Um Algoritmo Imunológico para a Solução do Problema do Caixeiro Viajante

Thiago A. S. Masutti[1], Leandro N. de Castro[2]

[1]Laboratório de Sistemas Inteligentes – Universidade Católica de Santos
R. Dr. Carvalho de Mendonça, 144, Vila Mathias – Santos – SP – Brasil – 11070-906

[2]Programa de Pós-Graduação em Engenharia Elétrica – Universidade Mackenzie
R. da Consolação, 896 – São Paulo – SP – Brasil – 01302-907

thiago.masutti@gmail.com, lnunes@mackenzie.br

Abstract. *Evolutionary and immune inspired algorithms are presented as efficient approaches to solve combinatorial optimization problems. This report summarizes an immune approach to solve the traveling salesman problem (TSP), one of the most studied vehicle routing problems in the literature. As a benchmark, a genetic algorithm to solve the TSP is also investigated. In this report, both algorithms are described, detailed and compared when applied to several standard instances from the literature. For both algorithms the obtained results show a good performance in relation to the set of instances used, detaching the copt-aiNet, which was able to achieve better quality results.*

Resumo. *Algoritmos evolutivos e imunológicos são apresentados como abordagens eficientes para a solução de problemas combinatórios. Este relatório apresenta um algoritmo imunológico para a solução do problema do caixeiro viajante (PCV). Para efeitos comparativos (benchmarking) também é investigado o uso de um algoritmo genético aplicado ao PCV. Neste relatório, a descrição de ambos algoritmos é apresentada e os resultados de experimentos computacionais com instâncias comumente utilizadas na literatura são descritos. Os resultados obtidos com as duas ferramentas demonstram um bom desempenho para o conjunto de instâncias avaliadas, com destaque para a copt-aiNet, que obteve resultados de melhor qualidade.*

1. Introdução

A Computação Natural [de Castro 2007] tem se destacado por diversas abordagens para a solução de problemas na qual a interação de seus agentes constituintes promove a emergência de algum fenômeno inteligente que leva a determinação de uma boa solução para um dado problema. Dentre estas abordagens, aquelas classificadas como algoritmos evolutivos (AE) [Eiben e Smith 2003] e algoritmos imunológicos [de Castro e Timmis 2002] têm se destacado pela eficiente solução de problemas de busca e otimização, principalmente os de caráter combinatório.

Através da inspiração na biologia evolutiva, principalmente na teoria da seleção natural, os algoritmos evolutivos propõem um paradigma de solução de problemas. Dada uma ou mais populações de indivíduos, os de melhor qualidade têm maior probabilidade de sobrevivência e reprodução, podendo, portanto, continuar no processo de busca

pela solução ótima. Este característica da evolução das espécies também é observada em alguns algoritmos de sistemas imunológicos artificiais baseados na teoria da seleção clonal e maturação de afinidade de células imunes [de Castro e Timmis 2002]. Neste sentido, há algoritmos imunológicos que também podem ser considerados algoritmos evolutivos, mesmo que desenvolvidos a partir de teorias imuno-evolutivas e não apenas evolutivas como no caso dos algoritmos evolutivos convencionais.

Este projeto de iniciação científica teve como objetivo o estudo de duas classes de algoritmos evolutivos para a solução de um problema clássico de otimização combinatória: o problema do caixeiro viajante (PCV). Para isso foi considerado um algoritmo genético modificado para tratar este problema e um algoritmo imunológico projetado para resolver problemas de natureza combinatória.

O restante deste relatório está organizado da seguinte maneira. A Seção 2 descreve o problema investigado; a Seção 3 descreve a implementação do algoritmo genético; a Seção 4 descreve a implementação do algoritmo evolutivo inspirado em sistemas imunológicos; a Seção 5 apresenta os resultados para os testes computacionais; e o relatório é concluído na Seção 6.

2. O Problema do Caixeiro Viajante

O PCV pode ser descrito da seguinte forma. Dado um conjunto de *M* cidades, o objetivo é determinar a rota de menor custo tal que: (1) ela inicia e termina na mesma cidade; e (2) cada cidade é visitada apenas uma vez. Apesar do fácil entendimento, o PCV é classificado como NP-hard [Garey e Johnson 1979], impedindo o uso de métodos exatos de solução para resolver instâncias grandes, pois seu espaço de busca tem um crescimento fatorial em relação ao número de cidades. Por causa desta e outras características, o PCV é um dos problemas de otimização combinatória mais estudados na literatura, tendo diversas abordagens de solução propostas [Johnson e McGeoch 1997].

3. Um Algoritmo Genético para a Solução do PCV

Os Algoritmos Genéticos (AG) compõem a classe mais estudada de algoritmos evolutivos. Sua versão mais popular foi apresentada por Holland (1992) como um meio de estudo de comportamentos adaptativos, embora atualmente eles sejam muito utilizados para a solução de problemas de busca e otimização [Eiben e Smith 2003]. A seguir são descritas as principais etapas de um AG e de sua modelagem ao PCV, de acordo com o pseudocódigo apresentado na Figura 1.

```
t = 0
População Inicial //gerar população de possíveis soluções ao PCV
Avaliação //avaliar a qualidade de cada solução da população
Até a t-ésima geração
   Seleção dos pais //selecionar, a partir da população, pares de soluções
   Recombinação //gerar novas soluções através de operadores de recombinação
   Mutação //gerar novas soluções através de operadores mutação
   Avaliação dos descendentes //avaliar a qualidade das novas soluções
   Seleção dos Sobreviventes //selecionar as soluções da próxima geração
   t = t + 1
```

Figura 1. Pseudocódigo para o algoritmo genético.

Representação dos indivíduos: A representação dos indivíduos é o modo pelo qual o contexto do problema é passado ao algoritmo de forma que ele possa manipular os candidatos à solução. Nos AGs esta representação é chamada de cromossomo (genótipo) do indivíduo e sua tradução ao contexto do problema é o seu fenótipo. No caso particular

do problema a ser resolvido, o fenótipo de um indivíduo corresponde à ordem de visita das cidades. Por causa disso, a representação genotípica de um indivíduo é feita por uma permutação de inteiros como a ilustrada na Figura 2.

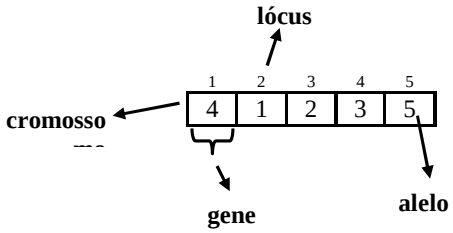

Figura 2. Representação de uma solução candidata no algoritmo genético.

População inicial: A população inicial é criada com um número *N* de indivíduos. Caso *N* seja menor ou igual ao número *M* de cidades, cada indivíduo é criado utilizando a heurística construtiva de vizinho mais próximo com uma cidade inicial diferente. Caso *N* seja maior que *M*, *M* indivíduos são criados como descrito acima e os $N - M$ demais indivíduos são criados com a permutação aleatória das cidades.

Seleção dos pais: Nesta implementação foi utilizada uma seleção *elitista* na qual 50% dos indivíduos melhores adaptados (com maior *fitness*) são selecionados deterministicamente e os outros 50% indivíduos são selecionados aleatoriamente a partir da população com reposição. Assim, cada par de pais é constituído por ao menos um indivíduo bem adaptado, dando maior probabilidade à permanência dos genes de bons indivíduos na população.

Recombinação: A etapa de recombinação contribui para que segmentos genéticos de diferentes indivíduos sejam combinados e propagados para os descendentes. Dado um par de pais, um método de recombinação gera dois novos indivíduos através do cruzamento de suas informações genéticas. Diversos operadores de recombinação foram desenvolvidos para o PCV [Larrañaga et al. 1999]. Nesta implementação do AG, foram utilizados os seguintes: (1) uma variante do *order crossover* [Davis 1991] que, ao invés de realizar cópia de apenas um bloco genético, realiza a cópia de *n* blocos genéticos dos cromossomos dos pais; e (2) o *partially mapped crossover* [Whitley 2000]. A realização da recombinação a um dado par de pais é probabilística da seguinte forma: (1) $1 - pr$ é a probabilidade deste par não sofrer recombinação; (2) $pr/2$ é a probabilidade dele sofrer recombinação do tipo *n-order crossover*; e (3) $pr/2$ é a probabilidade dele sofrer recombinação do tipo *partially mapped crossover*.

Mutação: A etapa de mutação consiste em criar variabilidade genética na população através de pequenas mudanças no código genético de um indivíduo, com o intuito de introduzir diversidade nas soluções candidatas. A mutação pode ocorrer de acordo com uma probabilidade de mutação *pm*. Nesta implementação são utilizados cinco tipos de operadores de mutação, com iguais probabilidades de ocorrência: (1) *swap*, que realiza a permuta de alelos entre dois genes; (2) *insert*, que consiste na inserção de um gene em uma nova posição; (3) *scramble*, que consiste em embaralhar a ordem de ocorrência de alelos entre dois genes; (4) *invertion*, que realiza a inversão da ordem de ocorrência de alelos entre dois genes; e (5) *displacement*, que funciona como a mutação *insert*, mas utilizando um bloco genético ao invés de um único gene. Estes operadores foram baseados na implementação proposta em [Eiben e Smith 2003].

Seleção dos sobreviventes: Esta etapa consiste em determinar os indivíduos que farão parte da população na geração seguinte, dando continuidade ao processo evolutivo. Nesta implementação, a manutenção dos indivíduos foi feita com o objetivo de manter aqueles melhores adaptados ao problema. Para gerar a nova população (t + 1) (os sobreviventes), são escolhidos 50% dos melhores indivíduos da população atual (t) e 50% dos melhores indivíduos gerados através da recombinação e mutação.

4. copt-aiNet: Algoritmo Imuno-Evolutivo para Otimização Combinatória

A copt-aiNet, acrônimo em inglês para *rede imunológica artificial para otimização combinatória*, é um algoritmo baseado em sistemas imunológicos artificiais para a solução de problemas de otimização combinatória. Este algoritmo foi originalmente apresentado por Souza et al. (2005) para a reordenação de dados de expressão gênica.

A copt-aiNet utiliza três conceitos importantes de sistemas imunológicos artificiais: (1) seleção clonal e maturação de afinidade; (2) teoria da rede imunológica; e (3) recombinação gênica. A teoria da seleção clonal e maturação de afinidade possui, intrinsecamente, características evolutivas, mas em uma escala de tempo muito inferior aquela da evolução das espécies. Isso permite a classificação da copt-aiNet como um algoritmo evolutivo inspirado no sistema imunológico dos vertebrados. A seguir são descritas as principais etapas da copt-aiNet, conforme pseudocódigo apresentado na Figura 3.

População inicial: Na copt-aiNet a população é composta por um conjunto de anticorpos, que no contexto de algoritmos evolutivos são os indivíduos da população, ou seja, os candidatos a solução do problema. A representação de um anticorpo é feita da mesma forma daquela apresentada na Figura 2. A população inicial da copt-aiNet é inicializada de forma semelhante à do AG da Seção 3.

```
Criar a população inicial
Avaliar a população inicial
Enquanto o critério de parada não é alcançado
   Realizar Seleção Clonal
   Se a população de anticorpos estabilizou
      Realizar a supressão
   Se o tamanho da população não atingiu seu limite
      Realizar inserção de anticorpos por recombinação
   Se não houve aperfeiçoamento nas K melhores soluções
durante um determinado número de gerações
      Realizar maturação fraca
```

Figura 3. Pseudocódigo para a copt-aiNet.

Seleção clonal: Cada anticorpo da população gera um conjunto de N_C descendentes através de clonagem, ou seja, são geradas células idênticas aos seus genitores (pais), chamadas clones. Cada clone sofre um número *ng* de mutações com a intenção de criar variantes a partir de células já existentes na população. O número *ng* de mutações que um clone sofrerá é inversamente proporcional ao *fitness* do anticorpo que o gerou. Clones de anticorpos de *fitness* alto tendem a sofrer um menor número de mutações. Já os clones de anticorpos de baixo *fitness* tendem a sofrer um maior número de mutações. Desta forma, os clones são expostos ao processo de mutação, que utiliza os mesmos operadores descritos para o AG. Realizadas as mutações em todos os clones de um anticorpo, o *fitness* de cada clone é calculado e aquele de maior *fitness* é selecionado. Caso seu *fitness* seja maior que o do seu anticorpo pai, este clone o substituirá na população.

Supressão: Caso a população alcance uma estabilidade em relação à qualidade das soluções propostas, a etapa de supressão realiza a eliminação de anticorpos com alto grau de

afinidade entre si, possibilitando a manutenção da diversidade da população. Dois anticorpos possuem alto grau de afinidade quando as suas estruturas são semelhantes. Dado o contexto do PCV, são considerados anticorpos de alta afinidade aqueles que apresentam uma taxa de co-ocorrência de adjacências entre cidades maior que um limiar pré-definido. Entre dois anticorpos com alto grau de afinidade, aquele que representa a solução de pior qualidade (menor *fitness*) é eliminado. O cálculo de afinidade é realizado par a par, até que todos os anticorpos da população tenham sido comparados.

Inserção de anticorpos por recombinação gênica: Caso a população não tenha atingido seu tamanho máximo, novos anticorpos são inseridos através da recombinação de anticorpos presentes na população. Esta recombinação é feita pelo *n-order crossover*.

Maturação fraca: Esta etapa se faz necessária quando nenhuma das *K* melhores soluções da população é aperfeiçoada durante um determinado número de gerações. Sendo assim, cada anticorpo da população é exposto ao processo de maturação fraca, que realiza alterações de forma mais eficiente que o processo de mutação. Nesta implementação da copt-aiNet, a execução de um determinado número de iterações do processo de busca local 2-opt [Lin e Kernighan 1973] faz o papel da maturação fraca.

Critério de parada: Caso as *K* melhores soluções não sejam aprimoradas durante um determinado número de gerações, seja por seleção clonal ou maturação fraca, o critério de parada é alcançado e o processo evolutivo da copt-aiNet é encerrado.

5. Testes computacionais

Para comparar o desempenho do algoritmo genético e da copt-aiNet, diversos testes foram realizados com um conjunto de instâncias comumente utilizado da literatura. A seguir são descritos os testes realizados e os resultados obtidos.

5.1. Material e Métodos

Antes de aplicar o AG a uma instância de PCV, faz-se necessária a definição de valores para alguns parâmetros. A Tabela 1 descreve estes parâmetros e apresenta os valores utilizados durante os experimentos. Tais valores foram obtidos empiricamente, sendo que alguns a partir de valores sugeridos em [Eiben e Smith 2003].

Tabela 1. Parâmetros e valores utilizados na execução do algoritmo genético.

Parâmetro	Valor	Parâmetro	Valor
Tamanho da população	120	Probabilidade de recombinação	0.80
Probabilidade de mutação	0.15	Número máximo de gerações	2000

Assim como o AG, o acerto de parâmetros precede a execução da copt-aiNet. A Tabela 2 descreve os parâmetros deste algoritmo e indica seus respectivos valores utilizados nos testes. Tais valores foram obtidos empiricamente, sendo que alguns a partir de valores sugeridos em [Souza et al. 2005].

Os algoritmos foram codificados em MATLAB e executados em um computador Intel Pentium Dual Core com 1,6 GHz e 1 GB de RAM. Foram utilizadas 26 instâncias do PCV simétrico da TSPLIB [Reinelt 1991] que variam em tamanho entre 51 e 198 cidades. Cada algoritmo foi executado 30 vezes para cada instância. A seguir, são apresentados os resultados em termos de tempo de execução e qualidade da solução obtida,

apresentada pela melhor, pior e solução média dentre as 30 execuções de cada algoritmo para cada instância.

Tabela 2. Parâmetros e valores utilizados na execução da copt-aiNet.

Parâmetro	Valor	Parâmetro	Valor
Tamanho inicial da população	25	Valor máximo da taxa de mutação de um clone	0,20
Tamanho máximo da população	60	Número de clones por anticorpo	10
Quantidade de melhores soluções que serão rastreadas	4	Número de gerações para checar a necessidade de supressão	5
Limiar de mudança na qualidade média das soluções da população para a ocorrência da supressão	0,001	Número de gerações sem aperfeiçoamento das K melhores soluções para a ocorrência da maturação fraca	15
Limiar de similaridade entre dois anticorpos	0,80		

5.2. Resultados

A Tabela 3 apresenta os resultados dos testes computacionais realizados com o algoritmo genético e com a copt-aiNet. Analisando os resultados obtidos com o AG, observa-se que, em média, a melhor solução que ele foi capaz de obter difere em 2,13% do ótimo. Avaliando a média das soluções obtidas, ele foi capaz de obter, no geral, uma solução que difere em 4,60% do ótimo. Já no pior caso, em média, a solução difere 7,63% do ótimo. Analisando os resultados obtidos com a copt-aiNet, observa-se que, em média, a melhor solução que ela foi capaz de obter difere em 0,98% do ótimo. Avaliando a média das soluções obtidas, ela foi capaz de obter, no geral, uma solução que difere em 2,47% do ótimo. Já no pior caso, em média, a solução difere 3,74% do ótimo.

Comparando os resultados obtidos pelos dois algoritmos, a copt-aiNet foi capaz de obter resultados de melhor qualidade para a maioria das instâncias, como pode ser observado pelos valores em destaque na Tabela 3. Analisando as melhores soluções obtidas pelos dois algoritmos, a copt-aiNet superou o AG em 22 instâncias. Avaliando as soluções médias, a copt-aiNet superou o AG em 25 instâncias. Já em relação ao pior caso, ou seja, para as piores soluções obtidas, a copt-aiNet obteve melhor desempenho para todas as instâncias.

O melhor desempenho da copt-aiNet em relação à qualidade da solução é contrastado com seu maior tempo de execução em comparação ao do AG. Isto é um exemplo do clássico *trade-off* entre qualidade de solução e tempo de execução [Michalewicz e Fogel 2000]. Caso a prioridade seja uma solução de melhor qualidade, a copt-aiNet é a melhor escolha entre os dois algoritmos estudados, abrindo mão de um menor tempo de resposta. Caso a prioridade seja uma resposta rápida, pode ser possível abrir mão de uma melhor qualidade da solução, escolhendo o AG.

6. Conclusões

Este relatório descreveu o estudo de duas ferramentas bio-inspiradas para a solução de um problema de otimização combinatória, o PCV. Apesar do fácil entendimento, o PCV é um problema comumente estudado por causa da dificuldade computacional em sua solução (o espaço de busca aumenta fatorialmente em relação ao número de cidades) e por causa do grande número de problemas práticos que podem ser modelados através dele (por exemplo, problemas de roteamento de veículos). Ambas as ferramentas estudadas são classificadas como algoritmos evolutivos, que têm como inspiração principal conceitos de biologia evolutiva. A primeira ferramenta, um algoritmo genético, compõe a classe de algoritmos evolutivos mais estudada. Já a copt-aiNet é uma ferramenta de

sistemas imunológicos artificiais com características que a permitem classificar como um algoritmo evolutivo. A descrição da proposta dos dois algoritmos para a solução do problema abordado foi apresentada.

Tabela 3. Resultados computacionais para o algoritmo genético e para a copt-aiNet. MSC é a melhor solução conhecida (ótimo) da instância; Tempo é a média do tempo de execução do algoritmo, em segundos; PDB, PDM e PDW são, respectivamente, os desvios percentuais da melhor solução, da solução média e da pior solução para a MSC. Os valores destacados são as melhores soluções entre os dois algoritmos.

Instância	MSC	Algoritmo genético				copt-aiNet			
		Tempo	PDB	PDM	PDW	Tempo	PDB	PDM	PDW
eil51	426	41,11	0,70	2,15	3,99	96,42	0,23	1,31	1,88
berlin52	7542	46,91	0,00	0,90	3,47	161,03	0,00	0,91	2,55
st70	675	71,24	1,48	4,35	8,30	178,11	0,30	1,77	2,81
eil76	538	76,05	1,49	4,63	8,74	150,89	1,30	2,77	4,46
pr76	108159	93,19	1,27	3,74	8,50	187,27	0,07	1,24	2,12
rat99	1211	120,22	3,63	5,30	7,93	155,39	0,91	3,29	5,37
kroA100	21282	119,58	0,05	1,72	5,38	217,80	0,41	1,43	2,52
kroB100	22141	121,70	0,47	4,21	8,10	167,86	0,17	1,59	3,00
kroC100	20749	127,93	2,72	4,83	6,85	157,05	0,61	2,07	3,63
kroD100	21294	129,84	2,01	8,05	12,16	224,32	1,42	3,15	4,69
kroE100	22068	122,55	0,58	2,61	5,17	200,54	1,11	2,41	3,45
rd100	7910	133,60	0,96	6,03	11,35	258,86	1,14	3,22	4,75
eil101	629	126,42	1,27	3,48	7,95	222,47	0,95	2,96	4,93
lin105	14379	133,93	0,67	3,37	7,45	303,67	0,53	1,49	2,55
pr107	44303	118,44	0,18	0,65	0,95	119,11	0,00	0,34	0,94
pr124	59030	167,53	0,82	3,11	6,09	143,25	0,08	0,98	1,84
bier127	118282	181,63	1,86	4,13	6,02	305,17	1,00	2,51	4,64
ch130	6110	194,39	3,88	6,52	10,88	234,53	0,77	3,54	4,44
pr136	96772	200,79	6,82	8,51	10,48	233,65	3,28	5,86	7,61
ch150	6528	210,40	0,83	2,67	3,72	197,50	0,70	2,21	3,09
kroA150	26524	226,25	5,58	8,03	10,67	293,21	2,36	3,72	4,82
kroB150	26130	223,68	4,56	7,01	11,47	352,87	1,95	3,31	4,50
pr152	73682	196,82	1,50	3,51	5,84	197,97	0,62	1,63	2,64
u159	42080	225,65	4,59	8,11	11,69	298,68	1,56	4,01	5,29
rat195	2323	269,95	4,48	6,88	8,22	261,48	2,84	4,24	5,64
d198	15780	287,50	3,07	4,97	7,08	406,90	1,24	2,18	2,97
Mínimo			*0,00*	*0,65*	*0,95*		*0,00*	*0,34*	*0,94*
Média			*2,13*	*4,60*	*7,63*		*0,98*	*2,47*	*3,74*
Máximo			*6,82*	*8,51*	*12,16*		*3,28*	*5,86*	*7,61*

Para avaliar o desempenho dos algoritmos estudados, diversos testes foram feitos com instâncias comumente usadas na literatura, das quais são conhecidas as melhores soluções, possibilitando uma avaliação e comparação mais precisa dos algoritmos. Os resultados demonstraram uma boa performance dos dois algoritmos, sendo que, em média, o AG obteve soluções que desviam 4,60% do ótimo e a copt-aiNet obteve soluções que desviam 2,47% do ótimo. Foi possível observar, através dos resultados, o *trade-off* clássico entre qualidade da solução e tempo de resposta, no qual a copt-aiNet foi

capaz de obter soluções de melhor qualidade que o AG, porém, com um esforço computacional maior.

Alguns assuntos podem ser investigados futuramente como, por exemplo, a análise da influência dos parâmetros e etapas do AG e da copt-aiNet na solução final; testes com instâncias maiores; comparação dos resultados obtidos com outros apresentados na literatura; e a avaliação do algoritmo para outras classes do PCV como, por exemplo, o PCV assimétrico, o PCV com janelas de tempo, o PCV com coleta de prêmios, entre outros.

Agradecimentos

Os autores agradecem à Fapesp, ao CNPq e ao MACKPESQUISA pelo apoio financeiro.

Referências

Davis, L. (1991). Handbook of genetic algorithm, Van Nostrand Reinhold Company.

de Castro, L.N. (2007). Fundamentals of natural computing: An overview, Physics of Life Reviews, 4, 1-36.

de Castro, L.N. e Timmis, J.I. (2002). Artificial immune systems: A new computational intelligence approach, Springer-Verlag, Heidelberg.

de Souza, J.S., Gomes, L.C.T., Bezerra, G.B, de Castro, L.N. e Von Zuben, F.J. (2004). An immune-evolutionary algorithm for multiple rearrangements of gene expression data, Genetic Programming and Evolvable Machines, 5, 157-179.

Eiben, A.E. e Smith, J.E. (2003). Introduction to evolutionary computing, Springer.

Garey, M.R. e Johnson, D.S. (1979). Computers and intractability: A guide to the theory of NP-completeness, W. H. Freeman & Co., New York.

Holland, J.H. (1992), Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control and artificial intelligence, MIT Press.

Johnson, D.S. e McGeoch, L.A. (1997). The traveling salesman problem: A case study in local optimization, In Local Search in Combinatorial Optimization, E.H.L. Aarts e J.K. Lenstra (eds.), John Wiley & Sons, 215-310.

Larrañaga, P., Kuijpers, C.M.H, Murga, R.H, Inza, I., e Dizdarevic, S. (1999). Genetic algorithms for the travelling salesman problem: A review of representations and operators, Artificial Intelligence Review, 13, 129-170.

Lin, S. e Kernighan, B.W. (1973). An effective heuristic algorithm for the traveling salesman problem, Operations Research, 21(2), 498-516.

Michalewicz, Z. e Fogel, D.B. (2000). How to solve it: Modern heuristics, Springer.

Reinelt, G. (1991), TSPLIB – A traveling salesman problem library, ORSA Journal on Computing, 3, 376-384.

Whitley, D. (2000), Permutations. In Evolutionary computation 1: Basic algorithms and operators, Bäck, T., Fogel, D.B. e Michalewickz, Eds., Institute of Physics Publishing, Bristol, 274-284.