2012 15th International IEEE Conference on Intelligent Transportation Systems
Anchorage, Alaska, USA, September 16-19, 2012

Estimation of missing values in traffic density maps

Jiri Petrlik[1], Pavol Korcek[1], Otto Fucik[2], Marian Beszedes[2] and Lukas Sekanina[1]

Abstract— **The traffic density map (TDM) represents the density of road network traffic as the number of vehicles per a specific time interval. TDMs are used by traffic experts as a base documentation for planning a new infrastructure (long-term) or by drivers for showing a current traffic status (short-term). We propose two methods for estimation of missing density values in TDMs. In the first method, the problem is formulated relatively strictly in terms of quadratic programming (QP) and a QP solver is utilized to find a solution. The second, more general method is based on a multiobjective genetic algorithm which allows us to find a reasonable compromise among several objectives that a traffic expert may formulate. These two methods can work automatically or they can be used by a traffic expert for an iterative density estimation. Results of experimental evaluation based on real and randomly generated data are presented.**

I. INTRODUCTION

The traffic density map (TDM) represents the density of road network traffic as the number of vehicles per a specific time interval. This interval can be given in minutes or hours. Usually, TDMs are used by traffic experts as a base documentation for planning a new infrastructure (long-term) or by drivers for showing a current traffic status (short-term). Such TDMs can be composed automatically – with the aid from standard surveillance technologies (e.g. various data sensors such as loop detectors or traffic cameras). Another approach, which can be used for TDM calcultion, is the manual counting on selected road segments. However, counting where people are involved in the process is usually quite inaccurate and also inefficient [14].

There is also a big effort to estimate the future traffic density. For example, paper [7] reports some techniques of density prediction for the congestion analysis under heterogeneous traffic conditions. The goal is to determine their feasibility under the Indian traffic scenario. Recently, a statistical approach to predict the density on any edge of such a network at some time in the future was also presented [8]. The method is based on short-time observations of the traffic history. In the two previous examples, however, complete data sets for the whole traffic network were required. The approaches require either a lot of traffic sensors or in the worst case, many people involved in the manual counting. In the situation where it is not possible to cover the whole traffic network with the field data, missing areas must be completely excluded from the traffic density estimation.

In this paper, we deal with a more realistic scenario in which TDM is not complete. We propose two methods for estimation of missing density values. In the first method, the problem is formulated relatively strictly in terms of quadratic programming (QP) and a QP solver is utilized to find a solution. The second, more general method is based on a multiobjective genetic algorithm which allows us to find a reasonable compromise among several objectives that a traffic expert may formulate. These two methods can work automatically or they can be used by a traffic expert [15] for an iterative density estimation as described later in this work. In present, estimation of missing values in TDMs is done by a traffic expert who is supposed to have a very good knowledge about traffic network in the observed area. We will show that our methods can significantly improve and accelerate this process.

The rest of the paper is organized as follows. The problem of missing values estimation in TDM is briefly introduced in Section II. Then, in Section III, the quadratic programming is explained followed by Section IV, where multiobjective evolutionary algorithms are described. In Section V the proposed multiobjective evolutionary approach is described. Results of experimental evaluation based on real and randomly generated data are summarized in Section VI. Discussion of results is provided by Section VII. Finally, conclusions and suggestions for the future work are given in Section VIII.

II. TDM REPRESENTATION AND DESCRIPTION

TDM can be viewed as a directed graph, where each node n represents a crossroad and each edge represents a particular road segment. The density on the edge, d_e, represents the number of incoming our outgoing vehicles per time interval on a given edge e. The historic value of density h_e (e.g. measured a year ago) can also be available for some edges. Nodes in TDMs are divided into two sets. In the first set, there are border nodes and we denote this set as N_b. All incomming and outgoing vehicles from the investigated traffic density map must go through these nodes. The second set contains the inner nodes. We denote this set N_i. For the inner nodes it also applies that the sum of the input vehicles minus the sum of the output vehicles should be near, or even better, equal to zero. Both sets are strictly disjunct ($N_i \cap N_b = \emptyset$). Every node n has a set of incoming edges I_n and a set of outgoing edges O_n, respectivelly. Fig. 1 shows a TDM consisting of 6 nodes and 12 edges, where

This work was partially suported by Camea spol. s r. o., the Czech Science Foundation under project P103/10/1517, the research program MSM 0021630528, FIT BUT grant no. FIT-S-12-1 and the IT4Innovations Centre of Excellence CZ.1.05/1.1.00/02.0070

[1]Jiri Petrlik, Pavol Korcek and Lukas Sekanina are with Brno University of Technology, Faculty of Information Technology, IT4Innovations Centre of Excellence, 612 66, Brno, Czech Republic {ipetrlik, ikorcek, sekanina}@fit.vutbr.cz.

[2]Otto Fucik and Marian Beszedes are with Camea spol. s r.o., Korenskeho 25, 621 00, Brno, Czech Republic {m.beszedes, o.fucik}@camea.cz.

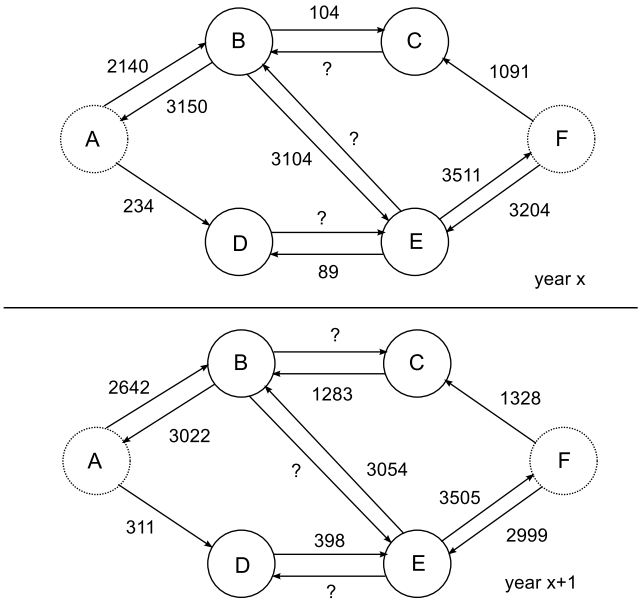

Fig. 1. Syntetic example of measured data for TDM in two years. Missing values are marked by question mark. Values on these edges should be estimated.

$N_b = \{A, F\}$ and $N_i = \{B, C, D, E\}$. In TDM which we consider in this work, one has to deal with hundreds of nodes and hundreds of edges. Typically 40 % of values are missing and have to be estimated.

III. QUADRATIC PROGRAMMING APPROACH

Quadratic programming is a method for solving optimization problems that can be formulated as follows:

$$\begin{array}{lll} \text{Minimize} & q(x) = \frac{1}{2}x^T G x + x^T d & \\ \text{subject to} & a_i^T x = b_i & i \in \xi \\ & a_i^T x \geq b_i & i \in I \end{array}$$

where x and d are vectors with n components and G is a positive semidefinite matrix. Expressions $a_i^T x \geq b_i$ and $a_i^T x = b_i$ represent some constraints [6].

In order to solve a particular TDM problem, the quadratic programming is used to minimize an absolute value of the difference betweeen the number of incoming and the number of outgoing vehicles for all nodes of TDM. Let E_n denote the error of an inner node, i. e.

$$F_1 = E_n = |\sum_{e \in I_n} d_e - \sum_{e \in O_n} d_e| \quad (1)$$

The squared sum of these errors for all those nodes is then our objective function:

$$E = \sum_{n \in N_i} E_n^2 \quad (2)$$

In this approach, we can define a set of constraints for edges e which we don't know the density value d_e for. As we may know historic values from the previous measurement h_e, it is possible to constraint the expected value to be in some range (e.g. $d_e \in [0.7h_e, 1.3h_e]$). The exact ranges must be specified by a traffic expert.

The main advantage of this approach is in its speed. It's guaranted that the QP method will always find a solution. Typical problem instances are solved in a few seconds. On the other hand, a quite strict problem formulation does not allow us to simply incorporate more domain knowledge into the quadratic programming method, for example, to define more objective functions for the optimization.

IV. MULTIOBJECTIVE OPTIMIZATION

A. Genetic algorithm

A genetic algorithm (GA) is a popular optimization algorithm based on principles of Darwinian evolution which has been applied to solve many hard optimization problems [12]. Each candidate solution is represented by an array of parameters. In the terminology of genetic algorithms these arrays are called chromosomes. A set of chromosomes forms a population.

The quality of each candidate solution is evaluated, by means of the objective (fitness) function. The selection of parents for the new population is based on individuals fitness values. Higher-scored individuals have a higher probability of selection. New population is then generated using crossover and mutation over the parent individuals. The GA is terminated when a suitable solution is discovered or predefined number of generations is exhausted. Elitism means that a given number of the highest-scored individuals is carried into the next population directly, without any modification.

B. Multi-objective optimization

In the single-objective optimization there is only one goal, i.e. finding one solution with the best fitness value. However, if there is more than one objective, a multiobjective optimization algorithm has to be applied. The main problem in this type of optimization is on how to compare the quality of two solutions. For example, solution a can be better then solution b according to one objective but worse according to another objective function and vice versa. To deal with this issue, the *Pareto dominance* relation is often utilized.

We can describe the multiobjective optimization more formally according to [2] as follows: The goal is to minimize (or maximize) M objective functions:

$$\begin{array}{lll} & f_m(x), & m = 1, 2, ..., M \\ \text{subject to} & g_j(x) \geq 0 & j = 1, 2, ..., J \\ & h_k(x) = 0 & k = 1, 2, ..., K \end{array}$$

where x is a vector of parameters $x = (x_1, x_2, ..., x_n)$, M is the number of objectives. Functions g_j define inequity constraints and functions h_k define equity constraints. The feasible solution is one which fullfills all given constraints.

For two feasible solutions a and b we can say that solution a dominates a solution b if and only if two conditions are fullfilled:

- Solution a is at least of the same quality as solution b for all objectives.

- Solution a is better than b in at least one objective.

We call solution a Pareto-optimal if and only if there is no solution b, which dominates a. In a multiobjective optimization problem with conflicting objectives, the goal is to find many or even all Pareto-optimal tradeoffs.

In the past, many methods for multiobjective optimization have been developed. For example, weighted sum method [16], ϵ-constrained method [17], weighted metric method, Benson's method [18], etc. However, the main disadvantage of these methods is that they provide only one solution. It's necessary to change parameters of the methods to obtain other solutions [2], [3].

One of the main advantages of truly multiobjective genetic algorithms is their ability to provide a set of different solutions in a single run. It's mainly because these GAs internaly create Pareto fronts (i.e. sort candidate solutions according to the dominance relation) in every single generation. In the past, there were proposed various kinds of multiobjective genetic algorithms, e.g. Vector Evaluated Genetic Algorithm (VEGA) [4], Strength Pareto Evolutionary Algorithm (SPEA) [5], NSGA, NSGAII [1] and many others [2].

NSGAII is a multiobjective genetic algorithm which was proposed by K. Deb in [1]. It uses a non-dominated sorting algorithm with an optimal complexity $O(MN^2)$, where M is the number of objective functions and N is the number of sorted solutions. It uses the elitism and provides diversity preservation mechanisms to find various different tradeoff solutions. One of the biggest advantages of this algorithm is that it's not neccessary to set numerous parameters such as the weights of objective functions and sharing parameter. NSGAII and non-dominated sorting mechanisms are described in greater detail in [1].

V. The proposed multi-objective approach

A. Encoding of paremeters

In the proposed approach which is based on NSGAII, each candidate solution is defined by a vector of real numbers. Every component of the vector represents a traffic density on one road segment, for which the density is not avalaible. The parameter value should be rounded to have the integer value. In the first generation of GA, components of vectors are initialized to positive randomly generated values $[0, 100000]$.

B. Objective functions

In the quadratic programming approach, it was possible to use only one objective function (see Eq. 2). NSGAII allows us to utilize more fitness functions directly. In our case there will be two objective functions. The first is the sum of errors on nodes (see Eq. 2) and the second is the sum of differences to historic values (see Eq. 2).

Let E_h be a set of edges which have not the density values available, but we know the historic values of density for them. The second objective function F_2 is then defined as

$$F_2 = \sum_{e \in E_h} |d_e - h_e| \qquad (3).$$

Similarly to the quadratic programming, it is possible in NSGAII to constraint the expected density to interval $[0.7h_e, 1.3h_e]$.

C. Genetic operators

A single point crossover and a normally distributed mutation are utilized. The single-point crossover swaps some parameters between two candidate chromosomes. Normally distributed mutations work as follows. For each gene g_i of chromosome a random number $r_i \in [0, 1]$ with the uniform distribution is generated. If this number is less than the mutation probability P_t, then a new randomly generated number (with the normal distribution $N(0, \epsilon)$) is added to gene g_i.

D. Self adaptation

In order to maximize performance of the genetic algorithm it is necessary to correctly set various control parameters such as the population size, the probability of crossoever, the probability of mutation etc. This can be considered as an optimization problem itself.

These control parameters can be determined by expert, or discovered by another genetic algorithm (this approach is called metaevolution). In our evolutionary approach we use a self-adaptive method, which enables to encode some control parameters of genetic algorithm in to the chromosome [9], [10] and [11]. We added a special gene g_σ to represent the deviation of mutations, gene g_c to represent the probability of crossover and gene g_m to represent the probability of mutation into the chromosome. Tab. I. shows permited values and deviations for mutation of these genes.

TABLE I
PERMITTED VALUES AND DEVIATIONS OF CONTROL GENES

Gene	Permitted values	Deviation
g_σ	$[0.00002, 0.2]$	0.013
g_c	$[0, 1]$	0.1
g_m	$[0.01, 1]$	0.1

As in [9] the mutation has two phases. In the first phase, the mutation of the control genes is performed. Then other genes are mutated according to the new values of the control parameters. The crossover probability control works as follows. Firstly two candidate solutions are selected by a tournament selection. Then the mean of their g_c genes is taken as the probability of crossover for these two solutions, i.e.

$$P_c = \frac{g_c^1 + g_c^2}{2} \qquad (4).$$

VI. Experimental results

A. Datasets

In order to evaluate the proposed methods, two data sources are utilized: (1) field data from annual manual counting from the city of Prague (counting in year 2008 and 2009). The data cover the central part of the city which

is modelled using 126 nodes (28 of them are border nodes) and 277 edges (117 of them without the traffic intensity). This data set was provided by TSK Prague [13].

(2) syntetic data where three random scenarios of incomplete TDMs were generated. These syntetic maps have 200, 500 and 1000 nodes.

B. Performance

For the real scenario from Prague, a solution produced by the first method based on QP is: $F_1 = 5.286681E8$ and $F_2 = 152450$. We used the Octave QP solver.

Three different variants of multiobjective GA were analyzed: (1) Genetic algorithm starts with a randomly generated initial population without self-adaptation; (2) GA starts with a randomly generated initial population, but self-adaptation is enabled; (3) GA starts with the initial population generated by quadratic programming and self-adaptation is enabled. All three variants optimized two objective functions: (i) the error on nodes as described in Eq. 2 and (ii) the sum of differences on edges against last year counting (Eq. 3). The population size was 50 for each run.

Tables II, III and IV give the mean of the best values of objective functions after a specified number of generations (1000, 2000, 5000, 10000, 25000 and 50000). Two situations were analyzed. One without hard constraints given and one with maximal difference of 30% against historic values as a hard constraint. It can be observed that variant (1) provides the worst results for both objective functions. The variant (2) provides better results against the history, but significantly worse results in node errors than the combined variant (3) approach when 30% constraint is applied. When method (3) doesn't use this constraint, it provides better results on nodes than the quadratic programming approach. However, it provides worse difference results against the historic values till 25000 generations, but better results after 25000 generations than the combined approach. Fig. 2 and 3 compare the speed of convergence for all three variants. One thousand generations takes 5.5 seconds on the Intel Xeon 2.66 GHz processor. It can be seen that GA is slower than QP. Traffic expert solves the same problem in a few days.

C. Pareto front

One goal of the multiobjective optimization is to have solutions widely spread along the Pareto front at the end of optimization. Table V shows the results obtained from a single run of the genetic algorithm. Figure 4 shows positions of individual solutions in the whole objective space. On the horizontal axis there is an error on nodes and vertical axis shows a difference against historic values. Every such point represents one tradeoff solution and it can be seen that solutions are quite well spread.

VII. Discussion

A. QP vs. multiobjective GA

It was shown that two-objective optimization process gives many tradeoff solutions situated on the Parreto front. This is usefull for iterative estimation, because one can choose

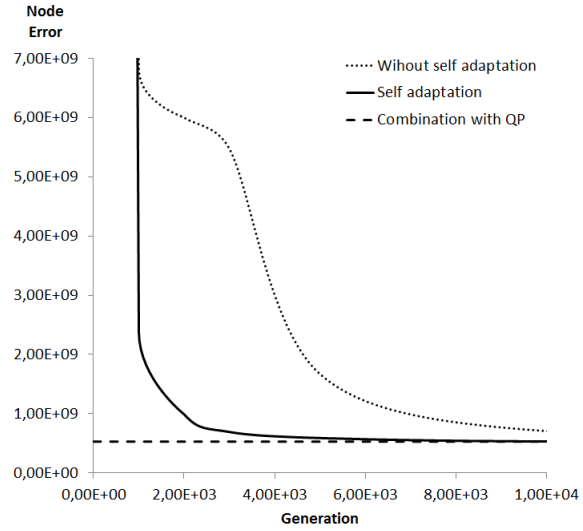

Fig. 2. Convergence of F_1 (error on nodes).

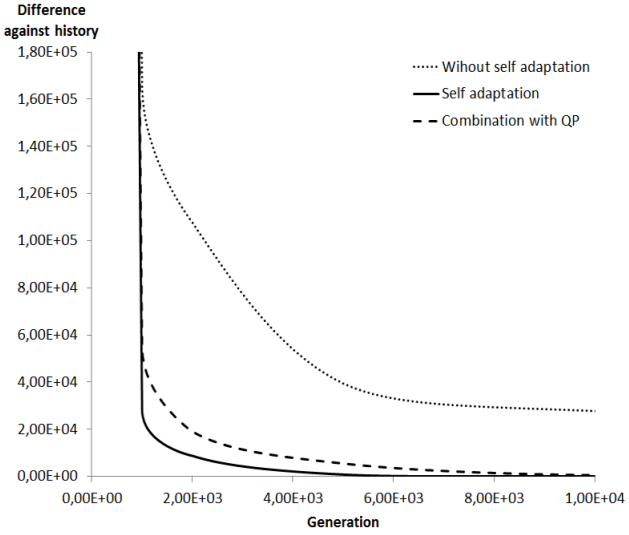

Fig. 3. Convergence of F_2 (history difference).

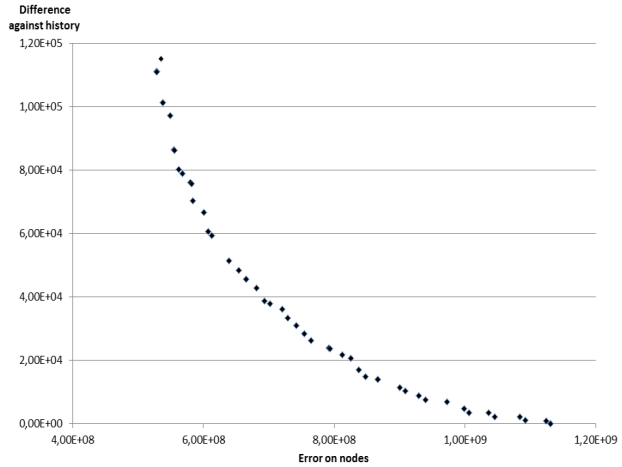

Fig. 4. Spread of solutions along Pareto front after a single run with 50 solutions in population. Each point represents one solution.

TABLE II

FITNESS VALUES FOR GENETIC ALGORITHM METHOD, VARIANT (1) WHICH DOESN'T USE THE SELF ADAPTATION

Generations	Error on nodes	History difference
Random scenario with 200 nodes		
2000	9.713179E9	26802.0
10000	9.680005E9	7771.2
25000	9.676344E9	5233.35
Random scenario with 500 nodes		
2000	3.370222E10	273080.4
10000	3.339072E10	54241.6
25000	3.336195E10	33805.9
Random scenario with 1000 nodes		
2000	9.354801E10	854547.4
10000	8.909628E10	244548.1
25000	8.884524E10	155489.4
Real data from Prague		
1000	6.813175E9	165515.3
2000	6.0065031E9	107727.4
5000	1.669960E9	39446.8
10000	7.082570E8	27780.7
25000	6.081299E8	21955.65
40000	5.92208E8	19916.1
50000	5.885092E8	18223.7

TABLE III

FITNESS VALUES FOR GENETIC ALGORITHM METHOD, VARIANT (2) WHICH USES THE SELF ADAPTATION

Generations	Error on nodes	History difference
Random scenario with 200 nodes		
2000	9.705400E9	7928.95
10000	9.672611E9	42.55
25000	9.672601E9	1.35
Random scenario with 500 nodes		
2000	3.353729E10	86722.9
10000	3.332559E10	2426.65
25000	3.332383E10	82.4
Random scenario with 1000 nodes		
2000	9.103709E10	450026.75
10000	8.855298E10	64569.45
25000	8.853087E10	15173.65
Real data from Prague		
1000	2.380441E9	28188.15
2000	9.907829E8	8765.05
5000	5.89444E8	817.5
10000	5.319566E8	57.4
25000	5.289430E8	9.45
40000	5.286752E8	0.1
50000	5.28668E8	0.2

the best tradeoff according to his/her knowledge. It's not necessary to define the objective functions in such a strict way as in the quadratic programming and it is possible to have more complicated objective functions (e.g. the objective function with different importance of errors on nodes). Also, it is possible to use the constraints in the same way discussed in the QP approach.

B. Combination of previous approaches

The best results results were obtained by combining both methods, when the initial solution is generated by the quadratic programming and then further optimized by GA.

TABLE IV

FITNESS VALUES FOR VARIANT (3) WHICH COMBINES QUADRATIC PROGRAMMING WITH GENETIC ALGORITHM

Generations	Error on nodes	History difference
Real data from Prague		
1000	5.286681E8	54754.95
2000	5.286681E8	19044.8
5000	5.286681E8	5488.6
10000	5.286681E8	550.8
25000	5.286681E8	7.35
40000	5.286681E8	1.5
50000	5.286681E8	0.75

TABLE V

FITNESS VALUES AFTER A SINGLE RUN OF GENETIC ALGORITHM.

Number	Error on nodes	History difference
1	528679433	111147
5	537616079	101211
10	567838646	78839
15	607468038	60657
20	665618956	45534
25	728551453	33265
26	742025879	30861
30	793220329	23726
35	866315327	14037
40	939494317	7520
45	1045551681	2242
50	1131328632	17

C. Expert knowledge

Recently built parking lots, shopping centres, etc. can introduce significant errors to TDMs with respect to historical data. The errors on nodes corresponding to these new entities are obviously much less significant than on other nodes. Automatically generated estimation can't deal with this fact. The proposed approach is able to incorporate this kind of domain knowledge into the solution. In one of supported scenarios, traffic expert is allowed to choose from three methods to define error on each single node n. These options are described by Eq. 5, where $k \in \{1, 2, 3\}$.

$$E_n = |\sum_{e \in I_n} d_e - \sum_{e \in O_n} d_e|^k \quad (5)$$

It is evident that higher value of k implies higher impact of error on current node to fitness F_1. We propose this method for incorporating of experts knowledge without any experiments. We developed a computer program in Java, which uses an incremental process of traffic density estimation. This process is supposed to be driven by a user – traffic expert. At the begining the user sets the values for measured edges and runs the multiobjective genetic optimization process. There are several optimized solutions at the end of this process. One of them can be chosen and eventually edited. The user can also change importance of the errors on nodes and constraints as mentioned previously. After this editing, the optimization process can be performed again and again. This iterative process continues until a sufficient estimation is reached.

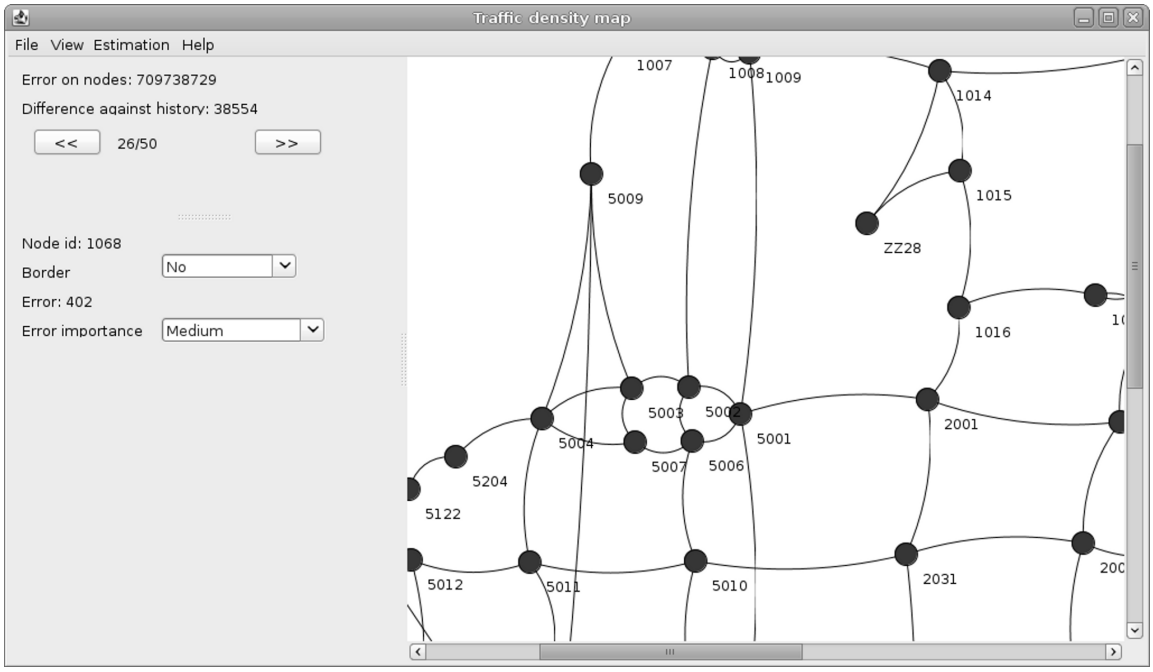

Fig. 5. A screenshot of application for estimation of missing values in TDM.

VIII. Conclusion

In this paper, we have presented that it is possible to effectively estimate TDM in situations where field data are not available. We have compared two advanced techniques QP and multiobjective genetic algorithm. We have proposed a method combining both techniques, which has also been implemented in the complete Java based software system. This software system can be used by traffic experts when the traffic density information is not available for a required region. The proposed methods could be exploited by local traffic authorities in decision making process (e.g. in situations when deciding where to install a new surveillance system). Our system can help to determine which crossroads are more or less important for the global network density estimation (where to place new data sensors for traffic monitoring).

In the future work, we are going to even more accelerate our estimation process, which is very important in case when smaller time interval than one year has to be used. Our aim is to use this density estimation process as a part of ITS in a real-time traffic congestion prediction.

References

[1] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evolut. Comput. 6 (2), pp. 182-197, 2002.

[2] K. Deb, Multi-objective Optimization Using Evolutionary Algorithms. Chichester, UK: Wiley., 2003, ISBN: 9780471873396.

[3] Y. Collete, P. Siarry, Multi-objective Principles and Case Studies. Berlin, Springer, 2003, ISBN: 9783540401827.

[4] J. D. Schaffer, Multiple objective optimization with vector evaluated genetic algorithms. In Proceedings of the First International Conference on Genetic Algorithms, 1985, pp. 93-100.

[5] E. Zitzler, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, Transaction on Evolutionary Computation, volume 3, issue 4, 1999, pp. 257-271.

[6] J. Nodcedal, S. J. Wright, Numerical Optimization, Springer-Verlag New York Inc., 2006, ISBN: 978-0387987934.

[7] A. Padiath, L. Vanajakshi, S. C. Subramanian,H. Manda: Prediction of traffic density for congestion analysis under Indian traffic conditions, in Proceedings of 12th International IEEE Conference on Intelligent Transportation Systems (ITSC 2019), St. Louis, 2009, pp.1-6, ISBN 978-1-4244-5519-5

[8] H. P. Kriegel, M. Renz, M. Schubert, A. Zufle: Efficient Traffic Density Prediction in Road Networks Using Suffix Trees, KI - Kunstliche Intelligenz, Springer Berlin / Heidelberg, 2012, pp.1-8, ISSN 0933-1875

[9] R. Hinterding, Gaussian mutation and self-adaption in numeric genetic algorithms, 2nd IEEE Conf. Evolutionary Computation, 1995, pp. 384-389.

[10] T. Back, The interaction of mutation rate, selection and self-adaptation within a genetic algorithm In proceedings of the 2nd European Conference Parallel Problem Solving from Nature, 1992, pp. 85-94.

[11] A. Eiben, R. Hinterding, Z. Michalewicz, Parameter Control in Evolutionary Algorithms, IEEE Transactions on Evolutionary Computation, volume 3, number 2, 1999, pp. 124-141.

[12] Goldberg, D., E., Genetic Algorithms in Search, Optimization and Machine Learning, 1st Edition, 1989, ISBN: 0201157675.

[13] Prague Technical Administration of Roads, TSK Prague, http://www.tsk-praha.cz

[14] S. Maerivoet and B. De Moore, Transportation planning and traffic flow models, Technical Report No. 05-155, Katholieke Universiteit Leuven, 2005.

[15] L. H. Immers and S. Logghe, Traffic Flow Theory, Katholieke Universiteit Lueven - Faculty of Engineering, Department of Civil Engineering, Section Traffic and Infrastructure, Course H111, en. version, 2002.

[16] I. C. Parmee, D. Cevtkovic, A. W. Watson, C. R. Bonham, Multiobjective satisfaction within an interactive evolutionary design environment, Evolutionary Computation Journal 8 (2), 2000, pp. 197-222.

[17] Y. Y. Haimes, L. S. Lasdon, D. A. Wismer, On a bicriterion formulation of the problems of integrated system identification and system optimization. IEEE Transactions on Systems, Man, and Cybernetics 1(3), 1971, pp. 296-297.

[18] H. P. Benson, Existence of efficient solutions for vector maximization problems. Journal of Optimization Theory and Applications 26(4), 1978, pp. 569-580.