PROSIDING
Seminar Nasional MIPA 2016
Naskah diseminarkan pada 5 November 2016 dan dipublikasikan pada
http://conf.unnes.ac.id/index.php/mipa/mipa2016/schedConf/presentations

Pengaruh Parameter *Learning Rate* dan Fungsi Ketetanggaan pada Pencarian Solusi *Travelling Salesman Problem* dengan Metode Kohonen *Self-Organizing Maps*

Muhammad Sulkifly Said[1], Indah Soesanti[2], Sri Suning Kusumawardani[3]
Departement Teknik Elektro dan Teknologi Informasi Universitas Gadjah Mada
Jl. Grafika No.2 Yogyakarta, Indonesia
email: sulkifly.ti14@mail.ugm.ac.id[1], indah@mti.ugm.ac.id[2], suning@ieee.org [3]

Abstrak

Travelling salesman problem merupakan permasalahan seorang salesman dalam menentukan rute perjalan yang dilalui untuk mengunjungi kota-kota untuk menjajakan barangnya. antara satu kota dengan kota lainnya telah diketahui jaraknya dengan, salesman harus dapat menentukan jarak tempuh dengan jarak tempuh minimum dengan aturan kota yang dilalui tersebut hanya dapat dikunjungi tepat satu kali oleh salesman. Penyelesaian masalah TSP salah satunya dengan menggunakan ANN (*Artificial Neural Network*) Kohonen *Self-Organizing Map*. Algoritma ANN merupakan metode yang meniru mekanisme biologis syaraf manusia. Pemilihan metode Kohonen *Self-Organizing Map* untuk menyelesaikan permasalahan NP-*Hard* dikarenakan kemampuannya dalam melakukan pemetaan dimensi tinggi ke dimensi yang lebih rendah tanpa merubah struktur suatu data. Namun terdapat beberapa kekurangan dalam pendekatan menggunakan metode SOM, salah satunya adalah menentukan parameter *learning rate* dan fungsi *neighborhood* untuk mendapatkan solusi global optimum. Pada penelitian ini dilakukan percobaan terhadap variasi nilai *learning rate* dan fungsi *neighborhood* untuk mengetahui pengaruh nilai parameter som dalam menghasilkan suatu solusi optimal. Pengujian pada penelitian ini menggunakan data eil51 dengan jumlah 51 kota TSPLIB. Hasil penelitian menunjukkan nilai parameter *learning rate* (alpha) terbaik yaitu 0.7 dengan jumlah iterasi sebanyak 680 dengan nilai *error* paling terkecil sebesar 3% dan nilai tetha adalah 0.5 dengan total jarak yang didapatkan 463 dengan error 8%.

Abstract

Travelling salesman problem is a problem of a salesman in determining the journey traversed to visit cities to peddle goods. Among the cities with other cities with known distances, the salesman should be able to determine the distance to the minimum distance traversed by the city rules can only be visited exactly once by a salesman. Problem solving TSP one using ANN (Artificial Neural Network) Kohonen Self-Organizing Map. ANN algorithm is a method that mimics the human neural biological mechanisms. Selection methods Kohonen Self-Organizing Map to solve NP-hard problem because the ability to perform high-dimensional mapping to a lower dimension without changing the structure of the data. However, there are some shortcomings in the approach of using SOM, one of which is to determine the parameters of learning rate and neighborhood functions to get the global optimum solution. In this research, conducted experiments on the variation of the learning rate and neighborhood functions to determine the effect parameter values som in generating an optimal solution. Tests on this study uses data eil51 with the number 51 city TSPLIB. The results show the value of learning rate parameter (alpha) that is best 0.7 with the number of iterations as much as 680 with the smallest error value by 3% and tetha value is 0.5 with a total distance of 463 obtained with an error of 8%.

***Keywords*: *Learning Rate; Neighborhood function; TSP; Combinatorial Optimization*;.**

PENDAHULUAN

Optimasi kombinatorial memiliki relevansi yang besar pada berbagai aplikasi rekayasa di bidang teknik. Secara umum, salah satu persoalan optimasi kombinatorial diskrit adalah *travelling salesman problem,* TSP terkait erat dengan meningkatnya kandidat solusi secara eksponensial dengan bertambahnya sejumlah kota. Penelitian beberapa dekade terakhir belum menghasilkan metode yang tepat untuk persoalan n dalam jumlah besar, sehingga dapat disimpulkan hampir tidak ada metode yang mampu untuk menghasilkan solusi terbaik untuk kelas *NP-Hard*. Dengan demikian, pencarian solusi pada TSP adalah daerah penelitian yang cukup menantang. banyak penelitian yang diusulkan dalam beberapa tahun terakhir serta meningkatnya minat dalam metode heuristik maupun metaheuristik untuk menemukan solusi terbaik untuk persoalan tersebut.

Ketika jumlah kota semakin besar maka algoritme deterministik sudah tidak efektif lagi Untuk persoalan dengan ruang pencarian yang sangat besar, oleh karenya diusulkan metode berbasis probabilistik salah satunya adalah ANN (*Artificial Neural Network*), ANN terinspirasi oleh fenomena biologi dan alam, salah satunya adalah SOM (Self-Organizing Maps) atau bisa juga dikenal sebagai *topological preserving map,* Kohonen Feature Map berbasiskan pada *unsupervised learning* atau pembelajaran tak terawasi. Topologi feature map membentuk sekumpulan *neuron* yang melakukan proses adaptasi dan *learning*, Dengan kata lain, disetiap iterasi terjadi proses pembaruan guna mendapatkan *neuron* dengan bobot paling dekat yang dapat merepresentasikan *vector*. Upaya untuk mencegah terjadinya divergen dalam penelitian ini yaitu dengan mengamati distribusi data dan menginisialisasi bobot awal dengan pola input yang akan dikelompokkan (*cluster*). dengan cara ini maka jaringan akan dapat belajar secara berlahan mengikuti input yang ada, yang pada akhirnya didapatkan bentuk pemetaan dan jalur TSP yang optimal.

METODE

Dalam penerapan pendekatan *Self-Organizing* untuk pencarian solusi untuk *Travelling Salesman Problem* topologi jaringan harus terdapat satu lapisan unit input dan satu lapisan unit output, jaringan tersebut terdiri dari unit input dua dimensi dan M output, unit masukan hanya terdiri atas dua dimensi yaitu koordinat-x dan koordinat-y atau dapat berupa *longitude* dan *latitude* yang mencakup posisi lokasi kota-kota, namun dimensi input tidak menuntut kemungkinan dapat lebih besar. Topologi standar *Competitive Network* yang terdiri dari sekumpulan neuron dimodifikasi membentuk arsitektur cincin (*ring*), dengan memilih $Y \geq K$ dalam hal ini K=1 atau berjumlah 1 klaster dengan menghubungkan unit keluaran membentuk membentuk sebuah Ring Y_1 - Y_2 - ... Y_n - Y_1 (setiap unit keluaran memiliki dua langsung tetangga).

Topologi di bawah membentuk sebuah lingkaran dengan jari-jari sebesar R terdiri atas Y neuron, neuron diposisikan pada lingkaran dengan jari-jari=1 dengan posisi sudut 360^0 dibagi dengan total Y neuron, sehingga membentuk sebuah kurva tertutup yang nantinya akan dapat membentuk sebuah garis penghubung sekaligus sebagai urutan kota-kota yang disinggahi. Topologi diatas merepresentasikan ketetanggaan linear, untuk setiap neuron memiliki maksimal 2 tetangga. kota pada layer input akan di*training* secara acak ke layer output berdasarkan *Euclidian Distance*. Untuk setiap pemenang neuron-j adalah simpul dengan minimal jarak pada kota ke-i, dengan persamaan berikut:

$$net_j = \left|X_i - Y_j\right|,$$

$$J = Argmin_j\{net_j\}$$

dimana X_i adalah koordinat kota-i, Y_j adalah koordinat neuron-j dan |.| adalah jarak

Euclidian, Selain itu nilai *Winning* neuron dan tetangganya akan ikut diperbaharui, hingga bergerak mendekati kota-i dengan fungsi pembaruan berikut:

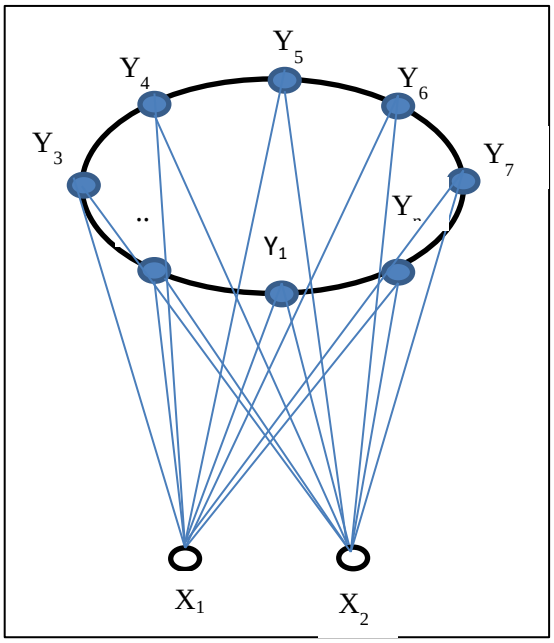
$$Y_j^{*new} = Y_j^{*old} + \alpha . f(\sigma, d)\left(X_i - Y_j^{*old}\right)$$

Dengan fungsi ketetanggaan yang digunakan adalah sebagai berikut:

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

α dan σ masing-masing adalah *learning rate* dan fungsi ketetanggaan gaussian. d=min{||j-J||, Y-||j-J||} adalah jarak kardinal sepanjang *ring* antara neuron j dan J dan ||.|| merupakan nilai absolut [8]. *Learning rate* dan fungsi ketetanggaan disini bersifat dinamis, nilainya akan berkurang selama bertambahnya proses pelatihan, nilai α umunya ditentukan dengan interval nilai [0 1]. Demikian pula dengan fungsi ketetanggaan *Gaussian* f(σ,d), fungsi ini bertugas untuk mempengaruhi tetangga neuron pemenang. Selama proses iterasi pelatihan, neuron akan cenderung bergerak mendekati kota masing-masing hingga akhirnya melekat dan membentuk sebuah path jalur. Setelah semua neuron melekat pada setiap kota akan dilakukan proses kalkulasi untuk mendapatkan urutan kota dengan cost terpendek. Variabel α menyatakan suatu konstanta belajar yang berharga antara 0 – 1. Nilai ini menunjukkan kecepatan belajar dari jaringan. Nilai yang terlalu tinggi menyebabkan jaringan menjadi tidak stabil sedangkan nilai yang terlalu kecil dapat menjadikan waktu belajar menjadi lama. Oleh karena itu pemilihan nilai α harus seoptimal mungkin agar didapatkan proses belajar yang cepat.

Jaringan kompetitif mengelompokkan input berdasarkan *similarity*, yaitu hubungan antara pola masukan X_i dan vektor bobot W_j. Dalam TSP pola input berupa koordinat dua dimensi dari kota, *Euclidian Distance* dapat digunakan sebagai ukuran kesamaan. Selain itu, tujuannya adalah untuk memetakan koordinat kota ke sebuah cincin pada unit output. Beberapa penelitian menyarankan menggunakan *Kohonen Self-Organizing Map*.

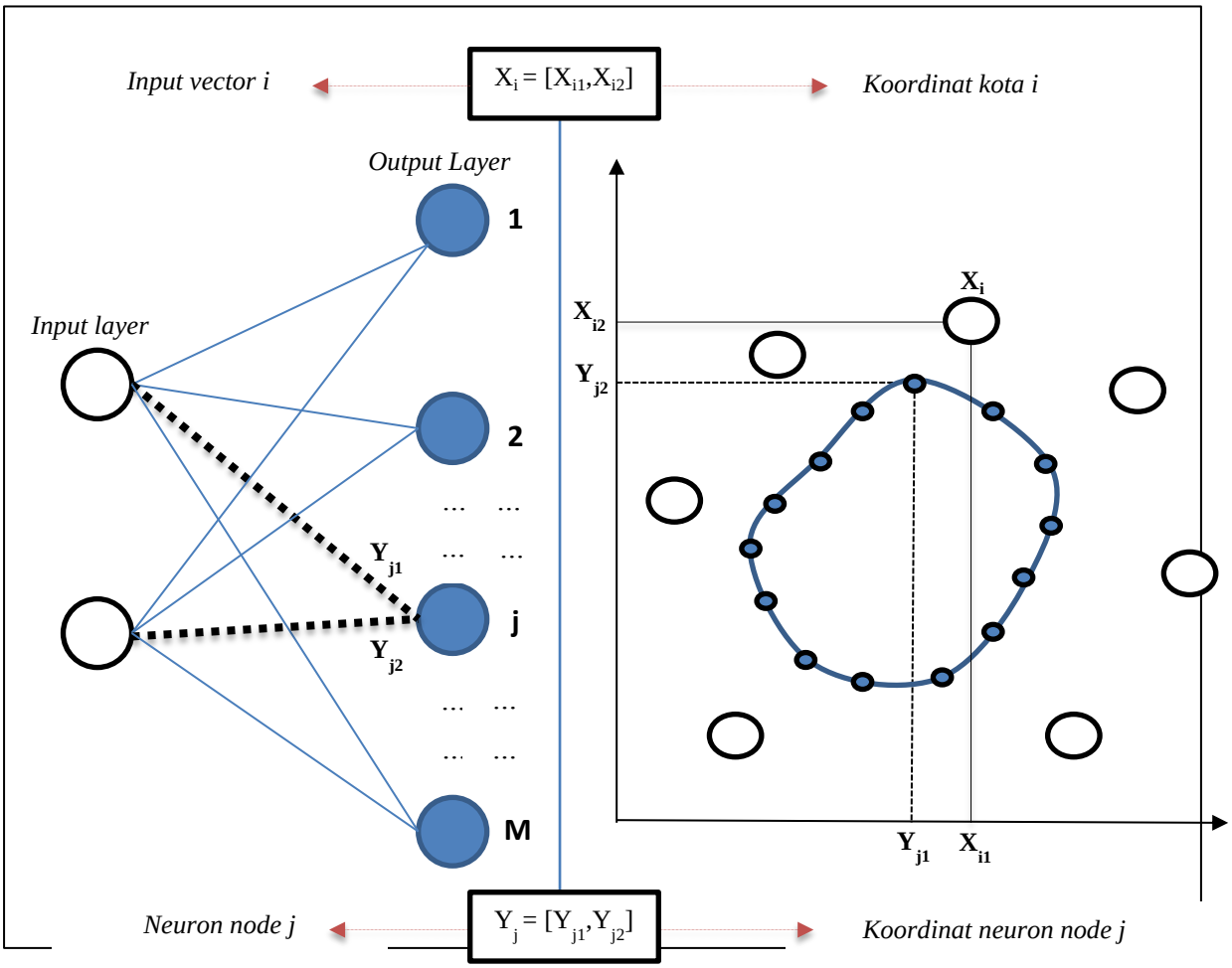


Dikarenakan kelebihan SOM dalam menghasilkan pemetaan dari ruang dimensi tinggi ke ruang dimensi yang lebih rendah, sehingga hubungan sejumlah input tercermin sebagai setia mungkin dalam output (pemetaan topologi). Akibatnya, sebuah hubungan topologi dihargai ketika bobot diperbarui. Yaitu, unit neuron pada *layer output* yang "dekat" menang Unit, menurut topologi, juga memperbarui vektor bobot mereka sehingga dapat bergerak ke arah yang sama sebagai unit menang.

Langkah pengimplementasian algoritme SOM-TSP Pada penelitian ini adalah menentukan parameter utama yang digunakan. Parameter tersebut adalah Jumlah Neuron, Alpha, Theta, Phi dan Momentum. Masing-masing parameter membutuhkan nilai-nilai untuk melengkapi perhitungan yang akan digunakan dalam proses pencarian jalur *routing*.

a. *SOM neighborhood function*

Neighborhood function atau fungsi ketetanggaan berperan dalam menentukan laju perubahan ketetanggaan di sekitar neuron pemenang. Fungsi ketetangaan mempengaruhi hasil pelatihan algoritme Self Organizing Map. Salah satu factor keberhasilan algoritme ini adalah menentukan fungsi Ketetanggaan yang tepat untuk suatu set kumpulan data beserta mengetahui distribusinya. Sama seperti *learning rate* terdapat banyak fungsi diantaranya *Bubble*, *Cut Gaussian, Epanechnikov* dan *Gaussian* yang secara luas digunakan dalam pelatihan SOM. *Bubble Function* adalah suatu fungsi konstan sementara *fungsi Gaussian* adalah fungsi menurun yang ditetapkan oleh neuron pemenang.



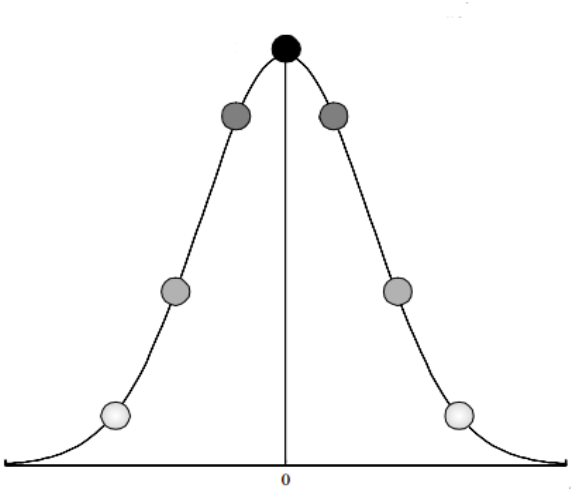
Tabel 1 Fungsi Ketetanggaan

Functions	Name	Expressions
Neighborhood functions	Bubble	$h_{cj}(t) = \max\{0{,}1 - (\sigma_t - d_{cj})\}$
	Gaussian	$h_{cj}(t) = \exp\left(\frac{-d_{cj}^2}{2\sigma_j^2}\right)$
	Cut Gaussian	$h_{cj}(t) = \exp\left(\frac{-d_{cj}^2}{2\sigma_j^2}\right) 1(\sigma_t - d_{cj})$
	Epanechnikov	$h_{cj}(t) = \max\left\{0{,}1 - (\sigma_t - d_{cj})^2\right\}$

Topologi ketetanggaan gaussian digunakan dalam proses bobot antar *neuron*, Topologi ketetanggaan gaussian maksimum berada pada titik D_{ij}=0, dalam proses training SOM, tinggi atau rendahnya kekuatan amplitudo ditandai dengan jarak antar kedekatan *neuron* terhadap koordinat kota. *winning neuron* adalah neuron dengan kekuatan amplitudo terbesar. fungsi tetangga Gaussian dapat dilihat pada gambar berikut:
Persamaan fungsi *Gaussian neighbourhood* dapat dinyatakan dengan:

$$h_{ci}(t) = exp\left(-\frac{(\| r_c - r_i \|^2)}{(0.2.\sigma^2(t))}\right)$$

r_c adalah posisi diskrit neuron pemenang dan r_i adalah posisi diskrit neuron ke-i. salah satu fungsi yang digunakan untuk memberi bobot kepada tetangga terdekat adalah gaussian function, bila sampel berada pada jarak 1 maka bobot voting diberikan kepadanya kurang lebih 0,4 jika sempel dengan pembelajaran berjarak 0,5 maka bobot yang akan diberikan kepada sampel tersebut 0.72, semakin dekat data pembelajaran sampel baru semakin besar nilai bobot yang diberikan dalam proses voting.



Gambar 2.7 Fungsi Ketetanggaan Gaussian

Tabel 2 *Learning Rate*

Learning Rate	Linear	$\alpha(t) = \alpha_0 \left(1 - \frac{t}{T}\right)$
	Power	$\alpha(t) = \alpha_0 \left(\frac{0.005}{\alpha_0}\right)^{t/T}$
	Inverse	$\alpha(t) = \alpha_0 \left(1 + \frac{100t}{T}\right)$

b. *Learning Rate Function*

Tingkat belajar atau *learning rate* merupakan parameter umum dalam proses belajar algoritme, dan mempengaruhi kecepatan di mana Jaringan Syaraf Tiruan tiba di solusi minimum. tingkat belajar adalah parameter trainng yang mengontrol ukuran vektor berat dalam belajar dari SOM. ada banyak fungsi tingkat belajar sementara linear, kebalikan dari waktu dan seri listrik sebagian besar digunakan dalam SOM [11]. linear, kebalikan dari waktu dan seri listrik didefinisikan dalam.

Nilai T adalah jumlah iterasi dan t adalah jumlah urutan iterasi saat, dalam SOM konvensional, itu adalah sangat penting bahwa fungsi tingkat belajar tertentu dan konsisten menurun dipilih. menurunkan laju pembelajaran terlalu cepat, peta tidak akan mendapatkan konvergensi dan kinerja SOM dapat mengambil langkah jatuh, dan jika terlalu lambat, prosedur akan mengambil sejumlah besar waktu untuk dilakukan.

c. *Similarity Measurement*

Dalam proses ini akan dilakukan pengukuran jarak (*distance*), di mana semakin tinggi nilai jarak antar dua buah objek, maka semakin berbeda dua objek tersebut, distance biasanya adalah ukuran dari ketidakimiripan (*Methods of Multivariate Analysis*, n.d.) untuk mengukuran ketidakmiripan dua data dengan beberapa atribut untuk setiap data digunakan kuantitas jarak (*distance*). Ada banyak model pengukuran jarak, dan yang paling sering digunakan adalah jarak *Euclidian* (Bezdek, Ehrlich, & Full, 1984). Jarak Euclidian memberikan jarak lurus antara dua buah data dengan N dimensi. Formula jarak yang lain adalah Manhattan/City Block dan Minkowsky (Singh, Yadav, & Rana, 2013). Pengukuran jarak pada ruang jarak Euclidian menggunakan formula

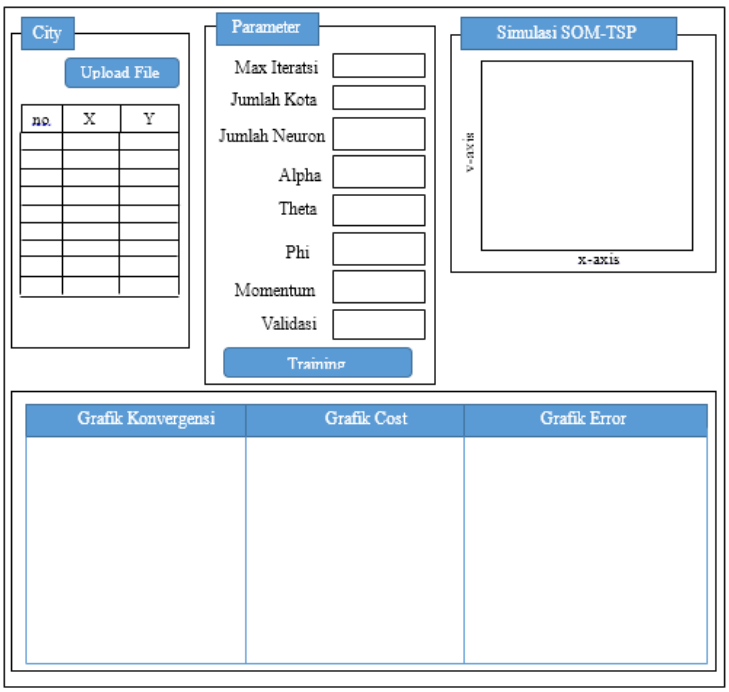
$$D(x, y) = \| x - y \|_2 = \sqrt{\sum_{j=1}^{N} | x - y |^2}$$

Dalam TSP-SOM, jarak antara kota, jarak antar kota terhadap neuron, dan jarak antar neuron dapat dihitung menggunakan persamaan *Euclidian distance*, jika jarak adalah jarak simetri, maka jarak antar kota A ke kota B sama dengan jarak dari kota B ke kota A, maka jarak total yang ditempuh adalah $d_{[A][B]} = d_{[B][A]}$ Karena tujuan yang ingin dicapai adalah jarak minimum maka persamaan tersebut dapat dinyatakan sebagai berikut:

$$f_{(\min)} = \sum_{i=1}^{n-1} d_{[i][i+1]} + d_{[n][1]}$$

HASIL DAN PEMBAHASAN

Pada design antarmuka perancangan ini, antar muka terdiri atas 4 panel utama, antara lain *panel city*, parameter, simulasi dan grafik. Pada *panel city* terdapat *combo box* "*Upload File*" yang merupakan *combo box* yang diguanakan untuk memilih *data set* yang akan diuji. hasil pemilihan pada dialog box selanjutnya akan di listkan pada *uitable* sesuai dengan banyaknya jumlah baris pada *data set* terpilih, kolom terdiri dari koordinat x dan y dalam 2 dimensi. Pada panel parameter adalah tahap inisialiasasi parameter pelatihan SOM, parameter pelatihan SOM tersebut antara lain maximum iterasi, jumlah kota, jumlah neuron, alpha, theta, phi, momentum dan validasi. *User* menginputkan banyaknya jumlah iterasi yang diinginkan, untuk *field* kota secara otomatis terisi sesuai dengan *file* TSP menggunakan perintah *set(handles.edit2,'string',CityN),* pada *field* jumlah neuron dapat diisi dengan sejumlah neuron yang diinginkan, umumnya neuron lebih besar atau sama dengan jumlah kota, ketika *user* mengklik *combo box training* maka proses simulasi neuron akan tampil pada panel simulasi SOM-TSP, pada proses ini neuron akan bergerak membentuk pola path terpendek. nilai alpha untuk parameter *learning rate*, nilai tetha untuk menentukan seberapa besar sebuah neuron mempengaruhi neuron lainnya, nilai momentum untuk menentukan seberapa banyak perubahan alpha dan phi setiap iterasinya. *training* SOM yang terdiri pemilihan kota secara *random*, BMU (*best matching unit*), *Updating* yang akan menampilkan pola *path* yang berubah seiring dengan bertambahnya jumlah iterasi. *Training* akan selesai jika jumlah iterasi = *max*. Dalam eksperimen ini akan dilakukan uji coba untuk mengetahui parameter yang optimal untuk menyelesaikan permasalahan TSP, data yang digunakan adalah eil51 dengan jumlah 51 kota dan nilai optimum adalah 426, percobaan dilakukan dengan mengganti nilai alpha, tetha, phi, momentum dengan standar 1000 kali iterasi dan 100 neuron. Jarak kota yang didapatkan melalui percobaan akan dibandingkan nilai optimum hingga dapat diketahui presentasi *error* daripada parameter masing-masing.



Tabel 1 Uji Nilai Optimal Alpha

No	Alpha	Konvergen pada iterasi ke -	Jarak kota	Error (%)
1	0.1	1000	629	47.65
2	0.3	1000	524	23
3	0.5	1000	466	9
4	0.7	680	441	3
5	0.9	666	463	8.68

Tabel 2 Uji Nilai Optimal Tetha

No	Theta	Konvergen pada iterasi ke -	Jarak kota	Error (%)
1	0.1	701	579	35.91
2	0.2	786	545	27.93
3	0.3	633	534	25.35
4	0.4	858	524	23.00
5	0.5	1000	463	8

Berdasarkan ilustrasi pada tabel 1 didapatkan parameter dengan nilai *learning rate* (alpha) terbaik yaitu 0.7 dengan jumlah iterasi sebanyak 680, dengan nilai *error* paling terkecil sebesar 3% dan nilai *learning rate* paling terburuk adalah 0.1 dan total jarak 629 dengan nilai *error* yang signifikan yaitu 47.65%. Dari penjelasan percobaan pada tabel 1 semakin rendah laju pembelajaran maka semakin lama algoritma SOM untuk menemukan solusi terbaik, misalnya pada percobaan 1 hingga 3 dengan nilai *learning rate* masing-masing 0.1, 0.3 dan 0.5 masing-masing membutuhkan seluruh iterasi sebanyak 1000 untuk mencapai konvergensi, nilai *learning rate* yang rendah juga tidak menjamin ditemukannya solusi terbaik namun sebaliknya pencarian solusi dapat terjebak pada solusi *local minima* hasil percobaan 1 hingga 3 secara berturut-turut menghasilkan nilai *error* yang cukup besar yaitu 47%, 23% dan 9%, dari tabel diatas penentuan pemilihan *learning rate* yang tepat sangat menentukan ditemukannya solusi *global minima* dengan sejumlah iterasi tertentu untuk mencapai titik konvergensi.

Pada penjelasan sebelumnya telah diketahui bahwa parameter nilai tetha berfungsi untuk mempengaruhi *update* ketetanggaan neuron di sekitar neuron pemenang, data percobaan pada tabel 2 di dapatkan nilai tetha terbaik pada percobaan ke-5 dengan nilai tetha 0.5 beserta total jarak yang didapatkan 463 dengan error 8% yang mengalami penurunan secara signifikan setelah dilakukan percobaan secara berturut-turut pada percobaan-1 hingga percobaan ke-4. Berdasarkan percobaan nilai tetha semakin besar nilai tetha yang diberikan maka semakin baik pula hasil yang didapatkan.

KESIMPULAN

Dalam pengimplementasi pemecahan persoalan TSP dengan menggunakan metode *Kohonen Self-Organizing Map* yang diuji pada data eil51 berjumlah 51 kota, dengan proses iterasi sebanyak 1000 dan neuron sebanyak 100, maka dapat diambil kesimpulan bahwa nilai parameter alfa dan tetha cukup mempengaruhi konvergensi dan hasil *rute* yang dihasilkan. pada proses pengujian nilai alfa, didapatkan hasil terbaik terbaik dengan nilai alfa = 0.7 dengan tingkat *error* sebesar 3% dan pada pengujian nilai tetha, didapatkan nilai tetha terbaik adalah = 0.7 dengan tingkat *error* 8%.

DAFTAR PUSTAKA

Bezdek, J. C., Ehrlich, R., & Full, W. (1984). FCM: *The fuzzy c-means clustering algorithm*. *Computers & Geosciences*, *10*(2–3), 191–203. http://doi.org/10.1016/0098-3004(84)90020-7

Methods of Multivariate Analysis. (n.d.).

Singh, A., Yadav, A., & Rana, A. (2013). *K-means with Three different Distance Metrics*. *International Journal of Computer Applications*, *67*(10), 13–17. http://doi.org/10.5120/11430-6785

Ã, J. Y., Yang, G., Zhang, Z., & Tang, Z. (2009). *Neurocomputing An improved elastic net method for traveling salesman problem*, *72*, 1329–1335. http://doi.org/10.1016/j.neucom.2008.09.011

Ã, P. H. S., Teresinha, M., & Steiner, A. (2007). *A new approach to solve the traveling salesman problem*, *70*, 1013–1021. http://doi.org/10.1016/j.neucom.2006.03.013

Applegate, D. L., Bixby, R. E., Chvátal, V., & Cook, W. J. (n.d.). *No Title*.

Aras, N., & Oommen, B. J. (1999). *The Kohonen network incorporating explicit statistics and its application to the travelling salesman problem* &, *12*, 1273–1284.

Brocki, L. (n.d.). *Kohonen Self-Organizing Map for the Traveling Salesperson Problem*, 2–8.

Chen, J. (2007). *Solving the Traveling Salesman Problem Using Elastic Net Integrate with SOM*, 2234–2237.

Kohonen, T., Barna, G., Chrisley, R., & Science, I. (n.d.). *Statistical Pattern Recognition with Neural Networks*: Benchmarking Studies, (x), 61–68.

Networks, W. N., Now, W., Are, H., & Networks, N. (n.d.). *No Title*.

Paper, I., Introduction, I., Models, N., & Maps, S. (1990). *The Self-organizing Map*, *78*(9), 1464–1480.