Learning and Nonlinear Models - Revista da Sociedade Brasileira de Redes Neurais (SBRN), Vol. 5, No. 2, pp. 81-98, 2007

UMA REDE NEURO-IMUNE APLICADA AO PROBLEMA DE MÚLTIPLOS CAIXEIROS VIAJANTES

Thiago Augusto Soares Masutti[1]
Leandro Nunes de Castro[2]

[1]Laboratório de Sistemas Inteligentes (LSIn)
Universidade Católica de Santos (UNISANTOS)
R. Dr. Carvalho de Mendonça, 144 – Vila Mathias
Santos/SP, Brasil, 11070-906
thiago.masutti@gmail.com.br

[2]Pós-Graduação em Engenharia Elétrica (PGEEL)
Universidade Presbiteriana Mackenzie
R. da Consolação, 896
São Paulo/SP, Brasil, 01302-907
lnunes@mackenzie.br

Resumo – Redes Neurais Artificiais têm sido alvo de estudo para a solução de problemas de otimização combinatória há mais de duas décadas. Uma das classes estudadas é aquela baseada em redes auto-organizáveis. Este trabalho apresenta um algoritmo híbrido entre uma rede neural auto-organizável e sistemas imunológicos artificiais para a solução do problema de múltiplos caixeiros viajantes (MTSP). O algoritmo investigado, nomeado de RABNET-MTSP, possui uma arquitetura construtiva e utiliza *m* sub-redes para representar cada um dos caixeiros envolvidos na solução. Diversos testes com instâncias comumente utilizadas na literatura foram realizados para avaliar a performance do algoritmo. O algoritmo proposto em Somhom et al. (1999) foi implementado e submetido a testes com as mesmas instâncias. Um comparativo da performance destes dois algoritmos, demonstrou que a RABNET-MTSP é capaz de obter bons resultados, porém, com um tempo de execução maior ao do algoritmo de Somhom.

Palavras-Chaves: Redes Neurais Artificiais, Redes Auto-Organizáveis, Sistemas Imunológicos Artificiais, Problema de Múltiplos Caixeiros Viajantes.

1 Introdução

O Problema de Múltiplos Caixeiros Viajantes (MTSP, do inglês *Multiple Traveling Salesmen Problem*) é uma generalização do Problema do Caixeiro Viajante, no qual é possível a utilização de mais de um caixeiro para se obter uma solução. O MTSP consiste em determinar uma rota para um conjunto de *m* caixeiros, que devem partir e voltar a um depósito específico, de tal maneira que todas as cidades intermediárias sejam visitadas apenas uma vez e por apenas um caixeiro e que o custo total destas rotas seja minimizado, sendo que o custo pode ser, por exemplo, a distância percorrida, o tempo empregado, ou outra grandeza que se queira minimizar. O MTSP é um problema que pode ser facilmente relacionado a aplicações de mundo real como seqüenciamento de tarefas, otimização de operações logísticas, otimização de produção e muitas outras [1],[16],[30],[33]. Dentre os critérios de minimização podemos destacar dois: 1) o *minsum*, que busca minimizar a soma do custo da rota de cada caixeiro; e 2) o *minmax*, que busca minimizar o maior custo entre as rotas, podendo levar à geração de rotas com custos eqüitativos.

Os Mapas Auto-Organizáveis (SOM, do inglês *Self-Organizing Maps*) de Kohonen [21] foram tema de diversos trabalhos para a solução do TSP ([2],[5],[7],[13],[23],[25],[28]) e os resultados destes trabalhos são muito satisfatórios. Porém, pouca atenção tem sido dada à aplicação de redes neurais auto-organizáveis para a solução do MTSP [29]. O objetivo deste trabalho é modificar a ferramenta proposta por Pasti & de Castro [25], denominada RABNET-TSP, para que ela seja capaz de resolver o MTSP *minmax*. A descrição do algoritmo proposto, um híbrido entre sistemas imunológicos artificiais e mapas auto-organizáveis, é dada e sua análise é feita. Diversos testes são executados para avaliar sua performance e, como comparativo, um algoritmo semelhante foi implementado e seus resultados comparados ao da ferramenta proposta.

Este artigo está organizado da seguinte maneira. A Seção 2 apresenta uma descrição de trabalhos relacionados à aplicação de redes auto-organizáveis à solução do MTSP. Na Seção 3 o algoritmo proposto, nomeado de RABNET-MTSP, é descrito. Na Seção 4 é feita uma análise dos parâmetros da RABNET-MTSP. Na Seção 5 os resultados computacionais, sua comparação com os resultados de outro algoritmo e a análise da complexidade da RABNET-MTSP são apresentados. Na Seção 6 o artigo é concluído com discussões sobre o trabalho apresentado e investigações futuras.

2 Trabalhos Relacionados

Nesta seção é feita uma revisão de trabalhos da literatura que relacionam mapas auto-organizáveis à solução do MTSP. Para uma revisão sobre o MTSP, suas aplicações práticas e outras abordagens de solução, indica-se ao leitor o trabalho de Bektas [4].

No trabalho de Hsu et al. [19] os autores apresentam uma extensão do algoritmo proposto por Angeniol *et al.* [2] para a solução do MTSP. No algoritmo apresentado, *m* 'bolhas' crescem a partir da cidade base na direção das cidades intermediárias definindo as rotas que cada um dos *m* caixeiros deverá seguir. Assim como o algoritmo de Angeniol, este algoritmo possui funções de criação e exclusão de nós. Um novo neurônio é inserido caso ele seja vencedor para mais de uma cidade e um neurônio é excluído da rede quando não for declarado vencedor durante três épocas consecutivas. Neste algoritmo a ordem de apresentação das cidades é escolhida aleatoriamente antes do início do processo iterativo. Uma restrição que parece não ser utilizada, porém existente no algoritmo de Angeniol, é a inibição de neurônios, através da qual neurônios que participaram do processo de duplicação são inibidos na época seguinte. Utilizando uma instância de 20 cidades, os autores compararam a performance de seu algoritmo com a de [32] e, de acordo com os resultados, o algoritmo proposto obteve melhor desempenho.

A rede elástica (do inglês, *Elastic Net*) foi originalmente proposta por Durbin & Willshaw [11] para a solução do TSP. Neste algoritmo, duas forças controlam a rede fazendo-a se alongar de uma forma parecida a de um elástico, na direção das cidades. Mais tarde, Vakhutinsky & Golden [31] propuseram uma extensão da rede elástica para a solução do MTSP. Nesse algoritmo *m* tiras elásticas se alongam a partir do depósito na direção das cidades intermediárias de acordo com duas forças: a primeira induz todos os nós da rede a se moverem na direção da cidade mais próxima e a segunda força faz com que cada nó se mova na direção de seus vizinhos. Este processo ocorre iterativamente até que cada cidade esteja suficientemente próxima de um nó da rede. De acordo com os autores, este algoritmo apresentou soluções aproximadamente 10% melhores que o algoritmo proposto por Ghaziri [15], mas com tempo de processamento maior.

Partindo de um trabalho para o TSP [28], Somhom et al. [29] propuseram uma rede baseada em SOM para a solução do MTSP com critério *minmax*. Para procurar minimizar a maior rota entre os caixeiros, a regra de competição considera o custo das rotas de cada caixeiro, punindo os caixeiros com rotas nas quais o custo supera a média. Para testar o desempenho do algoritmo, os autores realizaram testes com instâncias para o CVRP da TSPLIB [27] e compararam os resultados com a *Elastic Net* [31] e com o algoritmo 2-opt [26]. De acordo com os resultados apresentados o algoritmo proposto pelos autores obteve soluções de melhor qualidade que os demais. Em relação ao custo computacional o tempo de processamento do algoritmo proposto foi inferior ao da *Elastic Net*, mas superior ao do 2-opt.

No trabalho de Zhu & Yang [34], os autores utilizam uma versão modificada do algoritmo de Modares et al. [24] como o núcleo de seu algoritmo para o MTSP. No algoritmo proposto, cada rede contém *N* neurônios e cada neurônio é habilitado a vencer apenas uma vez durante uma época. Através da competição, restrições são impostas aos neurônios para propiciar a formação de rotas com custos eqüitativos. A performance do algoritmo proposto é avaliada para o objetivo *minsum* com a comparação dos resultados obtidos com aqueles apresentados pela implementação dos autores de [24]. De acordo com os resultados obtidos, o algoritmo proposto obteve melhor performance em relação à qualidade da solução e ao tempo de processamento.

3 A Ferramenta Proposta

O algoritmo proposto para a solução do MTSP é a RABNET-MTSP (*Real-Valued AntiBody NETwork to solve the Multiple Traveling Salesmen Problem*) que utiliza conceitos de redes auto-organizáveis e sistemas imunológicos artificiais. Este algoritmo é uma extensão da RABNET-TSP [23], [25].

As principais características da RABNET-MTSP são listadas a seguir.

- Processo competitivo e adaptativo inspirados em mecanismos elementares de processamento de informação existentes nos sistemas imunológicos biológicos.
- Treinamento de diversas sub-redes independentes, cada uma representando um caixeiro.
- Arquitetura construtiva com mecanismos de poda e clonagem de anticorpos (neurônios).
- Vizinhança topológica entre anticorpos da mesma sub-rede.

O objetivo da RABNET-MTSP é posicionar um anticorpo presente em uma de suas sub-redes próximo o suficiente de cada cidade da instância de MTSP a ser solucionada. Desta forma, a vizinhança topológica entre os anticorpos denotará o trajeto a ser feito pelos caixeiros.

Visualmente, a RABNET-MTSP apresenta *m* anéis circulares que se alongam iterativamente e simultaneamente, a partir do depósito, em direção às cidades intermediárias, o que é ilustrado na Figura 1.

Para facilitar a descrição do algoritmo proposto, este pode ser dividido nas seguintes etapas: 1) Inicialização das Sub-Redes; 2) Apresentação dos Antígenos (cidades do mapa); 3) Competição; 4) Cooperação; 5) Adaptação; 6) Clonagem de

Anticorpos (neurônios da rede); 7) Poda de Anticorpos; 8) Estabilização de Vencedores; e 9) Critérios de Convergência. Cada uma destas etapas é descrita a seguir.

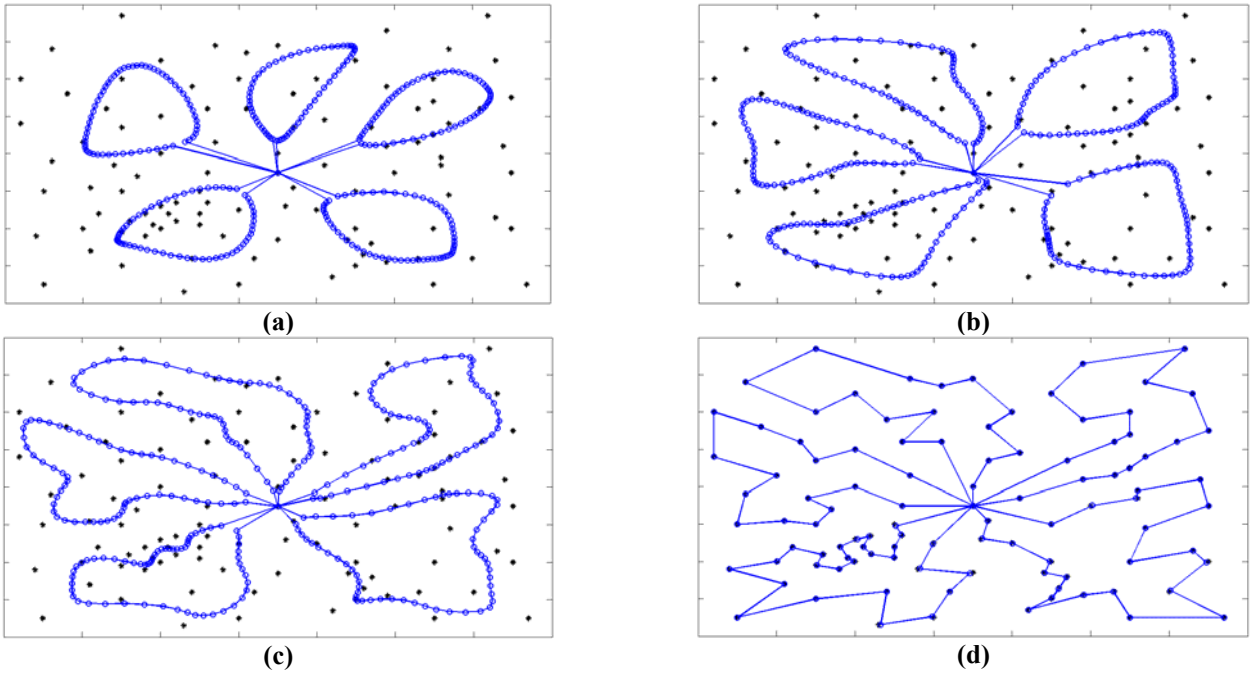


Figura 1: Evolução das sub-redes da RABNET-MTSP na proposição de uma solução para uma instância de 101 cidades e 5 caixeiros.

3.1 Inicialização das Sub-Redes

Como cada sub-rede representará um caixeiro, são inicializadas *m* sub-redes, cada uma com um número de anticorpos igual a round(2**N*/*m*), na qual *N* é o número de cidades, *m* é o número de caixeiros e round(·) é a função que arredonda um valor para o seu inteiro mais próximo. O vetor descrevendo as características do anticorpo de índice um de cada sub-rede é inicializado com valor igual ao vetor de coordenadas do depósito; os demais anticorpos são inicializados aleatoriamente no plano Euclidiano.

3.2 Apresentação dos Antígenos

Durante a evolução do sistema imunológico, um organismo pode encontrar um antígeno diversas vezes [10]. Como o problema a ser solucionado é o MTSP, cada antígeno representa uma cidade intermediária e o depósito, sendo que eles são apresentados iterativamente representando o encontro entre o organismo e o antígeno. Numericamente, o que descreve um antígeno é o vetor de coordenadas da cidade que ele representa. A apresentação de todos os antígenos é denominada como época. Para que a ordem de apresentação destes antígenos não influencie o processo de aprendizado do sistema, ela é alterada ao começo de cada época.

3.3 Competição

No sistema imunológico, o anticorpo é a substância responsável por ligar-se a um antígeno e, posteriormente, levar a eliminação do complexo antígeno-anticorpo do organismo. No entanto, os anticorpos são substâncias especialistas, sendo capaz de reconhecer, ligar-se e eliminar apenas um limitado grupo de antígenos. O que determina qual anticorpo reconhece certo antígeno (ou grupo de antígenos) é o grau de afinidade entre eles [10].

A etapa de Competição é responsável por determinar o anticorpo com maior afinidade ao antígeno apresentado de acordo com equação abaixo:

$$I, J = \arg\min[(\|\mathbf{Ab}_i^j - \mathbf{ag}\|)*bias^j];\ i = 1, ..., M^j;\ j = 1, \ldots, m; \qquad (1)$$

na qual *I* é o índice do anticorpo vencedor pertencente à sub-rede *J*, **Ab** é o vetor de atributos do anticorpo *i* pertencente à sub-rede *j*, **ag** é o vetor de atributos do antígeno apresentado, *bias* é um valor ponderador da sub-rede *j*, *M* é o número de anticorpos da sub-rede *j* e *m* é número de caixeiros.

É possível observar através da Equação (1) que, na RABNET-MTSP, a afinidade antígeno-anticorpo depende de dois fatores: 1) a distância Euclidiana entre eles; e 2) um fator ponderador que leva em conta uma proposição do custo da rota sendo formada, o qual foi inspirado na proposta de Somhom et al. [29] e é determinado pela seguinte equação:

$$bias^j = 1 + [(dist^j - avdist)/avdist], \quad (2)$$

na qual *bias* é o fator ponderador da sub-rede *j*, *dist* é o custo proposto para a rota sendo formada por esta sub-rede e *avdist* é o custo médio entre todas as sub-redes. Como mencionado anteriormente, o custo da rota sendo formada por uma sub-rede é uma proposição, a qual é feita através das informações contidas nos atributos dos anticorpos:

$$dist^j = \left(\sum_{i=1}^{M^j-1} \left\| \mathbf{Ab}_i^j - \mathbf{Ab}_{i+1}^j \right\| \right) + \left\| \mathbf{Ab}_{M^j}^j - \mathbf{Ab}_1^j \right\|. \quad (3)$$

Este valor ponderador da Equação (1) trata, de acordo com Somhom et al. [29], a requisição do objetivo *minmax*, propiciando a formação de rotas equivalentes em relação ao custo. Através de testes preliminares, a adição deste valor ponderador na RABNET-MTSP auxiliou na obtenção de boas soluções para o MTSP *minmax*.

O algoritmo proposto não impõe restrição alguma ao número de vezes que um anticorpo pode ser declarado vencedor a um antígeno apresentado. Desta forma, um anticorpo pode estar relacionado a zero, um ou mais antígenos. Esta informação, a quantidade de antígenos relacionados a um anticorpo, é nomeada aqui de (nível de) concentração, a qual é armazenada como um vetor **ρ** para cada sub-rede e é reinicializado ao início de cada época.

Caso o antígeno apresentado seja aquele que representa o depósito não há competição e os anticorpos de índice um de cada sub-rede serão utilizados nas etapas seguintes de Cooperação e Adaptação.

3.4 Cooperação

Na RABNET-MTSP, os anticorpos de cada sub-rede são interligados topologicamente por uma vizinhança. Com isto, um estímulo causado a um anticorpo é propagado, com intensidades diferentes, aos seus vizinhos. É importante destacar que as sub-redes são independentes umas das outras, ou seja, o estímulo a um anticorpo não é propagado a outros de uma sub-rede diferente. A etapa de cooperação consiste em determinar a intensidade do estímulo aos anticorpos da mesma sub-rede do anticorpo vencedor. Isto é feito de forma similar àquela utilizada em mapas auto-organizáveis de Kohonen [21] através da equação abaixo.

$$h_i = \begin{cases} \exp\left(-d_{iI}^{\,2} / 2\sigma(t)^2\right) & i \neq 1 \quad i = 1, ..., M_j, \\ 0 & \text{caso contrário} \end{cases} \quad (4)$$

na qual h_i é a intensidade do estímulo ao anticorpo i, $\sigma(t)$ é o parâmetro que controla a influência da vizinhança, t é a época atual e d_{iI} é a distância topológica entre o anticorpo i e o anticorpo vencedor determinada pela seguinte equação.

$$d_{iI} = \min([|i - I|, M_j - |i - I|]). \quad (5)$$

Para manter o anticorpo de índice um sempre relacionado ao depósito, a intensidade do estímulo causado a este anticorpo é nula.

A influência da vizinhança deve ser alta no início do processo de aprendizado e diminuir de intensidade ao decorrer deste processo [21]. Esta diminuição da influência da vizinhança é realizada através da equação abaixo.

$$\sigma(t) = \sigma(0)*\exp(-t/\tau_1), \quad (6)$$

na qual $\sigma(0)$ é o valor inicial da influência da vizinhança e τ_1 é a taxa de redução deste parâmetro.

3.5 Adaptação

Para apresentar uma resposta efetiva a agentes patogênicos o processo de aprendizado do sistema imunológico envolve o aumento da afinidade anticorpo-antígeno a cada encontro do organismo com este antígeno [10]. Na RABNET-MTSP, este o aumento da afinidade **Ab-ag** é realizado apenas para a sub-rede a qual pertence o anticorpo vencedor através da seguinte equação.

$$\mathbf{Ab}_i^J(t+1) = \begin{cases} \mathbf{Ab}_i^J(t) + \alpha(t) * h_{iI} * \left[\mathbf{ag} - \mathbf{Ab}_i^J(t)\right] & h_{iI} > \kappa \\ \mathbf{Ab}_i^J(t) & \text{caso contrário} \end{cases}, \quad (7)$$

na qual κ é o menor valor de h_{iI} para ocorrer a adaptação do anticorpo i e $\alpha(t)$ é a taxa de aprendizado definida a cada época pela equação abaixo.

$$\alpha(t) = \alpha(0)*\exp(-t/\tau_2), \quad (8)$$

na qual $\alpha(0)$ é o valor inicial da taxa de aprendizado e τ_2 é a taxa de redução deste parâmetro.

3.6 Clonagem de Anticorpos

O processo de clonagem de anticorpos é baseado no Princípio da Seleção Clonal do sistema imunológico [8]. Basicamente, o anticorpo mais estimulado do sistema imunológico é selecionado para a clonagem. Na RABNET-MTSP, o grau de estimulação de um anticorpo é definido pela concentração de antígenos reconhecidos por ele. Desta forma, a possibilidade de clonagem de anticorpos é realizada da seguinte forma:

- O anticorpo mais estimulado de cada sub-rede é selecionado. Caso haja, na mesma sub-rede, mais de um anticorpo com o mesmo grau de estimulação, um deles é selecionado aleatoriamente.
- Dentre os anticorpos selecionados, caso sua concentração seja maior que um, ou seja, ele mapeia mais de um antígeno, este anticorpo é clonado. Caso contrário, não ocorrerá mudança alguma na sub-rede a qual ele pertence.
- Para não sobrecarregar o algoritmo com um número excessivo de anticorpos, um anticorpo das sub-redes que apresentarão clonagem deverá ser excluído. O anticorpo selecionado para a exclusão deverá apresentar concentração igual a zero e fazer parte da região topológica com maior número de anticorpos que não mapeiam antígeno algum. Caso não exista um anticorpo com concentração igual a zero não ocorrerá a exclusão. No exemplo ilustrado na Figura 2, um dos anticorpos destacados deverá ser excluído para ocorrer a clonagem de outro anticorpo.
- Os vetores de atributos dos novos anticorpos são os mesmos que os de seus respectivos anticorpos pais e serão introduzidos em suas respectivas sub-redes como vizinhos de grau um deles.

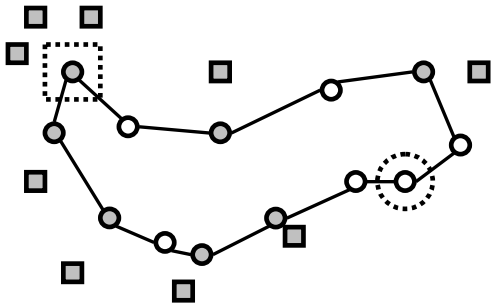


Figura 2: Exemplo de um caso de clonagem: os quadrados cheios são antígenos, os círculos cheios são anticorpos que mapeiam algum antígeno e os círculos vazios são anticorpos que não mapeiam antígeno algum. O anticorpo destacado pelo quadrado pontilhado foi selecionado para a clonagem, enquanto aquele destacado pelo círculo pontilhado foi selecionado para exclusão.

3.7 Estabilização de Vencedores

Para reduzir o tempo de processamento do algoritmo, a etapa de estabilização dos vencedores elimina a etapa de cooperação assim que nenhuma variação Δv de vencedores é detectada. Isto ocorre quando os anticorpos vencedores para todos os antígenos durante a época atual são os mesmos da época anterior:

$$\Delta v = \sum_{n=1}^{N} \left| I_n(t-1) - I_n(t) + J_n(t-1) - J_n(t) \right|, \tag{9}$$

na qual $I_n(t-1)$ e $I_n(t)$ é o índice do anticorpo vencedor para o antígeno de índice n na época anterior e na atual, respectivamente; e $J_n(t-1)$ e $J_n(t)$ é a sub-rede a qual pertence o anticorpo vencedor para o antígeno de índice n na época anterior e na atual, respectivamente.

A estabilização de vencedores é então detectada quando $\Delta v = 0$. A partir disto, não é realizada a etapa de Cooperação e a Adaptação ocorre apenas para o anticorpo vencedor, propiciando uma convergência mais rápida do algoritmo sem perda na qualidade da solução.

3.8 Critérios de Convergência

Dois critérios determinam a convergência do algoritmo a uma solução: 1) Cada anticorpo deve estar relacionado a no máximo um antígeno; e 2) Cada antígeno deve ter um anticorpo relacionado a ele a uma distância mínima λ.

3.9 Poda de Anticorpos

Ao final do processo de aprendizagem todos os anticorpos com nível de concentração igual a zero são excluídos da rede e toda a estrutura topológica das sub-redes é refeita. Isto faz com que o número total de anticorpos seja igual a $M + m - 1$.

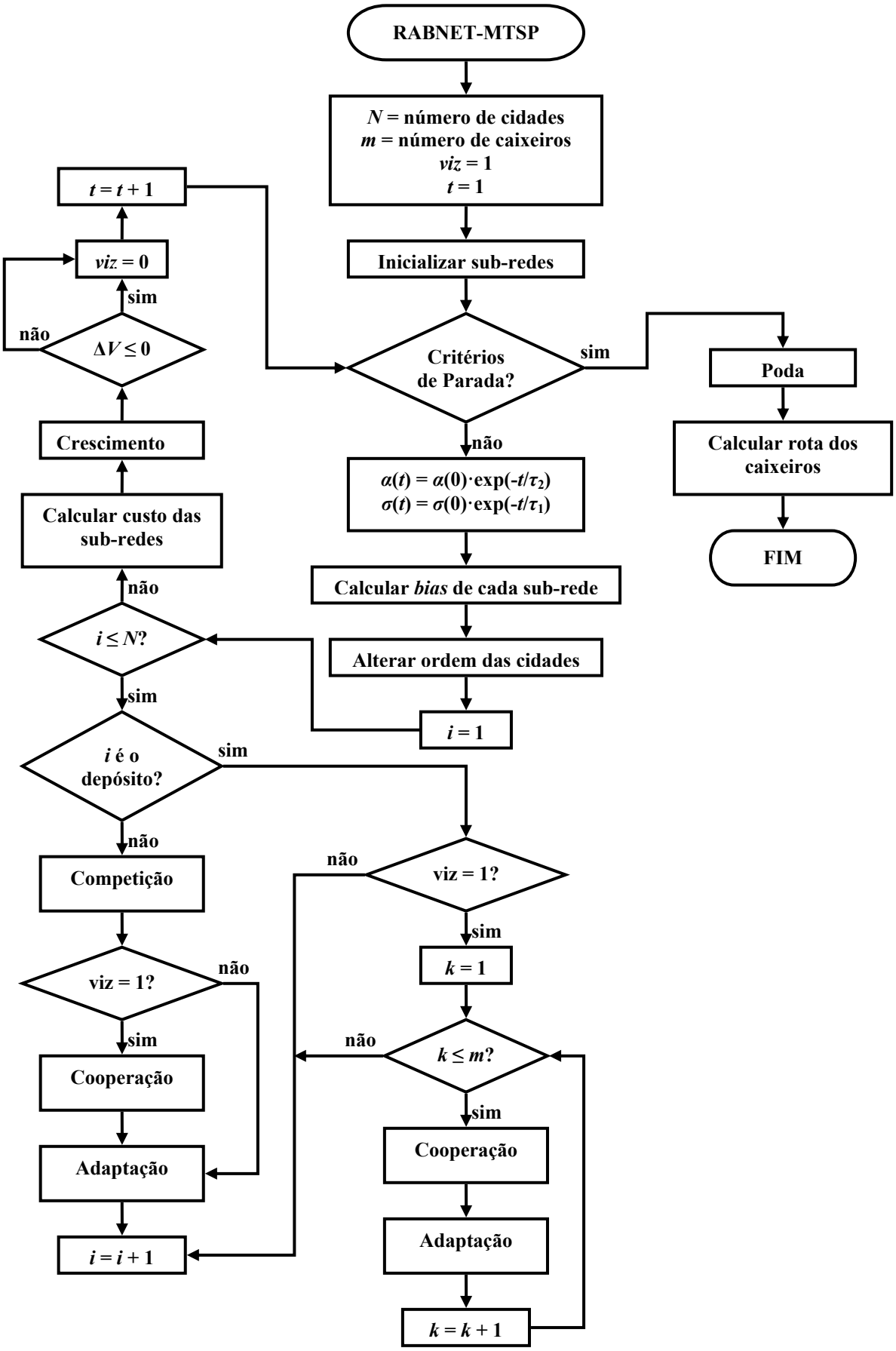


Figura 3: Pseudocódigo para a RABNET-MTSP.

4 Análise de Sensibilidade Paramétrica

Antes de ser aplicada a uma instância de MTSP, alguns parâmetros do algoritmo proposto precisam ser ajustados, sendo eles, λ, κ, $\alpha(0)$ e $\sigma(0)$. Dentre estes, os valores iniciais da taxa de aprendizado e da influência da vizinhança, $\alpha(0)$ e $\sigma(0)$, respectivamente, são aqueles que parecem interferir mais no processo de aprendizado das sub-redes. A seguir, é apresentada uma análise em relação aos parâmetros acima listados, com maior ênfase a $\alpha(0)$ e $\sigma(0)$, bem como a proposição de seus valores padrões.

Para parâmetros para os quais foram necessárias a realização de testes de sensibilidade do algoritmo, foi utilizada a instância eil101 da TSPLIB [27]. Tais análises foram feitas através de execuções do algoritmo com diversos valores destes parâmetros, sendo que, para cada valor testado o algoritmo foi executado 30 vezes e os outros parâmetros foram mantidos inalterados em seus valores padrões.

4.1 Distância mínima entre antígeno e anticorpo (λ)

Para garantir a formação de uma rota, cada antígeno deve ter seu anticorpo relacionado a uma distância mínima λ. Esta distância mínima deve ser inferior à menor distância entre as cidades da instância, *md*, o que é determinado pela equação abaixo:

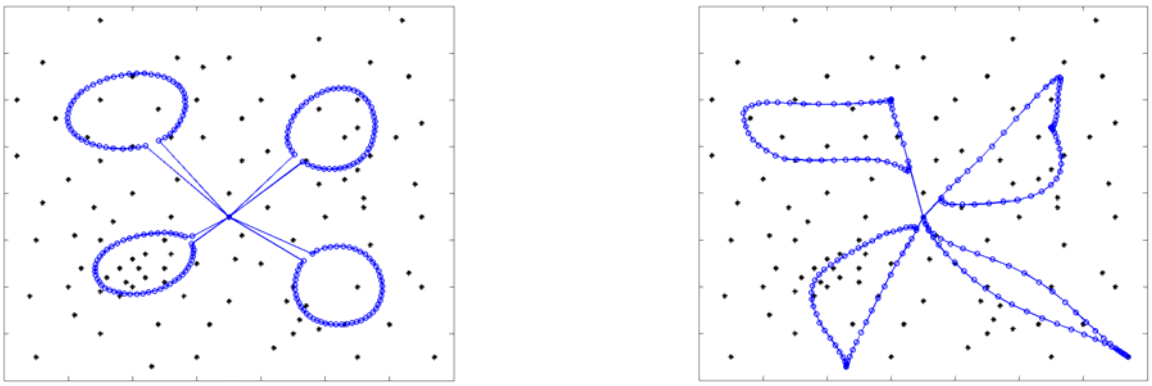
$$\lambda = 0{,}45 * md. \tag{10}$$

4.2 Limiar para atualização de anticorpos (κ)

O parâmetro κ controla o valor de h (Equação (4)) para que um anticorpo só seja atualizado caso o passo que ele dará seja significativo. Essa restrição não produz efeito expressivo na qualidade da solução final, mas economiza tempo de processamento do algoritmo. O valor padrão proposto para este parâmetro é $\kappa = 0.01$, ou seja, um anticorpo só será atualizado caso seu passo seja maior que 1% do passo dado pelo anticorpo vencedor.

4.3 Valor inicial da taxa de aprendizado $\alpha(0)$

A análise em relação a $\alpha(0)$ demonstrou que valores menores, por exemplo, $\alpha(0) = 0.2$, propiciam um crescimento mais ordenado da rede, como ilustrado na Figura 4(a). Para valores mais elevados, por exemplo, $\alpha(0) = 1.5$, a rede cresce de forma aleatória, como ilustrado na Figura 4(b).



(a) **(b)**

Figura 4: Situação das sub-redes após 200 épocas para (a) $\alpha(0) = 0{,}2$ e (b) $\alpha(0) = 1{,}5$.

A Figura 5 apresenta os resultados obtidos através da análise de sensibilidade em relação a $\alpha(0)$ para o tempo de execução, o número de épocas e a qualidade da solução. É possível observar que com o aumento do valor de $\alpha(0)$ é necessário um maior número de épocas para a convergência. Isto tem influência direta no tempo de execução do algoritmo, demonstrado pelas semelhantes tendências de curva apresentadas na Figura 5(a) e na Figura 5(b). Em relação à qualidade da solução, é possível observar que este parâmetro tem certa influência na obtenção de boas soluções, tanto pontuais (melhor, Figura 5(c)), quanto constantes (média, Figura 5(d)). No entanto, estes valores não seguem uma tendência ao variar as instâncias a serem resolvidas, mas foi possível observar a importância do correto ajuste deste parâmetro para a obtenção de soluções com boa qualidade. Em relação ao desvio padrão da qualidade das soluções obtidas, há resultados semelhantes ao variar as instâncias. O que ocorre, e pode ser observado na Figura 5(e), é que valores menores de $\alpha(0)$ apresentam menor desvio padrão e valores mais altos apresentam um desvio padrão mais elevado. Isto implica que valores elevados de $\alpha(0)$ propiciam uma diversidade maior de soluções, o que pode ser um facilitador na obtenção do ótimo global da instância.

A análise realizada indica que, para cada instância deve existir um valor ideal para $\alpha(0)$ e sua determinação só é possível empiricamente. Isto se torna inviável para algumas aplicações práticas e, portanto, é sugerido o valor padrão $\alpha(0) = 1.0$, que, empiricamente, propiciou, em geral, um bom desempenho do algoritmo.

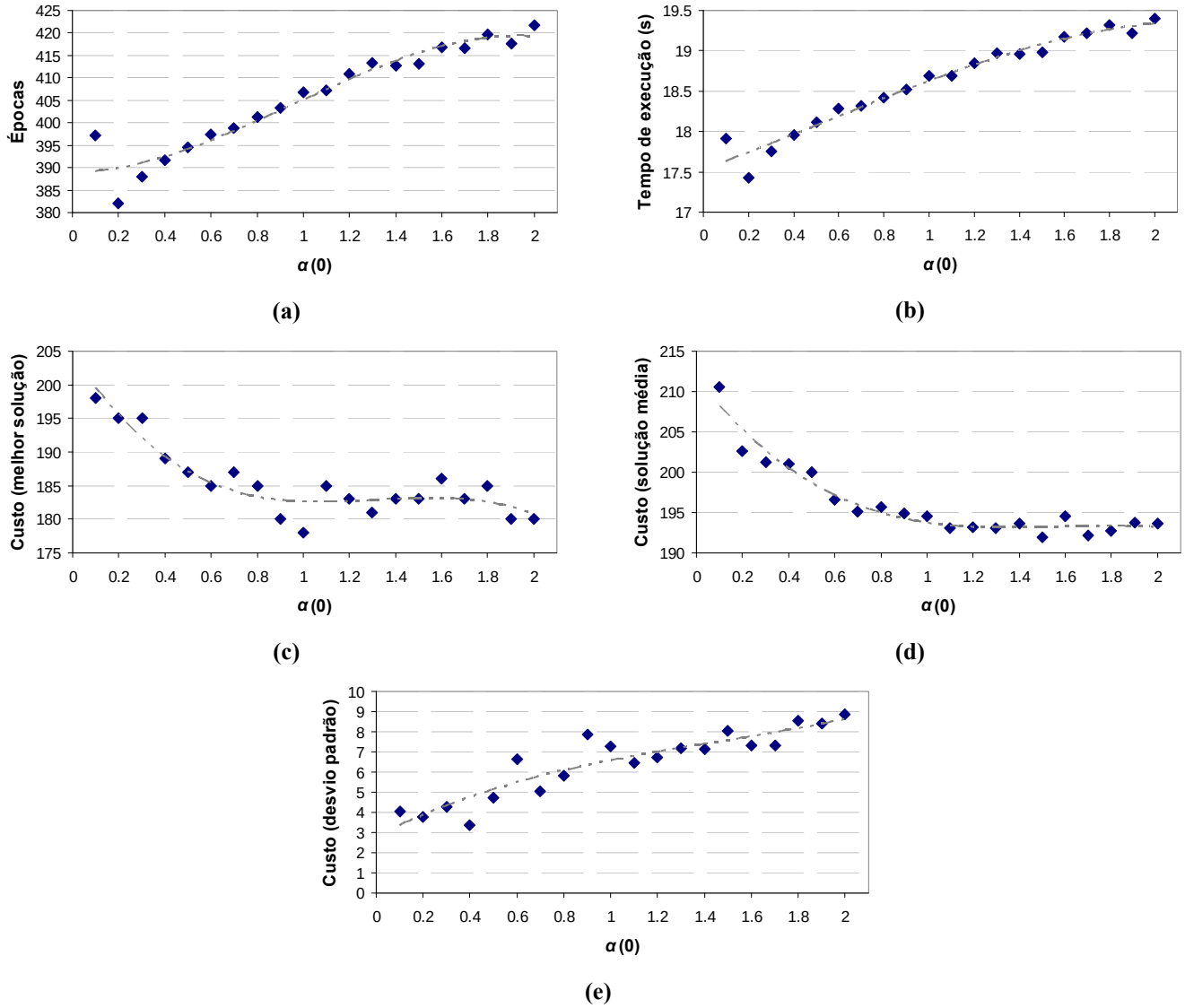
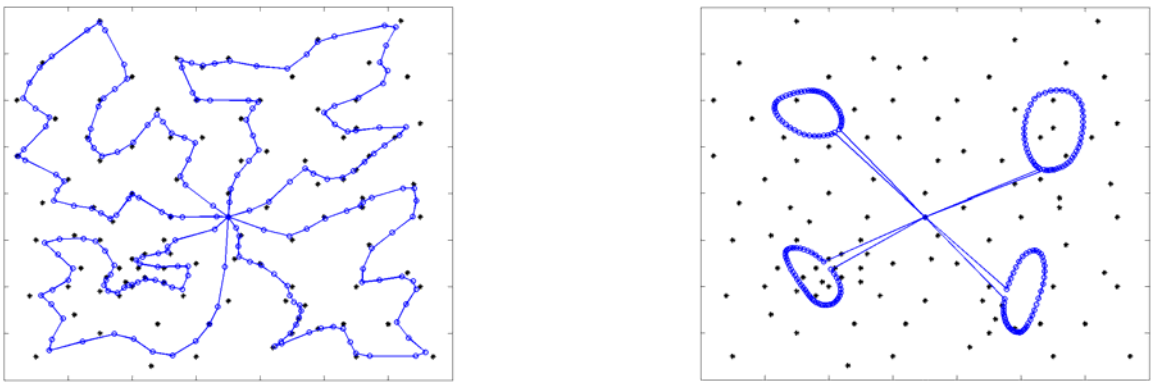


Figura 5: Resultados da análise paramétrica da RABNET-MTSP em função do valor inicial da taxa de aprendizado $\alpha(0)$ para (a) número de épocas para convergência; (b) tempo de processamento; (c) custo da melhor solução; (d) custo da solução média; e (e) desvio padrão do custo das soluções. Pontos na curva: valores obtidos nos experimentos; Curva tracejada: tendências utilizando funções de aproximação.

4.4 Valor inicial do parâmetro que controla a influência da vizinhança $\sigma(0)$

A análise em relação a $\sigma(0)$ demonstrou que valores menores, por exemplo, $\sigma(0) = 10$, permitem uma expansão mais rápida das sub-redes, como é ilustrado na Figura 6(a). Valores elevados deste parâmetro, por exemplo, $\sigma(0) = 100$, fazem com que as sub-redes mantenham-se retraídas por um número maior de épocas, como é ilustrado na Figura 6(b).



(a) (b)

Figura 6: Situação das sub-redes após 200 épocas para (a) $\sigma(0) = 10$ e (b) $\sigma(0) = 100$.

Este comportamento acima descrito faz com que o valor de $\sigma(0)$ influencie diretamente no número de épocas, no tempo de execução e na qualidade da solução, o que é ilustrado na Figura 7.

Através da Figura 7(a), é possível observar que para valores menores de $\sigma(0)$ são necessários um menor número de épocas e, conseqüentemente, menor tempo de execução. Já para valores elevados de $\sigma(0)$ é necessário um maior número de épocas, demandando um maior tempo de execução do algoritmo. Apesar do valor de $\sigma(0)$ influenciar na qualidade da solução, como é ilustrado na Figura 7(c), (d) e (e), esta influência apresenta diferente tendências para diferentes instâncias.

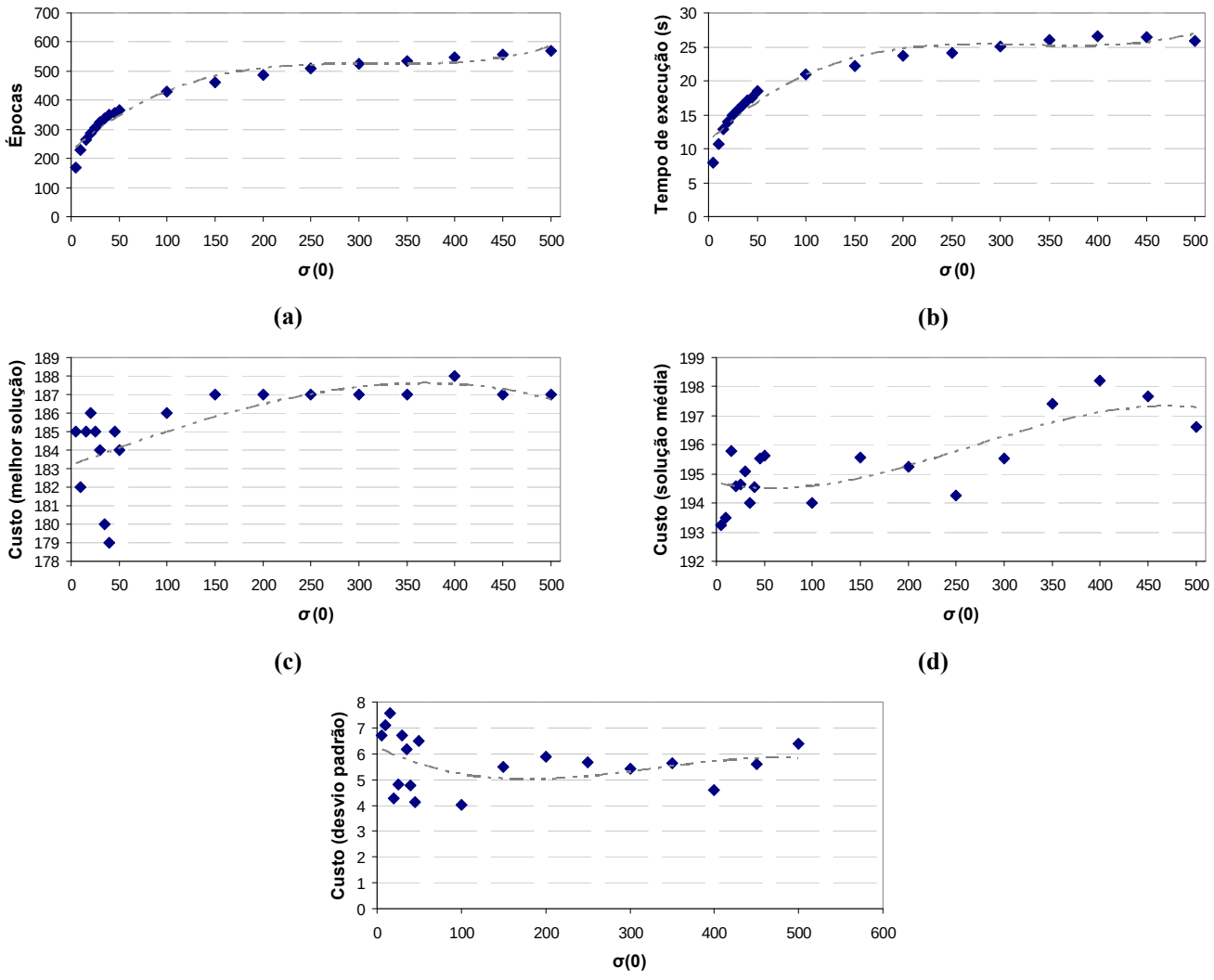


Figura 7: Resultados da análise paramétrica da RABNET-MTSP em função do valor inicial da influência da vizinhança $\sigma(0)$ para (a) número de épocas para convergência; (b) tempo de processamento; (c) custo da melhor solução; (d) custo

da solução média; e (e) desvio padrão do custo das soluções. Pontos na curva: valores obtidos nos experimentos; Curva tracejada: tendências utilizando funções de aproximação.

Assim como o valor inicial da taxa de aprendizado, cada instância pode apresentar um valor ideal de $\sigma(0)$, o qual pode ser determinado empiricamente. No entanto, caso esta determinação seja inviável, é proposto o valor padrão $\sigma(0) = 50$.

Como resumo, a Tabela 1 apresenta os parâmetros da RABNET-MTSP e seus valores padrões propostos.

Tabela 1: Parâmetros da RABNET-MTSP e seus valores padrões propostos.

Parâmetro	Símbolo	Valor Padrão
Distância mínima entre antígeno e anticorpo	λ	0.45**md*
Limiar para atualização de anticorpos	κ	0.01
Valor inicial da taxa de aprendizado	$\alpha(0)$	1.0
Valor inicial do parâmetro que controla a influência da vizinhança	$\sigma(0)$	50

Tabela 2: Resultados computacionais para o problema de múltiplos caixeiros viajantes com objetivo *minmax* obtidos com a RABNET-MTSP e a implementação própria do algoritmo proposto em Somhom et al. [29], nomeado aqui de SME. *m* é o número de caixeiros; 'tempo' é o tempo médio em segundos de uma execução; 'melhor' é o custo da melhor solução encontrada; 'média' é o custo médio das soluções obtidas; e dp é o desvio padrão do custo das soluções obtidas. Todos os resultados foram obtidos após 30 execuções dos algoritmos para cada instância e quantidade de caixeiros.

Instância	*m*	RABNET-MTSP				SME [29]			
		tempo	melhor	média	dp	tempo	melhor	média	dp
eil22	2	2,36	159	160,77	1,41	0,55	159	162,37	4,51
	3	2,75	117	**119,90**	0,55	0,66	117	121,13	4,82
	4	3,05	112	**113,40**	2,90	0,75	**110**	114,57	3,94
eil30	2	3,42	231	233,83	0,91	0,83	**230**	**233,60**	1,43
	3	3,97	**174**	**183,80**	1,92	0,97	177	193,30	18,01
	4	4,46	**160**	**169,83**	7,00	1,10	170	181,77	5,26
F-n45-k4	2	17,40	381	399,20	8,46	1,56	**380**	**392,37**	7,92
	3	31,39	265	**300,27**	42,82	1,77	**264**	321,50	38,31
	4	23,49	261	**261,00**	0,00	1,99	261	290,83	33,41
att48	2	5,86	18492	**19336,00**	422,93	1,49	**17829**	19426,00	1045,20
	3	6,44	**13461**	**13980,00**	422,39	1,69	13564	14206,00	351,59
	4	7,33	11448	11951,00	456,21	1,93	**11308**	**11947,00**	533,63
eil51	2	6,06	**224**	235,67	7,35	1,58	225	**235,47**	8,37
	3	6,70	164	**172,93**	6,75	1,80	**162**	174,27	6,84
	4	7,50	128	143,53	8,14	2,07	128	**139,23**	5,90
A-n54-k7	2	6,61	322	331,80	8,71	1,71	322	**328,80**	5,47
	3	7,38	269	284,40	8,31	1,95	**258**	**281,87**	12,37
	4	8,27	242	252,67	4,15	2,23	242	**252,30**	3,93
F-n72-k4	2	9,61	112	**113,87**	1,04	2,44	**110**	116,10	3,73
	3	10,35	85	**85,90**	0,55	2,66	85	86,40	0,97
	4	11,67	65	**65,37**	0,49	3,06	65	67,43	3,64
eil76	2	9,95	284	**293,93**	8,38	2,61	**282**	298,73	9,67
	3	10,79	199	**208,70**	6,56	2,98	**197**	212,13	7,35
	4	11,85	**153**	**161,87**	5,21	3,35	155	164,33	5,47
kroA100	2	14,36	11484	12352,00	390,59	3,98	11484	**12056,00**	514,46
	3	15,56	**8326**	**8661,50**	304,03	4,47	8438	9030,00	328,94
	4	16,71	**7279**	**7607,80**	145,66	4,96	7305	7627,70	250,79
M-n101-k10	2	14,73	**275**	**298,43**	13,94	3,89	276	308,63	17,87
	3	15,67	207	**211,33**	4,80	4,37	207	216,57	7,37
	4	17,06	173	**180,40**	6,19	4,88	173	185,07	13,48
eil101	2	14,77	**327**	**342,33**	9,14	3,94	334	347,77	9,60

	3	15,53	232	**244,90**	8,29	4,37	**228**	247,07	11,86
	4	16,91	**185**	**194,93**	5,65	4,89	186	198,37	7,00
kroA150	**2**	25,26	**14446**	**14926,00**	392,52	7,03	14484	14978,00	368,64
	3	26,33	**10049**	**10472,00**	175,83	7,64	10161	10891,00	462,77
	4	28,78	8365	8800,90	238,24	8,39	8365	**8675,10**	213,32
kroA200	**2**	37,95	15798	**16183,00**	197,80	10,57	**15712**	16255,00	306,13
	3	39,22	**11091**	**11512,00**	211,50	11,41	11242	11972,00	356,77
	4	41,21	**9063**	**9757,50**	484,48	12,41	9166	10392,00	508,57
gil262	**2**	56,50	1261	**1294,30**	27,43	16,00	1261	1323,70	42,86
	3	56,90	859	**894,27**	23,75	16,84	**856**	920,63	30,61
	4	58,96	**661**	**701,93**	29,75	18,17	680	754,40	39,17
fl417	**2**	121,41	6876	**7494,90**	311,58	32,78	**6835**	7641,50	313,57
	3	119,02	5474	5526,10	56,81	34,55	**5340**	**5507,30**	90,35
	4	123,90	**4768**	**4847,30**	137,26	35,37	4810	5007,90	147,29

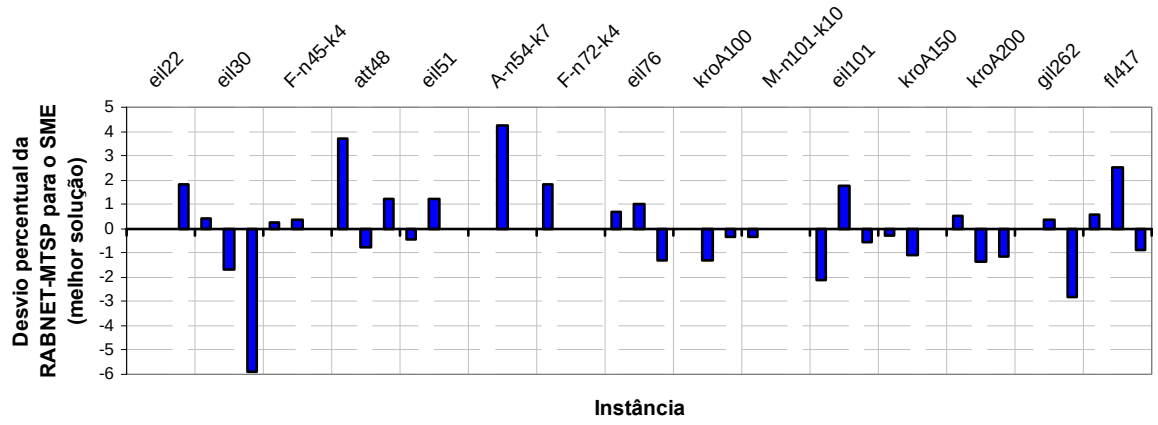
5 Resultados Computacionais

Para avaliar o desempenho da RABNET-MTSP, diversos testes foram realizados com instâncias provenientes de: TSPLIB [27], Fisher [12], Augerat et al. [3] e Christofides et al. [6]. Estas instâncias são utilizadas para o problema do caixeiro viajante e para o problema de roteamento de veículos capacitados. Para instâncias deste segundo problema a demanda dos clientes e a capacidade dos veículos foram desconsideradas. Estas instâncias são publicamente acessíveis através dos seguintes endereços eletrônicos: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/ e http://www.branchandcut.org/VRP/data.

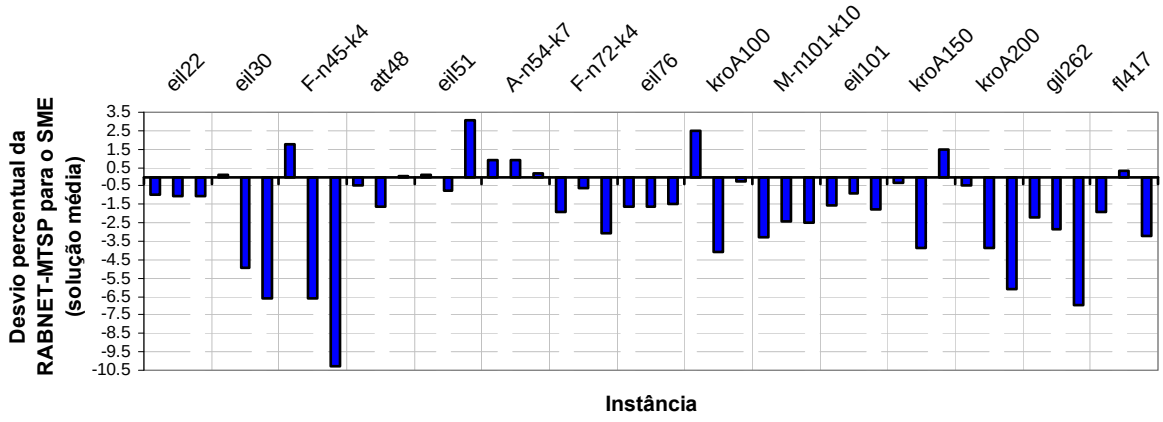
Foram utilizadas 15 instâncias variando em tamanho entre 22 e 417 cidades, sendo o depósito a primeira cidade. Foram realizados testes com 2, 3 e 4 caixeiros, respectivamente. Os testes foram feitos para o objetivo *minmax* e, como comparativo, foi implementado o algoritmo proposto por Somhom et al. [29], nomeado aqui de SME. Os dois algoritmos, RABNET-MTSP e o SME foram implementados em MATLAB e executados em um computador Intel Core 2 Duo 2.0 GHz com 2 GB de RAM.

A Tabela 2 apresenta os resultados computacionais para a RABNET-MTSP e para o SME em termos de esforço computacional e qualidade da solução, obtidos após 30 execuções de cada algoritmo para cada instância e quantidade de caixeiros. Em relação ao tempo de processamento, é possível observar que o SME teve menor tempo de execução que a RABNET-MTSP em todos os testes realizados. No entanto, em relação à qualidade das soluções, a RABNET-MTSP obteve melhores soluções que o SME em 16 testes, igualou a solução em 13 testes e obteve pior solução em 16 testes, totalizando 47 testes. Isto pode ser visto na Tabela 2, na qual as soluções realçadas são as melhores soluções obtidas por um ou pelos dois algoritmos e aquelas em negrito são as melhores soluções obtidas apenas por um dos algoritmos. Em relação à solução média, a RABNET-MTSP obteve melhores soluções em 34 testes, já o SME obteve melhores soluções em 11 testes.

A Figura 8(a) apresenta graficamente o desvio percentual das melhores soluções obtidas pela RABNET-MTSP em relação àquelas obtidas pelo SME. Barras acima da abscissa indicam o quanto as soluções obtidas pela RABNET-MTSP são piores que as obtidas pelo SME; barras abaixo da abscissa indicam melhores soluções obtidas pela RABNET-MTSP. O mesmo é apresentado na Figura 8(b) para as soluções médias.



(a)



(b)

Figura 8: Desvio percentual das soluções obtidas pela RABNET-MTSP para as soluções obtidas pelo algoritmo de Somhom et al. [29]. Barras abaixo da abscissa indicam soluções melhores encontradas pela RABNET-MTSP. (a) melhores soluções; (b) soluções médias.

5.1 Análise do Tempo de Execução dos Algoritmos

Dois parâmetros relativos a uma instância de MTSP influenciam o tempo de processamento do algoritmo: o número de cidades, N; e o número de caixeiros, m. Para determinar a complexidade da RABNET-MTSP e compará-la à do algoritmo proposto em Somhom et al. [29] diversos testes foram realizados para um conjunto de instâncias com valores diferentes dos parâmetros N e m. Esta seção apresenta uma análise da tendência de complexidade temporal dos algoritmos variando-se o tamanho de instâncias artificialmente geradas e criando-se funções de aproximação que se ajustem aos dados gerados.

As instâncias utilizadas nestes testes foram criadas com cidades uniformemente distribuídas no plano unitário ([0, 1]) e seu depósito é a cidade medóide desta instância. Para os testes variando o número de cidades, foram criadas 10 instâncias com $N = \{100; 200; 300; 400; 500; 600; 700; 800; 900; 1000\}$ e, para todas elas, $m = 4$. Para os testes variando o número de caixeiros, foi utilizada a instância com $N = 200$, previamente criada, e $m = \{2; 3; 4; 5; 6; 7; 8; 9; 10\}$. Tanto a RABNET-MTSP quanto o algoritmo proposto em [29] foram submetidos ao conjunto de testes envolvendo estas instâncias, sendo que cada algoritmo foi executado 30 vezes para cada uma delas.

Os valores do tempo de execução para cada instância e de cada algoritmo são utilizados para determinar uma curva de tendência e, conseqüentemente, aproximar a complexidade destes algoritmos. Para tanto o *toolbox* de *Curve Fitting* do MATLAB foi utilizado.

Tabela 3: Tempo médio de processamento em função do número de cidades e de caixeiros para a RABNET-MTSP e para o algoritmo proposto por Somhom et al. [29].

RABNET-MTSP			**Somhom et al. [29]**		
N	***m***	**tempo**	***N***	***m***	**tempo**
100	4	16,818	100	4	5,036
200	4	40,907	200	4	12,668
300	4	72,435	300	4	22,726
400	4	111,313	400	4	35,074
500	4	158,839	500	4	49,396
600	4	212,197	600	4	67,435
700	4	273,488	700	4	87,272
800	4	343,388	800	4	109,360
900	4	339,891	900	4	135,650
1000	4	404,981	1000	4	161,310
N	***m***	**tempo**	***N***	***m***	**tempo**
200	2	33,929	200	2	10,581
200	3	35,237	200	3	11,445
200	4	37,008	200	4	12,450
200	5	38,932	200	5	13,404
200	6	41,920	200	6	14,580
200	7	44,236	200	7	15,553
200	8	47,716	200	8	16,693
200	9	50,746	200	9	17,777
200	10	53,309	200	10	19,049

5.1.1 Análises para a RABNET-MTSP

A

Tabela 3 apresenta o tempo médio de processamento da RABNET-MTSP em função do número de cidades e do número de caixeiros. Em relação ao número de cidades, a aproximação feita através da ferramenta utilizada resultou no seguinte polinômio do terceiro grau:

$$\text{tempo}(N) = -5{,}90\times10^{-7}\cdot N^3 + 1{,}06\times10^{-3}\cdot N^2 - 9{,}40\times10^{-2}\cdot N + 18{,}5,\ N \neq 0. \quad (11)$$

Esta aproximação é ilustrada na Figura 9 e sugere uma estimativa de crescimento polinomial de ordem três da complexidade da RABNET-MTSP em relação ao número de cidades.

Fazendo a aproximação em função do número de caixeiros, foi obtido o seguinte polinômio do segundo grau:

$$\text{tempo}(m) = 1{,}2\times10^{-1}\cdot m^2 + 1{,}0\times10^{-2}\cdot m + 31{,}1,\ m \neq 0. \quad (12)$$

Esta aproximação é ilustrada na Figura 10 e sugere uma estimativa de crescimento polinomial de ordem dois da complexidade da RABNET-MTSP em relação ao número de caixeiros.

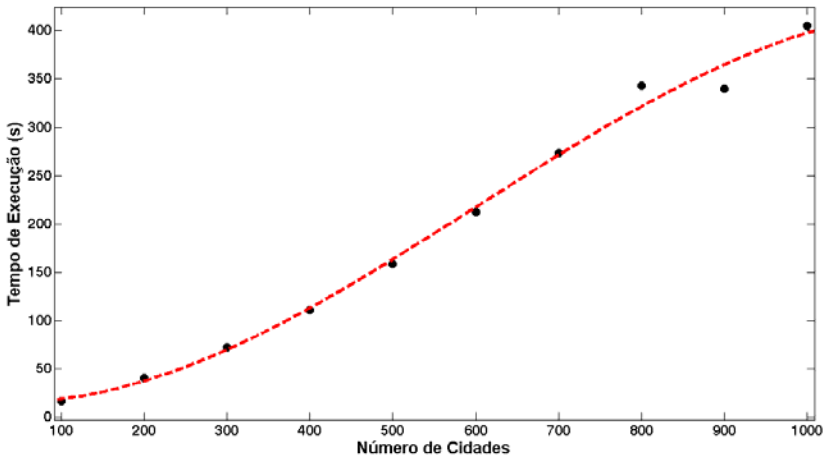


Figura 9: Tempo de execução da RABNET-MTSP em função do número de cidades da instância a ser resolvida. Os pontos indicam o tempo obtido em testes e a curva pontilhada é a aproximação polinomial de ordem 3 obtida.

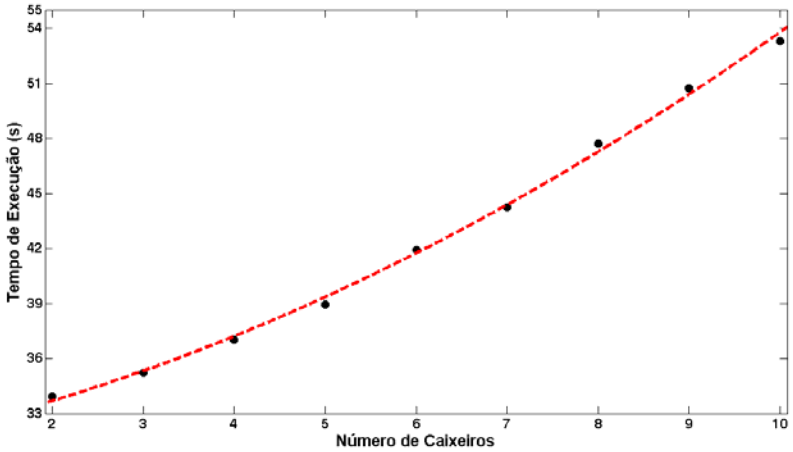


Figura 10: Tempo de execução da RABNET-MTSP em função do número de caixeiros. Os pontos indicam o tempo obtido em testes e a curva pontilhada é a aproximação polinomial de ordem dois obtida.

5.1.2 Análises para o algoritmo de Somhom

A

Tabela 3 apresenta o tempo de execução do algoritmo proposto por Somhom et al. [29] em função do número de cidades e de caixeiros. A aproximação feita em função do número de cidades apresentou o seguinte polinômio do segundo grau:

$$\text{tempo}(N) = 1{,}2\times10^{-4}\cdot N^2 + 3{,}7\times10^{-2}\cdot N + 1{,}1\times10^{-1},\ N \neq 0. \quad (13)$$

Esta aproximação é ilustrada na Figura 11 e sugere uma estimativa de crescimento polinomial de ordem dois da complexidade do algoritmo de Somhom et al [29] em relação ao número de cidades.

Em relação ao número de caixeiros, a aproximação obtida foi linear:

$$\text{tempo}(m) = 1.1\cdot m + 8.3,\ m \neq 0. \quad (14)$$

A ilustração desta aproximação é ilustrada na Figura 12, e sugere uma estimativa de crescimento linear da complexidade do algoritmo de Somhom et al. [29] em relação ao número de caixeiros.

6 Conclusões

Este trabalho apresentou uma heurística híbrida em sistemas imunológicos artificiais e mapas auto-organizáveis para a solução do problema de múltiplos caixeiros viajantes. Uma breve descrição de trabalhos relacionados também foi apresentada.

A ferramenta proposta é uma modificação da RABNET-TSP, sendo que as principais modificações necessárias foram: 1) utilização de *m* sub-redes independentes, cada uma representando um caixeiro; e 2) etapas de competição, cooperação, adaptação e crescimento que consideram as *m* sub-redes e a independência entre elas.

Para avaliar a performance do algoritmo diversos testes foram realizados com instâncias comumente utilizadas na literatura e, como comparativo, o algoritmo proposto por Somhom et al. (SME) foi implementado. A comparação dos resultados obtidos pela RABNET-MTSP com aqueles obtidos pelo SME demonstrou que este primeiro é capaz de obter, em geral, melhores soluções, porém, com maior tempo de processamento. O tempo de processamento dos algoritmos foi analisado empiricamente e utilizando técnicas de aproximação de funções, o qual se mostrou favorável ao algoritmo SME.

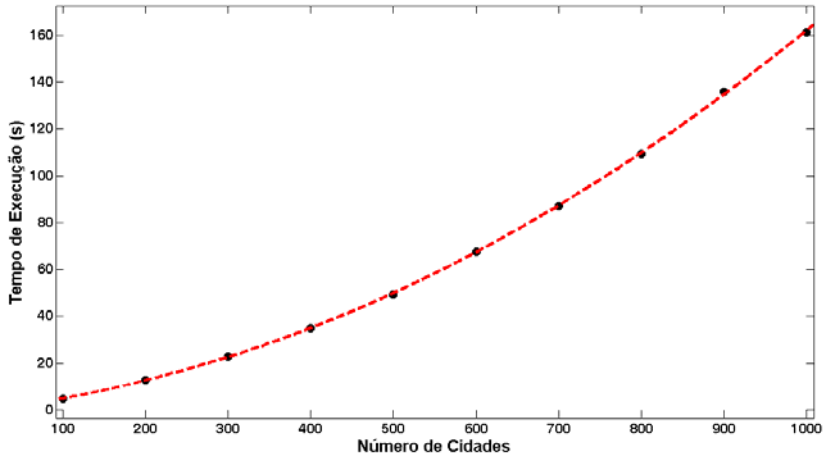


Figura 11: Tempo de execução do algoritmo de Somhom et al. [29] em função do número de cidades da instância a ser resolvida. Os pontos indicam o tempo obtido em testes e a curva pontilhada é a aproximação polinomial de ordem dois obtida.

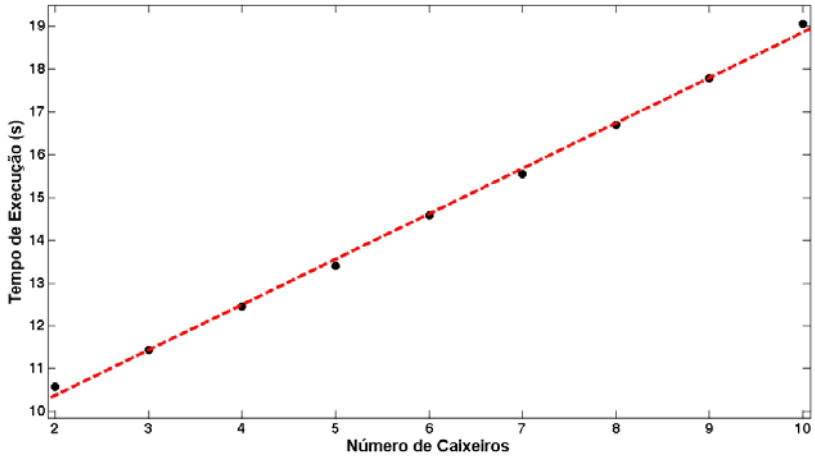


Figura 12: Tempo de execução do algoritmo de Somhom et al. [29] em função do número de caixeiros. Os pontos indicam o tempo obtido em testes e a curva pontilhada é a aproximação linear obtida.

Os bons resultados obtidos reafirmam o potencial de redes auto-organizáveis na solução de problemas de roteamento e motivam a modificação do algoritmo proposto para a solução de problemas mais complexos como, por exemplo, o problema de roteamento de veículos capacitados. Além disso, investigações futuras devem abordar propostas de diminuição do tempo de processamento da RABNET-MTSP sem perda significativa da qualidade da solução. Estudos iniciais demonstraram que grande parte do tempo de execução é utilizada durante a etapa de competição. Portanto, maior atenção deve ser dada à proposição de heurísticas de competição.

Agradecimentos

Os autores agradecem à FAPESP e ao CNPq pelo suporte financeiro.

Referências Bibliográficas

[1] Angel, R.D., Caudle, W.L., Noonan, R. & Whinston, A., "Computer-Assisted School Bus Scheduling", Management Science, vol. 18(6), pp. B279-B288 (1972).

[2] Angeniol, B., Croix Vaubois, G. & Le Texier, J-Y., "Self-Organizing Feature Maps and the Travelling Salesman Problem", Neural Networks, vol. 1(4), pp. 289-293 (1988).

[3] Augerat, P., Belenguer, J.M., Benavent, E., Corberán, A., Naddef, D. & Rinaldi, G., "Computational Results with a Branch and Cut Code for the Capacitated Vehicle Routing Problem", Research Report 949-M, Universite Joseph Fourier, Grenoble, France (1995).

[4] Bektas, T., "The Multiple Traveling Salesman Problem: An Overview of Formulations and Solution Procedures", Omega, vol. 34(3), pp. 209-219 (2006).

[5] Burke, L.I. & Damany, P., "The Guilty Net for the Traveling Salesman Problem", Computers and Operations Research, vol. 19(3-4), pp. 255-265 (1992).

[6] Christofides, N., Mingozzi, A. & Toth, P., "The Vehicle Routing Problem", Combinatorial Optimization, Wiley, Chichester (1979).

[7] Cochrane, E.M. & Beasley, J.E., "The Co-Adaptive Neural Network Approach to the Euclidean Travelling Salesman Problem", Neural Networks, vol. 16(10), pp. 1499-1525 (2003).

[8] de Castro, L.N. & Timmis, J., "Artificial Immune Systems: A New Computational Intelligence Approach", Springer-Verlag (2002).

[9] de Castro, L.N., "Fundamentals of Natural Computing: Basic Concepts, Algorithms, and Applications", CRC Press LLC (2006).

[10] de Castro, L.N, "Engenharia Imunológica: Desenvolvimento e Aplicação de Ferramentas Computacionais Inspiradas em Sistemas Imunológicos Artificiais", Tese de Doutorado, Universidade Estadual de Campinas (2001).

[11] Durbin, R. & Willshaw, D., "An Analogue Approach to the Traveling Salesman Problem Using an Elastic Net Method", Nature, vol. 326, pp. 689-691 (1987).

[12] Fisher, M.L., Optimal Solution of Vehicle Routing Problems Using Minimum K-Trees, Operations Research, vol. 42(4), pp. 626-642 (1994).

[13] Fort, J.C., "Solving a Combinatorial Problem Via Self-Organizing Process: An Application of the Kohonen Algorithm to the Traveling Salesman Problem", Biological Cybernetics, vol. 59, pp. 33-40 (1988).

[14] França, P.M., Gendreau M., Laporte, G. & Müller, F.M., "The m-Traveling Salesman Problem with Minmax Objective", Transportation Science, vol. 29(3), pp. 267-275 (1995).

[15] Ghaziri, H.El, **"**Solving Routing Problems by a Self-Organizing Map**",** In: Kohonen, T., Makisara, K. kangas, J., Artificial Neural Networks. North-Holland, Amsterdam, pp. 829-834 (1991).

[16] Gilbert, K.C. & Hofstra, R.B., "A New Multiperiod Multiple Traveling Salesman Problem With Heurist and Application to a Scheduling Problem", Decision Sciences, vol. 23, pp. 250-259 (1992).

[17] Haykin, S., "Neural Networks: A Comprehensive Foundation", 2nd Ed., Prentice Hall (1999).

[18] Helsgaum, K., "An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic", European Journal of Operational Research, vol. 126, pp. 106-130 (2000).

[19] Hsu, C., Tsai, M. & Chen, W., "A Study of Feature-Mapped Approach to the Multiple Travelling Salesmen Problem", IEEE International Sympoisum on Circuits and Systems, vol. 3, pp. 1589-1592 (1991).

[20] Knidel, H., de Castro, L.N. & Von Zuben, F.J., "Data Clustering with a Neuro-Immune Network", Lecture Notes in Computer Science, vol. 3610, pp. 1279-1288 (2005).

[21] Kohonen, T., "Self-Organizing Maps", 3rd Ed., Springer-Verlag (2000).

[22] Lin, S. & Kernighan, B. W., "An Effective Heuristic Algorithm for the Traveling-Salesman Problem", Operations Research, vol. 21, pp. 498-516 (1973).

[23] Masutti, T.A.S. & de Castro, L.N., "A Constructive Self-Organizing Network Applied to a Discrete Optimization Problem", Proceedings of the Seventh International Conference on Intelligent Systems Design and Applications (ISDA'07), pp. 52-57 (2007).

[24] Modares, A., Somhom, S. & Enkawa, T., "A Self-Organizing Neural Network Approach for Multiple Traveling Salesman and Vehicle Routing Problems", International Transactions in Operational Research, vol. 6(6), pp. 591-606 (1999).

[25] Pasti, R. & de Castro, L.N., "A Neuro-Immune Network for Solving the Traveling Salesman Problem", International Joint Conference on Neural Networks, pp. 3760-3766 (2006).

[26] Potvin, J., Lapalme, G. & Rousseau, J., "A Generalized k-opt Exchange Procedure for the MTSP", Information Systems and Operations Research, vol. 27(4), pp. 474-481 (1989).

[27] Reinelt, G., "TSPLIB - A Traveling Salesman Problem Library", ORSA Journal on Computing , vol. 3(4), pp. 376-384 (1991).

[28] Somhom, S., Modares, A. & Enkawa, T., "A Self-Organising Model for the Travelling Salesman Problem", Journal of the Operational Research Society, vol. 48(9), pp. 919-928 (1997).

[29] Somhom, S., Modares, A. & Enkawa, T., "Competition-Based Neural Network for the Multiple Travelling Salesmen Problem with Minmax Objective", Computers and Operations Research, vol. 26(4), pp. 395-407 (1999).

[30] Svestka, J.K. & Rinnooy Kan, A.H.G., "Some Simple Applications of the Traveling Salesman Problem", Operational Research Quarterly, vol. 26, pp. 717-733 (1975).

[31] Vakhutinsky, A.I. & Golden, B.L., "Solving Vehicle Routing Problems Using Elastic Nets", IEEE International Conference on Neural Network, pp. 4535-4540 (1994).

[32] Wacholder, E., Han, J. & Mann, R.C., "A Neural Network Algorithm for the Multiple Traveling Salesmen Problem", Biological Cybernetics, vol. 61(1), pp. 11-19 (1989).

[33] Zhong, Y., Liang, J., Guochang, G., Zhang, R. & Yang, H., "An Implementation of Evolutionary Computation for Path Planning of Cooperative Mobile Robots", Proceedings of the Fourth World Congress on Intelligent Control Automation, vol. 3, pp. 1798-1802 (2002).

[34] Zhu, A. & Yang, S.X., "An Improved Self-Organizing Map Approach to Traveling Salesman Problem", Proceedings of the 2003 IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, pp. 674-679 (2003).