Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 89 (2014) 618 – 625

Procedia Engineering

www.elsevier.com/locate/procedia

16th Conference on Water Distribution System Analysis, WDSA 2014

Implementing ΔQ Method to Accelerate the Optimization of Pressurized Pipe Networks

D. Ivetić[a,]*, Ž. Vasilić[a], D. Prodanović[a], M. Stanić[a],

[a]*University of Belgrade, Faculty of Civil Engineering, Belgrade, Serbia*

Abstract

To optimize the design of a pressurized pipe network, large number of possible solutions needs to be examined. This paper explores the option to accelerate the computation using ΔQ hydraulic method. The ΔQ method is firstly used in a standard form, as stand-alone hydraulic solver inside the evaluation function. Second version was to use ΔQ method once to solve the initial network, and during optimization procedure, unknown flow corrections were added to list of other unknowns for optimization. The suggested approach was tested on New York City distribution network reconstruction example using standard genetic algorithms (GA).

Peer-review under responsibility of the Organizing Committee of WDSA 2014

Keywords: Acceleration; Genetic Algorithms; ΔQ method; Efficiency; Looped networks.

1. Introduction

In the last decades modern engineering practice has been moving towards the higher degree of computer and software usage. As computer sciences experience fast progress, engineering practice is trying to catch up and utilize stronger processors and new software capabilities that are introduced. One of the engineering fields which benefits from this progress is hydraulics and water system analysis, especially numerical modelling of fluid flow and optimization of the design and performance of water systems. In this paper the focus is on the optimization of water distribution network systems, specifically the long computation time needed and the possibility to reduce it.

In order to accelerate the computation or optimization process, three different approaches can be utilized: use of more powerful computers (faster processors), use of parallel computing systems [1, 2] with adapted hydraulic solvers

* Corresponding author. Tel.: +381 11 337 0206; fax: +381 11 337 0206.
E-mail address: divetic@hikom.grf.bg.ac.rs

Peer-review under responsibility of the Organizing Committee of WDSA 2014
doi:10.1016/j.proeng.2014.11.486

or use of more efficient hydraulic method. First two, mostly hardware driven approaches, have been successfully addressed by many different authors, computer and communication scientists [3, 4]. In this paper third listed approach is being investigated and the results of first tests are presented. The idea was to promote the intelligent, hydraulically based solutions compared to brute force one.

When pressurized water supply network is to be optimized, in most cases the EPANET is used. EPANET is reliable, free software package, available as standalone EXE or DLL version, which can be easily integrated with most optimization programs. The software is using a so called hybrid node-loop algorithm originating from Todini and Pilati [5], for solving the continuity equation. Actually, the basis of this method originates from Hardy Cross and the method he originally called a method of balancing flows [6].

Hardy Cross has also introduced the ΔQ method for hydraulic calculation of network [6]. It was derived from his original moment distribution method, which he used for structural analysis purpose. It has been in a common use in 50's and 60's until computers arrived and node method took over the throne. Most significant difference is that in the node based method, number of non-linear equations is equal to number of unknown nodal heads plus unknown pipe flows while in ΔQ method it is equal to the number of loops in the network which is usually a much smaller.

In this paper, we explore the possibilities to use the ΔQ method in hydraulic calculation of network inside evaluation function of optimization algorithm. Method is firstly used in a standard format, where we simply employed it as a hydraulic solver. After that, we tried the modified version, named "Variable ΔQ method", where values of the flow corrections are not directly computed in evaluation function, but used as additional set of unknowns in optimization process. Paper will present results and obtained optimization accelerations for both options. Although number of variables used for optimization is larger in the Variable ΔQ method, obtained results are encouraging since acceleration is up to two orders of magnitude. To test the ΔQ methods, a well-known problem of the New York City distribution network reconstruction was used. Results and needed computation time were compared to ones obtained using EPANET as hydraulic solver inside the evaluation function. Used optimization method was standard genetic algorithm, efficient in handling a single objective optimization problem [7].

2. Methodology of ΔQ method implementation

It was already mentioned that ΔQ method requires solving less equations than usual node-loop methods. This implies already a modest reduction of computation time if it is presumed that a similar amount of resources is needed for both types of equations. Another big difference in usage of these two different methods is that here in the pre-processing stage of an optimization algorithms, analysis of network graph topology has to be performed. This is due to the nature of the hydraulic solver, where it is firstly necessary to detect all simple cycles, or minimal basis loops, in the network in which flow corrections need to be calculated. Flow corrections are introduced in the place where original loops were split. When every loop is split, network structure is changed. Instead of looped graph, tree-structured graph is obtained. Tree-structured or branch networks are quite easy to handle in water distribution analysis and unknown flows and nodal heads can be obtained in double sweep. In pre-processing stage after the branch network is derived, unknown flows are calculated for this new network. These flows satisfy continuity equation in the network nodes, which means that they can be used as starting assumption of flow for the ΔQ method. This is only done once in pre-processing stage, so latter calls of evaluation function will just include iterative solution for the flow corrections.

Since it is obvious that implementation of ΔQ method requires change from loop to branch structured networks, another variation of optimization algorithm can be derived in order to improve computation time. From previous experience it was concluded that the convergence to an optimal solution in branch networks is much better than in loop ones, even if new networks are made of several hundreds of pipes with unknown diameters. The tests were made with the version of the optimization algorithm in which values of flow corrections were kept fixed, equal to the initial values obtained in pre-processing stage [8]. Results were satisfying in the terms of the performance acceleration but there was the major applicability issue in the more complex and new networks. In complex networks, with large number of loops, starting assumptions for flow corrections can have a significant difference from their final, correct values. Also, for the new networks, starting assumptions for flow corrections cannot be calculated in the pre-processing stage which renders this approach inapplicable. In order to overcome these shortcomings, another approach was tested, and is presented in this paper. Initial values of flow corrections, as calculated during pre-processing stage, were assumed to be new unknown variables, subject to optimization together with other unknowns (for example, pipe

diameters, or pipe roughness). In each evaluation function call, no time consuming hydraulic computation has to be done. Only simple pressure distribution has to be calculated. Also, one more penalty function has to be added in fitness function calculation, which will guide the optimization of flow corrections. This version is named Variable ΔQ method.

In the next subsections, basics of the ΔQ method and minimal loop detection algorithm are recapitulated, followed by a brief description of the used optimization algorithm and test case. At the end of this section, the description of the both Standard and Variable ΔQ method implementations in optimization algorithm are given.

2.1. ΔQ method

The ΔQ method was originally presented by Hardy Cross in 1936. It was proposed as one of the two possible methods for analysis of flow in networks of conduits. In his original work this method is called "Method of balancing heads" [6]. Method is based on the fact that in every closed loop (circuit) of the water supply network sum of total head loos is equal to zero. This is derived from the conservation of energy equation for the closed loop.

In order to apply this method, initial distribution of flows needs to be assumed. This distribution likely won't satisfy the previously mentioned condition for the head losses in the loop (Fig. 1a).

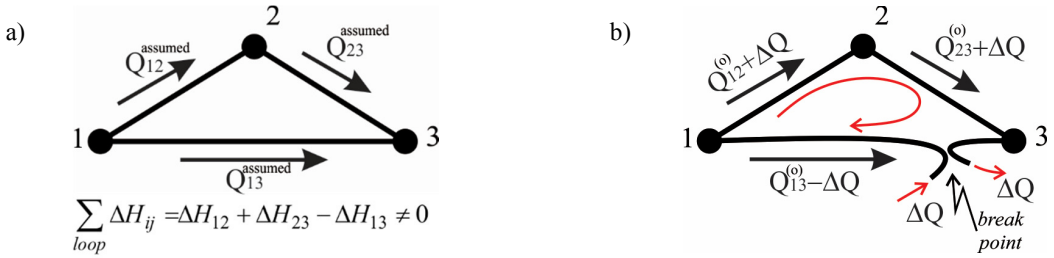

Fig. 1. a) Head loss sum in loop; b) Introduction of flow corrections.

This implies that corrections to that distribution must be made. To define the initial distribution of flows and make corrections to the distribution, loop needs to be broken at some point (Fig. 1b). Now we have branched network and with one pass backwards through that network, satisfying continuity equation in every node, the initial flow distribution is obtained. In this research graph, theory algorithms are used to obtain the initial flow distribution. Graph Breadth First Search (BFS) algorithm, started from the source node in the network, results in the spanning tree, which is essentially a branched network. Order in which spanning tree is formed is obtained as well, so this can be used to perform the passing backwards in order to determine initial flow distribution.

At the same break point flow correction ΔQ is introduced. Expanded form of the sum of head losses equation for the loop now is:

$$+R_{12}\left|Q_{12}^{(o)}+\Delta Q\right|\left(Q_{12}^{(o)}+\Delta Q\right)+R_{23}\left|Q_{23}^{(o)}+\Delta Q\right|\left(Q_{23}^{(o)}+\Delta Q\right)-R_{13}\left|Q_{13}^{(o)}-\Delta Q\right|\left(Q_{13}^{(o)}-\Delta Q\right)=0 \quad (1)$$

where R is the pipe flow resistance characteristic [s^2 m^{-5}] and $Q^{(o)}$ is the initial pipe flow [m^3 s^{-1}]. This nonlinear equation needs to be solved for the flow correction ΔQ. For this purpose we employ Newton Raphson iterative method and obtain general form of the solution:

$$\Delta Q^{i+1} = \Delta Q^{i} - \frac{\sum_{loop} sign \cdot K_{ij} \left| Q_{ij}^{(o)} + \sum_{pipe} sign \cdot \Delta Q_p^i \right| \cdot \left(Q_{ij}^{(o)} + \sum_{pipe} sign \cdot \Delta Q_p^i \right)}{2 \sum_{loop} K_{ij} \left| Q_{ij}^{(o)} + \sum_{pipe} sign \cdot \Delta Q_p^i \right|} \tag{2}$$

where i is the iteration number, ij is the ij-th pipe in the loop and ΔQ_p is the p-th flow correction that corrects initial flow in the pipe ij (there can be more than one, if pipe is common for two loops). *Sign* equals one (1) if direction of the introduced correction ΔQ_p is the same as the direction of the initial flow and zero (0) if otherwise. Iterative calculation is done until desired precision is reached.

Number of equations that need to be solved corresponds to the number of loops in the network. Loops are obtained as a result of graph theory algorithms. Aside from "ordinary loops"," term "pseudo loop" is introduced [9]. This is loop that is formed between two reservoirs/tanks with defined heads. Number of such loops is for one less than the number of reservoirs in the network. In that case one addition to the equation above has to be made. On the right side of the equation, in the numerator of the fraction, term $-\Delta H_{res}$ needs to be added, where ΔH_{res} is the difference of heads in the reservoirs.

2.2. Minimal basis loops detection

Prior to conducting calculations using the ΔQ method network loops need to be detected. This is done based on the results of the BFS algorithm mentioned before. Number of loops corresponds to the number of unused links during BFS propagation.

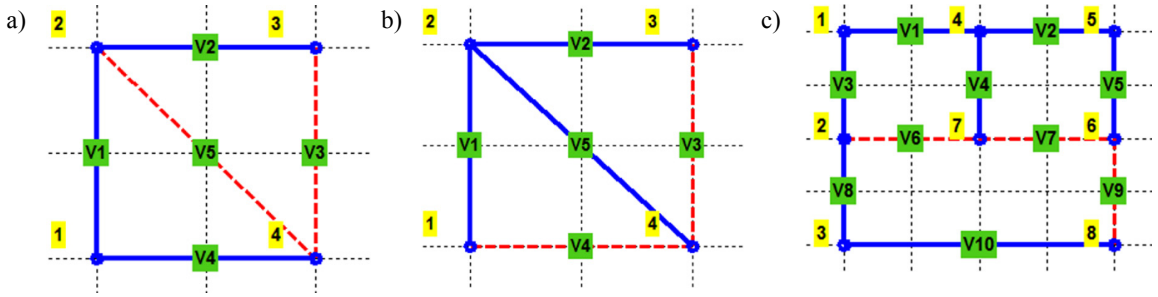

Fig. 2. Different configurations of Spanning Tree (ST).

These links (marked with dashed line) complete two loops – (1, 2, 4) and (1, 2, 3, 4). It is obvious that latter one is not a minimal basis loop but is a union of the loops (1, 2, 4) and (2, 3, 4) (Fig. 2a). Using this minimal basis loops configuration for the delta Q method means obtaining the simplest form of nonlinear equations to be solved, which would speed up the calculation. However, detecting them is not an easy task. Some algorithms are based on using outer or "back edges" of the network [10] but these have to be predefined. In the case of thousands of pipes this would be very demanding job so in these research another approach is proposed. Algorithm steps are as follows: 1) propagation with BFS algorithm to obtain spanning tree (ST); 2) transformation of ST to obtain optimal loops with minimal number of links; 3) final decomposition of overlapping loops if needed after step 2.

Transformation of ST will be explained on the simple example (Fig. 2a). Total number of links in loops for this configuration is 4+3 =7. In order to get better configuration, unused links will be swapped with the used ones. If link v5 is swapped with link v4, new configuration of ST gives the total number of links in loops 3+3 = 6 (**Errore. L'origine riferimento non è stata trovata.**2b). This is better than the previous one and it's marked as the new best configuration. Procedure is done until better configuration can't be reached. For the considered simple example best configuration, and with that minimal basis loops, is obtained with only one ST transformation. It could take more than

one but it doesn't mean that minimal basis loops will be obtained eventually (**Errore. L'origine riferimento non è stata trovata.**2c).

This requires step 3 of the algorithm – final decomposition. Decomposition is based on sorting the loops by their length (number of links) and creation of overlapping matrix which shows number of overlapping between any two loops. Decomposition starts from the loops with highest number of overlapping links. If this number equals one, decomposition is not needed.

2.3. Optimization algorithm and example problem

GA-s are employed as an optimization method which are efficient in terms of running time and finding suboptimal solutions [7]. GA-s are called inside an optimization algorithm, after pre-processing stage, with the task to generate coded solutions which are to be tested inside an evaluation function. For every solution examined, new value of fitness function is computed. Based on this value, examined solutions are ranked, where the solution with the lowest value of fitness function is ranked as suboptimal.

Testing example was extracted from literature [11], well known problem of New York's water distribution network reconstruction. Starting network for optimisation is presented in Fig. 3a.

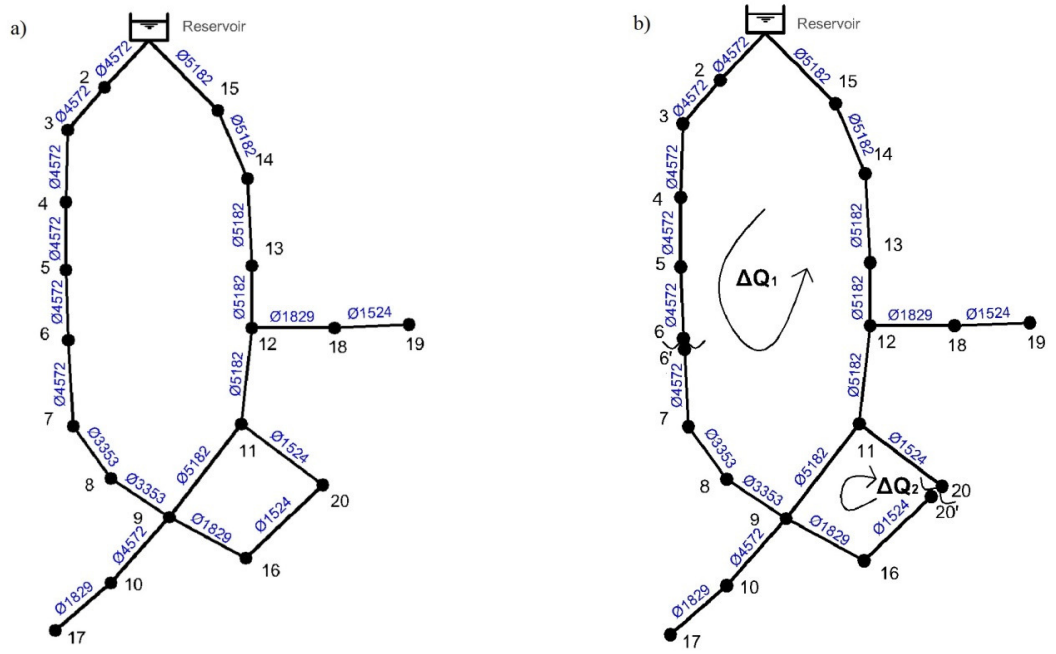

Fig. 3. a) Starting New York's water distribution network; b) Modified fictitious branch network.

It is an example of gravitational water distribution network made out of $n_n = 20$ nodes with $n_r = 1$ source node or reservoir. Nodes and reservoir are interconnected with $n_p = 21$ large pipes. Current disposition of the system cannot satisfy the minimal nodal pressure values of 20 meters of water column. Therefore it is necessary to reconstruct the network in order to meet the given condition in nodes. Changing the diameters of existing pipes is not possible due to the problem of excessive water demand shortfall so the remaining options are to: a) either duplicate the existing tunnel with some of the diameters offered, or b) do nothing. Number of diameters in catalogue for new pipes is 15. Together with the "do nothing" option, this makes 16 possible solutions for every pipe in the network. Fitness function f for this example is made of two parts, first one is investment in the water network I (eq.(3)) and the second is the penalty I_p for failing to meet the minimal nodal pressure values ($p^{min} = 20$ m H_2O).

$$\min_{\{D_k\}} f = I + I_p \; ; \qquad I = \sum_k C_k(D_k)L_k \; ; \qquad I_p = \sum_j \max(O,(p_j^{\min} - p_j)) \cdot C_p \tag{3}$$

In the equation above C_k, D_k and L_k are cost of the new pipe per meter, diameter and length of the pipe respectively, j is the number of a node, p_j is the pressure value in the node j while O is the function whose value is above zero if the $p_j < p^{min}$. Specific value of penalty function is C_p = 15 000 000 $/m.

2.4. Standard ΔQ method inside an optimization algorithm

Standard ΔQ method (section 2.1) is implemented as hydraulic solver inside the optimization algorithm. Equation (1) is solved using iteration. Pre-processing stage and evaluation function calculation includes:

- Pre-processing stage (preceded by a call of GA):

1. Topological graph sorting of network (Fig. 3a), resulting with directed graphs.
2. Minimal basis loop detection and fictitious branch network creation (Fig. 3b).
3. Hydraulic calculation of the branch network. The resulting ΔQ_1 and ΔQ_2 are stored as starting values for flow corrections ΔQ_1^0 and ΔQ_2^0 in further GA calls. It is presumed that ΔQ_1 and ΔQ_2 calculated this way will not differ much from their final values which will be calculated for every call of the evaluation function, thus, this should speed up the convergence of eq. (2).

- Evaluation function calculation:

1. For every tested alternative, "exact" values of flow corrections ΔQ are calculated using the eq. (2).
2. Using calculated flow corrections, the pressure distribution for the fictitious branch network is calculated in only one pass.
3. Fitness function (eq. 3) is calculated.

2.5. Variable ΔQ method inside an optimization algorithm

In standard ΔQ method (section 2.4) each evaluation function's calculation involves the iterative calculation of "exact" flow corrections. In variable ΔQ method, it is assumed that flow corrections are unknown variables, whose values are, together with pipe's diameters values, optimized. The pre-processing stage and evaluation function now includes:

- Pre-processing stage: same as in standard ΔQ method, all of the necessary analyses are done and flow corrections ΔQ_1 and ΔQ_2 for fictitious branch network.
- Evaluation function calculation:

1. For every tested alternative (having pipe's diameters and flow corrections as variables to optimize) only pressure distribution for the fictitious branch network is calculated in one pass.
2. Fitness function (eq. 3) is calculated.
3. Additional penalty function is added to the value of the fitness function, to compensate the fact that used flow corrections are not "exact". If pressure drop between neighbouring nodes where the loop is split (like 6 and 6') is positive in the direction of water flow, the value of penalty function is zero. This solution is feasible meaning that the pressure reducing valve should be installed in the place where the pressure drops occurs. However, if pressure drop between neighbouring nodes where the loop is split is negative, the solution would require a pump between these nodes, and it has to be penalized. The same penalty value as for pressure deficiency is used.

To reduce the search space for flow corrections, the value is entered as multiplication of flow corrections ΔQ_1 and ΔQ_2 computed in pre-processing stage, as $M_1*\Delta Q_1$ and $M_2*\Delta Q_2$. In previous tests it was confirmed that the correct values of ΔQ_1 and ΔQ_2 deviate not more than 25% from the first iteration [8], so multiplicative factors M_1 and M_2 can take a value in relatively narrow space from 0.75 to 1.25. However, the used search space is problem dependant and should be taken with the care.

Since the variable ΔQ method does not solve the "exact" flows, it tends to be the fastest method, but the accuracy of the hydraulic results is degraded compared to the standard method.

3. Results

Both presented approaches share the same modified fictitious branch network (Fig. 3b). There are two loops in the network, larger loop 1 contains reservoir along with the nodes numbered from 2 to 15, while the loop 2 has nodes 11, 9, 16 and 20. Location of the split is arbitrary and inconsequential. Loop 1 was split in the proximity of the node 6 where a new node 6' is introduced as a start node for the downstream pipe. Loop 2 was split close to node 20, with the new node 20' being generated. Flow corrections for loops 1 and 2, ΔQ_1 and ΔQ_2 respectively are introduced as demands in the nodes 6 and 20, and in the nodes 6' and 20' as negative demands or inflows.

Table 1: Obtained optimal solutions

Pipe id [/]	Start node [/]	End node [/]	Existing Diameter D_{old} [m]	New Diameter - **Standard ΔQ** $D_{new\text{-}St\Delta Q}$ [m]	New Diameter - **Variable ΔQ** $D_{new\text{-}Var\Delta Q}$ [m]
7	8	7	3.353	3.353	0
15	1	15	5.182	0	3.353
16	10	17	1.829	2.438	2.134
17	12	18	1.829	2.438	2.743
18	18	19	1.524	2.134	2.134
19	11	20'	1.829	1.829	1.829
21	9	16	1.524	1.829	1.829

Calculated optimal diameters are presented in the table 1 for both methods (last two columns). Pipes not affected by the presented solutions, are omitted from the table 1. Diameter of 0 m represents the "do nothing" option. Computation time is obtained through MATLAB function *cputime*. Standard ΔQ method presented in section 2.4. needed $t_{\Delta Qst}$ = 15 s, while the Variable ΔQ method presented in section 2.5. needed only $t_{\Delta Qvar}$ = 5.5 s.

4. Discussion

Results of two presented ΔQ methods are compared to results obtained using EPANET hydraulic solver: optimization algorithm calls the EPANET.DLL in every pass through evaluation function, gets the results of the simulation and uses them to derive the value for fitness function. Obtained optimal solution using EPANET is the same as one taken from the literature $f_{opt} = 38.6 \; x \; 10^6$ $ [11], proving it as a valid reference. Since ΔQ method approach presented in this paper was programmed in MATLAB environment, to compare the execution times, the same MATLAB environment and optimization algorithm was used with EPANET. Also, the GA settings were the same for every optimization algorithm tested, with maximum of 1000 generations and population of 50 coded solutions. Computer used was an Intel i7-2630QM CPU with 6 GB of RAM memory.

Table 2: Comparison of different optimization algorithms

GA optimization algorithm	Flow c. ΔQ_1 [m^3/s]	Flow c. ΔQ_2 [m^3/s]	Fitness function f [10^6 $]	Relative change of f [%]	CPU time t [s]	Relative change of t [%]
EPANET – based	/	/	38.6	/	390	/
Standard ΔQ - based	14.24	4.28	38.6	0	15	2500
Variable ΔQ - based	13.53	4.40	39.8	3	5.5	6991

Table 2 gives the two main compared parameters: quality of the obtained optimal solution (suboptimal value of the fitness function f) and total computation time (including pre-processing and optimization). In terms of the computation time, both ΔQ methods shown a significant reduction comparing to the computation time t_{epa} = 390s obtained with EPANET. Standard ΔQ method (section 2.4) needed only $t_{\Delta Qst}$ = 15s, which is 2500% faster than EPANET, and Variable ΔQ method (section 2.5) was even faster, taking in average $t_{\Delta Qvar}$ = 5.5s for the computation, which meant that the achieved acceleration is about 7000%. Acceleration in both ΔQ methods is primarily due to the way how evaluation function is handled. The pre-processing stage, called just once, performs a large portion of necessary analysis, therefore later calls for evaluation function take much less computation time than in reference algorithm. In

faster Variable ΔQ method, evaluation function solves only network pressure distribution in a single pass, while in Standard ΔQ method, computation of correct flow correction values is also done.

The reference EPANET-based algorithm founds the same optimal value of fitness function as given in the literature. The Standard ΔQ method also founds the same optimal value of the fitness function. This was expected, since the Standard ΔQ method uses the "exact" hydraulic solution. However, the Variable ΔQ method has found the suboptimal solution of fitness function $f_{\Delta Qvar} = 39.8 \; x \; 10^6$ \$, which is 3% higher than the optimal one. Reason for the suboptimal solution degradation is mostly due to hydraulic result's inaccuracy, which can be seen in Table 2 where both “exact” and “optimized” values of flow corrections ΔQ are presented. Having in mind the achieved speed acceleration, this can be satisfying and encouraging result.

5. Conclusions

During the optimization of water distribution networks, hydraulic computation of network inside an evaluation function consumes the most of the computation time. In this paper, use of the ΔQ hydraulic computation method inside an optimization algorithm is presented, through two different approaches. Each of these approaches requires a pre-processing stage, in which the loops in original water network are detected through minimal basis loop detection algorithm, split into the fictitious branch network and initial flow corrections computed. After that, in standard ΔQ method the correct values of the flow corrections are computed inside each evaluation function, while in Variable ΔQ method the values of flow corrections are, as multiplicative factors of their initial value, added to the list of variables for optimization. Presented optimization algorithms were tested on the example of New York water distribution network reconstruction. In both ΔQ methods significant computation time reduction was achieved, because ΔQ method solves fewer equations than EPANET's hybrid node-loop method. Also, in the Variable ΔQ method the network hydraulics is solved only once in the pre-processing stage which makes it even faster. However, only Standard ΔQ method managed to compute a global optimum. Second approach has slight solution's degradation caused by the hydraulic accuracy problem. However, it can still be used as hot-start for Standard ΔQ method.

Acknowledgements

The authors express the gratitude to the Serbian Ministry of Education and Science for the support through the project TR37010: "Rain water drainage systems as part of the urban and transport infrastructure”.

References

[1] G. Burger, W. Rauch, Parallel Computing in Urban Drainage Modelling: A Parallel Version of EPA SWMM, in Proc. The 9th international conference Urban Drainage Modelling, Belgrade, Serbia: Faculty of Civil Engineering, University of Belgrade, 2012.
[2] L. Smith, Q. Liang, P. Quinn, A Flexible Hydrodynamic Modelling Framework for GPUs and CPUs: Application to the Carlisle 2005 Floods, in Proc. International Conference on Flood Resilience: Experiences in Asia and Europe, Exeter, United Kingdom: Centre for Water Systems, University of Exeter, 2013.
[3] J. Marques, MC. Cunha, J. Sousa, D. Savić, Robust optimization methodologies for water supply systems design, Drinking Water Engineering and Science, 5 (2012) 31-37.
[4] F. Di Pierro, L. Berardi, S.T. Khu, D. Savic, Efficient multi-objective optimal design of water distribution networks on a budget of simulations using hybrid algorithms, Environmental Modelling & Software, 24 (2009) 202-213.
[5] E. Todini, S. Pilati, A gradient method for the analysis of pipe networks. International Conference on Computer Applications for Water Supply and Distribution, Leicester Polytechnic, UK, 1987.
[6] H. Cross, “Analysis of flow in networks of conduits or conductors”, Bulletin, University of Illinois, No. 286, 1936.
[7] J.H. Holland, Genetic algorithms, Scientific american, 1992, pp. 66-72
[8] M. Stanić, D. Ivetić, D. Prodanović, Ž. Vasilić, Improvement of application of genetic algorithms in optimization of pressurized pipe networks, Proc. 16th Conference of Society of Serbian Hydraulic Engineers, Donji Milanovac, Serbia, 2012.
[9] P. F. Boulos, K. E. Lansey, B. W. Karney, Comprehensive Water Distribution Systems Analysis Handbook for Engineers and Planners, second edition, American Water Works Association, 2006.
[10] K. Jha, Automatic minimal loop extraction and initialisation for water pipe network analysis, International Journal of Simulation Systems, Science & Technology, 8 (2007) 8-19.
[11] G.C. Dandy, A. R. Simpson, L. J. Murphy, An Improved Genetic Algorithm for Pipe Network Optimization, Water Resources Research, 32 (1996) 449–458.