A STUDY AND IMPLEMENTATION OF THE TRANSIT ROUTE NETWORK DESIGN PROBLEM FOR A REALISTIC URBAN CASE

M. Kalochristianakis[1] and D. Kosmopoulos[2]

[1]Department of Informatics Engineering,
Technological Educational Institution of Crete, Heraklion, Greece
kalohr@staff.teicrete.gr
[2]Department of Cultural Heritage Management and New Technologies,
University of Patras, Agrinio
dkosmo@upatras.gr

ABSTRACT

The design of public transportation networks presupposes solving optimization problems, involving various parameters such as the proper mathematical description of networks, the algorithmic approach to apply, and also the consideration of real-world, practical characteristics such as the types of vehicles in the network, the frequencies of routes, demand, possible limitations of route capacities, travel decisions made by passengers, the environmental footprint of the system, the available bus technologies, besides others. The current paper presents the progress of the work that aims to study the design of a municipal public transportation system that employs middleware technologies and geographic information services in order to produce practical, realistic results. The system employs novel optimization approaches such as the particle swarm algorithms and also considers various environmental parameters such as the use of electric vehicles and the emissions of conventional ones.

KEYWORDS

Public transport network, environmental optimization, particle swarm optimization, geographic informational systems, middleware

1. INTRODUCTION

The problem of optimizing the use of resources with respect to the environmental impact has been an area of focus during the last decade [1] [2]. The design of a public transportation network is a complex optimization problem, which involves a variety of design parameters (route structure, frequencies, vehicle types, etc) and assumptions on demand patterns, travel behavior and so on. Indeed, the associated Transit Route Network Design Problem (TRNDP) has been a topic of interest for over 40 years. The combinatorial nature of the TRNDP and the difficulty to formulate it analytically have resulted numerical optimization as the primary means of approaching the solutions over the last years. A review of the recent literature exhibits a variety

Jan Zizka et al. (Eds) : ICAITA, SAI, CDKP, Signal, NCO - 2015
pp. 59–65, 2015. © CS & IT-CSCP 2015 DOI : 10.5121/csit.2015.51506

of relevant techniques that consider routes, frequencies and other network parameters, based on preset objective functions, which are to be optimized. Widely used approaches include Genetic Algorithms (GA) [3], Simulated Annealing [4] and Ant Colony Optimization [5] besides others.

The Particle Swarm Optimization (PSO) is one of the most effective evolutionary algorithms inspired from social behavior of animals [6]. Its simplicity and efficiency makes this algorithm very popular. Due to these advantages, the PSO algorithm has been applied to many domains such as medical diagnosis, grid scheduling, robot path planning and computer vision. This algorithm is capable of solving problems with continuous search spaces, while some problems have discrete search spaces. The binary version of PSO (BPSO) was proposed by [7]. The TRNDP belongs to the discrete problems and probably this is the reason that the PSO algorithm has not been applied to this problem so far.

The rest of the paper presents the formulation of the TRNDP problem so that PSO optimization procedures can be used to approach its solution. The paper is structured as follows: section 2 analyses the formulation of the problem. Section 3 describes the component architecture of the proposed framework. Section 4 presents the conclusions of this work and the future perspective.

2. PROBLEM FORMULATION

This work has been based on the assumptions that there is a fixed number of S bus stops, a fixed number of bus lines L and that the bus lines have a maximum number of s bus stops. The solution is represented by a binary two dimensional matrix of L rows and s columns. The l-th row represents the l-th bus line. A "1" in position (l, σ) represents that the l-th bus line goes through the σ-th bus stop, while a "0" represents that the bus line does not include the bus 2stop. The solution must be in vector form, therefore, we vectorize the 2D matrix to formulate the hereafter mentioned as "*LN*" vector with $L \times s$ elements. To the previous vector we also have to append bits to encode bus frequencies per line (number of bits depends on what is the maximum bus frequency) hereafter denoted as "*f*" and whether the line is operated by electric or conventional bus hereafter denoted as "*G*". The following step is to minimize the objective function given by:

$$\begin{gathered} \min Z = w_1 D_u(\bar{L}\bar{N},\bar{f}) + w_2 T(\bar{L}\bar{N},\bar{f}) \\ + w_3 e(\bar{L}\bar{N},\bar{f},\bar{G}) + w_3 N_{cs}(\bar{G}) + w_5 V_c(\bar{L}\bar{N},\bar{f},\bar{G}) + \\ w_6 D_e(\bar{L}\bar{N},\bar{f},\bar{G}) \end{gathered} \quad (1)$$

where D_u is the unsatisfied passenger demand (not served under maximum transfers), T the average travel time, e the pollution emissions N_{cs} the number of charging stations, V_c the required number of conventional vehicles and V_e the required number of electric vehicles. The weights w_1 - w_6 are defined according to the policy we want to implement or according to values that can be statistically estimated. All the above quantities are straightforward to compute given *LN*, *F*, *G*. However, the question is: given the solution vector how do we define the sequence of the bus stops? Clearly the solution vector does not really capture the sequence in which the bus stops are visited. This is done deliberately and is one of our major contributions, because we significantly reduce the solution space. E.g. for S = 50 and s = 10 the number of possible permutations in which we are seeking optimum is $\sim 10^{16}$, while if we ignore the permutations as in our method this reduces the search space to $\sim 10^{11}$.

The answer is given by assuming that the next bus stop is the one closest to the current one. In other words we need to define the path that covers all the bus stops and at the same time has the minimum possible length. The answer to this problem is given by the Hamiltonian path, which solves exactly this problem [8].

3. The Binary PSO Algorithm

The binary version of PSO (BPSO) was proposed by [7]. The continuous and binary versions of PSO are distinguished by two different components: the transfer function and the different position updating procedure. The transfer function is used to map a continuous search space to a binary one, and the updating process is designed to switch positions of particles between 0 and 1 in binary search spaces. Several solutions have been proposed to the problem of getting trapped in local minima, e.g., [10], [11]. In [12], two different families of transfer functions, v- shaped and s-shaped were investigated. Let's start from the continuous PSO. Each particle *i* at time *t* corresponds to a single solution $x_i(t)$. To evolve towards a better solution the particle has to consider the current position, the current velocity $v_i(t)$, the distance to their personal best solution, *pbest*, and the distance to the global best solution, *gbest*. This is formulated as follows:

$$v_i(t+1) = w * v_i(t) + c_1 * r_1 * (pbest - x_i(t)) + c_2 * r_2 * (gbest - x_i(t)) \qquad (2)$$

where w is a weighting function, r_1, $r_2 \in [0,1]$ are random numbers and c_1, c_2 are acceleration coefficients. In the next iteration the particle will evolve to:

$$= x_i(t) + v_i(t+1) \qquad (3)$$

In binary space, due to dealing with only two numbers (''0'' and ''1''), the position updating process cannot be performed using eq. (3). Therefore, another definition of velocities is needed for changing positions from ''0'' to ''1'' or vice versa. This can be done by redefining the velocity to be the probability of a bit taking the value *0* or *1*. A sigmoid transfer function as in eq. (4) was employed in [1] to transform all real values of velocities to probability values in the interval [0,1].

$$T(\ v_i^k(t)\ ) = \frac{1}{1 + e^{-\ v_i^k(t)}} \qquad (4)$$

where $v_i^{\,k}(t\ )$ indicates the k-th dimension of the velocity vector. Then the position vectors are updated according to the following:

$$x_i^k(t+1) = \begin{cases} 0, & if\ \ r < T(v_i^k(t+1)) \\ 1, & if\ \ r \geq T(v_i^k(t+1)) \end{cases} \qquad (5)$$

where r is a random number in the interval [0, 1]. Variations of this strategy have been proposed in [10], [11], [12].

4. COMPONENT ARCHITECTURE AND ALGORITHMIC APPROACH

The design of our case study is based on the combination of presentation technologies, middleware and computational analysis, namely: HTML, Javascript, Google Maps API at the presentation tier, Hypertext Preprocessor (PHP), the known dynamic programming language for the middleware and Octave / Matlab for the computational analysis back-end. More specifically, the systems includes a graphical web interface capable of displaying the graph of the problem realistically and in real time through the programming interface of Google Maps engine management. The graph presented by Google Maps is processed by means of a computational analysis module which implement the TRNDP solver based on the Octave and Matlab environments. Middleware logic, is capable to handle requests directed towards the graphical interface, direct them to the computational analysis module and return results in appropriate form to be presented by the maps presentation engine.

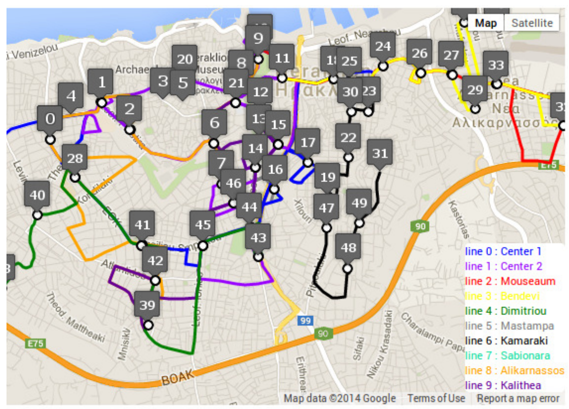
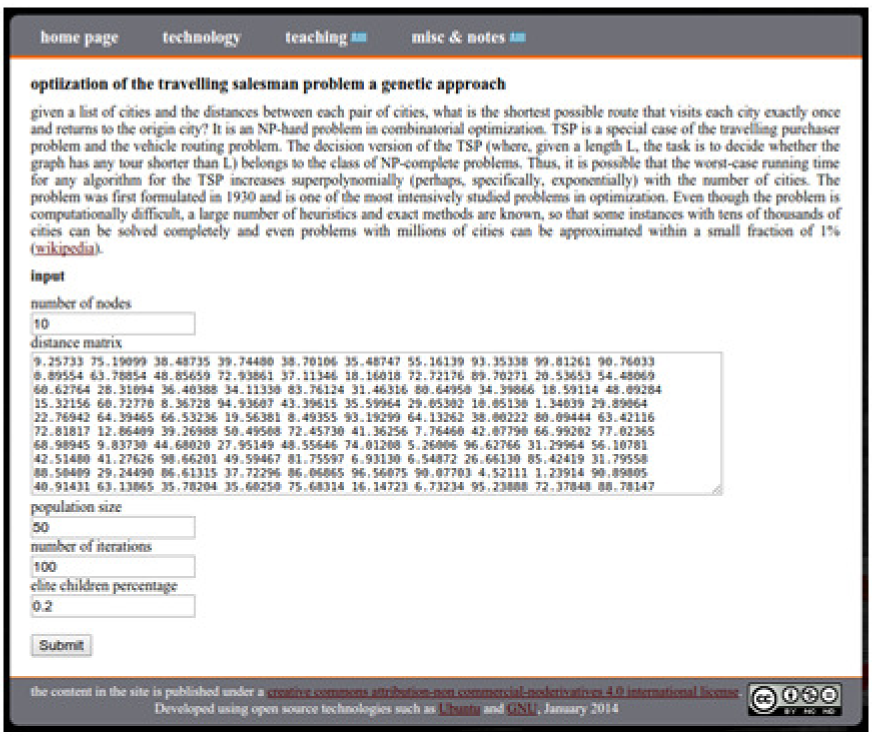


Fig. 1: an initial configuration of the system that is, a realistic selection of graph nodes and routes can be displayed on Google Maps and also information about distances and traffic can be retrieved and passed to the computational analysis system.

The Google Maps map management and presentation engine offers programming interfaces that are compatible with various programming languages. For the purposes of our work, we took advantage of the Javascript programming [9]. Necessary conditions for the use of the interface has been the knowledge of web technologies and principles of object-oriented programming design mode. This platform was chosen because it dominates the market of GIS and provides free, stable and reliable access. Google maps also offer interesting features such as real time traffic support, carbon dioxide emissions estimations, beside others. The use of the programming interfaces of the platform service is offered by means of subscription and the acquisition of appropriate application programming interface (API) keys that allow the service to monitor usage. Typical facilities include the DistanceMatrixService offering distance calculation service between start and destination nodes, DirectionsService offering directional calculation service between one or more locations, DirectionsRoute service offering route calculation between departure and destination which contains the sections of the route, among others, Map objects capable to illustrate maps. Features of the service include vehicle type specification, travel modes that is bicycling, driving, transit, or walking. As a commercial product, the Google Maps API allows limited use when not in subscription mode. In this context the design of our system took into account the respective restrictions. When the design took place the former allowed 25,000 map loads per day for 90 consecutive days, recovering 100 elements each performed search (query), recovering 100 elements per 10 seconds, recovery of 2,500 items per 24 hours. It is worth noting that requests are also subject to rate limits. The design of the system took into account all the above restrictions in order for the system to be capable to represent nodes (stops) as points in Google Maps, represent of realistic routes ie routes that take account of actual characteristics as e.g. one-way. An initial configuration of the system is illustrated in fig. 1.

GNU Octave is a high-level programming language, primarily intended for numerical computations. It offers a command line interface for solving linear and nonlinear problems numerically, and for performing other numerical experiments using an interpreted language mostly compatible with the well known MATLAB platform by Mathworks. It can also be used as a language oriented script execution. Octave is free software, distributed under the terms of the GNU General Public License. Besides its use for desktop computers that is, for personal scientific calculations, Octave is also used in academia and industry. Its features include that it is written in C ++ and uses the standard C ++ library, it uses an interpreter to execute the script language and it is expandable with the use of dynamic parts (modules). Versions 3.8.0 and later include a graphical user interface (GUI), other than the traditional command line interface (CLI). The architecture of our solution employs open source PSO packages compatible with Octave.



The architectural components and their integration was a challenge for the implementation of the system. Google maps was a valuable and suitable solution since it offers unique functionality, satisfactory front-end interface and sufficient, usable APIs. Matlab is both capable and efficient to execute the algorithmic logic but, by design, not suitable for the middle tier of the platform; Octave on the contrary is very easy to integrate but does not support either PSO or GA to the desirable extent. In order to produce realistic, applicable solutions the system needs to produce meaningful routes; to this end they needed to be optimized with respect to their total distance. Thus, besides optimizing with respect to the objective function analyzed above, it was decided to recover the shortest possible route that visits each node exactly once and returns to the origin. The aforementioned sentence is an expression of the Traveling Salesman problem [13] the well known non-deterministic polynomial time (NP) hard problem. This logic also needed to run in the middle tier.

Fig. 2: in order to ensure that bus routes are optimal with respect to traveling distance the systems solves the traveling salesman problem using a platform independent implementation of a genetic algorithm [14], illustrated in the figure.

5. CONCLUSIONS AND FUTURE WORK

The current paper outlines the engineering and algorithmic design of the DIANNA system that aims to solve the TRNDP problem using a PSO approach. The objective function of the optimization algorithm is very similar to [3]; it ultimately aims to produce solutions that optimize environmental parameters that is, vehicle emissions and vehicle types (electrical or conventional) besides more typical parameters of the problem such as distances, bus frequencies, demand. The design is innovative since the formulation of the solution is binary, designed to facilitate easy manipulation. Also, the architecture of the informational system is designed to interact with well known GIS services, relies on middleware logic that executes the optimization and presents the results using web technologies. Fig. 2 illustrates the implementation of a genetic algorithm that solves the traveling salesman problem; the implementation relies in platform independent server-side logic that can be called by any component of the system.

In the near future we expect to produce experimental results that exhibit realistic solutions for the case of Heraklion, Crete and, since the formulation of the problem allows it, we also expect to investigate the extent to which environmental policies can be applied that is, find optimal or next to optimal values for parameters w_1 to w_6 that correspond to minimum environmental costs.

ACKNOWLEDGEMENTS

This work is part of the DIANA project, funded by the Operational Program "Education and Lifelong Learning", action Archimedes III, co-financed by the EU and National funds (National Strategic Reference Framework 2007-2013).

REFERENCES

[1] Y. Jang, Y. Ko, "System architecture and mathematical model of public transportation system utilizing wireless charging electric vehicles", Intelligent Transportation Systems, 15th Internationalnal IEEE Conference on, pp.1055-1060, 2012

[2] T.H. Ortmeyer, P. Pillay, "Trends in transportation sector technology energy use and greenhouse gas emissions," Proceedings of the IEEE, vol.89, no.12, pp.1837-1847, 2001

[3] M. Pternea, K. Kepaptsoglou, and M. Karlaftis, "Sustainable urban transit network design", Transportation Research Part A, vol. 77, pp. 276–291, 2015

[4] F. Zhao and X. Zeng, "Optimization of transit route network, vehicle headways and timetables for large-scale transit networks", European Journal of Operational Research, vol. 186, no. 2, pp. 841–855, 2008

[5] J. J. Blum and T. V. Mathew, "Intelligent agent optimization of urban bus transit system design", Journal of Computing in Civil Engineering, vol. 25, no. 5 pp. 357–369, 2010

[6] R. Eberhart, J. Kennedy, A new optimizer using particles swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995.
[7] J. Kennedy, R. Eberhart, “A discrete binary version of the particle swarm algorithm”, in: Proceedings of the IEEE International Conference on Computational Cybernetics and Simulation, 1997.
[8] N. Biggs, "T. P. Kirkman, mathematician", The Bulletin of the London Mathematical Society vol. 13, no. 2, pp. 97–120, 1997
[9] Google MAPS documentation for the Javascript application programming interface https://developers.google.com/maps/documentation/javascript/, accessed on August 2015
[10] S. Lee, S. Soak, S. Oh, W. Pedrycz, M. Jeon, “Modified binary particle swarm optimization”, Progress in Natural Science, iss. 18, vol. 9, pp. 1161–1166, 2008
[11] L. Chuang, S. Tsai, C. Yand, “Improved binary particle swarm optimization using catfish effect for feature selection”, Expert Systems with Applications, iss. 38, pp. 12699–12707, 2011
[12] S. Mirjalili, A. Lewis, “S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization”, Swarm and Evolutionary Computation, pp. 91–14, 2013
[13] M. Tasgetiren, P. Suganthan, P. Quan-ke, L. Yun-Chia, “A genetic algorithm for the generalized traveling salesman problem”, IEEE Congress on Evolutionary Computation, pp. 2382 – 2389, 2007
[14] M. Kalochristianakis, http://83.212.103.151/~mkalochristianakis/techNote s/travelling.php, accessed on September 1, 2015

AUTHORS

Michael Kalochristianakis

Michael is currently an associate lecturer and researcher at the Technological Educational Institution of Crete (TEIC), at the Department of Informatics Engineering. His interests include but are not limited to technical project management and project management in general, innovative IT systems and services, software development methodologies, multimedia technologies and algorithms, web engineering, object oriented design, architectures, frameworks and patterns, in-band management and remote management, web and portal development, open source technologies, green technologies and energy conservation. Michael holds a Doctorate degree in Computer Engineering and Informatics, a Masters degree in Computer Science and a Diploma in Electrical Engineering and Computer Technology.

Dimitrios Kosmopoulos

Dimitrios Kosmopoulos is currently Assistant Professor at the University of Patras, Department of Cultural Heritage and New Technologies. Previously he was at Technical Educational Institute of Crete - Department of Informatics Engineering and before that, he was at Rutgers University, Computer Science Department, CBIM Lab and at the University of Texas at Arlington, Department of Computer Science and Engineering. He has been a research scientist in the Computational Intelligence Laboratory of the Institute of Informatics and Telecommunications in the NCSR Demokritos. He was also affiliated with the Department of Electrical and Computer Engineering of the National Technical University of Athens (Greece). He was employed as a researcher and developer for various companies and institutions.