Emerging Markets Queries in Finance and Business

Artificial Intelligence in the path planning optimization of mobile agent navigation

Sándor T. Brassai[a,b,]*, Barna Iantovics[b], Călin Enăchescu[a,b]

[a]*Petru Maior University of Tirgu Mures, str. Nicolae Iorga, nr.1, Tirgu Mures 540088, Romania*
[b]*Sapientia Hungarian University of Transilvania, O.p 9, C.p.4, Tirgu Mures, 540485 , Romania*

Abstract

Many difficult problem solving require computational intelligence. One of the major directions in artificial intelligence consists in the development of efficient computational intelligence algorithms, like: evolutionary algorithms, and neural networks. Systems, that operate in isolation or cooperate with each other, like mobile robots could use computational intelligence algorithms for different problems/tasks solving, however in their behavior could emerge an intelligence called system's intelligence, intelligence of a system. The traveling salesman problem TSP has a large application area. It is a well-known business problem. Maximum benefits TSP, price collecting TSP have a large number of economic applications. TSP is also used in the transport logic Raja, 2012. It also has a wide range of applicability in the mobile robotic agent path planning optimization. In this paper a mobile robotic agent's path planning will be discussed, using unsupervised neural networks for the TSP solving, and from the TSP results the finding of a closely optimal path between two points in the agent's working area. In the paper a modification of the criteria function of the winner neuron selection will also be presented. At the end of the paper measurement results will be presented.

Keywords: Mobile agent path planning, travelling salesman problem, self organizing map ;

* Corresponding author. Tel.: +40 265 206 210; fax: +40 265 206 211.
E-mail address: tiha@ms.sapientia.ro.

1. Introduction

Intelligent agents and multiagent systems represent a relatively novel field of artificial intelligence applicable for efficient solving of different problems Weiss, 2000; Ferber, 1999; Iantovics, 2007. The intelligence of the agents cannot be defined uniqualy based on the large variety of agents and the problems that they have to solve Weiss, 2000; Ferber, 1999; Iantovics, 2007. The intelligence of a system must give improvements in problems solving, like: efficient problem solving; flexible problem solving; solving difficult problem; efficient solving of large numbers of problems etc. In our research we consider the intelligence of the systems the "intelligence" in the problems solving based on the use of a computational intelligence algorithm. Important research directions are represented by mobile agents that could be software mobile agents Kirn, 2003; Kun, 2003, or robotic mobile agents Sgorbissa, 2012; López, 2007.

Robotic mobile agents have a wide range of applicability in different areas of life Wang, 2005; Kim, 2003; Chirillo, 2012, and Leitão, 2012. A large number of scientists are working on finding new solutions for different subsections of mobile agent applications such as navigation, localization, optimal path planning, object detection, movement of the robot. In our research, as mobile agent a mobile robot is considered. In this paper we will focus on path planning optimization of the mobile agent using a computational intelligence algorithm.

We will focus on discussing solutions for:
- TSP problem and a modified TSP when the agent does not have to get back to the starting point.
- Finding a closely optimal path from the resolved TSP.

For the TSP solving a Kohonen map was used with a proposed cost function in the winner neuron's selection. In the following section, the TSP problem , the artificial neural network structure and network training, results with the resolved TSP with Kohonen map and optimization of path planning between a starting node and a target node on the map will be presented.

2. TSP problem formulation

The following two tasks for a mobile agent were taken into consideration: In the first form the mobile agent has to supervise an area and to move back to the starting node. In the second form a mobile agent has to clean an area starting from one node and finishing the work in the end node. The second application can be used if we have a large area discomposed in subareas, and for each subarea the entering and outcoming nodes are defined. The agent has to visit each node, marked in the figure with a white square, avoiding the obstacles and move back to the starting respectively to finish the task in the ending node. Each node must be visited only one time.
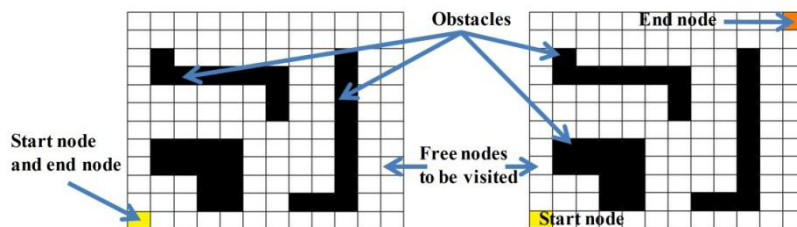

Fig. 1. (a) represents the map, the agent finishes the work at the starting node; (b) the agent starts the activity form the starting node and has to finish it in the end node

2.1. Different solutions for TSP solving

Many types of methods for solving TSP were developed: methods of total enumeration, branches and bounds, efficient algorithm of Clark and Wright, ant colony optimization Zhao, 2011; Pintea, 2007; Hui, 2012, particle swarm optimization algorithm Jiann-Horng, 2009, genetic algorithms Hui, 2012, LP-based solution methods for the asymmetric TSP Melkonian, 2007, approximative graph pyramid solution of the E-TSP Haxhimusa, 2009. Lust and Jaszkiewicz, 2010, discuss the Two-Phase Pareto Local Search method to speed up the heuristic resolution of the biobjective traveling salesman problem.

There are also several artificial intelligence based solutions for solving the TSP and in mobile robots path planning such as genetic algorithms Kao-Thing, 2007; Nouara, 2011, solutions based on artificial recurrent networks Yogita, 2012, Hopfield neural network Hui, 2012, Lotka–Volterra neural networks Li et.al., 2009, fuzzy clustering algorithm etc.

3. SOM structure for the TSP problem

3.1. TSP solving

The self organizing map was described as an artificial neural network by Teuvo Kohonen and often is called the Kohonen map. The Kohonen map is an artificial neural network with an unsupervised training algorithm Kohonen, 2011. Compared to the multi-layer feedforward network, the output of the Kohonen map is processed as a linear combination of the network weights and the network input. The general structure of the network is presented in the following Fig. 2 (a), (b), and (c). Several type of topology is used for artificial neurons mapping in Kohonen network
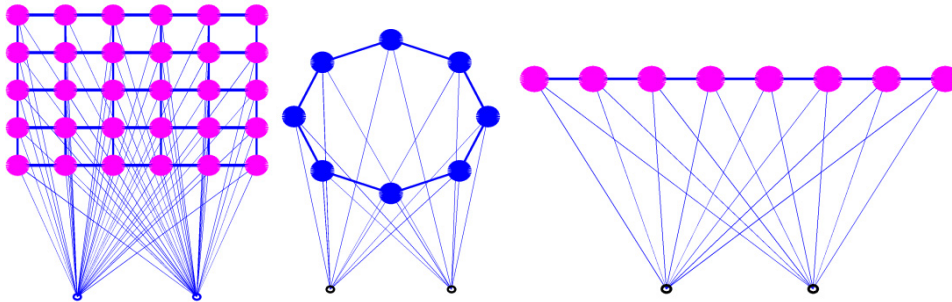

Fig. 2. (a) grid topology; (b) ring topology, (c) neurons placed along a line

The network output is composed in general accordingly to (1), but according to specific optimization applications, the network output can be processed using a cost function.

$$y_i = \sum_{j=1}^{P} w_{ij} \cdot x_j \qquad i = 1..K \tag{1}$$

$$y_i = \left\| \overline{w}_i - x \right\| = \sqrt{\sum_{j=1}^{P} \left(w_{i,j} - x_j \right)^2} \quad i = 1..K \tag{2}$$

$$y_i = \left\| \overline{w}_i - x \right\| = \alpha \sqrt{\sum_{j=1}^{P} \left(w_{i,j} - x_j \right)^2} + \left(1 - \alpha\right) \left(\sqrt{\sum_{j=1}^{P} \left(s_j - x_j \right)^2} + \sqrt{\sum_{j=1}^{P} \left(t_j - x_j \right)^2} \right), \quad i = 1..K \quad (3)$$

$$w_{i,j}[k+1] = w_{i,j}[k] + \mu \Phi\left(r, r^*\right)\left(x_j - w_{i,j}[k]\right) \quad i = 1..K, \quad j = 1..P \qquad (4)$$

Where y_i represents the network's i-th output, w_{ij}-the network weight between the i-th processing element and the j-th input, K the number of the network's processing elements respectively P the number of inputs of the network. The self-organizing map uses a neighborhood function to preserve the topological properties of the input space. The neurons of the network are placed based on a topology. The topology can be linear, hexagonal, a two or three-dimensional grid type or also a random type topology Fig. 2.

The definition, selection, of the most suitable topology specific to the input space is very important for resolving the task. For the self organizing map training, generally, an unsupervised learning algorithm is used. After the processing of the network output, based on a criteria function, the winner processing element will be defined. The weights of the winner processing element and the ones of the processing elements in the neighborhood of the winner artificial neuron are trained based on the Hebb or anti Hebb rule.

During the training process of the network, the neurons are organized according to the topology so that neurons with similar weights will be arranged closely to eachother according to the topology. The Mexican hat is frequently used as a neighborhood function, but several times the Gaussian function is considered.

In this paper multiple cost functions have been tested according to (2) and (3). In (2) the Euclidian distance is processed between the network input and the weights of the network. The network with the minimal value will be selected as the winner neuron. The cost function (3) was extended with a penalization member. With parameter α can be determined the extent to which prevails one or the other part of the cost function. Two other types of cost functions have been tested. The equations are not presented here, but can be deduced very simply by changing the Euclidian norm with the Manhattan norm.

For resolving the TSP problem with the self organizing network, the structure of the network is presented in fig. 2(b). The network neurons represent the nodes which the agent visits. A topology has to be defined so that it corresponds to the expectations of the TSP task. Each neuron can have two and only two neighboring nodes. One is from which the agent arrives and the other is where the agent will be in the next step. If the agent needs to get back to the starting point, this means that the first and the last neuron are the same. It can easily be concluded that this is equivalent to a ring-type topology.

As mentioned, the neuron represents a node where the agent arrives, and the weights of the network represent the position of the node on to the navigation map of the agent. The network structure for the TSP is presented in Fig. 2(b). The TSP is resolved based on the classification of the inputs of the network. As can be concluded from the normalized Hebb rule used for network training (4), weights are shifted towards the network current inputs. The neurons will be rearranged corresponding to the ring topology.

The neighborhood function defines/influences the neurons for which the weights will be updated. The neighborhood function value for the neuron close to the winner has a significant value close to 1 and the value of the neighborhood function for neurons far from the winner will have an insignificant value close to zero, and will block the update of weights for these neurons. As neighborhood function the Gaussian function was used, with the center point of the function at the winner neuron index on the topology map. The r and r^* represent the neuron positions on the topology map for the neuron for which the weight update is processed, respectively for the winner neuron (4).

3.2. Modified TSP solving

A modified task of the TSP considered in this paper is when the agent does not have to move back to the starting point, the target point is previously defined on the map, where the agent has to finish the task Fig. 1(b). The question is to solve this problem using the self organizing map.

If the agent does not have to move back to the starting point, the ring type topology does not function. For solving the problem, a one dimensional topology is proposed to be used. The neurons are placed along a line Fig 2.(c).

As resulting from the measurements, the agent doesn't move back to the target point. Unlike as it was expected, the starting and destination node of the solution do not coincide with the initially specified starting and target nodes. From the path resulted with the self-organizing map the starting and destination points should be searched for. The result of the self organizing map is the weight vector which contains the position of the nodes. By finding the starting node and the target node index from the vector, the path from the starting node to the target node is solved, and from the starting node all of the nodes can be reached. In case of the ring type topology two paths exist to the target node. In case of the linear topology only a single path exists.

These paths are not optimal, but with a simple searching algorithm a closely optimal solution can be found. In this application the cost function is the length of the path, the number of nodes. The problem to be resolved is that the first neuron weights to converge to the starting node coordinates respectively the last neuron weights to converge to the destination node coordinates. The solution lies in the training algorithm. The proposed solution in this paper is to not teach the first and last neuron weights and overwriting the first and last neuron weights with the starting and destination node coordinates. This simple amendment will result in that, that the starting and destination nodes of the solution will coincide with the initially specified start and target nodes.

3.3.Optimization for optimal path finding

All three forms of the network resolve the task to reach all the nodes on the map. The route from the starting node to the target node is resolved, but the solution is not optimal or close to the optimal. The proposed algorithm for finding a closely optimal route between the starting and the ending node is presented in the following:

- Elimination of the duplicate nodes. If the number of neurons is equal to the number of nodes on the map, the network does not find the solution, because a set of neurons will not be active during the network training process.
- Finding the indices from the resulted weight vector with the starting and ending nodes. The vector values which correspond to node coordinates on the map are compared with start and end node positions.
- Finding the nodes for which a neighborhood node with a lower cost to the target node exists.
- Calculating the cost from the start to the end node and from the neighborhood nodes to the end node for each node found in the previous step and storing the results in a table with the following fields: current node index, neighborhood node index, and the calculated cost.
- Ordering the table in ascending order according to the cost column, column number three
- Deleting the rows from the table for which the detected section with a higher cost overlaps the section with a lower cost.
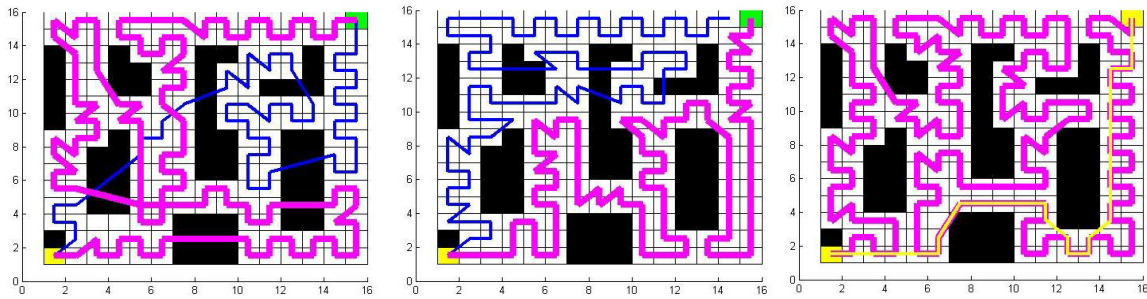- Finally extracting the closely optimal path from the start to the end point.

Fig. 3. (a) network training results using a ring type topology; (b) results using linear type placement, (c) results using linear type placement with overwriting of the first and last neuron weights with the start and end node coordinates

In the following figures the results of the self-organizing map training for solving the TSP and modified TSP are presented. In Fig 3.(a) as network topology a ring-type topology was used. The neurons were positioned on a circle. On Fig 3.(b) and 3.(c) a linear placement was used and in Fig 3. (c) the first and last neuron's weights were forced (overwritten) with the start and end node coordinates, as the result is how it was expected. In the Fig.4 (a) (b) (c) the evolution of the training process is presented in different training cycles. A neighbourhood degree was gradually decreased during the training cycles. In the first phase a large scale rearrangement of the neuron was allowed, finally reducing to the immediate vicinity of the winner.
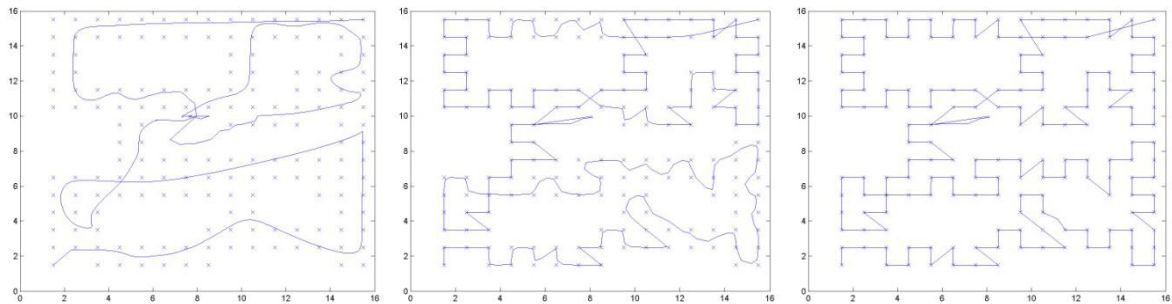

Fig. 4. Network training result: (a)-training cycle nr. 100, (b) training cycle nr.200, (c) training cycle nr 400

4. Conclusions

In this paper we have proposed an intelligent robotic mobile agent. The intelligence of the robotic agent is considered based on the use of a computational intelligence algorithm used for the optimization of the path planning. The robotic mobile agent uses an unsupervised neural networks for the TSP solving, and from the TSP results the finding of a closely optimal path between two points in the agent's working area. The criteria functions mentioned work well during the teaching. If the penalty part of cost function prevails, the number of learning cycles increases. At the tuning phase it must be taken into account that the teaching is started with neighboring values high enough. If the neighboring degree is low, most of the nodes will not be part of the solution. The calculation of distance with the Euclidean and Manhattan norms should be completed with the infinite norm and tested in the future. Using a simplified cost function, the complexity of the output processing is reduced, reducing the network output processing time.

Future research direction is the application of the self organizing map in economy and finance.

Acknowledgements

This paper is a result of the project `Transnational Network for Integrated Management of Postdoctoral Research in Communicating Sciences. Institutional building postdoctoral school and fellowships program CommScie" - POSDRU/89/1.5/S/63663, financed under the Sectorial Operational Programme Human Resources Development 2007-2013.

References

Cirillo, M., Karlsson, L., Saffiotti, A., 2012, Human-aware planning for robots embedded in ambient ecologies Original Research Article, Pervasive and Mobile Computing, Volume 8, Issue 4, p. 542-561.

Exnar F,ilip, Machac O.takar, 2011, The Travelling Salesman Problem and its Application in Logistic Practice, Wseas Transactions On Business And Economics, The Travelling Salesman Problem and its Application in Logistic Practice, Issue 4, Volume 8, p. 163-173.

Ferber, J. 1999, Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence, Addison Wesley, 1999.

Haxhimusa, Y., Kropatsch ,W. G., Pizlo Z., Ion A., 2009, Approximative graph pyramid solution of the E-TSP, Image and Vision Computing Volume 27, Issue 7, p. 887–896.

Hui W. 2012, Comparison of several intelligent algorithms for solvingTSP problem in industrial engineering, Information Engineering and Complexity Science - Part II, Volume 4, p. 226–235.

Iantovics, B., Chira, C. and Dumitrescu, D. 2007, Principles of the Intelligent Agents, Casa Cartii de Stiinta Press, Cluj-Napoca, 2007.

Jiann-Horng, L., Li-Ren, H., 2009, Chaotic bee swarm optimization algorithmfor path planning of mobile robots, Proceedings of the 10th WSEASinternational conference on evolutionary computing, Prague, CzechRepublic, pp. 84-89.

Kao-Ting H., Jing-Sin, L., Yau-Zen, C., 2007, A comparative study of smooth path planning for a mobile robot by evolutionary multi-objective optimization, Proceedings of the 2007 IEEE International Symposium on Computational Intelligence in Robotics and Automation, p. 254 – 259.

Kim, D.-H., Kim J.-H., 2003, A real-time limit-cycle navigation method for fast mobile robots and its application to robot soccer, ,Robotics and Autonomous Systems, Volume 42, Issue 1, p. 17–30.

Kirn, St., 2003, Ubiquitous Healthcare: The OnkoNet Mobile Agents Architecture, Proceedings of the conference: Objects, Components, Architectures, Services, and Applications for a Networked World, Netobjectdays NODe 2002, Lecture Notes in Computer Science, vol.2591, pp.265-277, 2003.

Kohonen, T., 2001, Self-Organizing Maps, Third Edition, Springer

Kun, Y., Galis, A., 2003,. Rule-driven Mobile Intelligent Agents for Real-time Configuration of IP Networks. Proceedings of the Knowledge Based Intelligent Information & Engineering Systems (KES'2003). University of Oxford, United Kingdom, p. 921-928.

Leitão, P., Barbosa, J., Trentesaux, D., 2012, Bio-inspired multi-agent systems for reconfigurable manufacturing systems Original Research Article, Engineering Applications of Artificial Intelligence, Volume 25, Issue 5, p. 934-944.

Li, M.,Yi, Z., Zhu, M., 2009, SolvingTSP by using Lotka–Volterra neural networks, Neurocomputing, Volume 72, Issues 16–18, p. 3873–3880.

López B., Salvi, J 2007, A multi-agent architecture with cooperative fuzzy control for a mobile robot Original Research, ArticleRobotics and Autonomous Systems, Volume 55, Issue 12, p. 881-891

Lust, T., Jaszkiewicz, A., 2010, Speed-up techniques for solving large-scale biobjective TSP, Computers & Operations Research, Volume 37, Issue 3, p. 521–533.

Melkonian, V., 2007, LP-based solution methods for the asymmetric TSP, Information Processing Letters, Volume 101, Issue 6, p. 233–238

Nouara A., Mohamed C., 2011, Mobile Robots Path Planning using Genetic Algorithms, ICAS 2011 : The Seventh International Conference on Autonomic and Autonomous Systems, pp 111-115.

Pintea, C.amelia M. Pintea, Pop, P.etric C. Pop, 2007, Dumitrescu, D., 2007, Dumitrescu: An Ant-based Technique for the Dynamic Generalized Travelling, Salesman Problem, Proceedings of the 7th WSEAS International Conference on Systems Theory and Scientific Computation, Athens, Greece, August 24- 26, 2007, pp. 255-259.

Raja, RP., and Pugazhenthi, S. Pugazhenthi, 2012, Optimal path planning of mobile robots: A review, nternational Journal of Physical Sciences , pp. 1314 – 1320.

Sgorbissa A., Zaccaria, R. 2012, Planning and obstacle avoidance in mobile robotics, Robotics and Autonomous Systems, Volume 60, Issue 4, April 2012, p. 628-638

Sivaram, Kumar M.P., Rajasekaran, S., 2012, A Neural Network based Path Planning Algorithm for Extinguishing Forest Fires, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 2, No 2, March, p. 563-568 .

Wang, M., Liu, J. N.K., 2005, Interactive control for Internet-based mobile robot teleoperation, Robotics and Autonomous Systems, Volume 52, Issues 2–3, 31, p 160–179.

Weiss, G. (Ed.), 2000, Multiagent Systems A Modern Approach to Distributed Artificial Intelligence, MIT Press Cambridge, Massachusetts London, England, 2000.

Yogita G., Kusum G., 2012, Artificial Intelligence in Robot Path Planning, International Journal of Soft Computing and Engineering (IJSCE), Volume-2, Issue-2, May, p. 471-474.

Zhao, J., Fu, X., Jiang Y., 2011, An Improved Ant Colony Optimization Algorithm for Mobile Robot Path Planning, Intelligent Systems and Applications (ISA), 3rd International Workshop on, 28-29 May 2011, p. 1-4.