Technion – Israel Institute of Technology

Stephen and Nancy Grand Water Research Institute

POWADIMA – Final Report

Uri Shamir, Elad Salomons, Alexander Goryashko

January 2004

Introduction

The aim of the **POWADIMA** (**PO**table **WA**ter **DI**stribution **MA**nagement) project is to develop a prototype real-time optimal-control system for a large water distribution network. The project was conducted during the years 2001-2003 for the European Union, under its Fifth Program, as a collaborative project of four Universities and their water company sub-contractors (see web page http://www.ncl.ac.uk/powadima/):

- The University of Newcastle-upon Tyne, UK
  Professor Derek Jamieson – Project Coordinator
  Thames Water – evaluation of the SCADA systems
- Grand Water Research Institute, Technion–Israel Institute of Technology
  Prof. Uri Shamir – Team Leader
  Subcontractor: Department of Water, Sewage and Drainage, Haifa Municipality - Yaron Ben-Ari, Director
- Universidad Politecnica de Valencia
  Prof. Fernando Martinez – Team Leader
  Subcontractor: Aguas de Valencia - subcontractor
- Universita degli Studi di Ferrara, Italy
  Professor Marco Franchini – Team Leader
  Subcontractor: META, Castelfranco Emilia

This report contains a detailed description of the work carried out by our team at the Grand Water Research Institute with application to part of the Haifa water distribution system – called Haifa-A – and the results that were obtained for each POWADIMA Work

Package during the project duration 2001-2003. The report is also to serve as input to the final report of the project as a whole.

The Haifa-A Network and Problem Formulation

Haifa is a port city in the north of Israel, with a population of about 270,000. It covers a topography that rises rapidly from sea-level to elevations above 350 meters. The Haifa city water distribution system is fed from three connections to the national water system operated by Mekorot (Israel National Water Company). The annual water supply to the city is about 30mcm. There are about 150km of main pipes, 35 tanks with a total volume of about 58,000m^3, 18 pumping stations and some 35 pressure zones (due to the very pronounced topography). A Master Plan is being prepared for the entire city's network, but there still is no hydraulic model available for the system.

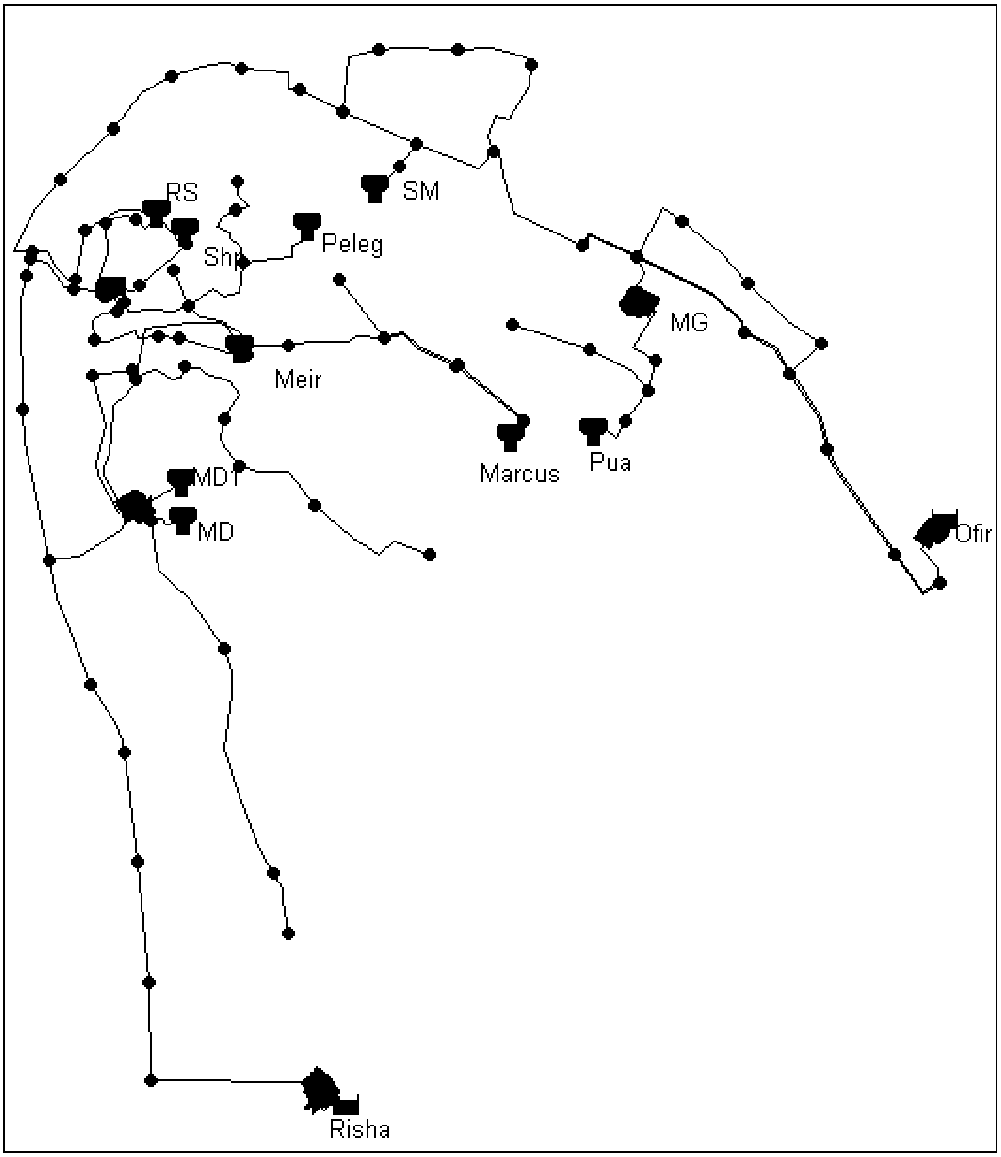
A portion of the system – called Haifa-A (shown in Figure 1) - was selected for this study. This part of the system was selected for the following reasons:

- It is defined distinctly by its supply points.
- It has limited influence on other parts of the system.
- Most of the required information is available.
- The ongoing upgrade of the city's SCADA has begun in this area.
- Haifa-A is "interesting" enough for our purposes in the POWADIMA project.

The Haifa-A network is shown in Figure 1. It contains the supply points, the main feeder line, six pressure zones with their tanks and about 20% of the city distribution network. (See also a schematic of Haifa-A, in a later section).

The Haifa-A network is located on the western and northwestern side of the city. It is fed from two major pumping stations: Vadi-Risha in the south and Ofir in the north, and serves a population of about 60,000. The peak daily consumption, over the year, is about 21,000 m^3/day, while the average daily consumption is about 13,000 m^3/day. The peak hourly demand is estimated to be about 10% of the peak daily demand, 2,100 m^3/h. The total storage is 12,900 m^3 which is 61% of the peak daily consumption.

Figure 1: Haifa-A network layout.



Layout, hydraulic elements and data

The Haifa-A network model consist 126 pipes, 112 junctions, 9 tanks and 17 pumping units located in 5 pumping stations. The current network has no remotely-controlled valves of any kind. There are some valves that are locally controlled. The 126 pipes range in diameter from 4" to 24", with a total length of about 41,500 meters. The CHW coefficient of all pipes is estimated to be 120. Of the 112 junctions only 65 are demand junctions. The topography is quite pronounced – the lowest junction is at an elevation of 5 meters above sea level and the highest is 230 meters above sea level. Due to the topography there are 6 pressure zones, each fed by a dedicated set of pumps and otherwise separated from the other zones. These pressure zones are also Demand

Management Areas (DMAs) – see below. The 9 tanks have a total storage of 12,900 m^3. Tanks are located in all six pressure zones but most of the storage - 9,150 m^3 - is located in the lowest zone (DMA 1). The additional 3,750 m^3 are spread over the other 5 pressure zones, as shown in Table 1.

Table 1: Tanks in the Haifa-A Network

Number	Name	Bottom Elevation (m)	Height (m)	Volume m^3)
1	Pua	114.00	7.00	1,250
2	Mahane David 1	65.00	6.00	2,000
3	Mahane David 2	112.90	6.70	1,000
4	Markus	240.50	4.50	500
5	Shprinzak	102.00	5.00	200
6	Stela Maris	70.00	5.00	2,150
7	Peleg	163.80	4.50	500
8	Meir	102.00	5.50	300
9	Ramat Shaul	71.50	6.70	5,000
Total	**-**	**-**	**-**	**12,900**

There are 17 pumping units, grouped in 5 pumping stations, as shown in Table 2.

Table 2: Pumps in the Haifa-A Network

Number	Station	Electricity connection (A)	Pump	Power (HP)
1	Vadi-Risha	3 X 400	1	125
			2	125
			3 (reserve)	125
2	Mahane David	3 X 250	1	150
			2 (reserve)	150
			3	25
			4	25
3	Shprinzak	3 X 125	1	50
			2	50
			3	30
			4	20
4	Anilevichz	3 X 160	1	60
			2	50
			3 (reserve)	50
5	Ofir	3 X 630	1	200
			2	200
			3 (reserve)	200

As can be seen from Table 2, the pumping units in each pumping station are not all identical, so each is modeled separately, and they cannot be grouped. In some pumping stations there are reserve pumps for cases when a working pump fails or for the option of cycling between the pumps.

There are no controlled valves in the system. There is a valve (link 9 in the EPANET hydraulic model) that closes when Mahane David pumping station is switched on. For modeling proposes this control valve was removed and a PRV was entered (link V1 in the model). The optimization design process includes this valve as a decision variable for each hour.

The network covers 6 demand management areas (DMAs), as shown in Table 3. DMA1 is supplied from the two sources, and in turn supplies water to the other five DMAs.

Table 3: Demand Management Areas (DMAs)

DMA number	DMA name	Pumping in	Pumping out
1	Ramat Shaul	Vadi Risha Ofir	Anilevichz Shprinzak Mahane David
2	Markus	Mahane David 1+2	
3	Mahane David	Mahane David 3+4	
4	Shprinzak	Shprinzak 3+4	
5	Peleg	Shprinzak 1+2	
6	Pua	Anilevichz	

The total daily demand, for a peak demand day, of Haifa-A is about 21,000 m^3/day. The total consumption is allocated, among the DMAs as shown in Table 4, and within each DMA equally among its nodes, which are listed in Table 4.

Table 4: DMAs, Peak Day Demands, and List of Nodes in each DMA

DMA Number	Peak Daily demand (m^3/day)	Demand nodes
1	13,313	3, 4, 5, 6, 7, 8, 9, 30, 33, 34, 35, 108, 109, 36, 37, 110, 111, 38, 39, 68, 40, 67, 66, 65, 41, 69, 70, 72, 73, 74, 76, 75, 77, 81, 80
2	2,325	56, 106, 107, 25, 29, 26, 64, 27
3	775	11, 17, 18, 19, 22, 23
4	1,288	62, 61, 58, 45, 1, 13
5	1,136	49, 50, 51, 52, 53, 54, 55
6	2,126	12, 21
Total	**20,963**	**-**

Current operating routine

At present the system is operated mainly using "set points", i.e., local control for the pumping units based on water levels in the tanks. Each pump is assigned two water levels, one to switch on and one to shut off. There is no consideration of energy use and no special attention is given to the electricity tariff. The current operation approach is considered by the operators to be satisfactory, and there is no special effort at this time to try and reduce the operating cost. An EPANET model with these simple control rules in it was constructed. It has been used to compare the optimization results with the current operation costs, which is done in Work Package 7 below.

Constraints

The following constraints are to be considered in the optimal operation design:

- A minimum pressure of 25 meters should be maintained at all demand nodes.
- Minimum and maximum tank levels: The operators wish to keep the water level in the tanks within some range. These constraints are to keep storage for emergency uses, such as fire fighting, and to avoid waste of spilling water from the tanks and to absorb the effects of communication delays and errors in the SCADA system. How these limits are determined is not explicit, and we have simply accepted the values as given. An evaluation of the tradeoff between the safety margins (emergency storage at the bottom, free space to avoid spill at the top) and energy cost could be the subject of another study.
- Tank levels at "6 am": For reasons of supply reliability the operators wish to have the tanks practically full in early morning, at the end of the low demand period and before the demands increase and the energy tariff rises. This means that we should maintain high water levels in the tanks at the beginning of the day – at 6 am. In Haifa the "6 am" is actually 7 or 8 am (depending on the time of year), since this is the time that the electricity tariff changes. The water levels at this time should be higher than a specified value. The constraint was introduced in 6 tanks out of the 9. It was removed from the small 200 to 500 cubic-meter tanks since these tanks do not actually function as storage tanks and can be filled and emptied in an hour or two. It should be noted that the constraint is not imposed at a fixed hour throughout the year, since the low tariff period ends at different hours (sometimes at 7am and sometimes at 8am) in different months.

The following “6 am” constraints were used:

Table 5: Minimum Levels Required in Tanks at “6 am”

Number	Name	“6am” level (m)
1	Pua	3
2	Mahane David 1	3
3	Mahane David 2	3.1
4	Markus	-
5	Shprinzak	-
6	Stela Maris	3.5
7	Peleg	2.7
8	Meir	-
9	Ramat Shaul	5

- Energy: The capacity of the electricity connection in some pumping stations is limited. For example the connection at “Shprinzak” station is 3X125 Amperes. This means that only pumping units with a total of about 115HP can be operated at any given time. At this station there are 4 pumps with a total capacity of 150HP so the optimization will seek for each time the best combination of pumps within this constraint.

Work Package 2: Use of an ANN to Replicate a Simulator of the Water Distribution Network

The objective of WP2, as stated in the project proposal, was to use of an ANN to replicate both water-quantity and quality aspects of a hydraulic simulation model and its application to the experimental networks. As the project progressed, water quality was dropped, and emphasis was placed on water quantity and saving of energy costs.

In order to gain experience with the development of an ANN for a water distribution system, the first phase was to do so for the simple AnyTown network (Walski, 1987). Initial experimentation indicated that this network in its original form did not provide sufficient diversity of function for the purpose of testing the suitability of an ANN. Some modifications were therefore introduced in the model of the network, and the resulting model was called AnyTown Modified (ATM). The details of this model, the ANN and the results obtained with it, will be described below.

We begin with a discussion of several generic aspects of constructing an ANN for a water distribution system, as this will set the stage for considering the specifics of developing the ANN for both ATM and Haifa-A. Let us begin by emphasizing that the ANN is to serve in a particular role, namely to replace the simulator in a real-time on-line control system. We therefore require the ANN to satisfy two criteria:

1. It is more computationally efficient than the simulator it replaces, which is EPANET in our case. The ANN's computation time must be such that the optimization for the operation planning horizon (see below) is acceptable.
2. It is sufficiently accurate and robust to serve as an emulator of EPANET in the context of the optimization scheme for operation of the network over a planning horizon of one day ahead (possibly somewhat longer, to deal successfully with the issue of end-state, see below).

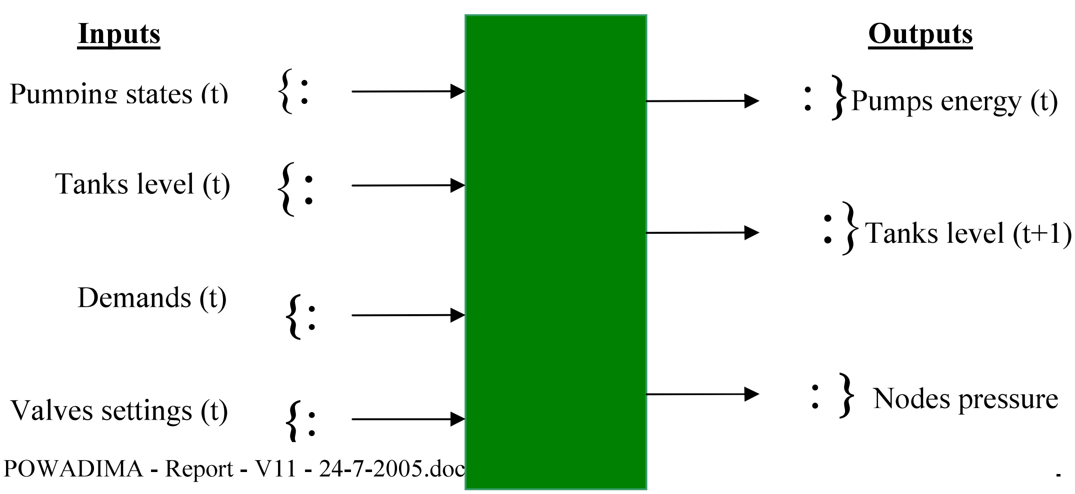
Since the optimization routine is based on a Genetic Algorithm, the network simulator or its replacement ANN has to be run very many times for each optimization: in the case of Haifa-A, between 4,000 and 24,000 times for each optimization (population size = 8, between 500 and 3,000 generations), i.e. every hour. A simulator, such as EPANET, would take more computation time than can be allowed.

<u>ANN architecture</u>

It should be emphasized that there is no standard way of constructing the architecture of ANNs, nor are there fixed ways for selecting the set of training and testing data and for choosing the training algorithm. These choices must be guided by an intimate familiarity with the system being modeled and the application for which the ANN is to be used.

Based on experience with an EPANET model of the Haifa-A network, the input-output structure shown in Figure 2, with 29 input variables and 23 output variables (listed in a later section) was selected for the ANN.

Figure 2: Input-Output Structure of the Haifa-A ANN



This is a "feed-forward" ANN, i.e. it produces the output by moving ahead from the input through the network. The ANN has a single hidden layer. The number of nodes in this layer is varied during the experimentation between 40 and 90, as will be described later.

As a rule it is useful to scale the inputs and outputs so that they always fall within a specified range – usually [0,1] or [-1,1]. In first case [0,1] a *sigmoid activation function* is used for the neuron:

$$f_s(S) = 1/(1+\exp(-\alpha S)) \tag{1}$$

For the range [-1,1] the sigmoid is the same as the *hyperbolic tangent* activation function:

$$\varphi(S) = (\exp(\alpha S) - \exp(-\alpha S))/(\exp(\alpha S) + \exp(-\alpha S)) \tag{2}$$

In (1) and (2) $S = \sum_i w_i x_i + b$; w_i is the weight of the connection i to neuron and b is the neuron bias.

Other suitable ranges and functions can also be used. The reason for the choice of the sigmoid or hyperbolic tangent functions is that they are "squashing" functions – since their slope approaches zero as the input gets large, they compress an infinite input range into a finite output range. While this is a desirable property, it causes a problem when a steepest descent algorithm is used to train the ANN, since the gradient can be very small; which results in small changes in the weights and biases, even when the weights and biases are far from their optimal values. A possible way of avoiding this effect is to limit the range of outputs to [(0+delta) , (1-delta)], where delta is a small number, for example <0.1.

Another approach for scaling network inputs and outputs is to normalize the mean and standard deviation of the training set so that they will have zero mean and unity standard deviation. Unfortunately, there is no single "correct" scaling procedure, and the ultimate test is whether the network is successful.

Training and Testing

Development of an ANN is carried out in two steps: *training* and *testing*. Both use vectors of inputs and outputs that are generated by the hydraulic simulator, which is considered to be the true system. Obviously, the simulator itself is not completely accurate, but we rarely if ever have input-output pairs measured on the real system, and definitely not in sufficient numbers to allow proper training of the ANN. Therefore, the simulator is our "real system".

In the training phase, a set of input and corresponding output vectors is used. In the testing phase, another set of input vectors is used to generate outputs, which are compared with the known outputs. Error measures are computed in both phases.

ANN Performance

Measures of performance of the ANN are established by comparing its output to that of the simulator. Before we describe and discuss the performance measures used in our study, it is worth considering accuracy of the ANN in the context of the optimization package. As indicated above, the ANN is run many times in each optimization run. Most of these runs have only a "transient" value, in the sense that they merely serve to drive the solution towards a solution that it is feasible (satisfies the constraints) and efficient (has a "good" value of the objective function, hopefully one that is close to the global optimum, which in a GA application is not known with certainty).

Thus, the true value of an ANN is established only in the context of this use, for which it is designed. More precisely, the absolute accuracy of an ANN in itself is not as important as the test whether it is accurate enough, robust and fault-tolerant in simulating the real system in the context of the GA optimization. Still, the accuracy of the ANN itself should exceed some threshold level, otherwise it may perform poorly. Our experience with Haifa-A indicates that an RMSE (see definition below) of 3% is a minimum.

Developing an ANN is basically a heuristic process, in the sense that it is somewhat of an art. Experience does provide guidance, but the development of a successful ANN for a particular application requires trial-and-error, keeping in mind during this process the specific purpose for which the ANN is being designed.

Performance measures are calculated either with the original values of the outputs or with *normalized data*. The normalization in based on the range of output values, which is commonly transformed into a range [0,1] or [-1,1]. In this project, the range [0.05,0.95] was also used. This latter range may have an influence on the ANN training process. This is because the sigmoid functions in the ANN are flat near the ends, thereby causing an accumulation of normalized values from a wide range of original values, and effect that is modified if the selected range of the normalized values is less than the range of definition of the sigmoid function.

Error Measures

We use several errors measures to evaluate the fitness of the ANN as a replacement of the simulator in the optimization package. *P* vectors of inputs are used. Each input vector *p*=1,…,*P* generates a computed output vector t_{pj}, j=1,…,*M*, for which we have the vector of true outputs, O_{pj}. *P* here stands for the number of training or testing vectors, since the error is computed for each of them separately. The standard performance measure is:

$$\textit{Root mean square error (RMSE)} = \varepsilon_{rms} = \left(\sum_{p=1}^{P}\sum_{j=1}^{M}(t_{pj} - O_{pj})^2 / PM\right)^{1/2} \qquad (3)$$

Where:

- t_{pj} is the true (EPANET) value of output *j* when the input vector is *p;*
- O_{pj} is the computed (ANN) value of output *j* when the input vector is *p*.

The RMSE for the normalized values does not provide information about the absolute values of the errors. A measure that captures this information is computed with the original output values:

$$\textit{Relative RMSE (j)} = \left(\frac{1}{\mathrm{P}} \cdot \sum_{\mathrm{i}} \left((\mathrm{Q}_{\mathrm{i,j}} - \mathrm{t}_{\mathrm{i,j}}) / \max(\mathrm{t}_{\mathrm{i,j}}, 1)\right)^2\right)^{1/2} \qquad (4)$$

Where:

- *max(* t_{pj}*,1)* in the denominator is to avoid very large deviations, which occur when t_{pj} (by chance) is small ,i.e. less than 1.

The value (2) is an average error for each output variable *j*, taken over all input vectors.

In addition to the average value of the Relative RMSE we compute the *Maximum Relative RMSE* over all input vectors for each output vector *j* as:

$$\textit{Maximum Relative RMSE (j)} = \max\left(\left((\mathrm{Q}_{\mathrm{i,j}} - \mathrm{t}_{\mathrm{i,j}}) / \max(\mathrm{t}_{\mathrm{i,j}}, 1)\right)^2\right)^{1/2} \qquad (5)$$

The values (4) and (5) depend not only on the absolute error but on the value of the true variable itself. For example, let the true tank level value be 0.5m and the ANN output is 1m. Then the absolute error will be 0.5m and the relative error will be 50%. On the other hand, if the true value is 6m and the ANN output is 5.5m, then the absolute value is the same 0.5m but the relative value is only 8.6%. Thus a large value for the maximum relative RMSE does not directly indicate large absolute errors.

Training should not be continued beyond the point where an acceptable error is achieved. Further training may result in *over-fitting.* This means that while the resulting ANN has a good error measures, it has less robustness to forecast outputs for vectors which were not in the training set. This is analogous to using a polynomial of high degree to fit through points in a correlation study, which results in a curve that approximates the points well but has such oscillations as to poor in predicting the dependent variable at points not included in the training set.

Training Algorithms

Once the network weights and biases have been initialized, the network is ready for training. The training process requires a set of examples of network input vectors **X** and corresponding output vectors **Y**. During training the weights and biases of the network are adjusted iteratively to minimize the network performance measure – the RMSE of the normalized outputs of the training set.

Training is performed with sets pairs of input-output vectors. The process is iterative: successive pairs are used to "feed" the iterative process. A back-propagation algorithm for feed-forward ANNs is usually implemented in *batch mode* – i.e., the weights and biases of the network are updated only after the entire training set has been applied to the network. Each iteration of the batch training is named *an epoch*. The gradients calculated with each training pair of the epoch are added to produce the change in the weights and biases.

Several options are available for training algorithms. Basically, these are optimization methods that use the gradient of an objective function – which measures the goodness of fit of the predicted output to their known values - to make successive moves, in which values of the parameters (weights) are adjusted in a way that reduces the value of the objective function.

We considered three training algorithm:

- Standard back propagation - represented by the gradient descend algorithm with momentum (see below). Momentum is an acceleration factor, which multiplies the correction term in the steepest descent algorithm, designed to improve convergence.

- Resilient back propagation - a heuristic modification of the standard steepest descent algorithm (Riedmiller, M., and H. Braun, 1993).
- A version of the conjugate gradient algorithm by Polak and Ribiere (Hagan, et al., 1996).

The classical back propagation training algorithms have one common disadvantage: they adjust the weights in trained ANN in the steepest descent direction (negative of the gradient). But it turns out that, although the function decreases most rapidly along the negative of the gradient, this does not necessarily produce the fastest convergence. The Polak-Ribiere has better convergence properties as it uses *conjugate directions*.

The classical approach for adjusting the weights w_{ij} in the training process is the gradient descent algorithm with momentum:

$$\Delta w_{ij}(n) = -\lambda(\partial E/\partial w_{ij}) + \mu\Delta w_{ij}(n\text{-}1) \qquad (6)$$

Where $\Delta w_{ij}(n)$ and $\Delta w_{ij}(n\text{-}1)$ are increments (adjustments) to the weight of the link between node i and j of the ANN during epochs n and $(n\text{-}1)$. λ and μ are called *learning rate* and *momentum*, respectively (ASCE Task Committee I, 2000, Eq. 4, p. 117). They are selected, heuristically, to accelerate the convergence of the iterative optimization process.

As mentioned earlier, the flat parts at the ends of the sigmoid function slow down the iterative training process. To overcome this drawback, a *resilient back-propagation procedure* uses only the sign of the derivatives in determining the direction of the weight update. The update value for each weight (bias) is increased by a factor δ_+ when the derivative of the performance function with respect to that weight has the same sign for two successive iterations. The update value is decreased by a factor δ_- when the derivative with respect to that weight changes sign from the previous iteration.

Standard back-propagation algorithms adjust the weights in the direction of the steepest descent (negative of the gradient). But it turns out that while the function decreases most rapidly along the negative of the gradient, this does not necessarily produce the fastest convergence. Faster convergence is achieved by conjugate gradient algorithms, where each move direction is orthogonal to the one in the previous step. It has been proved that conjugate gradient algorithm converge quadratic ally, i.e., the error decreases as the square root.

One versus Many ANNs

The ANN converts a vector of inputs to a vector of outputs. It is possible to do so with a single ANN for the entire output vector, or create separate ANNs for groups of output variables. The advantage of separate ANNs is that their accuracy can be expected to be better than that of a single ANN, since they concentrate on a group of output variables with similar properties. The disadvantage may be in the amount of work required to create and then use the ANNs. A decision regarding the use of a single or several ANNs must be based on experimentation, since it depends on structure of the ANNs, and the number of input and output vectors.

For the AnyTown Modified case (see below) we tried with a number of ANNs, which indeed showed better accuracy than a single ANN for all output variables. However, for the Haifa-A system we reverted to using a single ANN for the entire system, due to considerations of computational efficiency.

ANN for AnyTown Modified

As stated above, in order to gain experience with training an ANN of a water distribution network and use in with the GA algorithm, we began with the simple AnyTown model (Walski, 1987). However, some modifications were introduced. The main deficiency of the original network for our project is the lack of adequate storage capacity to allow the system to be operated efficiently. Other changes were also found useful, to make the AnyTown Modified (ATM) model more suitable and challenging for using the ANN and GA in combination. The modifications made are as follows:

- The units were converted into SI units.
- An additional storage tank was introduced.
- A small number of pipes were added (mainly in the northern part of the network).
- A time-variable electricity tariff was introduced.

Modified AnyTown model

The new data are:

- A third tank was introduced at node 265. The new tank has the same properties as the two old tanks. It is connected to junction 140 with a 30.48 ft., 203.2mm, chw=120 pipe.
- All tanks in the system have the following properties:

Property	Value (m)
Elevation	65.532
Initial level	3.048
Minimum level	3.048
Maximum level	12.192
Diameter	15.24

- The demands at each of the junctions 55, 75, 115 is 24.384 m^3/hour.
- The diameter for all new pipes (54, 68, 70, 72, 74 and 74) is 304.8mm, and their coefficient is Chw=120.
- The electricity tariff changes during the day, as follows:

Hours	Rate (cost units/KWh)
0 – 7	18.14
7 – 17	35.28
17 – 21	80.97
21 – 24	18.14

- For cost calculation purposes the pumps efficiency curve is given by:

$$\eta(\%) = -5.589\mathrm{E}-5 \cdot Q^2 + 0.1216 \cdot Q$$

Where Q is the pump flow in m^3/hour.

The problem definition is: to design the operation of the three pumps during 24 hours so the minimum pressure at any of the junctions will not fall below 30 meters at all times, and the electricity cost is as low as possible.

The EPANET toolkit was used as a hydraulic solver. Four ANNs were trained to perform predictions of different outputs:

1. ANN for tank level prediction:

   12 Inputs: tank1 level(t), tank2 level(t), tank3 level(t), pump1 state(t), pump2 state(t), pump3 state(t), total demand(t), time1(a.m/p.m) , time2 (1…12), dem.pat.mult(t), dem.pat.mult(t+1), dem.pat.mult(t+2);

   8 Hidden units

   3 Outputs: tank1 level(t+1), tank2 level(t+1), tank3 level(t+1);

2. ANN for pump energy warning prediction.

   7 Inputs: tank1 level(t), tank2 level(t), tank3 level(t), pump1 state(t), pump2 state(t), pump3 state(t), total demand(t);

   6 Hidden units

   1 Output: warning (t+1);

3. ANN for prediction of the pressure at junction 170 – which is considered a critical location in the system;

   7 Inputs: tank1 level(t), tank2 level(t), tank3 level(t), pump1 state(t), pump2 state(t), pump3 state(t), total demand(t);

   4 Hidden units

   1 Output: pressure value

A fourth ANN was also tried: to predict *warnings* that are generated by the EPANET. There are several types of warnings that are created by EPANET, indicating that something unacceptable has occurred in the numerical solution of the simulator.

The input vectors for training and testing were generated by the EPANET toolkit program under the following conditions:

- Total length of the experiments = 1200 (50 cycles times 24 hours).
- Initial states of the tanks were chosen randomly from a uniform distribution over the range [1.4,…,6].
- The state of each pump (on/off) was chosen randomly for each hour.
- For ANN predictors of the tanks level, pump energy and pressure at Node 170 (ANNs 1,3 and 4):
  - # of training sets = 480 (20 days).
  - # of testing sets = 720 (30 days).

The number of training epochs did not exceed 500 for any of the four ANN's. The prediction accuracy of the testing stage of four output variables (except "warning") is shown in Table 6.

Table 6: Error Measures for AnyTown ANN

Prediction value	**Relative RMSE**	**Maximum relative RMSE**
Tank level 1	0.11	0.59
Tank level 2	0.1	0.41
Tank level 3	0.09	0.33
Pressure at node 170	0.1	0.3
Energy	0.05	0.12

Sample results are shown in Figures 3-5, for the first 100 sets (only these are shown, for clarity of the presentation). The results are shown for pressure at a junction (Junction 170), level of a tank (Tank 2) and the energy used by a pump (Pump 3). Results of EPANET are shown in blue, ANN results in red.

Figure 3: ANN and EPANET Results for pressure at Junction 170

(RMSE = 9%; Relative RMSE=10%)

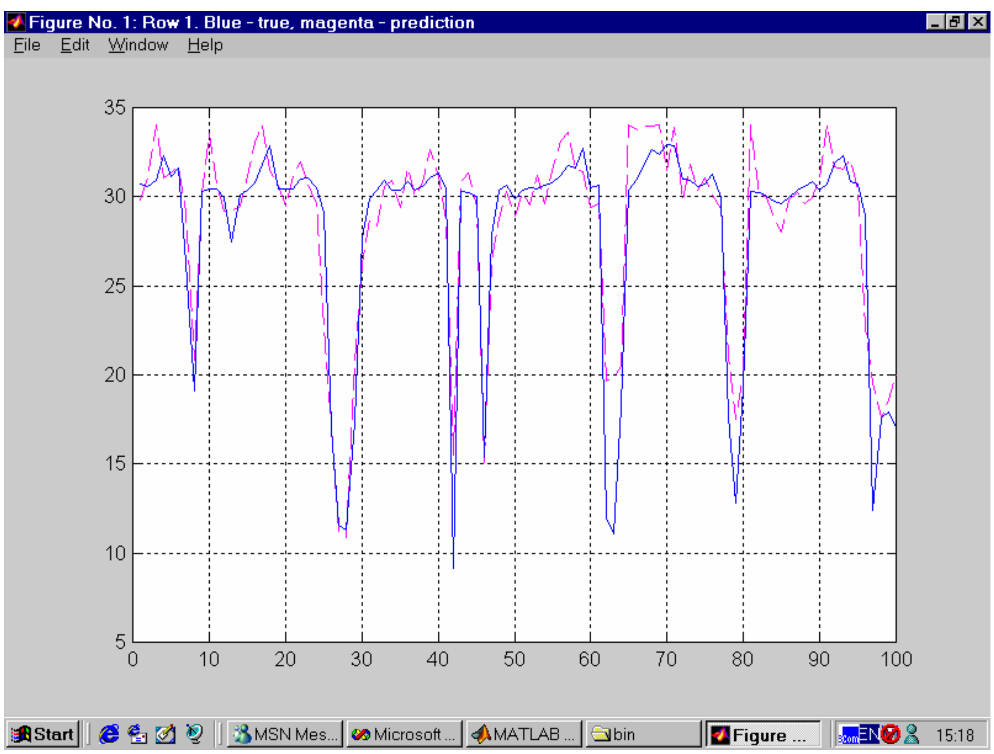


Figure 4: EPANET and ANN results for level in Tank 2 level

(RMSE=13%; Relative RMSE =12%)

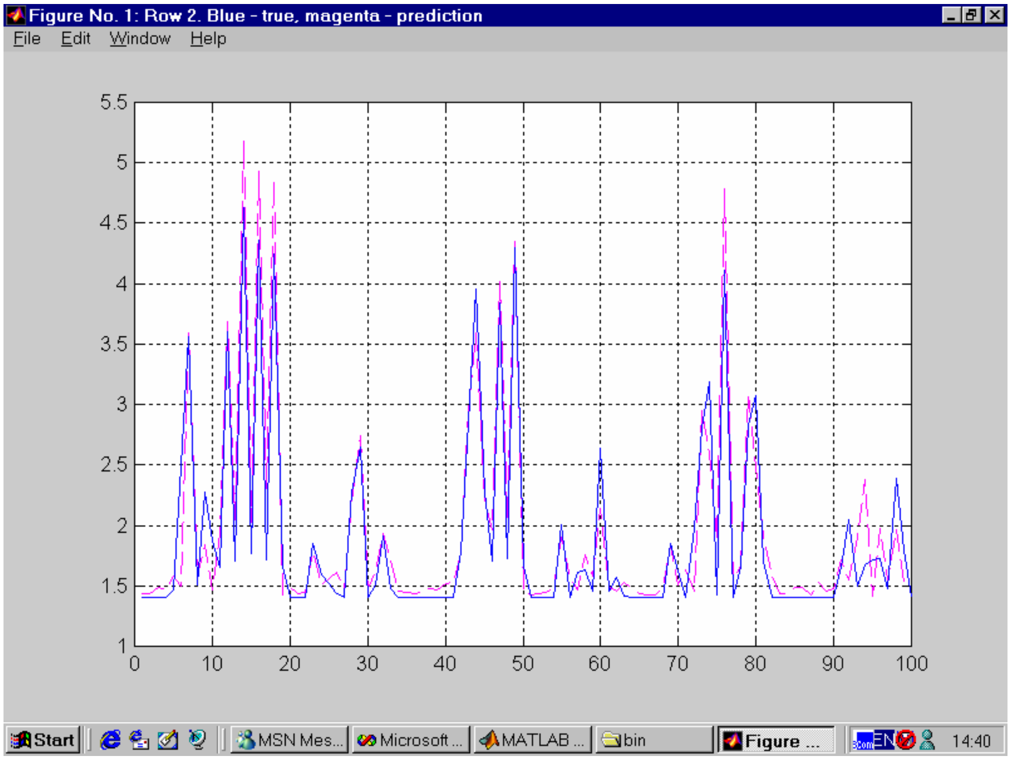
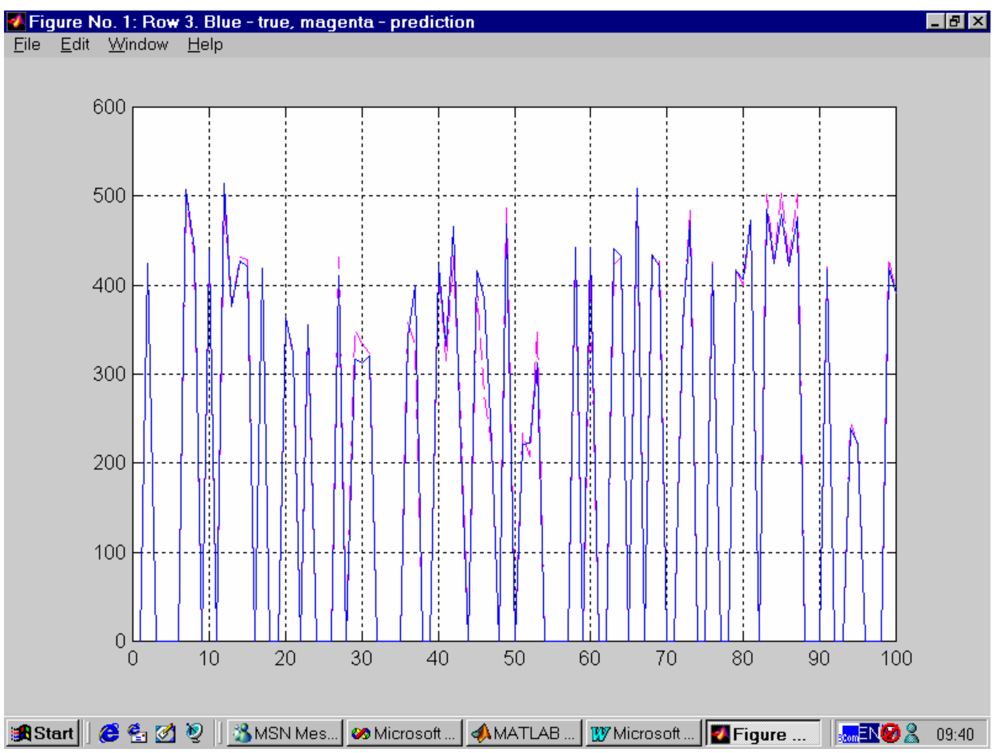


Figure 5: EPANET and ANN Results for Pump 3 Energy

(RMSE =4%; Relative RMSE =5%)



Conclusions Resulting from Development of the ANN for AnyTown Modified

Based on the experimental results above, it may be concluded that:

- On this stage our main aim was to demonstrate that very simple ANN's (and as consequence very computational effective) could achieve not very big prediction errors. However, to prove that the errors are "not very big" indeed, realization of the GA+ANN optimization package and estimates of the accuracy corresponding optimization process is absolutely necessary condition.
- The ATM model is quite small. It must be expected that training an ANN for a large real system would be much more demanding, since the complexity of the ANN increases substantially with its size.
- The lowest prediction accuracy was obtained for tank levels. This is due to the fact that the levels appear in the ANN twice: once as outputs and then as inputs for the following hour (see Figure 2 above). This means that errors for tank levels are propagated in a different way than for variables that are only outputs and do not "feed back" into the next hour.

In addition to the errors of the normalized values, we have also examined the errors of the absolute values of the variables. This is important for the overall performance of the GA+ANN procedure. We will return to this issue when the results of the Haifa A simulation by ANN are discussed

ANN for Haifa-A

Experience with the AnyTown model proved quite conclusively the advantage of creating separate ANNs for parts of the system, in terms of computational efficiency. Still, for Haifa-A we designed a single ANN, since we found that it provides the best tradeoff between computational efficiency and accuracy when it is combined with the GA for optimization.

Tables 7 and 8 list the data for the ANN of Haifa-A. It has 29 inputs - 13 pumps, one valve, 6 hourly demands in the 6 DMAs, 9 tank levels at the beginning of the hour – and 23 outputs – the power consumption at 5 pumping stations, pressure at 9 network nodes, and levels in the 9 tanks at the end of the hour.

Table 7: The 29 Inputs of the Haifa-A ANN

Input #	Input description	Hydraulic element
1.	Pump #1 state (0=off, 1=on)	Ofir1
2.	Pump #2 state (0=off, 1=on)	Ofir2
3.	Pump #3 state (0=off, 1=on)	MG1 (Anilevichz 1)
4.	Pump #4 state (0=off, 1=on)	MG2 (Anilevichz 2)
5.	Pump #5 state (0=off, 1=on)	Risha1
6.	Pump #6 state (0=off, 1=on)	Risha2
7.	Pump #7 state (0=off, 1=on)	MD1 (Mahane David 1)
8.	Pump #8 state (0=off, 1=on)	MD3 (Mahane David 3)
9.	Pump #9 state (0=off, 1=on)	MD4 (Mahane David 4)
10.	Pump #10 state (0=off, 1=on)	Shp1 (Sprinzak 1)
11.	Pump #11 state (0=off, 1=on)	Shp2 (Sprinzak 2)
12.	Pump #12 state (0=off, 1=on)	Shp3 (Sprinzak 3)
13.	Pump #13 state (0=off, 1=on)	Shp4 (Sprinzak 4)
14.	Pressure reducing valve (PRV)	V1
15.	DMA #1 hourly demand	-
16.	DMA #2 hourly demand	-
17.	DMA #3 hourly demand	-
18.	DMA #4 hourly demand	-
19.	DMA #5 hourly demand	-
20.	DMA #6 hourly demand	-
21.	Tank level #1 at time t	Pua
22.	Tank level #2 at time t	MD_UP (Mahane David 2)
23.	Tank level #3 at time t	Marcus
24.	Tank level #4 at time t	MD_D (Mahane David 1)
25.	Tank level #5 at time t	Sph (Shprinzak)
26.	Tank level #6 at time t	RS (Ramat Shaul)
27.	Tank level #7 at time t	SM (Stela Maris)
28.	Tank level #8 at time t	Peleg
29.	Tank level #9 at time t	Meir

Table 8: The 23 Outputs of the Haifa-A ANN

Output #	Output description	Hydraulic element
1.	Pumping station #1 power	Ofir (Ofir1, Ofir2)
2.	Pumping station #2 power	Anilevichz (MG1, MG2)
3.	Pumping station #3 power	Risha (Risha1, Risha2)
4.	Pumping station #4 power	David (MD1, MD3, MD4)
5.	Pumping station #5 power	Shprinzak (Shp1, Shp2, Shp3, Shp4)
6.	Pressure node #1	54
7.	Pressure node #2	3
8.	Pressure node #3	81
9.	Pressure node #4	107
10.	Pressure node #5	13
11.	Pressure node #6	25
12.	Pressure node #7	19
13.	Pressure node #8	109
14.	Pressure node #9	21
15.	Tank level #1 at time t+1	Pua
16.	Tank level #2 at time t+1	MD_UP (Mahane David 2)
17.	Tank level #3 at time t+1	Marcus
18.	Tank level #4 at time t+1	MD_D (Mahane David 1)
19.	Tank level #5 at time t+1	Sph (Shprinzak)
20.	Tank level #6 at time t+1	RS (Ramat Shaul)
21.	Tank level #7 at time t+1	SM (Stela Maris)
22.	Tank level #8 at time t+1	Peleg
23.	Tank level #9 at time t+1	Meir

Training and testing the ANN

The input training set for Haifa-A was generated as follows:

- For each pumping station the value of the pumping hours is chosen randomly from a given range, as listed in Table 9.

Table 9: Range from which pumping hours were selected at random

#Pumping station	Pumping station	Min. pumping hours	Max. pumping hours
1	Ofir	9	14
2	Anilevich	10	14
3	Risha	9	14
4	Mahane_David	26	32
5	Shprinzak	15	20

- The times at which a pump is turned on is selected at random over the 24 hours.
- The demand of the DMA is distributed equally among all junctions of the DMA. Two sets of experiments were conducted. In the first, the demand for each DMA is deterministic and set at 0.7 of the maximum demand for the DMA. In the second set the demand for each DMA was chosen randomly in the range 0.6 times the maximum demand to 0.8 times the maximum demand. The training results for these cases are presented below.
- The state of the single valve is chosen at random over the range (10 m,…,50 m).
- The initial state of each tank is chosen at random in the range (min. tank level +1m, max. tank level –1 m.) before each simulation cycle.
- The range for the tank levels was chosen as (-1 m., maximal head + 4 m.). If a tank level occurs outside of this range the corresponding input vector was not included in the input training set.
- All training/testing vectors were normalized in the range [-1,1].

A suggested rule of thumb for the number of the training vectors is:

No. of Training vectors = c x Total number of ANN weights, where $1 < c < 3$

For Haifa-A, with one hidden layer, the number of hidden nodes was changed between 40 to 90, so the corresponding number of weights ranged between 2080 and 4680. Following the inequality above, the required number of training input sets is 6000. Beyond this number the training did not improve significantly.

The set of the testing vectors contains about 1000 vectors, which is sufficient for stable results regarding of the testing (prediction) errors.

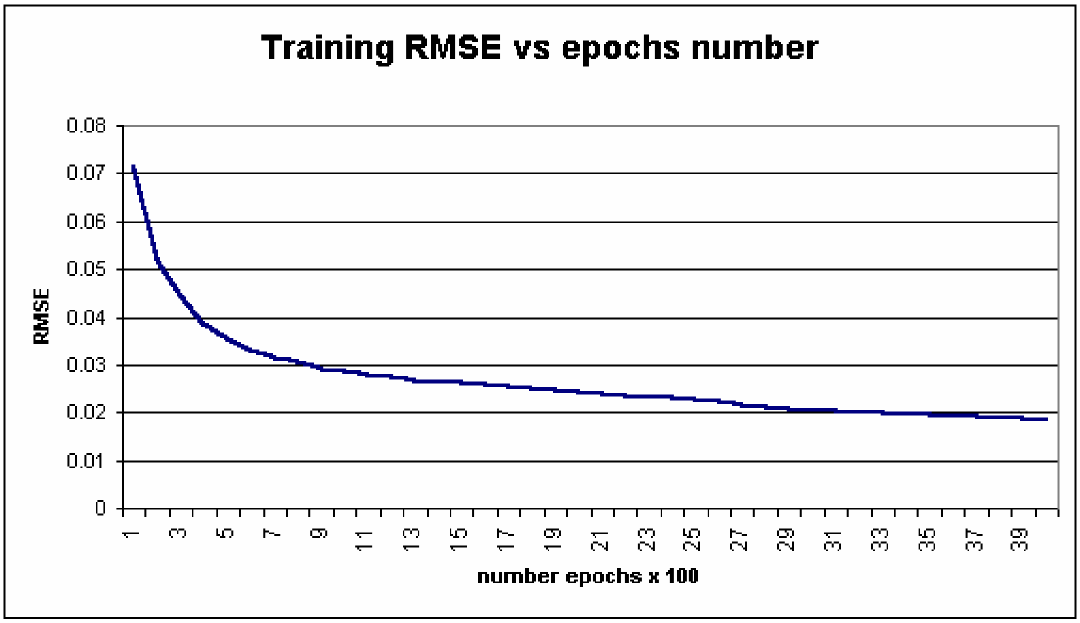
Training of the ANN

The basic problem of the function's approximation by ANN is the optimal tradeoff between the training complexity (number of the weights in ANN and number of vectors in the training set) on the one hand, and the achieved prediction error, on the other. The greater the complexity (which is essentially also a flexibility), the better the chance of obtaining a good fit (low prediction errors). The price of increased complexity is a greater computational effort.

We began to teach the ANN with different back-propagation training algorithms with momentum. Initial attempts to obtain RMSE values less than 5% - at least with less than 5,000 epochs (training vectors) - failed. A distinct improvement of the prediction accuracy was achieved by using the Polak-Ribiere conjugate gradient algorithm.

The result is a set of ANNs with 29 inputs, 23 outputs and between 40 and 90 hidden nodes. Figure 6 demonstrates the decrease of the RMSE with the number of epochs in training the ANN with 90 hidden nodes.

Figure 6: Evolution of the RMSE in training the Haifa-A ANN with 90 hidden nodes



The number of training epochs is limited, to avoid over-fitting (see explanation in a previous section).

The Testing RMSE for normalized output values are shown in Table 10.

Table 10: Testing RMSE values for normalized [-1,1] outputs of the Haifa-A ANN with deterministic and random demands

Hidden units	**40**	**50**	**80**	**90**
Deterministic Demand	2.67%	2.15%	1.56%	1.38%
Random Demand	2.85%	2.54%	1.79%	1.83%

Increasing the number of hidden nodes above 90 did not decrease the testing RMSE, although the training errors did decrease. It was therefore decided to stay with 90 as the maximum number of hidden nodes.

The greatest sensitivity of level prediction experienced with the AnyTown ANN was repeated with the Haifa-A ANN. Errors in level prediction can result in two types of errors in the optimization. The first is a *false negative*, when a feasible solution is found infeasible due to level violations, and the optimization procedure will then penalize a valid candidate solution. The second is a *false positive*, when a solution that is in fact infeasible is deemed to be feasible due to error in the predicted level and is retained by the optimization procedure without a penalty term. The first type is of greater concern, since in the first case the solution is made less attractive, while in the second case the infeasible solution may be thrown out at a following iteration. It is intuitively clear that the optimal solution is to have a level trajectory that is close to the constraints (in fact, the constraints should be binding). Avoiding *false negative errors* is achieved by a good penalty function, which does not cause a false negative solution to be thrown out.

Table 11 lists testing errors for the 9 water levels in the tanks. The relative error is the absolute difference between the ANN and ("true") EPANET values divided by the (true) EPANET value (when the EPANET value is less than 1 we divide by 1).

Table 11: Relative and absolute testing errors of tank levels

Tank	Maximum absolute relative error	Average absolute error	Maximum absolute error
	(-)	(meter)	(meter)
Pua	0.38	0.05	0.71
MD_UP	0.71	0.08	0.71
Marcus	0.29	0.03	0.29
MD_D	0.92	0.11	0.92
Shp	0.29	0.05	0.54
RS	0.43	0.06	0.45
SM	0.37	0.05	0.41
Peleg	0.17	0.03	0.46
Meir	0.19	0.03	0.45

For use in the DGANN procedure (see Work Package 4 below) we used the ANN (29,90,23) which was trained with deterministic demand.

<u>Work Package 3:</u> Development of an adaptive optimization procedure based on a GA, specifically for real-time control of water-distribution networks

The objective of Work Package 3 is to develop a GA tool for optimizing the operation of a network, which is simulated by an ANN. The TWRI team provided a supporting role in this work, and we have therefore not constructed a special purpose GA tool. Instead, we used optiGA (www.optiwater.com), a GA engine that is available commercially, and is at our disposal free, by virtue of the presence of its developer and owner, Elad Salomons, on our team. A basic overview of genetic algorithms, different parameters and methods is available in optiGA's user manual (http://www.optiwater.com/optiga/manual.html).

Here we discuss some of the general considerations in developing a real-time optimization procedure for operating a water distribution network. The optimization implies that at any point in time we look ahead over a time period which is the "operation planning horizon" and wish to determine the operating plan of the controlled elements (pumps, valves) for each hour over this time horizon (or for smaller time increment, if it is relevant to change the operation so often).

To do so we need to know the capabilities and constraints of the network and the demands that it has to supply over the planning time horizon. The operating plan is implemented for the first hour, at the end of which the state of the system is measured by a SCADA system. The state of the network is compared to that which was calculated. Either the old operating plan remains in effect or a new one is calculated, depending on whether the new measured state variables have deviated from their planned value, and/or if there is new information about the system (such as equipment failure) or a new demand forecast.

We first discuss some important considerations in developing an adaptive on-line optimal operation procedure, and then proceed to describe in detail the methodology developed in this project.

Demand Forecasts

Operation of a distribution system for a planning period ahead requires forecasts of the demands. It is well known from experience that such forecasts have a high degree of uncertainty. Therefore, operation planning has to be repeated rather frequently, to allow for new data as they become available. Unforeseen events in the system such as pipe or equipment failure may also require re-calculation of the operation plan. This topic is covered in a later section (Work Package 5).

Demand forecasting for the POWADIMA project was developed by the Ferrara team, and is outside the scope of this report. Here we merely emphasize that the uncertainty in demand forecasting is of major concern in development of the overall optimization package. The sensitivity to variations in the demands is somewhat alleviated by the fact that the optimization is re-run every hour, with new data. These data include the new demand forecast (which may be the same as in the previous hour) and the actual levels in the tanks. These levels, read in via the SCADA system, may deviate from those calculated one hour ago, due to differences in the demands, to inaccuracies of the hydraulic model and to the inaccuracies of the ANN in replicating it. The actual tank levels become the starting point for the new optimization, for the next hour, and by accepting them errors made in previous times are now eliminated from affecting the new solution. Thus, the operation plan is executed one hour at a time, and modified in the new hour.

This may cause some instability in the operation, a "hunting" process as it were, which causes the pumps to turn on and off rather more frequently than is practical. Experience has shown that this is not a real problem in the methodology we have developed. Still, if in another application this does turn out to be a deficiency, then some measure can be taken to prevent rapid changes in operation, or penalize them in the objective function so that the solutions tends to avoid them.

Length of the Operation Planning Period

Urban water distribution systems typically operate with daily and weekly cycles which tend to vary in shape in different seasons of the year. The typical operation planning horizon is 24-hours ahead, in view of this cyclic behavior. When there is large storage in the system, it may have a weekly cycle, being lowered each day slightly, to be filled up during the weekend, and some consideration must be given to this weekly behavior. In the case of Haifa, and most specifically Haifa-A, the size of overall storage is not very large (see above) and daily operation is sufficient.

Thus, it would seem that the optimization should cover a period of 24 hours. However, consideration was given to extending the planning period beyond 24 hours, and periods up to 32 hours were tried. This has to do with setting the final state (levels) in the tanks, an issue that will be discussed below. The optimization algorithm must include something about the end state of the system, and it was felt that extending the planning period beyond 24 hours, up to a time at which we can set the system state (tank levels) at a known value, is required in order to recognize the fact that we are optimizing over a time horizon that is but a "slice" out of an infinitely long horizon.

End Point Determination

The problem of the "end state" is one that is encountered in most water resources models (and other inventory problems), and has attracted much attention by researchers and practitioners. We avoid getting into the general issue, and restrict ourselves to the specific approach taken in this project.

Practically all approaches to determining an optimal operating plan for a water resources system encounter the difficulty of dealing with the state of the system at the end of the planning period. This time period is merely a “slice” out of the infinite time horizon over which the system will be operated, but for practical reasons we cannot expect to determine the operating policy for all times to come. We must therefore limit the time horizon to something manageable and do something to include considerations of future times in the optimization over this limited time horizon.

When the objective is to minimize operating costs, as in our case, the optimization will tend to deplete the resource (empty the tanks, use the water and leave the tanks empty) towards the end of the planning horizon – unless the end-state either constrained or has some value that mitigates against this depletion.

There are a therefore a few ways for handling the issue of the end-state. In our case these are the levels in the tanks. The options are:

1. The end-state is either set to a fixed value or constrained to be above some value. This value can be, for example, the initial state, or some other value. For example, if there are known consideration for having the tanks filled at the end of the period or allowing them to empty by a certain amount.
2. Having such a constraint not at the end of the planning period but at some intermediate time. IN our case this might be full tanks at the end of the low-tariff period.
3. Giving a value (opposite of a cost) to the state (levels) of the tanks at the end of the period. It is difficult to set this value, which should in fact reflect the incremental operating cost of the system from that point on as a function of this state (which becomes the initial state for the future).
4. Extending the time horizon to the point where the end-state has little effect on the operation during the first few time steps. The operation during later time steps will anyhow be re-calculated in the next time step, so the operating plan for later times is not really binding. There merely to ensure that the operation for the immediate time steps is not "myopic", and does have consideration of the costs to be incurred in later times.

Suffice it to say that this issue must be considered carefully in any application of the optimization to other systems. Some of the dilemma associated with selecting the planning period is alleviated by the fact that the optimization is re-run every hour.

<u>Work Package 4:</u> Combining the GA with the ANN to form an optimal-control process for water distribution and its application to the experimental networks

The work on this task began in April 2002 with the development of the GA+EPANET and the GANN (GA+ANN) programs for the AnyTown Modified (ATM) network. Following the results obtained for ATM the program was enhanced for the use with the Haifa-A network, and used in a dynamic optimization mode, which tracks the system over time, hour-by-hour for a selected period of time. We have done it for a whole year - 2000.

Optimal operation for the AnyTown network

A GA model was built in order to solve the operation problem. The GA engine that was used is optiGA. The GA module was connected to the EPANET toolkit, which solves the ATM network and generates the results for each operational schedule proposed by the GA. The combined module yields good results. A feasible solution is reached already in the first GA generation. Below are two figures showing results. Figure 7 shows the progress of the fitness function over 100 generations.

Figure 7: progress of the ATM fitness function (GA+EPANET)

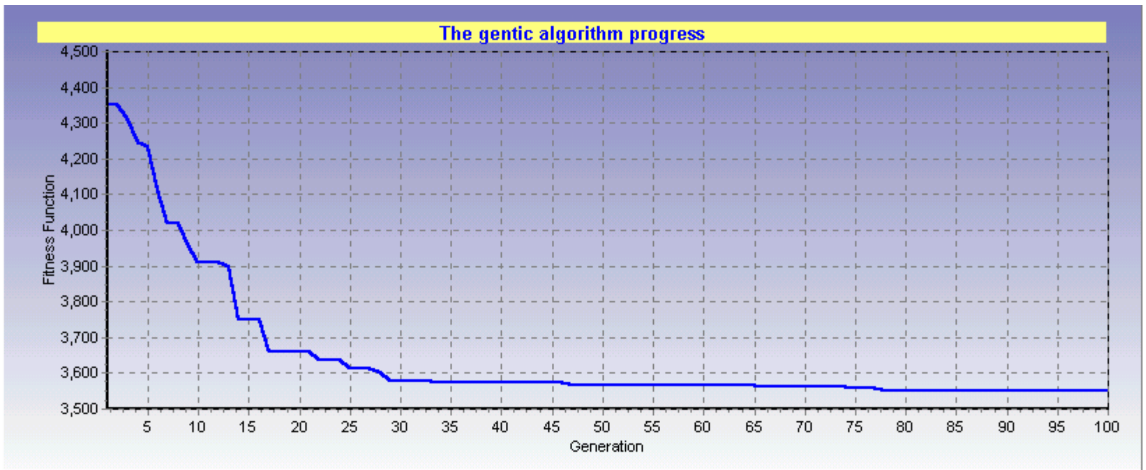
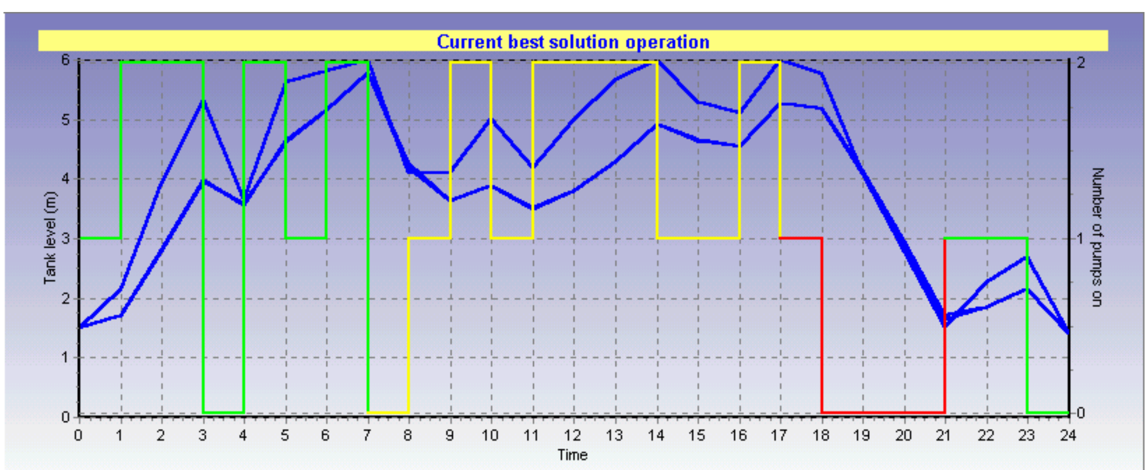


Figure 8 shows the results of the best solution found. The three blue lines show the water levels in the storage tanks (left ordinate, only two are seen, as two of the tanks follow exactly the same pattern - tanks 165 and 265), the colored block line shows the number of pumps operating in each time step (right ordinate). The different colors are for the different electricity tariff times (green – low cost, yellow – medium cost, and red – high cost). Note that a maximum of only two out of the three pumps operate. This is the result of the relative magnitude of the demands, pump capacities and tank volumes.

Figure 8: the results of the best solution found for ATM (GA+EPANET)



The genetic algorithm model, for ATM, consists of the parameters presented in Table 12.

Table 12: GA parameters for ATM model

GA parameter	Parameter value
Binary string	3 pumps X 24 hours = 72 bits
Population size	50
Number of generations	100
Mutation probability	1/72
Crossover type	Two points
Selection type	Top mate
Penalties type	Linear

In the next stage the EPANET was replaced with the four ANNs trained (as described in Work Package #2). Below are two figures showing results. Figure 9 shows the progress of the fitness function over 100 generations and Figure 10 shows the results of the best solution found. In both figures the blue line indicates the EPANET solution while the cyan line indicates the ANN prediction.

Figure 9: progress of the ATM fitness function (GA+ANN)

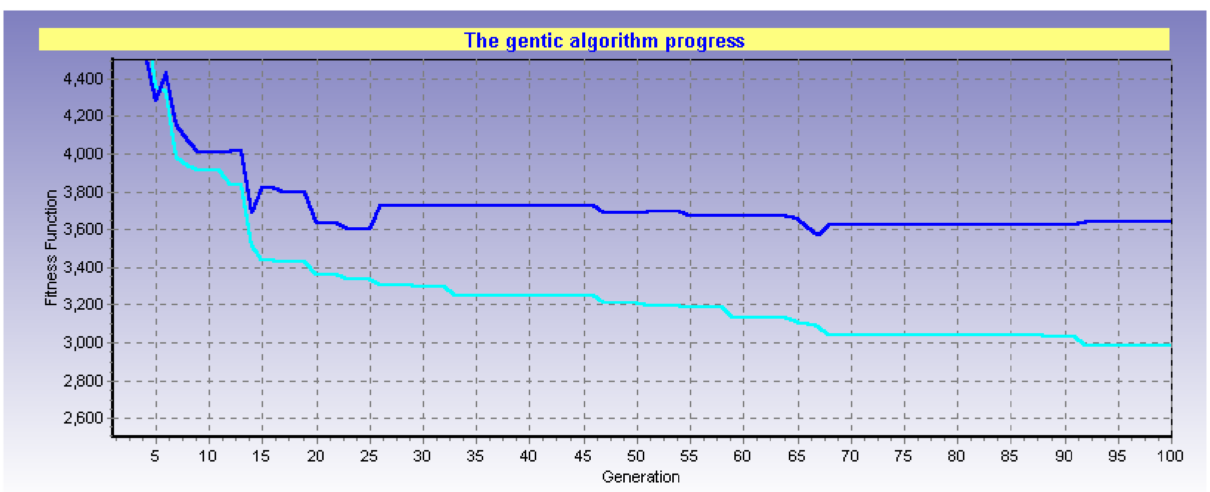
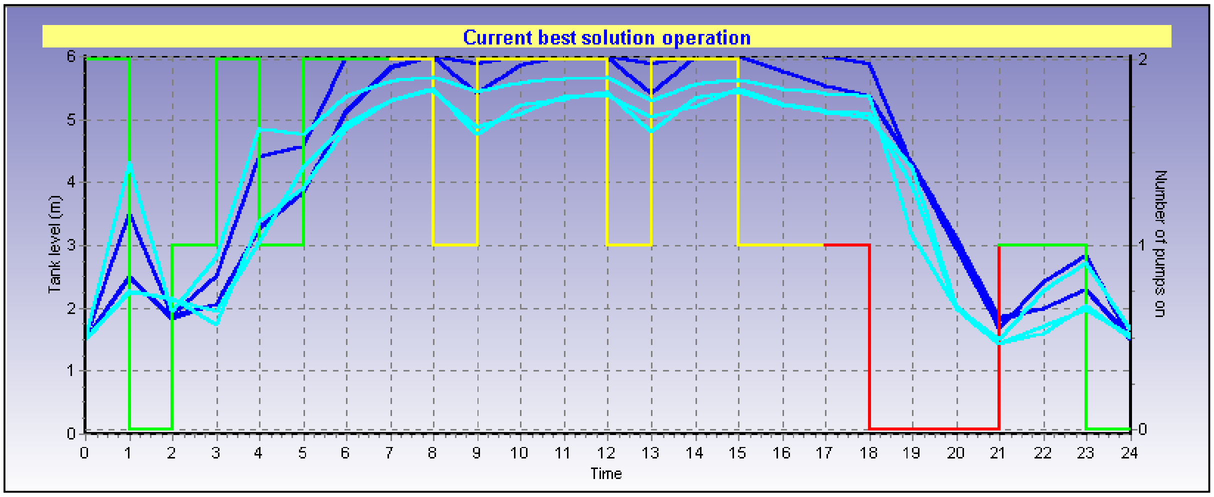


Figure 10: Results of the best solution found for ATM (GA+ANN)



It should be noted that EPANET does note play a role in the optimization process itself, which is performed entirely with the GA+ANN. The EPANET program is run, as a separate module, in order to identify and keep the best EPANET-feasible solution found by the GA+ANN procedure. While the GA+ANN fitness, the cyan line in the first figure, monotonically decrease due to the elitism feature of the genetic algorithm, the EPANET solution, the blue line, may increase or decrease as the control settings of the system change. After the optimization has ended, the best feasible solution found, as simulated by EPANET, is maintained. In the case shown in Figure 8 it is the solution at generation 67, with the cost of about 3,500 units.

In order to compare the performance the GA+EPANET and the GA+ANN procedures, a series of 10 runs of each were made. The results are shown in Table 13: the value of the objective function (cost) and the running times.

Table 13: Comparison of GA+EPANET and GA+ANN results for ATM

Run #	GA + EPANET		GA + ANN	
	Cost	Run time	Cost	Run time
1	3,561	244	3,548	71
2	3,542	247	3,638	69
3	3,561	237	3,570	71
4	3,565	239	3,578	69
5	3,553	249	3,591	69
6	3,515	242	3,578	71
7	3,567	245	3,662	71
8	3,560	252	3,565	70
9	3,541	238	3,652	71
10	3,535	245	3,699	71
Average	3,550	243.8	3,608	70.3

While the GA+ANN average cost is less then 2% higher than the GA+EPANET average cost, the GA+ANN running time is on average 3.5 faster than the GA+EPANET.

GANN program for Haifa-A

Having good results for ATM, the GANN program was updated and enhanced for use with the Haifa-A network. The main features that were implemented in the program are:

- Network zoning for DMAs with different demand patterns.
- Minimum, maximum, end of simulation and specific hour levels constraints for each tank.
- Valves type includes PRVs and On-Off valves.
- Each pump can have a different electricity tariff structure.
- Each pump can have central, local or manual control. Central - the pump is operated from the control center via the optimization program. Local – the pump is operated according to local control rules, such as set-points. Manual - the pump is operated manually by the operator.
- Cost of water and of energy is assigned to each of the sources. This option was implemented but not used for Haifa-A. It was added since in the case of Valencia there are treatment costs at the sources.
- Maximum energy consumption constraints can be imposed on pump stations.
- The program can be run in EPANET mode or ANN mode.
- A run can be started from a previous design. This provides the GA with a solution that has a good chance of being feasible and close to optimal.
- Constraints on the total number of pumping hours for each pumping station can be imposed. This option was added to help the process converge. It is clear that a range of pumping hour can be estimated since a specified volume of water should be pumped to each DMA according to the zone demand
- An enhanced and friendly GUI was created.
- Validation procedure with EPANET was added to ensure that the optimized solution found by the GANN is EPANET-feasible.

Before applying the optimization program to Haifa-A, the network data were updated with the latest information available from the Haifa municipality. This included daily demands for each DMA, pump operation settings and a few corrections to the network connectivity.

Until this stage, the steady-state version of the program was used: 24 separate hourly optimization without the dynamic feature. The objective was to minimize operational costs for the 24 hours horizon while keeping the following constraints:

- Supply all demand.
- Pressure above minimum at demand nodes.
- Maximal power at pumping stations.
- Tank levels at specific hours.
- Final tank levels at the end of the design horizon – usually set to the initial tank levels.

Since constraints without some tolerances have no meaning in the practical world (1mm of violation in a water level is not like a violation of 1m), a set of constraint tolerances were determined and shown in Table 14.

Table 14: Constraint tolerances for Haifa-A

Type of Constraint	Tolerance level
Pressure violation	~ 3% (accuracy of measurement equipment) ~ 1 meter
Maximum tank level violation	~ 3% (accuracy of measurement equipment) ~ 30 cm
Total tank level violations	In each tank, each hour, no more then the maximal violation allowed
Energy violation	< 3% (accuracy of measurement equipment) ~ 3 kW

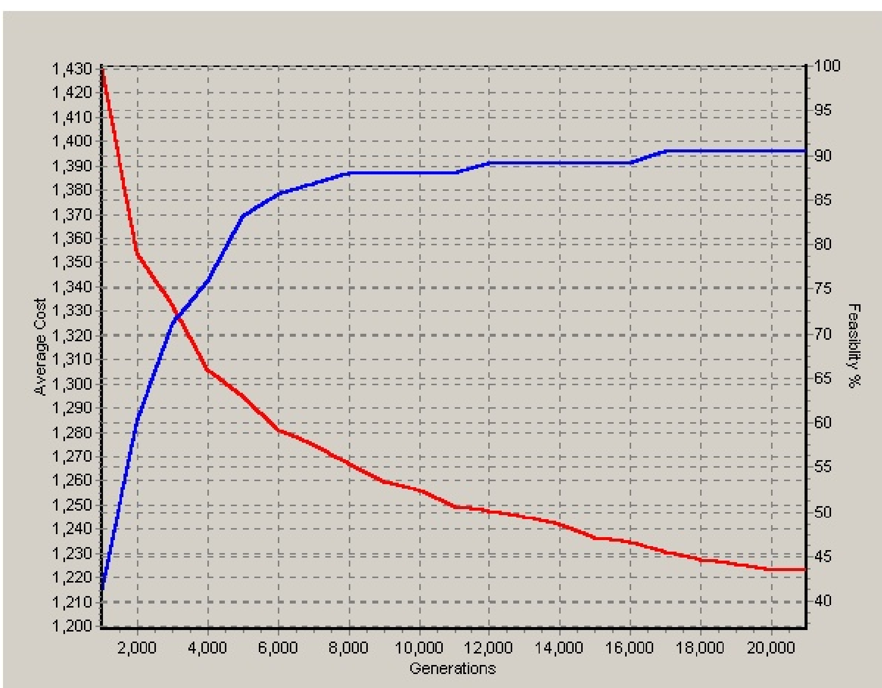
The genetic algorithm model uses the parameters shown in Table 15.

Table 15: GA parameters for Haifa-A GANN

GA parameter	Parameter value
Binary string	13 pumps X 24 hours = 312 bits
Real string	1 valve X 24 hours = 24 values
Population size	8
Number of generations	Variable (parametric runs)
Mutation probability	0.5%
Crossover type	Two points
Selection type	Top mate
Penalties type	Linear

The results are summarized in Figure 11. A series of runs was made with different number of generation (horizontal axis). The red line (left vertical axis) is the average cost of the optimized operational control settings. The blue line (right vertical axis) is the probability of finding a feasible solution with the ANN as a hydraulic simulator. It can be seen that as the number of generation increases the average cost decreases and the probability of finding a feasible solution increases.

Figure 11: Haifa-A GANN results: Average cost and probability of having a feasible solution, as a function of the number of generations



The DGANN program for Haifa-A

Following the project meeting in Ferrara on January 2003, the GANN program was further developed to include dynamic capabilities. The result is DGANN+, which stands for Dynamic Genetic Algorithm – Artificial Neural Network enhanced. The + stands for "enhanced", which indicates an additional constraint that was imposed on operating pumps during high tariff periods, to improve convergence to the optimum solution (see below).

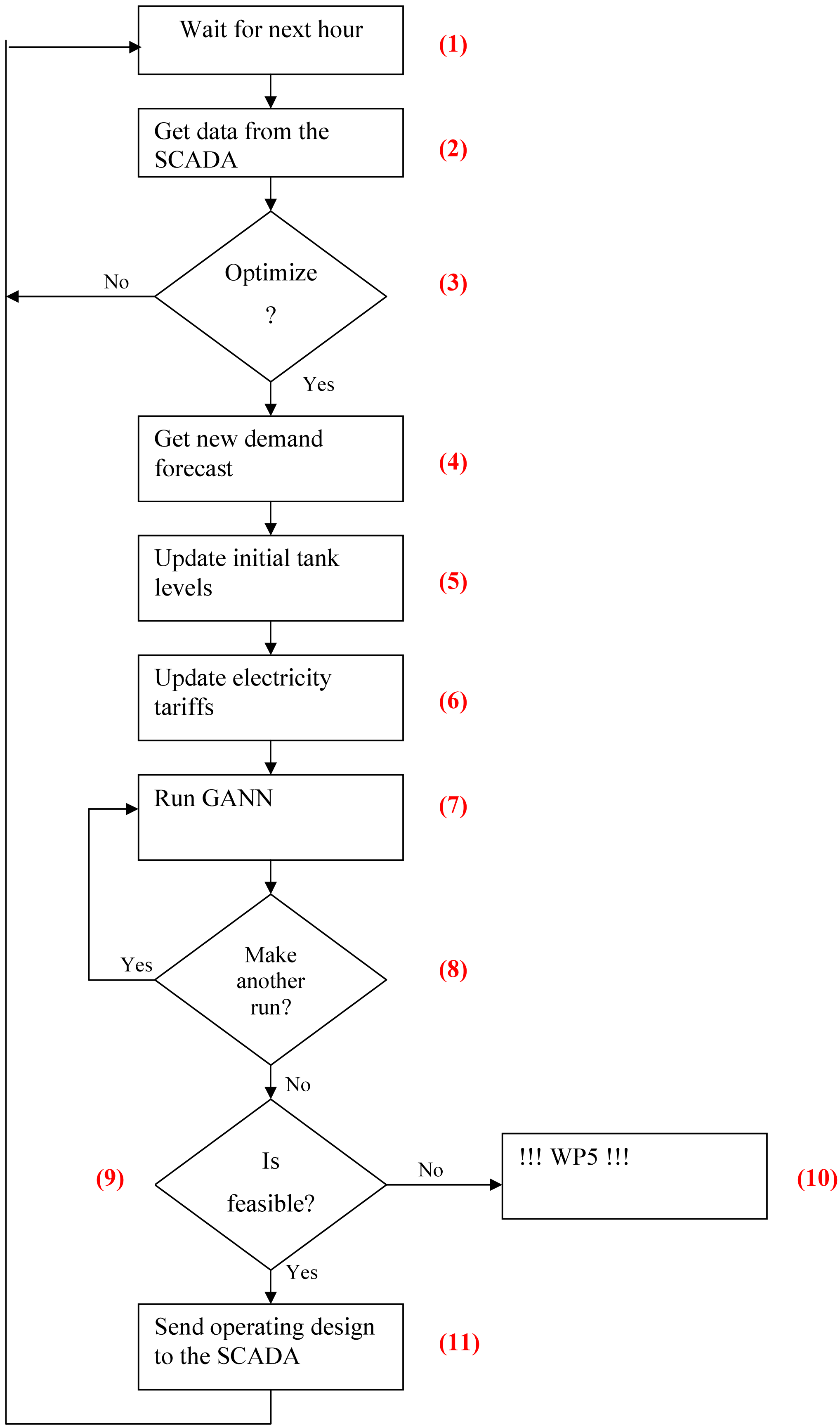
The program has the following features (beyond those of GANN):

- The operating horizon is now a user-controlled parameter set to the default value of 32 hours. A 32 hours time horizon was determined to avoid the need to specify the tank levels at the end of the planning horizon. Instead of setting this level it was decided to have a "floppy end level", without any constraint on it. This would result in the tanks being emptied towards the end of the planning horizon in order to lower pumping costs. To avoid this from affecting the feasibility of the operation over the next 24 hours, the time period is extended to 32 hours and the tank levels at the end of the low tariff period are set to their required maximum. There is therefore at least one such point in the planning horizon, which fixes the trajectory of the tank levels at this time.
- A dedicated subroutine was added to connect to a SCADA system. At the moment, since we are not really working with a SCADA system, the subroutine sends EPANET the operating instructions for the coming hour and receives back the resulting tank levels at the end of the next hour, which are then viewed as if they were read by the SCADA system. These levels are then used as the new initial condition for the next hour's optimization. This allows us to “ground” the tank levels each hour and avoid cumulative errors. It in fact emulates what would occur in a real system with SCADA.
- A dedicated subroutine was added to communicate with the demand-forecasting module. Since the demand-forecasting module was not available as a stand-alone module, our subroutine extracts the 32 values for each DMA from the file of the forecasted demands generated by the demand-forecasting module.
- A file containing the electricity tariff each hour of the year was prepared, taking into account the different tariff periods over the day, week, months and holidays. At

each hour the main program extracts from this file the 32 values for the next operating horizon.

- A penalty on operating pumps at high tariff times was introduced. This option helps to guide the optimization “away” from operating in the high tariff periods. In spite of this penalty, pumps can still operate during the high tariff periods, and there is no influence on the feasibility of the solution. A + was appended to the program name, resulting in DGANN+. It is our final version.
- The optimization module is activated every hour according to the following flow chart in Figure 12, whose stages are explained below.

Figure 12: Flow Chart of DGANN+



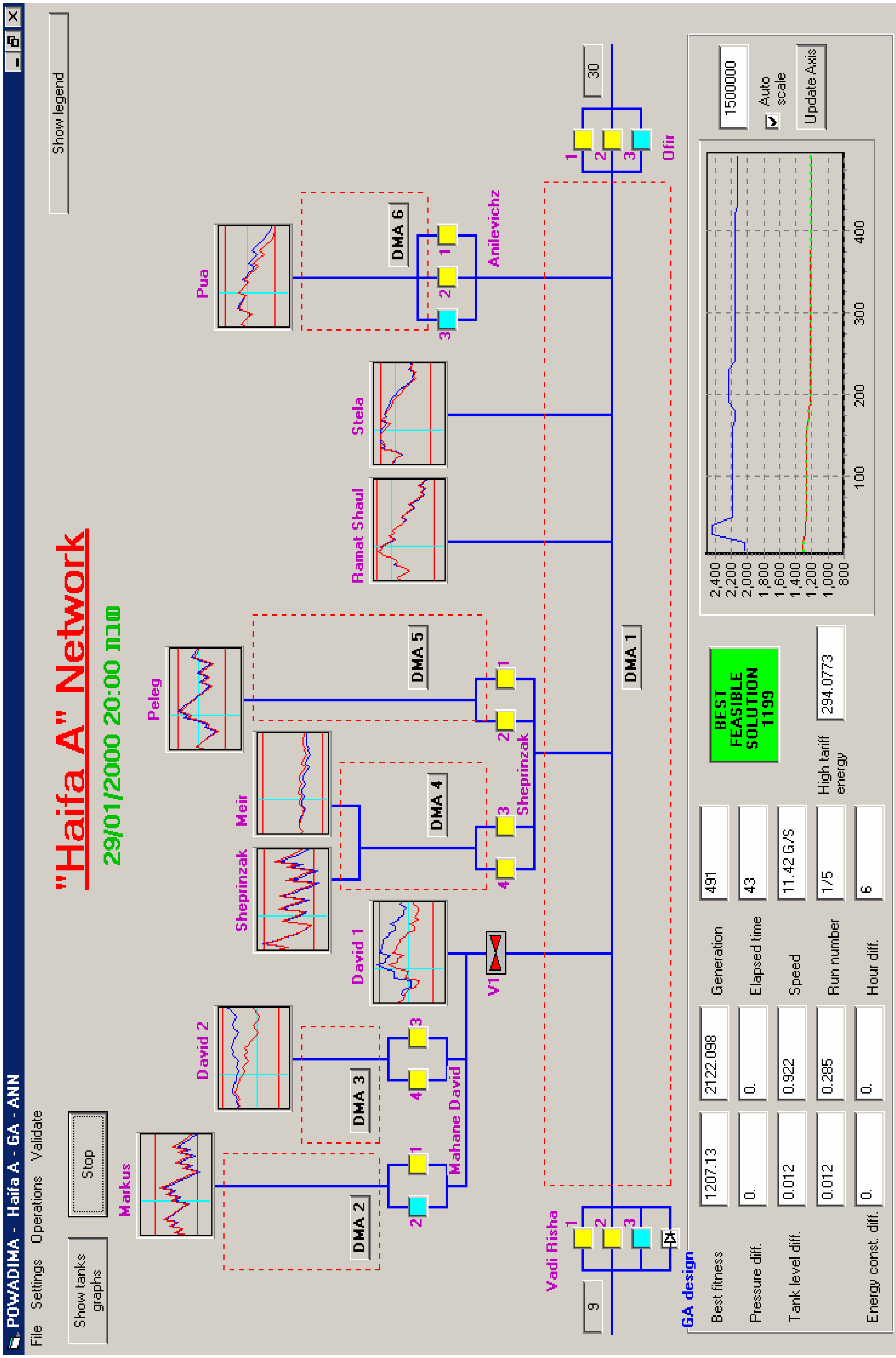
The DGANN+ program stages:

1. The program is waiting for the next hour activities and information. Once the set time arrives the next steps are performed:
2. The SCADA module is called, and the updated tanks levels and flows that are needed for calculation of the actual demands in the last hour by the forecasting module are received back.
3. A decision whether to compute a new operating plan is made. At this stage the answer is "Yes" every time since it was agreed to do so. It would be possible to introduce some criteria for this decision (for example: the true tanks levels are as predicted and no change in the operating design is needed).
4. The forecasting module makes new demand forecast and the main program is updated.
5. The initial tanks levels are updated from the data received from the SCADA system.
6. The electricity tariffs for the next operating horizons are updated.
7. The GANN optimization module is called. The best solution of the previous hour is used as one member in the initial population of the GA. The operating plan for the remaining parts of the planning horizon (the first hour has passed, so it is dropped, and a new hour at the end has been added). This has improved the speed of the runs substantially, since it can be expected that the optimal operation in the new time horizon would be close to the one found in the pervious hour.

   During the optimization, at a user defined interval, the best solution found to that point by the GANN optimization is validated with the EPANET model for feasibility. The best, low cost, feasible solution is kept in parallel with the optimization procedure. This validation process has no influence on the optimization process, which is made entirely by the GA with the ANN. This method insures that at the end of the run, a real feasible solution, if one has been found, is available.
8. When results are available from the GANN module, a decision is made whether another GANN run is to be made (mainly influenced by the time available for the run). If the answer is "Yes" then stage 7 is repeated. At this point, if a feasible solution was found the answer is "No".

9. The outcome of the GANN is checked to determine if the solution is feasible. If not, stage 10 is invoked (Work Package 5). If the solution is feasible then we proceed with step 11.
10. A set of alternative actions is considered as described in the next section (Work Package 5).
11. The SCADA system is updated with the new operating design for the next hour, the program returns to stage 1 and waits for the next hour.

The main screen of the DGANN+ program is shown on the next page:


POWADIMA - Haifa A - GA - ANN
File Settings Operations Validate
Show tanks graphs
Stop
Show legend
"Haifa A" Network
29/01/2000 20:00
Markus
David 2
David 1
Sheprinzak
Meir
Peleg
Ramat Shaul
Stela
Pua
Anilevichz
Ofir
Vadi Risha
Mahane David
V1
DMA 1
DMA 2
DMA 3
DMA 4
DMA 5
DMA 6
GA design
Best fitness
Pressure diff.
Tank level diff.
Energy const. diff.
Generation
Elapsed time
Speed
Run number
Hour diff.
High tariff energy
BEST FEASIBLE SOLUTION 1199
1207.13
2122.098
0.012
0.922
0.285
491
43
11.42 G/S
1/5
6
294.0773
1500000
Auto scale
Update Axis

Results

Each simulated day takes about one hour to run (on average it takes about 2 minuets to make a single hour run on a 733MHz PIII computer). A comparison with GA+EPANET runs was made. The results are that using EPANET instead of ANN is about 10-12 times slower for the Haifa-A network. The run-time of the GANN program is considered suitable for our purpose since it is much shorter than the update time intervals of the SCADA system.

At each hour the EPANET file (INP file) that contains the operating design is saved. In addition, two daily files are saved: "Solutions.dat" and "Hour_Cost.dat". The first contains information about the results of each operating design at each hour, while the latter holds information about the actual operating costs each hour as received from the SCADA system (or in our case from the EPANET). This is the cost of the first hour only, of the operating design made by the program each hour. Below is a sample print out of the "Hour_Cost.dat" file for April 11th, 2000:

Time	Hour cost
11/04/2000 00:00	19.32366
11/04/2000 01:00	44.85899
11/04/2000 02:00	14.58509
11/04/2000 03:00	66.7654
11/04/2000 04:00	33.11203
11/04/2000 05:00	49.96976
11/04/2000 06:00	47.48326
11/04/2000 07:00	91.5051
11/04/2000 08:00	51.60624
11/04/2000 09:00	50.07917
11/04/2000 10:00	62.36257
11/04/2000 11:00	90.94405
11/04/2000 12:00	28.35306
11/04/2000 13:00	109.1501
11/04/2000 14:00	35.84935
11/04/2000 15:00	90.25778
11/04/2000 16:00	99.71409
11/04/2000 17:00	19.58369
11/04/2000 18:00	10.03182
11/04/2000 19:00	47.27864
11/04/2000 20:00	113.0292
11/04/2000 21:00	56.03272
11/04/2000 22:00	42.32941
11/04/2000 23:00	55.55387

Explanation of the table:

- At 00:00 there is an optimal operating plan for the next 32 hours, with a level set for 07:00 that morning and again 24 hours later.
- The cost of operation according to this plan, in the hour between 00:00 and 01:00, is 19.32366 NIS.
- The program is run again at 01:00, with tank levels computed from the operation during the first hour. The cost for the first hour of that run, between 01:00 and 02:00, is 44.85899 NIS.
- The large difference between the two costs is probably due to the following reason: The operating plan is insensitive to the hour in which pumps are operated, as long as it is during the period of fixed tariff, so in one plan pumps may be operated in one hour while in the subsequent one the timing of operation may be shifted by one or more hours, resulting in the difference from one hour to the next is the hourly cost. This results in a flat response surface (i.e., similar, here actually identical, values of the objective function as several points around the optimum) so the optimization is indifferent among these points.

All 24 hours INP files obtained from the DGANN+ program were checked with EPANET. Tank level violations (>0.3m from nominal) and pressure violation at critical point (>1m) and all warning message from EPANET were recorded for the entire simulated year.

Three full year runs were made with two demand-forecast sets (see Work Package 6 below):

1. DGANN with UNIFE forecasted – no violations, no warning messages.
2. DGANN with WDA forecast – no violations, no warning messages
3. DGANN+ with UNIFE forecast – one violation in SM tanks (level 70.63m whereas minimal level is 71m), no warning messages.

Total number of the one hour experiments = 366 days x 24 hours x 3 years = 26,352.

These results indicate that DGANN is stable.

A detailed presentation of the different runs made and results obtained are to be found in the Work Package 7 section.

EPANET as a Validation Module

As described in step 7 of the DGANN+ flowchart above, an EPANET validation module was introduced. The methodology used for the validation is as follows:

- At predefined GA generation intervals, the best GA+ANN solution is checked for feasibility with the hydraulic model used to train the ANN.
- A feasible solution is kept, and replaced only by a better feasible solution (if found).
- At the end of the GA+ANN optimization process the best feasible solution is used.

Note: EPANET is not part of the GA+ANN optimization process, nor does it influence this process in any way.

The main reasons and benefits for this validation procedure are:

- Since the EPANET model was used for the training of the ANN and there is no 100% accurate ANN, by definition the EPANET model is more accurate then the ANN. Even if the ANN is close to perfect it is still less accurate then the EPANET model used for training it.
- The optimized operating scheme is feasible according to the best available model.
- This process is more robust, as dependence on the accuracy of the ANN is reduced. In the general case, for city "C", we will not have to be so depended on a highly accurate ANN if one is difficult to achieve.
- There is a feasible operating scheme for the entire planning horizon when there is a failure of the SCADA.

TWRI software and final project software

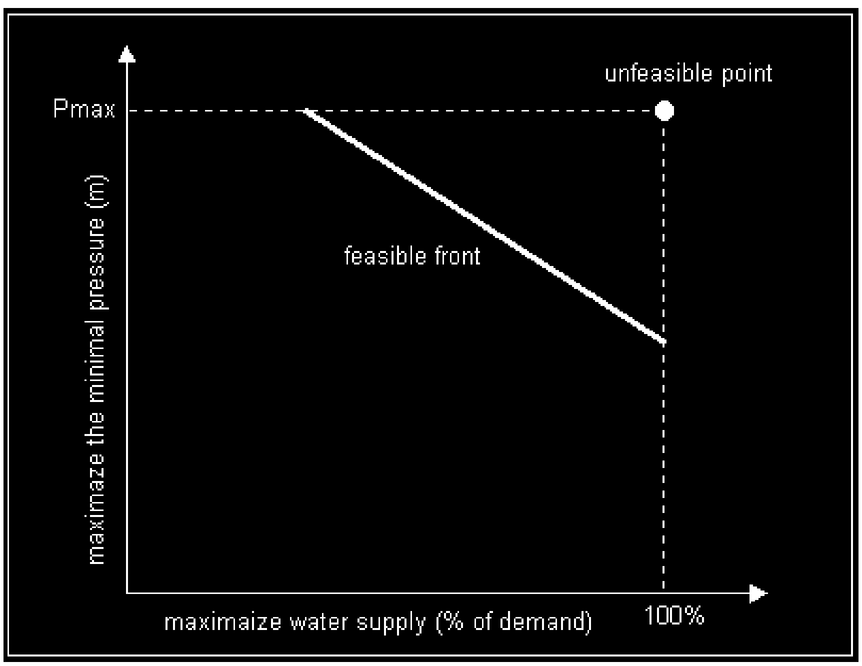
After our team developed this DGANN, the final package for the project was developed by the UNEW team. This is reported separately, by the UNEW team.

Work Package 5: Extension of the optimal-control process to include emergency procedures

The TWRI Team took the lead in this Work Package. The objective of WP5 is to incorporate procedures for dealing with emergency conditions such as pipe bursts, sticking valves, pump failures, loss of electricity etc. We are interested in emergency situations that cause a decline in system performance, which is defined as a failure.

Our initial proposal for handling emergency situations was as follows. To formulate a multi-objective problem (MOP) with two objective functions: maximize the amount of water supplied and maximize the minimum pressure at any of the consumer nodes. There is a tradeoff between these two – as more water is supplied the minimum pressure will be reduced. An emergency occurs when it is not possible to supply the required demand with pressures at all consumer nodes above the required minimum pressure.

Figure 13: Multiple objective (MO) space

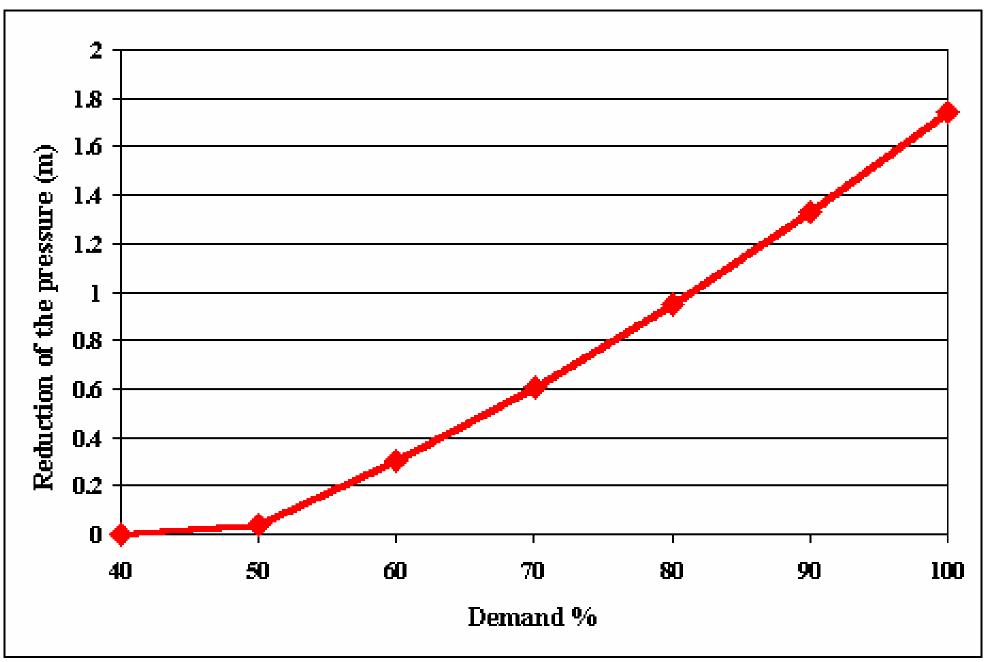


In Figure 13 Pmax is the original minimum pressure that must be maintained and 100% of the water supply is the original water demand. A failure occurs when the point [100%, Pmax] is infeasible, since this demand level and minimum pressure cannot be provided at the same time; this is the definition of an emergency condition.

Our attempts to cause a failure situation in the simple case of the AnyTown model and the more complicated network of Haifa-A showed that this happens only when some part of the system becomes disconnected from the sources. Otherwise the systems tend to have a connectivity and capacity that overcomes failures of single or even several components at a time. In most cases, breakdown of a single hydraulic element does not cause a network failure. Similarly, it was reported by the Valencia team that the only major failure in Valencia in the past few years was due to shortage of water from one of the treatment plants, and there were no failures resulting from other types of component failures.

During our attempts to cause a failure in the ATM network we closed a number of main pipes and followed the pressure changes at a critical node as the demand changed. Such multiple component breakdowns are very rare in real systems. A summary of the results appears in Figure 14. It shows that once the supply level goes above 40% of the demands the pressure at the most critical node begin to drop below the required minimum (30m). However, even when the full 100% of the demand is supplied, the maximum pressure drop is only about 1.7m, which would not be viewed in a real system as a failure, merely an inconvenience.

Figure 14: Reduction of the pressure vs. % demand supplied



Having obtained these results, it was suggested to partially return to the original project proposal, namely capturing the knowledge of the system operators how to deal with emergency situations. In cases of failures that do not cause a part of the system to be disconnected (which cause supply in that part to be curtailed), the DGANN+ (or GA+EPANET) is used to obtain a feasible operating plan within the original constraints.

When a component breakdown occurs and a feasible solution cannot be obtained, the design time horizon is decreased and the DGANN+ program is initiated again. If now a feasible solution is not obtained, a predefined, library based, operating scenarios is used.

The following scheme was embedded in the DGANN+ program:

- Reduced design time horizon when a feasible solution can not be found.
- Setting pumps to “manual off” when they are not available.
- In case of a change in the water network that cannot be simulated by the ANN, we switch to GA+EPANET.
- Library of operational steps (available off-line).

Work Package 6: Inclusion of real-time demand forecasting and its application to the experimental networks

Our team provided a supporting role in this work package, and our main task was to supply the UNIFE team with demand data for designing the real-time demand forecasting procedure. This procedure was to be embedded in the DGANN+ program.

As real hourly demand data for the city of Haifa are not available, a set of hourly demand data from a similar size city was used. This data, for the year 2000, was normalized by dividing each value by the maximum value in the series, and multiplying by the real peak hourly demand data of Haifa, in order to produce “real hourly demand data”. This set was then sent to the UNIFE team for demand forecast analysis. The forecasted demand were obtained and analyzed.

We define for each hour t the Average Forecasting Relative Error (AFRE):

$$\mathrm{AFRE}(t) = \frac{1}{32} \cdot \sum \left| \left(\mathrm{fD}(t, t+i) - \mathrm{trueD}(t, t+i) / \mathrm{trueD}(t, t+i) \right) \right|$$

Where:

$\mathrm{fD}(t, t+i)$ is the forecasted demand value for hour (t+i),

$\mathrm{trueD}(t, t+i)$ is the true demand value for hour (t+i) (i=1,…32)

The results of the UNIFE forecasting algorithm for DMA1 are:

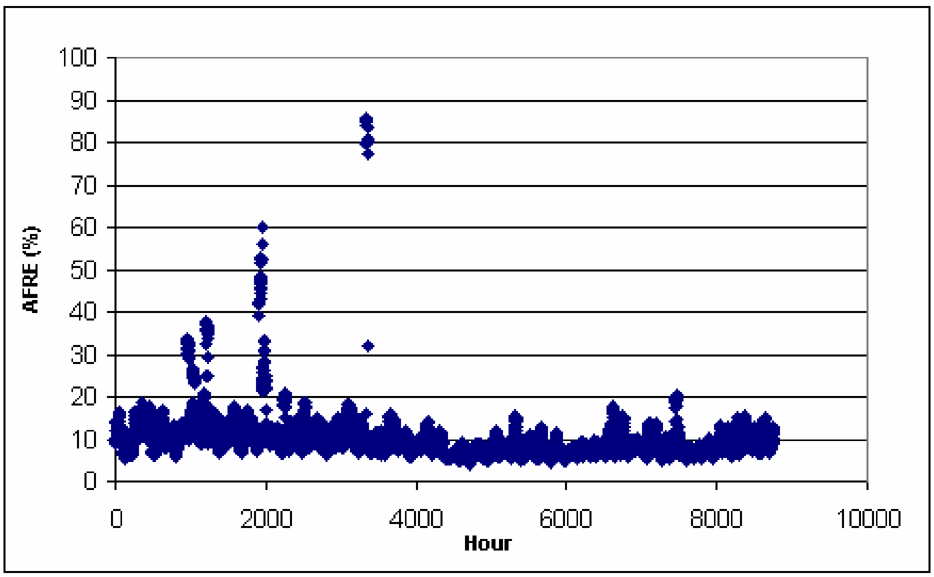
Minimal AFRE = 4.1%; Maximum = 85.8%;

Average (over the year) AFRE = 10.2%;

St.Dev (over the year) AFRE = 6.1%

The AFRE for DMA 1 over the year 2000 is shown in Figure 15.

Figure 15: UNIFE AFRE for DMA 1 over the year 2000



Results for UNIFE forecasting algorithm for the next hour only (DMA1) are:

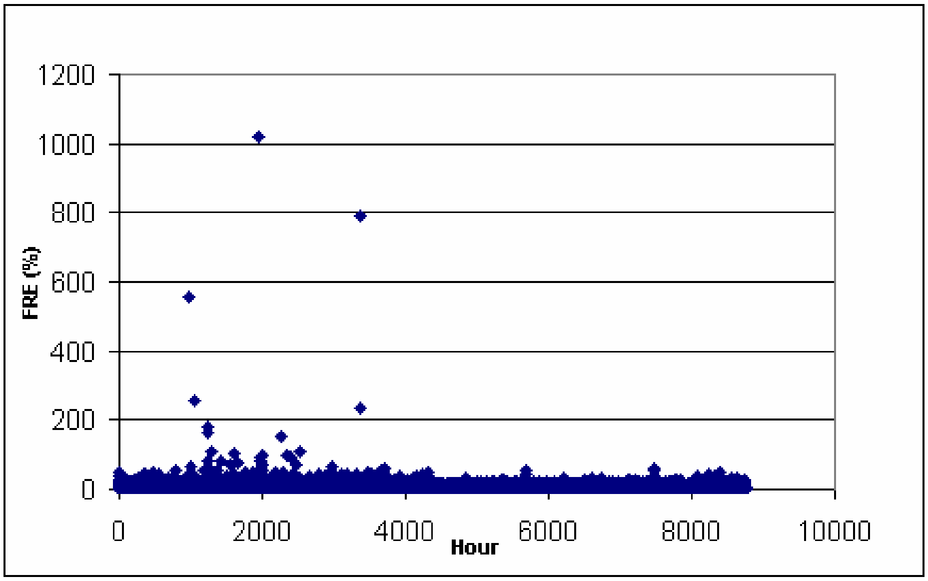
Minimum FRE = 0%; Maximum FRE = 1017%;

Average (over the year) FRE = 8.6%;

St.Dev (over the year) FRE = 17.3%

The FRE for next hour only (DMA1) over the year 2000 is shown in Figure 16.

Figure 16: FRE for next hour only (DMA1) over the year 2000



In the same manner, the “real hourly demand data”, for year 2001 were generated and sent to the UNIFE team for forecasting. The results provided by the UNIFE team had about the same as for 2000: the Average FRE is about 9%, with rare events of very large (hundreds percent) forecasting errors.

Exhaustive experiments with the DGANN+ procedure indicate that low accuracy of demand forecasting does not cause serious deterioration in the performance of the DGANN+ optimization – provided the values of last hour's demands are used to calculate the "true" value of the tank levels at the end of the hour, which are then used as initial conditions for the next hour. This procedure is used in the simulated environment, as an equivalent of bringing in via the SCADA the correct current values of the tank levels. Since these tank levels are not available in the simulated environment, the calculated tank levels, using the actual demand values, act as the equivalent of this measurement.

This approach was demonstrated it when we used a different and simpler demand forecasting algorithm in the DGANN+ optimization, one we called "Weekly Delay Algorithm (WDA). In this simple algorithm, it is assumed that the demand in the current hour is the same as it was exactly 7 days before.

The results with the WDA forecasting algorithm for DMA1 are:

Minimum AFRE = 5%; Maximum AFRE = 79%;

Average (over the year) AFRE = 14%;

St.Dev (over the year) AFRE = 6%

We then found that the operating cost and the number of constraints violations with this demand forecast remained practically as they were with the UNEF method, when the "actual" (last hour's) demands were used to calculate the initial tank level for the next hour. This will be seen when the WP7 results are presented in the next section.

Work Package 7: Evaluation of benefits arising from real-time optimal-control of water-distribution networks, definition of its SCADA requirements and documentation of methodology

The main objective of WP7 is to evaluate the real-time optimal-control system for water distribution networks in terms of its benefits/costs. The comparison is made between the current way of operation and the proposed optimal control program. The criteria for comparison will include satisfying the demands, cost savings, pressures in the network and the ability to meet operational constraints. On the other hand, some investments in a new or upgraded SCADA system are needed.

Base run – current operation

As mentioned earlier, the system is currently operated by local controls of the pumping units based on water levels of the tanks (set points). There are no energy considerations and no special attention is given to the electricity tariff. An EPANET model with the current control rules was built and used to calculate the current estimated operational cost. The demands used with this model are the “real” demands, as explained in the WP6 section. The monthly operating costs are shown in Table 16.

Table 16: Monthly operating costs with current operation

Month	Operating cost (NIS)
January	42,449
February	40,077
March	35,376
April	35,829
May	40,488
June	50,030
July	51,447
August	49,513
September	45,317
October	36,953
November	36,331
December	43,505
Total	**507,315**

DGANN runs

The above operating costs are to be compared with the operating cost suggested by the DGANN program. The following set of yearly runs was made with the DGANN program (the monthly costs are shown in Table 17):

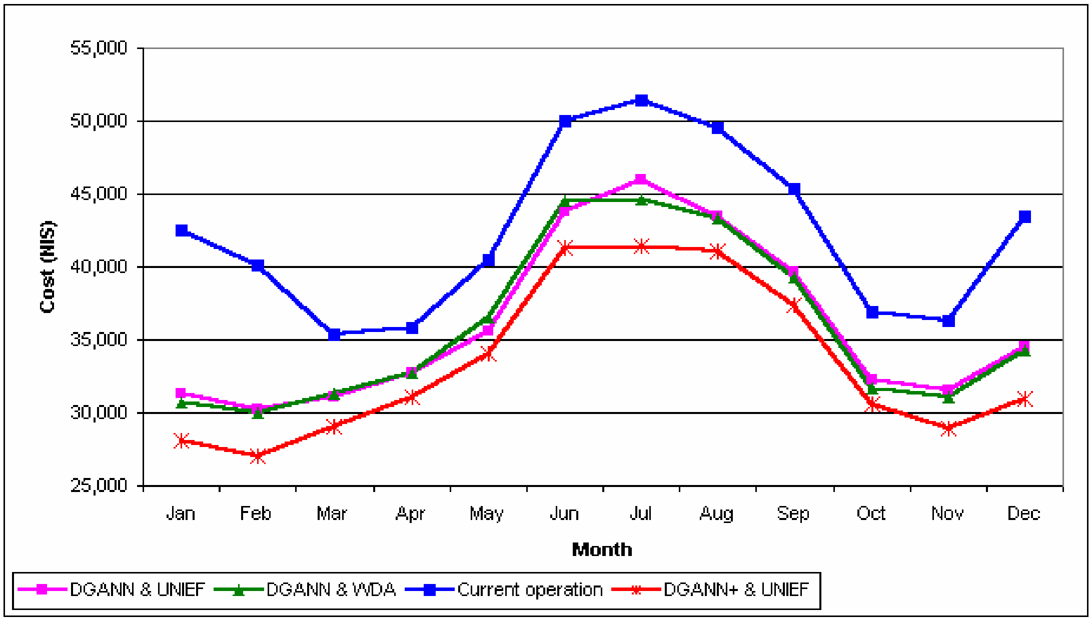
- DGANN and UNIFE Demand Forecast
- DGANN and WDA Demand Forecast
- DGANN+ and UNIFE Demand Forecast

Table 17: Monthly costs for DGANN runs

Month	DGANN and UNIFE Demand Forecast (NIS)	DGANN and WDA Demand Forecast (NIS)	DGANN+ and UNIFE Demand Forecast (NIS)
January	31,283	30,720	28,148
February	30,288	30,055	27,081
March	31,041	31,285	29,006
April	32,760	32,760	31,060
May	35,595	36,548	34,030
June	43,773	44,467	41,316
July	45,906	44,636	41,409
August	43,418	43,316	41,109
September	39,589	39,291	37,370
October	32,288	31,677	30,574
November	31,566	31,063	28,871
December	34,544	34,324	30,907
Total	**432,051**	**430,142**	**400,881**

A graphical comparison of the current operation costs and the DGANN results are shown in Figure 17.

Figure 17: Monthly costs for DGANN and DGANN+ runs with different demand forecasting algorithms



The total cost savings due to shifting of pump operation to low electricity tariff periods are listed in Table 18.

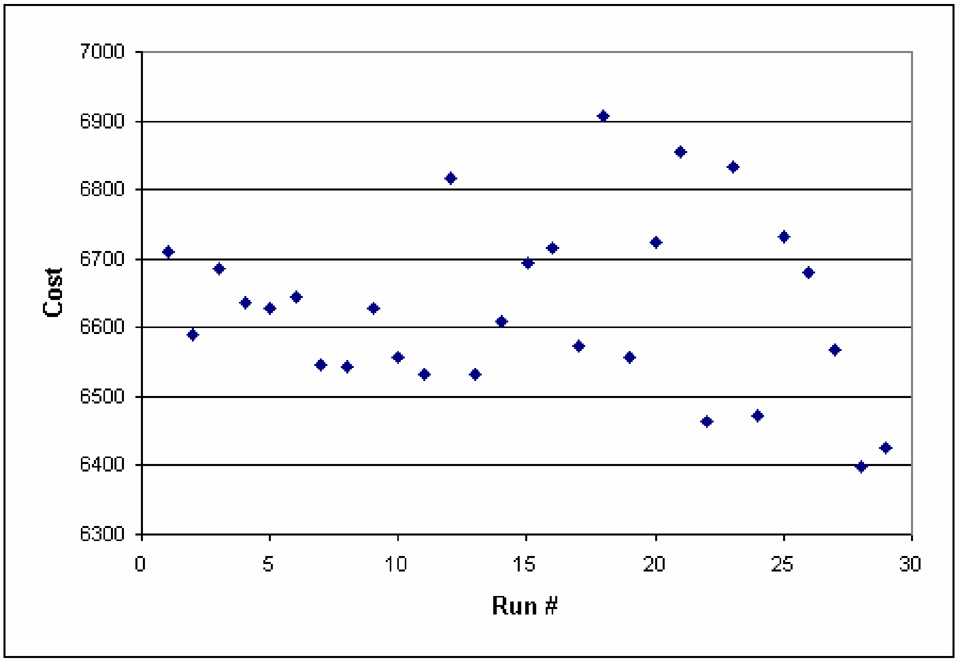
Table 18: Savings in operation costs

Run	Cost savings (NIS)	Cost savings (%)
DGANN and UNIFE Demand Forecast	75,264	15
DGANN and WDA Demand Forecast	77,173	15
DGANN+ and UNIFE Demand Forecast	106,434	21

Estimation of the confidence interval for the cost results

Since the genetic algorithm is a search method and in each run a different solution may be obtained, the optimal cost is essentially a random variable, whose value depends on the results of a particular run. It is therefore important to estimate the confidence interval of the optimal cost value. This is done by simulation. However, it is not practical to conduct a full-year experiment many times due to the long computing time required (a few weeks of computer time for each simulated year). So we made 29 runs for one representative week (1/11/2000–7/11/2000) with identical parameters in DGANN. The cost results of the 29 runs are shown in Figure 18.

Figure 18: estimation of the confidence interval for the cost results



The resulting statistics are:

Minimum cost = 6,399

Maximum cost = 6,907

Average cost = 6,629

Standard Deviation (Cost) = 126.4 (1.9%)

A few runs were also made for one full month, and a similar range of values of the optimal cost was obtained. Conclusion: the error estimation for the monthly electricity cost for DGANN is in the range of +/- 1.1%

Estimation of the computational advantage of the ANN used as a network simulator

A main objective of the POWADIMA project is "to develop a computationally-efficient alternative to a hydraulic simulation model". In this section we address the relative computational advantage of the ANN, as compared to a network simulator EPANET. A theoretical framework is developed first, and its results are then compared with computational experience.

Let A_1 be the algorithm used in EPANET to solve the hydraulic equations - an iterative gradient solution method with fixed accuracy (e.g. 0.001, see EPANET Manual, 2000).

It can be shown that the computational burden (expressed as the number of multiplications) for one time step of the EPANET algorithm can be estimated as follows

$$C(A_1) \geq \tau \cdot \Delta \cdot \vartheta ,$$

Where τ is the number of the iterations to achieve convergence, Δ is the number of links in the hydraulic network, and ϑ is the total number of nodes (including tanks, sources). The number of iterations of this method depends strongly on specific parameters of the water network, and cannot be estimated in general terms. For each specific water network the number of iterations in the EPANET simulation is set, according to experience. We have used 100.

Now let A_2 be the algorithm that simulate the trained ANN_ε (I,H,M), where I is number of the inputs , H is number of hidden nodes and M is number of outputs. Each of the outputs replicates the behavior of a hydraulic variable (flow, pressure or head) at a specific location of the water network with accuracyε.

The computational burden (number of multiplications) for this case is estimated as follows

$$C(A_2) \sim H^*(I + M) + c_m (H+M)$$

The first term represents the total number of connections in the ANN. For each link there is just one multiplication. The second term represents the number of iterations to approximate a sigmoid function. c_m is a constant that depends on the number of multiplications required to compute the exponential sigmoid function (which is the neuron activation function). Experience shows that this constant is approximately 10. (H+M) is the number of nodes, hidden plus output, at which a sigmoid is to be calculated.

Comparison of the computational burden of the ANN and EPANET requires estimation of the required size of the ANN. For Haifa-A we used ANN(29,90,23). So from estimation above it is clear that

$$C(A_2) \sim 6 \text{ x } 10^3 \text{ multiplications}$$

In order to estimate $C(A_1)$ it is necessary to know the parameters of the water model. In Haifa-A there are 216 pipes and 138 elements (nodes, tanks and pumps). Suppose that τ=100 iterations, then we have the following approximation:

$$C(A_1) \sim 1.7 \text{ x } 10^3 \text{ multiplications}$$

Thus the ratio of the number of multiplications in EPANET and this ANN is 283. This is an approximation of the relative efficiency of the ANN as compared with EPANET – for one application (one hour) of the two algorithms.

The results of running the EPANET toolkit for Haifa-A and the ANN(29,90,23) model on the same computer are as follows:

- It takes 5.4 x 10^{-2} seconds to compute one hour's operation of the Haifa-A water network (with 100 iteration);
- It takes 2.2 x 10^{-4} seconds to compute one output vector replicating one hour's operation of Haifa-A.
- The ANN is thus 245 times faster. This figure compares favorably with the theoretical ratio of 283 given above.

In the optimization process we have the following figures:

- Running GA with all program components except simulator (ANN or EPANET) **- 13** generations/second
- Running GA+ANN **- 8.6** generations/second
- Running GA+EPANET **- 0.5** generations/second

Let **T** be the time required for the simulator (EPANET <u>or</u> ANN) to perform one generation in the combined (GA + simulator) optimization process. Then from the figures above:

T(EPANET) = (1/0.5) **-** (1/13) = 1.924 seconds

T(ANN) = (1/8.6) **-** (1/13) = 0.04 seconds

These figures are shown in the following table:

Optimizer	**GA**	**GA+ANN**	**GA+EPANET**
One generation (sec)	1/13 = 0.076	1/8.6 = 0.116	1/0.5 = 2
Simulator		**ANN**	**EPANET**
One generation (sec)		0.04	1.924

The computational efficiency of the ANN relative to EPANET is:

T(EPANET) /**T**(ANN) = 1.924/0.04 = 48.1

The large difference between this figure and comparison above for running times of EPANET and ANN standing alone requires further investigation.

A more significant conclusion from these figures is as follows: improved computational efficiency of the GA+ANN optimization process depends only mildly on the ANN running time. For example, if the ANN could be 10 times faster the GA+ANN optimization process would be only 1.5 times faster.

Estimation of SCADA costs

The task of estimating the investments needed to install a new SCADA system for the Haifa-A network was made by Themes-Water as a sub-contractor of the UNEW team.

In order to have a water system controlled by an optimization program the following requirements are to be met:

- A modeling computer for the program is to be connected to the SCADA system in the control center.
- The SCADA system should have the ability to receive operational information form the optimization program.
- Each pump should have at least flow, power and status measurements.
- Each pumping station should have pressure measurements.
- Each tank should have a level measurement with conversion to the volume.
- Critical pressure nodes should have pressure measurements.
- The SCADA should make a reading cycle for all RTU's at least every 15 minutes.
- All readings should be available for at least two years.

Water pressure and leakage

It is clear that there is a connection between the pressure in a water distribution system and the leakage from it. As the pressure increases so does the leakage. If we can, by using the optimization program lower the overall pressure in the system we can have additional benefits, as the leakage will decrease. The comparison is shown in Table 19: the difference (in meters) between the values with the current operation and those obtained by the optimization by the DGANN+ program. The average values are mostly positive, indicating that the pressure is indeed reduced somewhat in the optimized solution. However, the difference is not significant and cannot be considered a real benefit.

Table 19: pressure comparisons (current operation – optimization values)

Node	54	3	81	107	13	25	19	109	21
Average (m)	0.3	2.9	-0.9	0.3	1.0	0.3	2.4	0.1	0.2
Min (m)	-5.1	-54.5	-36.5	-4.8	-5.2	-3.1	-21.7	-8.2	-3.0
Max (m)	4.3	77.1	30.3	5.6	4.2	3.6	26.6	10.7	4.9

POWADIMA web site http://www.ncl.ac.uk/powadima/

Designed by Derek Jamieson

Implemented by Ofer Menchell

Under supervision of Uri Shamir at TWRI

Installed on the NCL Server

Summary, Conclusions and Recommendations

Creating software for automatic real-time optimal control of a large real water distribution system is a difficult task. It is a problem that has been addressed over the years by many researchers and developers, and still there is no readily available methodology, nor its implementation in software for standard application.

Much of the difficulty stems from the paucity of adequately complete and sufficiently accurate data for modeling the physical system and its actual performance over time. Another source of difficulty, one that is common to all real-time systems, is the random nature of the demands imposed on the system by the consumers. As a consequence of these difficulties, the best one can hope for is a good approximate model of the physical system, a valid method for forecasting the demands, and a robust optimization methodology, one that leads to significant savings of operational costs and is accepted by the operators as providing adequate reliability against supply shortages.

Since our objective is development of a real-time on-line optimization software package, the most important feature of the ANN is its *robustness*, namely its ability to generate good candidates for the optimization program, whether GA or any other search routine, to yield jointly an effective GANN algorithm. It is also important to have a good demand forecasting procedure, so that in combination with the GANN it becomes practical to apply the combined package to real water distribution systems.

The only reason for having an ANN instead of a network simulator such as EPANET is the computational speed of the ANN. For the Haifa-A network each optimization step takes about 2-3 minutes with the ANN and about 10-12 times slower with EPANET. It might be possible to use an EPANET model and not an ANN by:

- Using a much faster computer, or otherwise speeding up the calculations (e.g., parallelization);
- Optimizing the EPANET solver for making many similar runs;
- Skeletonizing the system into a smaller model, still suitable for the optimization.

Another consideration regarding the combined GANN package is the possibility of tailoring the GA to the specific problem we are solving. We have used a general purpose GA, and a specially designed GA for our problem might have better performance. The UNEW team stated that they have introduced improvements into the GA they used, but we have no information.

References

ASCE Task Committee on Applcation of Artificial Neural Networks in Hydrology (2000), "Artificial Neural Networks in Hydrology. I: Preliminary Concepts", Journal of Hydrologic Engineering, ASCE, 5(2), 115-123.

ASCE Task Committee on Applcation of Artificial Neural Networks in Hydrology (2000), "Artificial Neural Networks in Hydrology. II: Hydrologic Applications", Journal of Hydrologic Engineering, ASCE, 5(2), 124-137.

Bhattacharya, B., Lobbrecht, A.H. and Solomanite, D.P. (2003), "Neural Netowkrs and Reinforcement Learning in Control of Water Systems", Journal of Water Resources Planning and Management, ASCE, 129(6), 458-465.

EPANET 2 Users Manual (2000)

Hagan, M. T., H. B. Demuth, and M. H. Beale (1996) "Neural Network Design", Boston, MA: PWS Publishing, 1996.

Riedmiller, M., and H. Braun (1993) "A direct adaptive method for faster backpropagation learning: The RPROP algorithm", Proceedings of the IEEE International Conference on Neural Networks, 1993.