



APLICAÇÃO DE CAOS EM REDES NEURAIS AUTO-ORGANIZÁVEIS PARA RESOLVER PROBLEMAS DE OTIMIZAÇÃO

Josué Ervin Musial

Paulo Henrique Siqueira

ajmusial@gmail.com

paulohs@ufpr.br

Universidade Federal do Paraná

Rua Cel. Francisco Heráclito dos Santos, 100 - Centro Politécnico - Jardim das Américas, Caixa Postal 19081, CEP 81531-980, Curitiba, Paraná, Brasil

Abstract. *This study aims to present a modification to the algorithm of Self-Organizing Maps networks (SOM) using the concept of deterministic chaos for a comparison with results found in literature. Efficient problem solving as the Vehicle Routing is fundamental in reducing costs, efficient logistics and distribution products. Since its formulation in 1959, when Danzig and Ramser describe him as a generalization of the Travelling Salesman Problem (TSP), the second a classic optimization problem. Given a list of cities and the pair of distances including the TSP is to find the shortest possible route passing through every city exactly once. The solution to the TSP is computationally classified as a combinatorial optimization problem that demands a great computational effort and thus, there is a great interest in developing efficient heuristics to solve it. In the literature there are many heuristics that are used in the solution of TSP as Genetic Algorithms, Tabu Search, Simulated Annealing, Kohonen networks (SOM), among others. As this study focuses on the SOM algorithm, it is noteworthy that to solve the TSP using this network is necessary to make an adjustment creating a mesh of connected weights involving, or not, the geometric position of the cities. This algorithm produces an approximation of this mesh that adapts to a possible solution to the TSP. In the search for better solutions are added to some changes in the algorithm of SOM networks. Therefore this paper uses the concept of deterministic chaos in order to modify certain standards associations (winner neuron), and the adjustment weights of neighboring neurons for improvements in the solutions found for the TSP*

Keywords: *Neural Networks, Chaos Theory, Traveling Salesman Problem, Metaheuristics*

1 INTRODUÇÃO

As redes neurais tem sido amplamente aplicadas nos mais diferentes campos científicos para a solução dos mais variados problemas, bem como as mais variadas meta-heurísticas baseadas em algum comportamento biológico. Essas técnicas, em geral, tem alcançado um ótimo desempenho na solução de problemas de alta complexidade computacional, em especial nos problemas da pesquisa operacional.

Admitindo heurísticas baseadas no modelo de aprendizado do cérebro humano e que a formação dessa experiência se dá de maneira aleatória, os modelos heurísticos clássicos se mostram rígidos (Dingle et al., 1993) e um tanto quanto limitados se comparados com a capacidade de aprendizado dos organismos biológicos (Haykin, 2001). A melhor maneira de simular e/ou explicar o paradigma do aprendizado para aproximar os algoritmos heurísticos da capacidade cognitiva das estruturas biológicas é a utilização de caos nas redes neurais de qualquer natureza (Dingle et al., 1993).

O Problema do Caixeiro Viajante (Travelling Salesman Problem – TSP) é utilizado neste estudo, porque serve como plataforma de testes para investigar as diversas propostas heurísticas, além de ser um problema de alta aplicabilidade no mundo real, de fácil compreensão e descrição. Entretanto de difícil solução, pois pertence à classe de problemas NP-hard e que demanda um tempo computacional que pode ser exponencial dependendo da instância a ser solucionada. Assim, vários métodos tem sido desenvolvidos com o propósito de se resolver instâncias cada vez maiores desse problema em um tempo computacional menor.

Os mapas auto-organizáveis são conhecidos por sua capacidade de agrupar padrões de entrada devido ao modelo de sua função de densidade de probabilidade gerar uma ordem espacial. O trabalho de Dingle et al. (1993) apresenta simulações que ilustram a capacidade que o mapa auto-organizável caótico tem para executar cada uma destas tarefas. O caos é usado para escolher o neurônio vencedor de forma probabilística mas, o efeito não é examinado em relação aos problemas difíceis como por exemplo na resolução do TSP.

Este trabalho apresenta uma proposta de melhoria para a solução do clássico problema do Caixeiro Viajante, para tanto utilizou-se a meta-heurística clássica Redes de Kohonen, que foi adaptada para resolver este problema. Além dessa adaptação foi implementada a característica caótica nesse algoritmo com o objetivo de testá-la e comprovar sua eficiência. No final deste artigo apresentamos os primeiros resultados obtidos.

2 REDES NEURAIS AUTO-ORGANIZÁVEIS

A rede de Kohonen, mais conhecida como rede SOM (Self-Organizing Map) ou ainda, rede de mapas auto-organizáveis, foi desenvolvida pelo finlandês Teuvo Kohonen na década de 80. Seu algoritmo tem como princípio a auto-organização, um processo muito semelhante ao que ocorre no interior do cérebro humano (Kohonen, 2001). O modelo de Kohonen pertence a uma classe de algoritmos de codificação vetorial e, gera um mapeamento topológico dos vetores de entrada.

A rede SOM emprega uma mistura dinâmica de competição e cooperação para permitir a formação emergente de um isomorfismo entre um espaço de característica e uma matriz de

neurônios (Haykin, 2001). Ela simplesmente inspeciona os dados de entrada, buscando regularidades e padrões e organiza-os de tal maneira a formar uma ordenada descrição dos dados de entrada. Esta descrição pode levar a uma solução do problema em questão (Lueug, 2004).

O algoritmo se resume em alguns passos elementares: apresentação de uma entrada, cálculo do neurônio vencedor e atualização dos pesos até que o mapa de características esteja completo, ou seja, não sofra modificações significativas em seus pesos a cada interação. Os elementos fundamentais do algoritmo são os seguintes:

1. *Inicialização.* Atribua valores iniciais aleatórios e distintos para cada um dos pesos $w_j(0)$, onde $j = 1, 2, \dots, l$ e l é o número de neurônios da rede.
2. *Apresentação de um padrão de entrada.* Sorteie um vetor de entrada x ainda não apresentado à rede no tempo n .
3. *Cálculo do neurônio vencedor.* Encontre o neurônio vencedor $i(x)$ para o padrão de entrada x na interação n utilizando o critério da distância Euclidiana.

$$i(x) = \arg \min_j \|x(n) - w_j\|, \quad j = 1, 2, \dots, l \quad (1)$$

4. *Atualização.* Ajuste o valor dos pesos sinápticos de todos os neurônios usando a seguinte equação de atualização:

$$w_j(n+1) = w_j(n) + \eta(n)h_{j,i(x)}(n)(x - w_j(n)) \quad (2)$$

onde $\eta(n)$ é o parâmetro de aprendizagem e $h_{j,i(x)}(n)$ é a função de vizinhança onde o centro é o neurônio vencedor $i(x)$.

5. *Cálculo do Erro.* Continue até que não ocorram mudanças significativas no ajuste dos pesos.

3 CAOS

Caos Determinístico ou simplesmente Caos é o fenômeno pelo qual um comportamento imprevisível, aparentemente aleatório é produzido por um sistema completamente determinístico. Tal atividade é sensivelmente dependente das condições iniciais e, por consequência, imprevisível. Deste modo, pequenas diferenças nas condições iniciais podem gerar grandes diferenças nas saídas dentro de um tempo finito (Ott, 2002).

Um dos sistemas clássicos mais simples, capazes de produzir caos determinístico é a equação logística:

$$y(n+1) = 4\mu y(n) [1 - y(n)] \quad (3)$$

O comportamento dessa equação de diferença de primeira ordem muda drasticamente quando o parâmetro de bifurcação μ é alterado, conforme mostra a Figura 1. Para $2 < \mu < 2,9$ a saída converge para um único valor diferente de zero. Quando $\mu > 2,9$ a saída começa a oscilar primeiro entre 2 valores, depois 4 valores, em seguida, 8 valores e assim por diante, até que para $\mu > 3,6$ a saída se torna caótica.

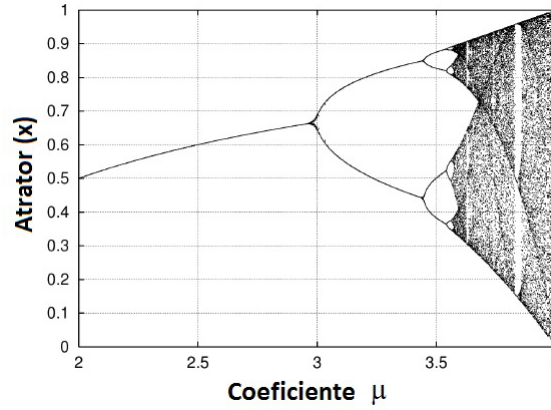


Figura 1: Diagrama de Bifurcação para o mapa logístico.

Num regime caótico um sistema apresenta sensibilidade às condições iniciais, ou seja, mesmo se as condições iniciais são próximas o resultado das iterações pode divergir exponencialmente, com o decorrer do tempo. Uma maneira de medir essa divergência é através do *expoente de Lyapunov* λ (Ott, 2002).

O expoente de Lyapunov $\lambda(x_0)$ é definido na equação (4).

$$\lambda(x_0) = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^{N-1} \ln |f'(x_n)| \quad (4)$$

onde N é o número de iterações, $x_1 = f(x_0)$, $x_2 = f(x_1) \dots$. Assim se $\lambda(x_0) > 0$ o sistema apresenta regime caótico e se $\lambda(x_0) \leq 0$ o sistema apresenta regime regular (Ott, 2002).

Outro conceito utilizado neste estudo é a unidade neural caótica fundamentada na equação logística invertida, na qual a entrada convencional da unidade neural é dada por:

$$net = \sum_i w_i x_i \quad (5)$$

onde x_i são as entradas da rede neural e w_i são os pesos associados. A próxima saída $y(n+1)$ da unidade neural depende da entrada da saída anterior como segue:

$$y(n+1) = 1 - 4[1 - net]y(n)[1 - y(n)] \quad (6)$$

O comportamento desta unidade neural depende da sua entrada, se $net < 0,11$ a unidade exibe atividade caótica neural, pois satisfaz as mesmas condições da Equação (5) que apresenta atividade caótica para $\mu > 0,89$. Para $0,11 < net < 0,25$ produz atividade periódica, enquanto que para $net > 0,25$ produz um único valor que cresce com o aumento do valor net e é igual a unidade para $net > 0,75$.

Um dos pioneiros na utilização de caos em redes neurais foi Aihara et al. (1990). Em seu trabalho é proposto o modelo com um único neurônio com dinâmica caótica, considerando as seguintes propriedades dos neurônios biológicos: memória, refratariedade e soma espaço-temporal das entradas (Aihara et al., 1990). O modelo contempla os modelos convencionais de

um neurônio como um dos seus casos especiais; ou seja, a dinâmica caótica é apresentada como uma extensão natural dos modelos de neurônios não-caóticos. Em resumo o trabalho mostra a viabilidade da utilização de caos determinístico em redes neurais.

4 METODOLOGIA

4.1 Problema do Caixeiro Viajante (TSP)

Utiliza-se amplamente nos experimentos dos mais diversos métodos de otimização o TSP, pois trata-se de um problema de fácil descrição e compreensão. Entretanto apresenta uma grande dificuldade na obtenção da solução por que é um problema NP-Difícil (Karp, 1975), apesar disso apresenta uma aplicabilidade ampla.

O TSP é um típico problema de otimização combinatória, ele pode ser entendido como uma busca pelo menor caminho fechado que visita cada cidade uma vez e apenas uma vez. A forma de problema de decisão do TSP é um NP-completo, por isso o grande interesse em heurísticas eficientes para resolvê-lo.

Existem muitas das heurísticas que utilizam o paradigma da computação neural ou noções relacionadas recentemente. A primeira abordagem para o TSP via redes neurais foi no trabalho de Hopfield e Tank (1985), que foi baseado na minimização de uma função de energia, e os mínimos locais devem corresponder a uma boa solução de uma instância do TSP. No entanto, esta abordagem não garante a viabilidade, ou seja, nem todos os pontos de mínimos da função energia representam solução viável para o TSP.

A formulação matemática para o *problema do caixeiro viajante* é a seguinte:

$$\min C(x) = \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \quad (7)$$

sujeito às restrições

$$\sum_{i=1}^n x_{ij} = 1, \quad \text{para } j = 1, 2, \dots, n, \quad (8)$$

$$\sum_{j=1}^n x_{ij} = 1, \quad \text{para } i = 1, 2, \dots, n, \quad (9)$$

$$x_{ij} \in \{0, 1\}, \quad \text{para } i, j = 1, 2, \dots, n, \quad (10)$$

$$\tilde{x} \text{ forma um ciclo Hamiltoniano} \quad (11)$$

onde x_{ij} são as variáveis de decisão e c_{ij} é o custo de designar a distância da cidade i para a cidade j .

Outras técnicas clássicas são utilizadas para resolver o TSP, por exemplo, o método Simplex e os métodos exatos como a inserção mais econômica e a inserção do nó mais distante. Além das

técnicas existem outras maneiras para resolver o TSP, destacando-se os Algoritmos Genéticos (Goldberg, 1986) e as Redes Neurais Artificiais (Fausett, 1994).

Uma técnica para a solução do TSP apresentada em Siqueira (2005), utiliza a resolução clássica do problema da Designação como uma fase preliminar da resolução do TSP, na qual considera-se a solução ótima para a matriz de custos do problema da Designação como uma solução inicial para o TSP. Se a solução inicial determina uma rota viável então esta já é a solução ótima do TSP, caso contrário, um método “Winner Takes All” com rede recorrente é aplicado para viabilizar a solução.

4.2 Algoritmo SOM para Resolver o TSP

O algoritmo SOM para ser aplicado na resolução do TSP deve sofrer uma adaptação para que se obtenha como resposta, após o final do processo, uma sequência de cidades a serem visitadas, ou seja, uma rota solução para o TSP. Para tal, é criada uma rede bidimensional com m neurônios de saída e entrada de dados bidimensional com $2m$ pesos dispostos em forma de circunferência ao redor das entradas do TSP em questão conforme exemplificado na Figura 2.

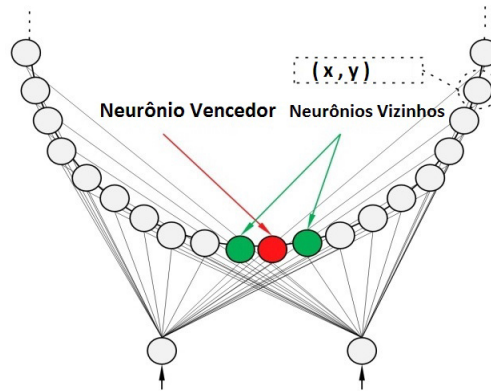


Figura 2: Distribuição dos neurônios.

Para simplificar a implementação os dados de entrada bem como a camada de pesos, estes foram convertidos em uma escala que varia no intervalo $[0, 1]$ utilizando a equação (12)

$$x_{escala} = \frac{(x - x_{min})}{(x - x_{max})} \quad (12)$$

onde x é o valor da entrada a ser convertida e x_{min} e x_{max} são respectivamente os valores de mínimo e máximo das entradas antes da conversão para a escala (Bai et al., 2006).

A arquitetura SOM consiste além das entradas bidimensionais das coordenadas espaciais das cidades e de um anel sobre o qual os neurônios estão espacialmente distribuídos como pode ser observado na Figura 2. Dispor os neurônios em forma de anel é a maneira mais citada na literatura de adaptar o algoritmo SOM para resolver o TSP.

Os pesos dos neurônios definem a posição do neurônio no espaço, e inicialmente são definidos do seguinte modo: assumindo um anel de m neurônios, os neurônios são igualmente posicionados em um círculo de raio unitário, utilizando a medida em graus, a posição do ângulo

de dado neurônio que é igual a 360° dividido por m . Os dados de entrada (um conjunto de n cidades) são apresentados ao SOM em uma ordem aleatória e uma competição baseada em distância euclidiana é realizada entre os neurônios no anel. O neurônio vencedor é o neurônio i^* que possui a distância mínima para a cidade apresentada.

$i^* = \min_j \{\|x_i - w_j\|_2\}$, onde x_i é a coordenada do i -ésima entrada, w_j é a posição do neurônio j e $\|\bullet\|_2$ é a distância euclidiana. Assim, o neurônio vencedor i^* , bem como os neurônios vizinhos movem-se na direção desse neurônio, usando a função de vizinhança:

$$f(\sigma, d) = \exp\left(\frac{-d^2}{\sigma^2}\right). \quad (13)$$

Os neurônios são atualizados segundo a seguinte função de atualização:

$$y_j^{novo} = y_j^{atual} + \alpha \cdot f(\sigma, d) \cdot (x_i - y_j^{atual}) \quad (14)$$

onde α e σ são a taxa de aprendizagem e o parâmetro de vizinhança respectivamente e $d = \min\{\|j - J\|, m - \|j - J\|\}$ é a distância cardinal calculada ao longo dos neurônios j e J onde $\|\cdot\|$ representa o valor absoluto. Taxa de aprendizagem tem característica dinâmica e diminui gradualmente durante o processo de aprendizado, assumindo valores no intervalo $(0; 1)$. Do mesmo modo, a função de vizinhança $f(\sigma, d)$ assume um valor muito grande no início do treinamento, e diminui lentamente, em abrangência, com o progresso do treinamento. Depois de muitas iterações os neurônios tendem a se aproximar das entradas e finalmente convergem para elas. Uma vez que todos neurônios convergiram para os valores de entrada, basta ler as coordenadas na ordem em que aparecem. O resultando dessa sequência constitui uma solução para o TSP em questão. A Figura 3 ilustra a evolução do algoritmo a partir do estado inicial do anel, (a) atingindo uma fase intermediária, após algumas iterações (b) e estabilizando no estado final (c).

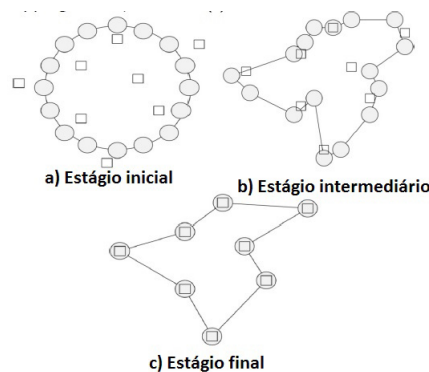


Figura 3: Exemplo.

O número de neurônios deve ser maior do que o número de cidades para evitar a oscilação de um neurônio e seus diferentes vizinhos. Uma boa alternativa é definir o número de neurônios m como sendo o dobro do número de vértices, ou seja, $m = 2n$.

4.3 Caos em Redes de Kohonen

A aplicação de caos determinístico em redes de Kohonen foi feita pela primeira vez por Dingle e Jones (1993). Os mapas auto-organizáveis convencionais utilizam unidades lineares:

$$y_{pj} = net_{pj} = \sum_{i=1}^M w_{ji}x_{pi} = w_j \text{ e } x_p = \|w_j\| \|x_p\| \cos \phi_{pj} \quad (15)$$

onde ϕ_{pj} é o ângulo entre o padrão de entrada x_p e o vetor de peso w_j . Tipicamente os vetores de entrada e de peso são unidades de comprimento, logo $y_{pj} = \cos \phi_{pj}$. Quando um padrão de entrada x_p é apresentado à rede, as unidades neurais competem para ter o privilégio de aprender e a unidade neural vencedora será aquela com a maior saída y_{pj} . Como $y_{pj} = \cos \phi_{pj}$, tem-se que a unidade neural vencedora é aquela cujo vetor de peso é mais próximo do padrão de entrada. O vetor de peso da unidade neural vencedora e os seus vizinhos são modificados de acordo com a seguinte regra, que gira no sentido w_j o padrão de entrada x_p :

$$w_j = \frac{w_j^{atual} + \lambda x_p}{\|w_j^{atual} + \lambda x_p\|} \quad (16)$$

A taxa de aprendizado λ , geralmente, diminui de forma linear ao longo da sessão de treinamento de um valor inicial $\lambda(0)$ para um valor final $\lambda_{\min}$.

O caos pode ser introduzido no mapa auto-organizável por substituição dos neurônios lineares da rede convencional por neurônios caóticos (6). Assim, quando um padrão de entrada x_p é apresentado para a rede, cada neurônio produz uma sequência de saídas $y_{pj}(n)$. Tal como acontece com o mapa auto-organizável convencional, os neurônios competem para ter o privilégio de aprender, mas, neste caso, **o neurônio vencedor é escolhido para ser o único com a maior produção média $\bar{y}_{pj}$ calculada sobre os elementos T da sequência de saída:**

$$\bar{y}_{pj} = \frac{1}{T} \sum_{n=1}^T y_{pj}(n). \quad (17)$$

Na prática, T é relativamente pequena e o $\bar{y}_{pj}$ média de saída depende do valor de entrada net_{pj} e o valor inicial $y_{pj}(0)$. Assim, a competição para aprender efetivamente se torna probabilística, com a unidade neural possuindo uma maior entrada líquida (ou seja, mais próximo do padrão de entrada), geralmente tendo maior probabilidade de ganhar. Tal como acontece com o mapa auto-organizável convencional, o neurônio vencedor e seus vizinhos ajustaram seus vetores de pesos de acordo com (16).

5 RESULTADO E DISCUSSÕES

O algoritmo SOM possui dois parâmetros adaptativos, o coeficiente de aprendizagem α_n e a função de vizinhança (normalmente a função gaussiana é adotada) e o coeficiente de variância σ_n propostos por Kohonen apresentam um decréscimo exponencial da seguinte forma:

$$\alpha_n = \alpha_0 \times \left(-\frac{n}{\tau_1} \right) \quad (18)$$

$$\sigma_n = \sigma_0 \times \left(-\frac{n}{\tau_2} \right) \quad (19)$$

em que $n = 1, 2, 3, \dots$ é o número de iterações e τ_1 e τ_2 são constantes de tempo exponencial.

Uma adaptação ao algoritmo SOM para a solução do TSP, proposta por Costa et all. (2003), adapta o algoritmo clássico para obtenção dos coeficientes de aprendizagem e raio de vizinhança pelas seguintes equações:

$$\alpha_n = \frac{1}{\sqrt[3]{n}} \quad (20)$$

$$\sigma_n = \sigma_{n-1} \times (1 - 0.01 \times n) \quad (21)$$

e ainda

$$\sigma_0 = \frac{l}{4 \times c} \quad (22)$$

em que l é o número de neurônios e c é uma constante, neste trabalho foram adotados os seguintes valores: $c = 8$ ou $c = 10$.

O algoritmo modificado segue os seguintes passos:

1. Inicialização: as entradas do algoritmo são formadas pelas coordenadas cartesianas das cidades, juntamente com o número de neurônios escolhidos para os cálculos. O número de neurônios devem ser maiores que o número de cidades e inicialmente são dispostos em um retângulo que envolve a posição geométrica das cidades.
2. Parâmetros de adaptação: α_n e σ_n calculados de acordo com as equações 20 e 21.
3. Avaliação da vizinhança: que é obtida pela avaliação dos parâmetros de vizinhança e aprendizagem através de sua atualização.
4. Competição: Cálculo do neurônio vencedor
5. Cooperação O neurônio vencedor tem sua vizinhança adaptada de acordo com a evolução do raio e coeficiente cognitivo descritos no passo 3 e acrescido da soma de um termo com característica caótica.
6. Adaptação: Retorne ao passo 2 enquanto $\alpha_n > \alpha_{\min}$.

Para o teste inicial foram escolhidas algumas instâncias do TSP com um número pequeno de cidades, descritas brevemente na Tabela 1.

Tabela 1: Instâncias utilizadas nos experimentos computacionais.

Instâncias	Número de Cidades	Tamanho da rota ótima
ulisses22	22	70.13
Eil51	51	426
Eil76	76	538
eil101	101	629
gr96	96	55209
st70	70	675

Os primeiros resultados foram obtidos utilizando o algoritmo SOM e o algoritmo SOM modificado com característica caótica SOM/Caos. O método foi aplicado 5 vezes para cada instância e resultou em uma solução média. Os resultados são apresentados na Tabela 2.

Tabela 2: Instâncias utilizadas nos experimentos computacionais.

Instâncias	SOM	SOM/Caos	Erro em (%)
ulisses22	80.12	76.53	9
Eil51	464.28	455.14	6.8
Eil76	581.64	571.52	6.2
eil101	661.68	656.54	4.3
gr96	55391.3	55384.56	3.2
st70	712.47	709.23	5.0

Inicialmente os neurônios são dispostos em uma malha com formato parabólico, envolvendo as cidades e o algoritmo SOM/Caos modela essa malha ajustando a rota solução do TSP apresentada pelo algoritmo. Na Figura 4 apresenta-se o comportamento do algoritmo após a primeira iteração para a instância Eil51.

A Figura 5 apresenta o resultado do algoritmo SOM/Caos, aplicado na instância Eil51, após a inclusão das cidades que ficaram fora da rota e aplicação do refinamento, caso tenha ocorrido cruzamento de arcos.

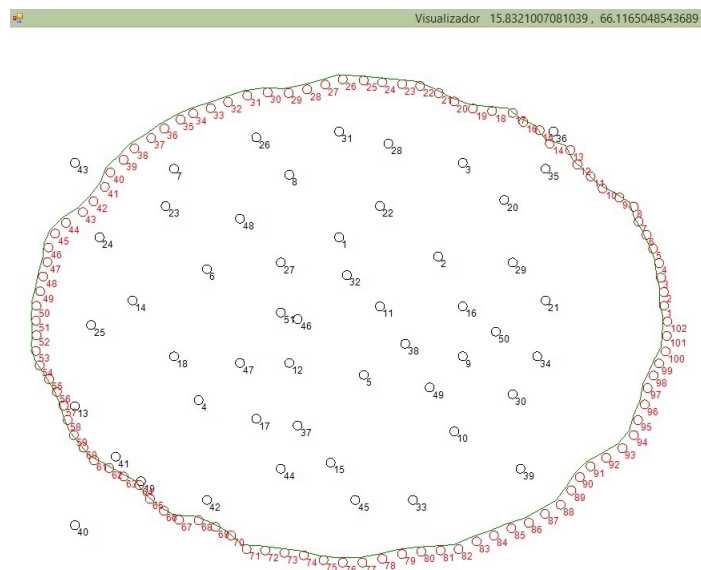


Figura 4: Primeira iteração do SOM/Caos para a instância Eil51.

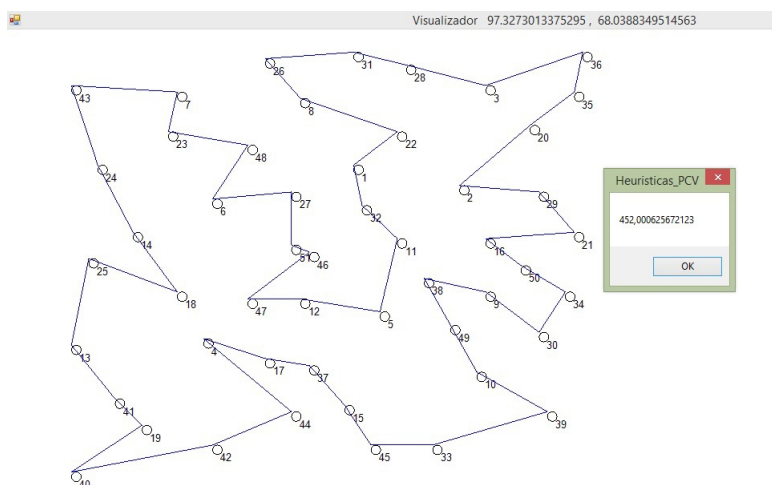


Figura 5: Resultado final do algoritmo SOM/Caos para a instância Eil51.

5.1 Considerações Finais

Neste trabalho é feita uma proposta acrescentando a característica caótica ao algoritmo heurístico clássico de mapas auto-organizáveis, na tentativa de tornar essa busca mais semelhante aos processos naturais biológicos.

O foco inicial deste trabalho é a solução do TSP devido a pelo menos três características, que são: sua grande aplicação prática, uma enorme relação com outros modelos Laporte et al, (2000) e sua grande dificuldade de solução exata. Dessa forma, a importância do modelo é indiscutível, tanto sob o aspecto prático, como o teórico.

A implementação inicial dos algoritmos foi programada na linguagem computacional Microsoft Visual Basic, versão 2010 e posteriormente transcritos para a linguagem C++. Foi desenvolvido um software utilizado como uma ferramenta capaz de testar as instâncias do TSP extraídas da biblioteca TSPLIB95 e, resolvidos pelos algoritmos clássicos de mapas auto-organizáveis e mapas auto-organizáveis com introdução de caos. O aplicativo produz uma solução em um tempo computacional que permite realizar testes para instâncias com milhares de cidades.

Os primeiros resultados mostram que o algoritmo converge para a solução, e os neurônios inicialmente dispostos em um retângulo (ou anel) envolvendo as cidades se aproximam de uma solução. O algoritmo tem um bom potencial e possibilidade de melhoria substituindo a função com característica caótica ou fazer uma combinação com heurísticas de refinamento 2-opt ou 3-opt.

Os resultados se mostraram satisfatórios para continuar utilizando a técnica para a solução de outros problemas, como por exemplo, na solução do Problema do Roteamento de Veículos (PRV). No trabalho de Gomes e Von Zuben (2001) encontra-se o pseudo-código para a utilização da rede SOM na solução do PRV. Uma adaptação neste pseudo-código permite a implementação da técnica que utiliza caos, conforme apresentado neste trabalho, para resolver o PRV.

REFERENCES

- Aihara K.; Toyoda, M. Chaotic neural networks. *Physics Letters A*, v. 144, p. 333–340, 1990.
- Bai Y.; Zhang, W. J. Z. An new self-organizing maps strategy for solving the traveling salesman problem. *Chaos, Solutions and Fractals*, v. 28, p. 1082–1089, 2006.
- Castro L. N. de ; Masutti, T. A. S. A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem. *Information Sciences*, v. 179, p. 1454–1468, 2009.
- Dantzig G.B.; Rasmer, J. The truck dispatching problem. *Management Science*, v. 6, n. 1, p. 80–91, 1959.
- Dingle A. A. ; Andreae, J. H. J. R. D. The chaotic self-organizing map. *Proc. of First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, p. 15–18, 1993.
- Fausett, L. *Fundamentals of Neural Networks*. New Jersey: Prentice Hall, 1994.
- Goldberg, D. *Genetic Algorithms in Search, Optimization and Machine Learning*. New Jersey: Menlo Park, Addison-Wesley, CA, 1986.

- Gomes L.C.T.; Von Zuben, F. Uma abordagem baseada em mapas auto-organizáveis de kohonen aplicada ao problema de roteamento de veículos. Simpósio Brasileiro de Automação Inteligente. Canela: [s.n.], 2001. v. 5.
- Haykin, S. S. Redes Neurais: Princípios e Prática. Canada: Pearson Education Inc., 2001.
- Karp, R. On the computational complexity of combinatorial problems. *Networks*, v. 5, p. 45–68, 1975.
- Kohonen, T. The self-organizing map. *Proceedings of the IEEE*, v. 78, p. 1464–1480, 1990.
- Kohonen, T. *Self-Organizing Maps*. New York: Springer, 2001.
- Laporte G.; Gendreaub, M. P. J. S. F. Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research*, v. 7, p. 285–300, 2000.
- Murty, K. *Linear and combinatorial programming*. Florida: Springer, 1985.
- Ott, E. *Chaos in Dynamical Systems*. Maryland: Cambridge University Press, 2002.
- Siqueira, P. *Uma Nova Abordagem na Resolução do Problema do Caixeiro Viajante*. Tese (Doutorado) — Universidade Federal do Paraná, 2005.