz $C = [c_{ij}]$simétrica, isto é, considera-se que $c_{ij} = c_{ji}$, assumindo ainda que $c_{ij} = +\infty$, $\forall i \in N$ e considerando a matriz $X = [x_{ij}]$ das variáveis de decisão do problema, onde $x_{ij} = \begin{cases} 1, \text{ se o arco } a_{ij} \in \text{ rota} \\ 0, \text{ se o arco } a_{ij} \notin \text{ rota} \end{cases}$ tem-se a seguinte formulação de programação linear inteira para o problema, devida a Golden (1977), (Bodin *et al*., 1983) :

$$\text{Minimizar} \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{s.a.}: \sum_{i=1}^{n} x_{ij} = 1 \quad (j = 1, 2, \ldots, n)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad (i = 1, 2, \ldots, n)$$

$$X = (x_{ij}) \in S$$

$$x_{ij} = 0 \text{ ou } x_{ij} = 1 \quad (i, j = 1, 2, \ldots, n)$$

Os dois primeiros grupos de restrições garantem que exatamente um arco(i, j) emana de cada nó da rota e exatamente um arco (i, j) é direcionado para cada nó j da rota. A penúltima restrição contém um conjunto S que pode ser qualquer conjunto de restrições que impeça a formação de subrotas. Estas restrições são chamadas restrições de quebra de subrotas e podem ser, entre outras :

$$S = \left\{ (x_{ij}) : \sum_{i \in Q}\sum_{j \notin Q} x_{ij} \geq 1, \text{ para todo subconjunto próprio não vazio Q de N} \right\};$$

$$S = \left\{ (x_{ij}) : \sum_{i \in R}\sum_{j \in R} x_{ij} \leq |R| - 1, \text{ para todo subconjunto não vazio R de } \{2, 3, \ldots, n\} \right\}$$

$$S = \left\{ (x_{ij}) : y_i - y_j + n x_{ij} \leq (n-1), \text{ para } 2 \leq i \neq j \leq n \text{ para alguns números reais } y_i \right\},$$

onde $y_i = t$ se o nó *i* é visitado no passo t da rota ou $y_i = 0$, caso contrário.

1.2 Um Algoritmo Genético para o Problema do Caixeiro Viajante

Existem diversas abordagens para o problema do caixeiro viajante utilizando algoritmos genéticos. Elas diferem entre si não apenas na questão dos parâmetros, mas também na forma de representar as soluções viáveis, de selecionar os indivíduos para reprodução e na maneira de definir os operadores genéticos.

Considere-se $X = \{x_1, x_2, \ldots, x_n\}$ o conjunto das n cidades pelas quais o caixeiro viajante deve passar, retornando ao ponto de partida no final da jornada. Assumindo que o custo da viagem entre cada par de cidades é conhecido e dado por $w(x_i, x_j)$, $\forall x_i, x_j \in X$, sendo $w(x_i, x_j) > 0$, define-se seguir a estrutura do cromossomo, a função de *fitness* e os operadores de *crossover* e mutação utilizados neste trabalho.

1.2.1 Estrutura do Cromossomo

Existem várias maneiras de codificação do espaço de busca para o PCV. Dentre as principais representações estão: a ordinal, por caminho (ou inteiros) e por adjacência. Neste trabalho optou-se em utilizar a Representação por Caminho, onde o cromossomo é formado pela sequência dos nós na solução.

Por exemplo, o cromossomo c_1 = {1, 2, 3, 4, 5, 6, 7} representa diretamente a solução r_1 {1, 2, 3, 4, 5, 6, 7}. Esta representação é talvez a mais natural de uma solução para o PCV e de simples implementação.

1.2.2 Função de Aptidão

A função de aptidão ou função de *fitness* representa a capacidade de um indivíduo de se adaptar ao meio ambiente. Neste algoritmo adotou-se como *fitness* de um cromossomo r_i, o custo total do circuito, onde n é o número de cidades a ser visitadas.

$$Fitness(r_i) = w(x_n, x_1) + \sum_{j=1}^{n-1} w(x_j, x_{j+1})$$

Para problemas de minimização, quanto menor for o custo total do circuito, maior será o *fitness* do cromossomo correspondente e conseqüentemente maiores serão as chances deste indivíduo sobreviver e vir a se reproduzir (Mayerle, 1994).

1.2.3 Processo de Seleção

A principal idéia do processo de seleção é permitir que os indivíduos mais adaptados (melhor *fitness*) tenham maior chance de se reproduzir. Barboza (2005) afirma que a seleção elitista consiste em copiar ou reproduzir os melhores indivíduos da população atual para a próxima geração, garantindo que estes cromossomos não sejam destruídos nas etapas de recombinação e mutação. Sua vantagem é que se no caso ótimo global for descoberto durante o processo de busca, o AG deve convergir para tal solução.

Considerando que a população possui m cromossomos dispostos em ordem crescente dos seus custos, isto é $C_1 \leq C_2 \leq ..... \leq C_m$, onde C_i é o custo associado ao cromossomo r_i, a escolha de um indivíduo para ser submetido aos operadores genéticos é feita considerando uma distribuição de probabilidade inversamente proporcional ao índice dos cromossomos. Assim, quanto menor for o índice de um cromossomo, maior é a probabilidade do mesmo ser selecionado. De acordo com esta distribuição, a função de seleção é a seguinte (Mayerle, 1994):

$$Select\,(R) = \left\{ r_j \in R \,/\, j = m + 1 - \left\lceil \frac{-1 + \sqrt{1 + 4.Rnd\,(m^2 + m)}}{2} \right\rceil \right\}$$

onde $\mathrm{Rnd} \in [0,1)$ é um número aleatório uniformemente distribuído, $\lceil b \rceil$ é o menor inteiro maior do que b e $R = \{r_1, r_2, ..., r_m\}$ é o conjunto ordenado dos cromossomos, de modo que $C_1 \leq C_2 \leq ..... \leq C_m$ e $C_i = \mathrm{Fitness}(r_i)$.

1.2.4 Operador de Cruzamento

De acordo com Potvin (1996) existem diversos operadores de cruzamento desenvolvidos para manipular as representações descritas acima e estão classificados em: os que preservam a posição absoluta das cidades e os que preservam a ordem relativa.

Dentre os operadores que mantém a posição absoluta estão: o PMX (*Partially Mapped Crossover)* proposto por Goldberg e Lingle (1985 apud POTVIN, 1996) e o CX (*Cycle Crossover),* proposto por Oliver (1987 apud POTVIN, 1996). E entre os operadores que mantém a ordem relativa está o OX (*Order Crossover)* desenvolvido por Davis (1985 apud POTVIN, 1996)

Além destes operadores, Silva e Oliveira (2006), fizeram um estudo comparativo utilizando uma versão adaptada do HX (*Heuristic Crossover),* o qual destacou-se sobre os demais operadores, sobretudo quando o número de nós aumenta.

- **Operador HX:** O operador HX utiliza a distancias entre as cidades, isto é, o tamanho dos arcos. O mesmo pode ser descrito da seguinte forma:

Algoritmo:

P1: Escolha uma cidade aleatória inicial ***i*** de um dos pais.

P2: Calcular as distâncias entre a cidade aleatória ***i*** escolhida e suas vizinhas (i - 1 e i + 1) em ambos os pais.

P3: No cromossomo filho, escolher para suceder a cidade i, a vizinha da mesma cuja distância até i seja a menor possível, a qual formará uma subrota.

P4: Se a inserção de qualquer das cidades vizinhas de i fecharem um ciclo, antes que todas as cidades sejam incluídas, sortear outra cidade (dentre aquelas que não pertencem a atual subrota). Esta cidade sorteada sucederá i na subrota.

P5: Repita os passos 2, 3 e 4 até que todas as cidades sejam incluídas na rota.

1.2.5 Operador de Mutação

Segundo Malaquias (2006), o algoritmo genético pode convergir muito rapidamente para uma região específica do espaço de busca se nenhum mecanismo seja implementado para que evite isso. Existe uma tendência de convergência rápida para uma região de mínimos locais ao invés de minimos globais. Para que isso não ocorra, impõe-se uma rotina para explorar outras áreas do espaço de busca por meio de alterações nos genes por meio da mutação

Os operadores de mutação introduzem uma alteração aleatória no individuo.A probabilidade de se efetuar uma mutação deve ser relativamente baixa, caso contrário o algoritmo se comportará fazendo uma busca aleatória, dificultando a convergência. Assim sendo, pode-se considerar os operadores de mutação como uma espécie de "apólice de seguro" contra perdas acidentais deste material genético de boa qualidade.

A mutação pode ser feita por troca *(Swap),* queconsiste na troca aleatória de posição entre dois genes. Por exemplo, temos o indivíduo {1, 2, 3, 4, 5, 6, 7} e que após a mutação fica {1, 2, 6, 4, 5, 3, 7}.

1.2.6 Descrição do Algoritmo Proposto

Considerando as definições anteriores, e usando os princípios da evolução das espécies, pode-se construir um algoritmo genético, cujos principais passos são os seguintes (Mayerle, 1994; Bezerra, 1995):

P1: *Construção da população inicial*: gere os m cromossomos aleatórios representando os roteiros Hamiltonianos. Calcule os custos $C_i = [Fitness(r_i)]$, $(1 \le i \le m)$ de todos os cromossomos gerados, construindo uma lista $R = (r_1, r_2, ..., r_m)$, de forma que $C_1 \le C_2 \le ..... \le C_m$; faça $k = 0$, defina o erro ε e o número máximo de iterações k_{max} ;

Obs. Os cromossomos são construídos de modo que a probabilidade de uma cidade suceder outra, seja inversamente proporcional ao quadrado da distância que as separa;

P2: *Teste*: se $C_n - C_1 \le \varepsilon$ ou $k \ge k_{max}$, então PARE e apresente o cromossomo r_1;

P3: *Seleção Natural*: selecione dois cromossomos $r_p = Select(R)$ e $r_q = Select(R)$ com $r_p \ne r_q$;

P4: *Reprodução*: faça $r_f = \text{Crossover}(r_p, r_q)$;

P5: *Mutação*: aplique o operador de mutação com uma probabilidade *p*;

P6: Calcule $F_f = \text{Fitness}(r_f)$ e caso o fitness do cromossomo filho for melhor do que o fitness do pior cromossomo,elimine esse cromossomo, inserindo o cromossomo r_f na lista R, mantendo a ordem crescente dos custos; faça $k = k + 1$, e volte ao P2.

1.3 Simulated Annealing

A origem da técnica de otimização conhecida por *Simulated Annealing* vem de 1953, quando foi usada para simular em um computador o processo de "*annealing*" de cristais. A idéia de aplicar este método para resolver problemas de otimização combinatória surgiu bem mais tarde (KIRKPATRICK *et al*., 1983), (ARAGON *et al*., 1984).

O método surgiu da seguinte observação da mecânica estatística:

O resfriamento gradativo de um material a partir de uma alta temperatura inicial leva o material a estados mínimos de energia. Informalmente esses estados são caracterizados por uma perfeição estrutural do material congelado que não se obteria caso o resfriamento não tivesse sido gradativo. Sob outras condições menos cuidadosas de resfriamento, o material se cristalizaria com uma energia "localmente mínima", apresentando imperfeições estruturais. A esse processo cuidadoso de resfriamento dá-se o nome de "*annealing*".

A referida simulação a uma temperatura fixa T consiste em dar um pequeno deslocamento a um dos átomos, computando a variação ΔE da energia do sistema. Se $\Delta E \leq 0$, o deslocamento é incorporado ao estado do sistema, que é então utilizado no passo seguinte. Caso contrário, isto é, se $\Delta E > 0$, a aceitação ou não do deslocamento passa a ser uma decisão probabilística.

A "intuição" que está por trás da proposta de se utilizar *Simulated Annealing* como ferramenta de otimização combinatória é a seguinte:

- Identifica-se a função energia do sistema com a função objetivo que se quer otimizar, por exemplo minimizar, e os átomos do sistema são associados às variáveis do problema.
- Para cada temperatura de uma sequência de temperaturas decrescentes realiza-se a simulação descrita. No final do processo, espera-se que o sistema estacione em um estado de energia globalmente mínima, por analogia com a física do problema.

O fato do método *Simulated Annealing* permitir a aceitação de configurações intermediárias do problema em que cresce o valor da função objetivo que se deseja minimizar é crucial. Essa aceitação temporária de soluções "piores", significa que o método admite caminhar "morro acima", na esperança de encontrar "vales" mais profundos.

Na proposta original de *Simulated Annealing* a função r que promove a redução de temperatura foi dada por:

$$r(k) = \alpha^{k+1} T_o \quad , \quad \text{para todo } k \geq 0 \quad \text{e} \quad \text{algum } 0 < \alpha < 1$$

A probabilidade de o algoritmo aceitar j como solução factível a partir de uma solução i corrente é dada por:

$$\text{Prob}\{\text{ j vir depois de i }\} = \begin{cases} 1 & \text{se } \Delta \leq 0 \\ e^{\frac{-\Delta}{T}} & \text{se } \Delta > 0 \end{cases} \quad \text{onde} \quad \Delta = f(j) - f(i)$$

Quando a temperatura T possui um valor relativamente alto, praticamente todo deslocamento é aceito, uma vez que a probabilidade é praticamente igual a 1. À medida que o valor de T decresce, a aceitação de soluções com maior valor na função objetivo torna-se cada vez mais improvável.

Algumas decisões e escolhas de parâmetros dependem do problema particular em questão. Por exemplo, a escolha da solução inicial x_o e a escolha do conjunto $A(x) \subset D$ de soluções dentre as quais uma será selecionada para o teste de aceitação, quando a simulação se encontra no ponto x, sendo D o conjunto dos valores factíveis, podem variar de problema para

problema. Segundo Barbosa (1989), e Mitra*et al*., (1986), um dos principais resultados relativos às condições sob as quais se pode garantir a convergência para um mínimo local, é selecionar T_o e r(k) de tal forma que: $T_o = \frac{\xi}{\log(1+k_o)}$ e $r(k) = \frac{\xi}{\log(2+k_o+k)}$ para $k \geq 0$, onde k_0 é qualquer parâmetro que satisfaça $1 \leq k_0 \leq \infty$ e ξ é uma constante que depende das características da cadeia de Markov associada ao Processo de "*Simulated Annealing*".

O maior "mistério" do algoritmo é decidir quando a temperatura T será adaptada. A verificação da verdadeira condição de equilíbrio é realmente bastante difícil. Uma solução simples é introduzir outro parâmetro H, fixando o número máximo de iterações para ser usado com a temperatura T, provavelmente dependendo do número de variáveis do problema.

Os passos do algoritmo contidos em Barbosa (1989) são os seguintes:

Seja $x_0 \in D$ a solução inicial e T_0 a temperatura inicial;

$k = 0$; $x_k = x_0$; $T_k = T_0$

While $T_k \geq T_{min}$ ***do***

begin

While ainda não em equilíbrio a temperatura T_k ***do***

begin

selecione aleatoriamente um ponto $x' \in A(x_k) \subset D$

$\Delta = f(x') - f(x)$

if $\Delta \leq 0$ ***then***

$x_k = x'$

else

$x_k = x'$ com probabilidade $e^{\frac{-\Delta}{T_k}}$

end

$T_{k+1} = r(k)$; $x_{k+1} = x_k$; $k = k+1$

End

A solução é x_k .

2. Testes e Resultados computacionais

Os parâmetros utiliazados para o Algoritmo Genético foram de 50 cromossomos, no máximo 10000 iterações,a diferença máxima entre o melhor e o pior fitness foi de 0,00001 e a probabilidade de mutação de 0,01. O tempo computacional girou em torno de 3 segundos aproximadamente.

Para o *Simulated Annealing* os parâmetros utilizados foram de no máximo 1000 iterações, 1000 perturbações por iteração e fator de redução de temperatura de 0,95. A temperatura

inicial foi calculada por $T_o = \frac{\Delta E}{\log(\xi)}$, sendo $\xi = 0{,}03$ e ΔE a média das diferenças de custos obtidos para algumas perturbações efetuadas em uma rota inicial. O tempo computacional girou em torno de 4 minutos aproximadamente.

Os algoritmos foram implementados em MATLAB 5.3 e os testes computacionais foram realizados em um notebook com processador Intel(R) Core(TM) 2 Duo T5550, 1,83 GHz e 2,99 GB de RAM.

Para realizar a comparação das metaheuristicas explanadas na seção anterior, utilizou-se cinco conjunto de pontos de visitação, no qual cada um desses conjuntos representa um dia da semana de um representante da distribuidora de produtos. Os endereços destes pontos foram inseridos no *Google Earth*no qual obtivemos as coordenadas geográficas dos pontos de visitação. A utilização dessas coordenadas nos fornece um maior grau de precisão e com isso resultados mais significativos para a pesquisa.De posse de todos os pontos mapeados geograficamente, parte-se para a conversão destes pontos em um plano cartesiano. Para que isso seja feito, deve-se converter as coordenadas geográficas que são dadas em latitudes e longitudes, sendo referência para localização global em coordenadas cartesianas, que se referem a distâncias métricas a partir de determinada referência. Para fazer esta conversão foi utilizada uma planilha no Excel, no qual ao entrarmos com a latitude e longitude, estas eram transformadas em latitude decimal e longitude decimal, para que fossem transformadas em coordenadas cartesianas X e Y, para que pudessem ser calculadas as distâncias euclidianas entre os pontos que o representante precisa visitar.

Na tabela 1 temos os dados das rotas atuais praticadas pelo representante e os resultados computacionais obtidos com as Metaheuristica Algoritmo Genético e *Simulated Annealing*.

.

Dia da semana	**Segunda 25 pontos**	**Terça 26 pontos**	**Quarta 28 pontos**	**Quinta 20 pontos**	**Sexta 33 pontos**
Rota atual	15808,84	25198,19	25185,87	36156,98	20691,49
Algoritmo Genético	9842,52	**16451,17**	18798,64	**21309,23**	**18409,83**
Simulated Annealing	**9811,77**	16600,13	**18691,17**	22348,60	18808,38

Tabela 1. Comparação dos resultados (custos em metros) das metaheuristicas

3. Conclusões

Neste trabalho foram realizados testes com as Metaheuristicas implementadas. Em todos os dias da semana houve melhoria na rota já praticada pelo representante.

Na maioria dos casos o desempenho do Algoritmo Genético foi melhor, e quando isso não ocorreu, a diferença não foi tão significativa em comparação com o *Simulated Annealling*.

Com estes resultados, concluí-se que a aplicação destas metaheurísticas é viável para melhorar os roteiros já utilizados pela empresa. Os tempos computacionais são baixos, o que permite rápidas atualizações dos roteiros quando forem alterados os pontos dos mesmos. A implementação de algoritmos de melhorias de rotas como 2-Opt, poderá certamente melhorar ainda mais a qualidade das soluções, exigindo para isto um tempo computacional pouco significativo.

Referências

ARAGON, C.R.; JOHNSON, D.S.; McGEOCH L.A.; SCHEVON C., *Optimization by simulated annealing: an experimental evaluation*; Workshop on Statistical Physics in Engineering and Biology, 1984.

BARBOSA, V.C.;*Redes Neurais e Simulated Annealing como Ferramentas para Otimização Combinatória*; Investigación Operativa, Vol.1, N.2, 1989.

BARBOZA, A.O.; Simulação e Técnicas da Computação Evolucionária Aplicadas a Problemas de Programação Linear Inteira Mista**.** Tese de Doutorado, UTFPR, 2005.

BEZERRA, O.B.; Localização de postos de coleta para apoio ao escoamento de produtos extrativistas - Um estudo de caso aplicado ao babaçu. Dissertação de Mestrado, UFSC, 1995

BODIN, L.; GOLDEN, B.; ASSAD, A.; BALL, M.; Routing and Scheduling of Vehicles and Crews – The State of the Art; Computers Ops Res; Vol. 10, Number 2; Pergamon Press, 1983.

KIRKPATRICK, S.; GELATT Jr, C.D.; VECCHI, M.P.; *Optimization by Simulated Annealing*; Science, Vol 220, N.4598, 1983.

MALAQUIAS, N.G.L.; Uso *dos Algoritmos Genéticos para a Otimização de Rotas de Distribuição;* Dissertação de Mestrado. Universidade Federal de Uberlândia, 2006

MAYERLE, S.F.; *UmAlgoritmo Genético para o Problema do Caixeiro Viajante***.** Artigo de Circulação Interna do Departamento de Engenharia de Produção e Sistemas da UFSC, 1994.

MITRA, D.; ROMEO, F.; SANGIOVANNI-VINCENTELLI, A.; *Convergence and finite-time behavior of simulated annealing*; Advances in Applied Probability, N.18, pp.747-771, 1986.

POTVIN, J. Y. *Genetic algorithms for the traveling salesman problem*. Annals of Operations Research 6, p.339 - 370, 1996. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=F65927180919D4CDA044AFF23E372615?doi=10.1.1.93.2179&rep=rep1&type=pdf>. Acesso em 03/05/2011.

SILVA, A. F., OLIVEIRA, A.C. *Algoritmos genéticos: alguns experimentos com os operadores de cruzamento ("Crossover") para o problema do caixeiro viajante assimétrico***.**XXVI ENEGEP – Fortaleza, 2006. Disponível em:http://www.abepro.org.br/biblioteca/ENEGEP2006_TR460314_7093.pdf. Acesso em 13/04/2011.

WHITLEY, D.; STARKWEATHER, T.; FUQUAY, D.; *Scheduling Problems and Traveling Salesman: The Genetic Edge Recombination Operator*; Proceedings of the Third International Conference on Genetic Algorithms, George Mason University, 1989.