Application Of Heuristics, Genetic Algorithms & Integer Programming At A Public Enterprise Water Pump Scheduling System

Christos Gogos[1,3], Panayiotis Alefragis[1,2] and Efthymios Housos[1]

[1] Computer Systems Lab., Dept. of Electr. & Computer Eng., University of Patras
[2] Dept. of Telecom. Systems & Networks, TEI of Messolonghi, Varia, Nafpaktos
[3] Dept. of Finance & Auditing, TEI of Eipirus, Preveza
{cgogos, alefrag, housos@ece.upatras.gr}

Abstract

In this paper the problem of minimizing the electricity cost required by a water storage and disposal system is analyzed. Three solution approaches are presented focused in the way that various pumps will be scheduled to operate and satisfy prospected water demand while at the same time respect availability of water and reservoirs capacity. The first solution uses a heuristic approach closely related to the way a human operator might have used to solve the problem. The second solution uses mathematical programming, formulates the problem as an Integer Programming problem and solve it by using a Branch and Bound commercial solver. Finally the third solution uses Genetic Algorithms and defines a fitness function describing the attractiveness of each solution for a set of solutions and through a number of evolution steps creates a population with desirable characteristics. A comparative study of the three approaches is presented. The actual formulas that compute the electricity cost is used for the comparisons. Those formulas state the fact that demand peaks especially during high demand periods result in high electricity cost. Data used in our experiments were provided by the municipal enterprise of water supplies and sewage of Chania (Crete).

Keywords: heuristics, genetic algorithms, integer programming, pump scheduling, water reservoir management, optimization.

1. Introduction

Water is said to be one of the most valuable assets for the 21st century. Availability of high quality drinking water without interruptions is considered to be a measure of achievement for societies. Many cities are served for their water needs by water resources that are not always in abundance and this situation is expected to be worse in the future. Usually a group of water customers are served by water pipes that form networks which are directly or indirectly connected to a reservoir. A reservoir concentrates water to serve demand and this is usually done by operating electrical pumps. The planning of the pumping intervals for each pump in order to reduce the cost of the electricity consumed while maintaining certain level for the water storage reservoirs is a challenging exercise for every water department. This stems from the fact that the energy company pricing policy promotes the balanced use of energy throughout the day. A number of different approaches have been used in various situations describing similar problems. In [Ormsbee, Lansey (1994)] a number of linear and nonlinear models for the pump scheduling problem are presented and solved. The use of a genetic algorithm for a similar problem is described in [Mackle et. Al. (1995)]. [Savic et. Al. (1997)] introduces the concept of additionally examining wearing of the pumps because of an extended number of activations and deactivations resulting in a model of a multiobjective optimization problem. The use of a mixed integer nonlinear programming formulation is presented in [Biscos et. Al (2003)] while [Kelner et.Al. (2003)] uses genetic algorithms for optimal pump scheduling in water supply. Finally [Baran et. Al. (2005)] models water distribution and pump scheduling as multiobjective optimization problems. In section 2 the pump scheduling problem is presented, the electricity charge model is analyzed and a mathematical description of the problem is given. The solution methodology of the heuristic approach is presented in section 3 followed by description of solution methodologies for the mathematical programming approach and the genetic algorithm approach. In section 4 software design issues of the system and libraries and frameworks that were used are examined. In section 5 the solution methodologies are compared. Finally conclusions and future work are presented.

2. Pump Scheduling Problem

The problem that has been modeled involves the creation of an operation schedule for a duration of N periods for a system consisting of a number of reservoirs each having a number of pumps with different characteristics. Typical values for N and period length are 48 and 30 minutes respectively. A profile exists about the available water per period which is comprised by forecasted values. All pumps are provided with water from the same resource so pump operation should be organized so as to avoid request of water in cases where the available water volume has already been drained by other pumps operating in parallel. Each reservoir with its associated pumps serves a number of water customers and a profile about the prospected water demand per period is known to the system. Every reservoir has a specified capacity volume imposed by its construction characteristics. Additionally lower level values for each period define the desired lower water level for each reservoir and often implement various policy and security issues and concerns. These limits represent the accumulated knowledge with respect to the time of the day and the season of the year. Finally initial water volume and desirable water volume at the end of the complete planning period for each reservoir is specified. The schedule will specify whether each pump will be operating or not during each period of the time horizon. The effective pumped water and the electricity used when a specific combination of pumps are utilized for a given time period, is situation specific and their values are experimentally estimated. In Table 1 a sample of actual values used for experiments are presented. It is worth noticing that the model is non linear and simultaneous operation of pumps decrease the volume of water that each pump can contribute to the reservoir.

Table 1. *Pumped water – electricity demand*

Reservoir 1 [3 pumps]				
Pump Combination			Water inflow (m3/hour)	Power demand (KW)
Pump1	Pump2	Pump3		
OFF	OFF	OFF	0	0
ON	OFF	OFF	300	120
OFF	ON	OFF	500	210
OFF	OFF	ON	550	240
ON	ON	OFF	700	285
ON	OFF	ON	750	295
OFF	ON	ON	900	325
ON	ON	ON	1000	340

2.1. Electricity Charge Model

The electricity pricing model promotes the use of electricity during certain periods of the day (night zone) and discourages high demand during certain other periods (peak zone). ZAX is the maximum value of the requested power demand during peak zone intervals, ZNYK is the is the maximum value of the requested power demand during night zone intervals and ZHM is the maximum value of the requested power demand during non peak or night zone intervals.

KMZ is the maximum value between ZAX and ZHM and di is a discount factor computed by the following formula.

$$di = 50.0 - 50.0 * ZAX / ZHM \tag{1}$$

Finally φ is the angle between the active and reactive requested power for the location. XZ is the chargeable demand and can be computed from the following formula.

$$XZ = \begin{cases} MRD * di * \dfrac{0.8}{\cos\phi}, & \cos\phi \le 0.8 \\ MRD * di, & 0.8 \le \cos\phi \le 0.85 \\ MRD * di * \dfrac{0.85}{\cos\phi}, & \cos\phi \ge 0.85 \end{cases} \tag{2}$$

In Figure 1 the effect of increasing requested demand during peak hours is depicted. Cost increases in a piecewise linear function with greater slope for values of ZAX greater than the maximum value between ZHM and ZNYK.

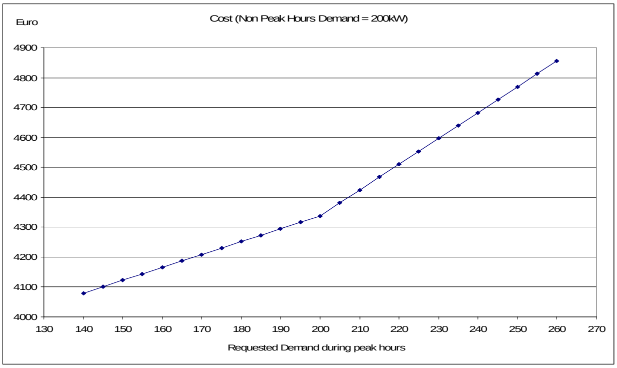

Figure 1. *Effect of increasing ZAX*

2.2. Mathematical representation

The mathematical representation of the problem is as follows. Let $r_i \in R$ be the individual reservoirs, $t_j \in T$ be the time units and $x_{ij} \in X$ be the pump combination number active in reservoir r_i at time period t_j. The number of feasible pump combinations is installation specific. Let $AW_j \in A$ be the available water for all functional units in time unit t_j, $IV_i \in I$ be the initial volume of water at the beginning of the scheduling period at reservoir r_i, $UV_i \in U$ be the maximum volume of water at reservoir r_i and $EV_i \in E$ be the lowest acceptable volume of water at the end of the scheduling period at reservoir r_i. Also, let $LV_{ij} \in L$ be the lowest acceptable volume of water at time period t_j for reservoir r_i, $D_{ij} \in D$ be the forecasted water demand for time period t_j and for reservoir r_i, $W_{ik} \in W$ be the consumed electricity of reservoir r_i when pump combination k is used, $PW_{ik} \in P$ be the pumped volume of water of reservoir r_i when pump combination k is used and $V_{ij} \in V$ be the actual volume of water at the end of time period r_j at reservoir r_i. It can easily be observed that

$$V_{ij} = \begin{cases} IV_i - D_{ij} + PW_{ik}, x_{i1} = k, \ j = 1, i:1...n \\ V_{ij-1} - D_{ij} + PW_{ik}, x_{ij} = k, \ j > 1, i:1...n \end{cases} \quad \textbf{(3)}$$

The model must follow a set of constraints.

Maximum and minimum volume limit constraints

$$LV_{ij} \le V_{ij} \le UV_i, i:1...n, \ j:1...m \quad \textbf{(4)}$$

Lowest Acceptable water volume at the end of the scheduling period

$$V_{ij} \geq EV_i, i:1...n \quad \textbf{(5)}$$

Maximum allowable pumped water

$$\sum_{i=1}^{n} PW_{ik} <= AW_j, x_{ij} = k, j:1...m \quad \textbf{(6)}$$

For the solution methodology of this paper, a cost/fitness function that takes into account three factors which measure the cost and applicability of the proposed solution is used. These factors are:

1. Requested Demand (RD), which sums the requested power
2. Spike Demand (SD), which counts the difference in the requested power in consecutive periods
3. Maximum Demand Avoidance (MDA), which measures the difference in the requested power from the maximum

For every time unit *t*, the corresponding RD_t is calculated as the sum of the requested power of all the pumping functional units that where operational during a specific time unit.

$$RD_t = \sum_{i=1}^{n} W_{ik}, x_{it} = k \quad \textbf{(7)}$$

The *RD* is the weighted sum of all RD_t. The weight multipliers c_t are user defined and during the night hours $c_t \in [0, 0.5]$, while during peak hours $c_t \in [2, 5]$. Thus

$$RD = \sum_t c_t RD_t, t:1...m \quad \textbf{(8)}$$

SD sums the increase of the requested power in consecutive time periods.

$$SD = \sum_t \max(0, c_{t+1} RD_{t+1} - c_t RD_t), t:1...m \quad \textbf{(9)}$$

It should be noted that it is desirable to minimize SD because a large difference in the demand of two consecutive periods, creates demand spikes and thus significantly bigger values for MRD. Moreover, it is desirable that the pumps usage pattern is smooth in order to maintain medium to high reservoir levels, thus creating a robust water availability state. Medium to high water availability will also benefit future power consumption, as in the case of high water demand, the available water will act as a buffer and fewer pumps will start operating concurrently in order to maintain the water level in the reservoir.

MDA counts the difference of MRD and the theoretical maximum requested demand MRDmax. MRDmax can be calculated as the requested demand when all pumps are operated concurrently.

$$MDA = MRD_{\max} - MRD \qquad \textbf{(10)}$$

The fitness objective function calculates the weighted sum of the three factors.

$$O = \min(c_1 RD + c_2 SD + c_3 MDA), \sum c_i = 1 \qquad \textbf{(11)}$$

3. Solution methodology

The user has the option to select any of the three solvers to generate an initial solution or improve upon the current solution. The user selects or creates new profiles about demand per reservoir per period, availability of water per period and desirable lower volume per reservoir per period. Initial volume and desirable final volume per reservoir and night and peak periods can be selected or altered.

3.1. Heuristic Solver

The heuristic solver implements a simple strategy which tries to rearrange operational pump configurations from high demand periods to night periods. In this strategy, for every water reservoir and for every time period, it selects the pump combination that can provide the water volume which approaches the volume of water for the same period that is forecasted to be the demand.

```
forall (r in reservoir)
    for all (t in periods)
        min=MAXDOUBLE
        x[r,t] = null
        forall(c in comb[r])
            if (abs(demand[t] – pw[r,c]) < min)
                min = abs(demand[t] – pw[r,c])
                x[r,t] = c
if(!feasible(x))
        repair(x)
```

This strategy provides feasible solutions most of the time by maintaining a nearly constant water level and emulates the strategy that has been used in the past by the users.

Repair algorithm

In case that a constraint is violated the following algorithm is used to repair the solution. Symbol x[r,t] refers to the selected combination per reservoir r per period t.

```
tries=0;
While (!repaired(x)) and (tries <= ACCEPTABLE_NR_OF_REPAIRS )
        If (underflow(x))
                Try firstly in night zone, secondly in
                normal zone and finally in peak zone
                one_more_pump(x)
        Else if (overflow(x))
                Try firstly in peak zone, secondly in
                normal zone and finally in night zone
                one_less_pump(x)
        Else if (request_more_than_available_water(x))
                Randomly choose a Reservoir r
                locate period t with problem
                x[r, t]=0
        increment(tries)
EndWhile
If (repaired(x))
        Accept solution
Else
        Reject solution
```

When a lower level constraint is violated the algorithm activates a combination of pumps in the offending reservoir. It selects the combination that more closely resembles the previously active combination. A nice feature of the repair algorithm is that while trying to deactivate pumps it does so by giving priority to peak hours while in the other hand in trying to activate new pumps it prefers night hours. Subroutine one_more_pump(x) locates a Reservoir r with underflow problem by randomly selecting a period rk between the first period and the period that manifested the problem. In the subroutine's code symbol aw[t] refers to the available volume of water per period t, d[r,t], lv[r,t], pw[r,t] and v[r,t] are respectively the demand, the lower volume, the pumped water and the volume per reservoir r per period t. Finally symbol tpw[t] refers to the totally pumped water per period t and uv[r] is the upper volume per reservoir r.

```
repair_action = false
k = rk
While (repair_action == false) && (k >=1)
        mindiff = MAXDOUBLE
        forall(c in comb[r])
                If ((pw[r,c]>pw(r, x[r,k]))
                && (abs(pw[r,c]- pw(r,x[r,k])) < mindiff)
                && (v[r,k]+ pw[r,c] - pw(r,x[r,k]) <= uv[r])
                && (tpw[k]+pw[r,c]- pw(r,x[r,k]) <= aw[k]))
                        mindiff = abs(pw[r,c]-pw(r,x[r,k]))
                        new_selected_combination = c
        If (new_selected_combination <> x[r,k])
```

```
            x[r,k] = new_selected_combination;
            repair_action = true
        decrement(k)
    EndWhile
```

Subroutine one_less_pump(x) locates a Reservoir r with overflow problem by randomly selecting a period rk between the first period and the period that manifested the problem. The only difference compared with one_more_pump is the if statement condition that is presented below:

```
(pw[r,c]<pw(r,x[r,k])) && (abs(pw[r,c]-pw(r,x[r,k]))< mindiff)
            && (v[r,k]+pw[r,c]pw(r,x[r,k])<=lv[r,k])
```

3.2. Integer Programming Solver

Another implemented solution strategy was to solve the scheduling problem using mixed integer linear programming. The problem was formulated to include a number of linear constraints and an objective function. The decision variables x_{it} represents the active combination of water pumps for each reservoir and period. Model constraints were easily mapped to the underline solvers. The only problem we encountered was that the introduction of constraint (9), referring to the differential increase of the requested power in consecutive time periods, created problem instances that were quadratic in nature and as a consequence the solution time was not acceptable. Therefore, we relaxed the formulation of the problem by removing this constraint To formulate and solve the problem, we used the commercial integer modeling platform ILOG CPLEX 9.1[http://www.ilog.com/products/cplex/] as the optimization engine through ILOG Concert Technology. Moreover, we experimented with the open source mathematical programming platform GLPK [http://www.gnu.org/software/glpk/]. Although, mathematical programming can provide a proven optimal solution, for our application this was not a requirement, as the goal was to provide good quality solutions in a timely manner. We selected to provide the user with the option of selecting three stopping criteria. The first one was that difference between the current solution and the lower bound was below a user defined threshold. The second criteria, was the time spend in the solution process. The final one was the differential progress in the objective value per number of processed nodes in the branch and bound tree. The user can create any combination of stopping criteria, which are OR enforced.

3.3. Genetic Algorithms Solver

The final implemented solution strategy was to solve the scheduling problem by using Genetic Algorithms (GA) as the solver engine [Goldberg (1998)]. GAs use techniques inspired by evolutionary biology such as inheritance, mutation, natural selection, and recombination (or crossover). GAs require the definition of solution instances as chromosomes. A chromosome consists of genes. Every gene of the chromosome corresponds to a reservoir and period combination and its value (allele) is the number of the active pump combination in the reservoir for the specific period. A population of chromosomes is initially constructed. With higher probability better solutions are combined to give a new population of the same size which will be the next generation. If by combining two solutions the new one is infeasible then a repair phase happens. The repair phase may lead to an accepted feasible solution or to a rejected still infeasible solution. The evolution happens for a number of generations and finally the best solution of the last generation is presented to the user. More information about the Genetic Algorithm Solver can be found at [Gogos et. Al. (2005)]. A fitness function is formulated giving low values for good solutions, higher values for less good solutions and MAXDOUBLE for infeasible solutions. The fitness of each chromosome is computed by aggregating the three factors that has been analyzed in the objective function (10). Every chromosome gets a fitness value. The same repair procedure that was described in paragraph Repair Algorithm of (3.1) is used. To implement the solution we used the open source package JGAP[http://jgap.sourceforge.net/] which is a java genetic algorithm package. JGAP provides basic genetic mechanisms that can be used to apply evolutionary principles in order to achieve high quality solutions to a wide range of problems. Various parameters of the genetic algorithm like population size, number of generations, mutation probability, elitism etc had to be tuned in order to get acceptable results.

4. System Design & Implementation

One of our requirements was that a family of interchangeable algorithms could be implemented and made available for use. Moreover, from a software engineering point of view, we would like to create interchangeable components for communicating with data sources, data acquisition mechanisms and reporting tools that could be implemented independently and integrated based on the requirements of the end user. Our approach was to create a generic solution model that could integrate different solution strategies and incorporate different solver engines. In our effort, we used the principles of Strategy and Model View Control design patterns. Strategy lets the algorithm vary independently from the clients that use it. Model View Control separate data handling and presentation from requests for calculations. Our implementation, fulfills our requirement and provides us with the means to easily create various usage scenarios, combine or compare different algorithms, create various reports and compare problem solution properties like quality, feasibility and robustness.

5. Comparative study

A sample model instance with 2 reservoirs each with 3 pumps each was solved using all three of the available solvers. Night periods were specified to be 0-13 and 46,47 while peak periods were 21-26 and 36-41. Each solver found a feasible solution for a 48 period schedule time and calculated the electricity cost due to power demand projecting the results of the scheduling period to a full month which is the typical time between electricity payments. The results of executing the algorithms are summarized in table 2.

Table 2. *Results*

Solver	ZAX	ZHM	Discount	Estimated Cost
Heuristic	396	407	1,35%	3.687 €
Integer Programming	285	400	14,38%	3.145 €
Genetic Algorithms	210	407	24,20%	2.833 €

The integer programming solver used a threshold of 1.5% for reaching the optimal solution to the relaxed linear problem and needed about 1 minute to find an integer solution within that range. The genetic algorithms solver used a population of 300 solutions that was evolved for 3000 generations. The best solution of each generation was preserved(elitism) and the mutation probability was 0.1%. Solution time was less than 5 minutes. A graphical representation of the aggregated power demand per period for the Genetic Algorithms Solver can be seen in Figure 3. Genetic algorithm solver gave the best results but that happened only after fine tuning factors that were included in the fitness function and giving more than 60% weight to the factor that refers to the maximum demand avoidance.

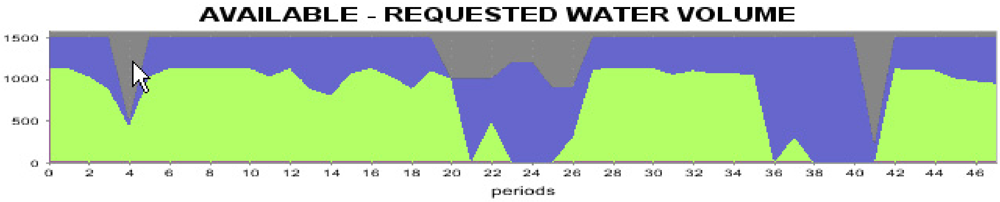

Figure 2. *Available Water vs Aggregated Pumped Water*

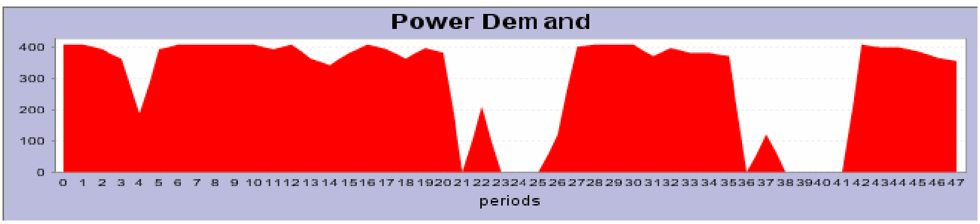

Figure 3. *Genetic Algorithms Solver*

6. Conclusions

In this paper, an application that solves the problem of activating and deactivating pumps in a set of related reservoirs was presented. We demonstrated that significant cost reductions can be achieved without changing existing pumps and reservoirs configuration. Our system suggests a program that cuts energy peaks and promotes operation during the night hours resulting to cost savings. It is easy to use, fairly fast and can be applied to a number of different configurations. As a future improvement we are planning to use the IP solver in a hybrid approach as a chromosome repairing mechanism which tries to minimize the number of pump switching while repairing.

References

Baran B., Lucken C., Sotelo A. 2005, *Multi-objective pump scheduling optimization using evolutionary strategies*, Advances in Engineering Software, Elsevier, Vol. 36, pp 39-47.

Biscos C., Mulholland M., Lann M., Buckley C., Brouckaert C. 2003. *Optimal Operation of Water Distribution Networks by Predictive Control using MINLP*, Water SA, Vol. 29 No.4.

Coello C. 2000, *An updated survey of GA-Based Multiobjective Optimization Techniques*, ACM Computing Surveys, Vol. 32, No. 2.

Goldberg D 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley.

Gogos C., Alefragis P., Housos E. 2005, *Public Enterprise Water Pump Scheduling System*, IEEE ETFA05, Conference Publication.

Kelner V., Leonard O. 2003, *Optimal Pump Scheduling for Water Supply Using Genetic Algorithms*, Eurogen.

Mackle G., Savic D., Walters G. 1995, *Application of Genetic Algorithms to Pump Scheduling for Water Supply*, GALESIA '95, Conference Publication 4/4, pp 400-405.

Ormsbee L., Lansey K. 1994, *Optimal Control of Water Supply Pumping Systems*, Journal of Water Resource Planning and Management, 120(2).

Savic D., Walters G., Schwab M. 1997, *Multiobjective Genetic Algorithms for Pump Scheduling in Water Supply*. Lecture Notes In Computer Science; Vol. 1305, pp 227-236.