PIPE NETWORK DESIGN USING MIXED SIMULATED ANNEALING AND TABU SEARCH - MSATS

André Jacomel Torii, ajtorii@hotmail.com
Universidade Federal do Paraná
Jucélio Tomás Pereira, jucelio@utfpr.edu.br
Universidade Tecnológica Federal do Paraná

Abstract. *The problem of designing a pipe network can be stated as: find the pipes diameters of a given network that leads to a minimum building cost; when operating at given boundary conditions (nodal pressures and discharges) and respecting some prescribed pressures (necessary for the correct use of the network). However, this design can become very complex since the equations governing the analysis of pipe networks are nonlinear. Consequently, small changes in the diameters can lead to significant changes in the behavior of the network, which difficults an iterative design process. Therefore, the use of optimization methods for the design of pipe networks is an interesting alternative. However, many difficulties arise when classical convex numerical optimization techniques are applied to this problem. In this context, the use of metaheuristics for the design of pipe networks appears evident, since these methods present answers to most difficulties related to the use of classical optimization techniques. These methods don't need information about gradients of the objective function; they can be successfully applied to problems with several local minima; and they can handle discrete design variables naturally. Among the methods grouped as metaheuristics, the Mixed Simulated Annealing and Tabu Search (MSATS) presents some characteristics which make attractive its use for the design of pipe networks, as already shown by other authors. This paper presents the description of the design process of pipe networks based in the MSTAS heuristics; together with some examples to demonstrate the application of the method.*

Keywords: *pipe network, water distribution, optimization, simulated annealing, tabu search, mixed simulated annealing and tabu search*

1. INTRODUCTION

The design of pipe networks is a classical problem in engineering practice. However, since the equations associated with the analysis of pipe networks are nonlinear, several complications arise when designing such networks. These equations are sensible to small changes in pipe diameters and the design of large networks may become a complex process, since changing the diameter of a single pipe may lead to very different pressures in the entire system.

Many optimum design problems can be stated as optimization problems, and consequently be solved by optimization techniques (Arora, 2004; Nocedal and Wright, 1999). In the case of pipe networks, the objective function of the problem can be defined as the building cost. The optimization algorithm will then search for a network with minimum cost. Besides, constraints on minimum nodal pressures, necessary for the correct use of the network, are defined. The set of all solutions which respect these constraints are then called the feasible domain of the problem. Finally, the parameters to be modified by the algorithm are the pipes diameters, thus defining the design variables of the problem. The optimization algorithm will then change the design variables, in order to obtain an optimum solution for the problem which respects the constraints.

However, most convex optimization techniques (Nocedal and Wright, 1999) can't be successfully applied to the design of pipe network for a series of reasons. First, most convex optimization techniques need information about the gradient of the objective function and the constraints. However, the efficient evaluation of the gradient of the constraints of this problem is difficult, since the relation between the pipes diameters and the nodal pressures is defined by an implicit relation, given by a system of nonlinear equations. Second, the optimization problem defined by the pipe network design can present several local minima solutions. Since most convex optimization techniques generally converge to the closest local minimum, these methods may give solutions which are far from global optimum solutions. Finally, the pipe diameters available for construction are, in most cases, to be chosen from an array of discrete values. However, most convex optimization techniques were conceived for optimization problems with continuous variables, and handle discrete variables only after significant modifications.

The optimization techniques commonly grouped as metaheuristcs, however, present good answers for the needs of the design of pipe networks. These methods don't need the evaluation of gradients; may skip local minima by several different strategies; and can handle discrete variables naturally. In fact most of these methods were originally conceived for discrete variables. Consequently, the use of optimization methods such as Genetic Algorithms (GA) and Simulated Annealing (SA) prevail over the use of convex optimization methods for the problem here studied. However, it must be noted that these techniques may not find the real optimum solution of the problem in some cases, due to the probabilistic nature of the search procedure (Dréo et al., 2006).

Several references in literature can be found describing the use of GAs to the problem of pipe network optimization (Savic and Walters, 1997; Gupta et al., 1999, Reca et al., 2008), since this is one of the most popular metaheuristics.

GAs present excellent results in finding promising regions of the feasible domain (i.e. regions were a "good" solution can probably be found), but may present, in some cases, difficulty in actually converging to a true solution of the problem. GAs work with several solutions at the same time; and new solutions are conceived basically by crossing characteristics of two solutions and applying mutations (Goldberg et al., 1987; Dréo et al., 2006). Consequently, when dealing with a network composed of several pipes, it may be difficult to find the exact optimum diameters of a given pipe, since it is difficult to change one design variable at a time. The Simulated Annealing (SA) technique, instead, may not be so efficient in finding promising regions of the feasible domain, since it works with a single solution at a time. However, for the same reason, it can, in some cases, succeed in finding solutions closer to the true solution of the problem, since the SA algorithm is able to modify one design variable at a time.

Since the pioneering work of Kirkpatrick et al. (1983) on the SA, some works used this technique for solving the problem of pipe networks optimization (Reca et al., 2008). However, the recently proposed optimization technique called Mixed Simulated Annealing and Tabu Search (MSTAS) (Gil et al., 2002) distinguished itself from other metaheuristic when applied to the problem here discussed (Reca et al., 2008). This heuristic is the result from incorporating some concepts from Tabu Search (TS) into a basic SA algorithm. This may prevent the algorithm from visiting already visited solutions, by defining some tabu moves. Consequently, this approach can lead to more efficient algorithms than the SA or TS alone (Gil et al., 2002; Reca et al., 2008).

Following the concepts presented by Reca et al. (2008) and Formiga (2005), this paper applies the MSATS to the design of pipe networks. It is important to note that the MSATS is not radically different from the SA. In practice, it turns out that only few modifications are made to a basic SA algorithm. However, this paper shows that a more efficient MSATS algorithm may be developed if more concepts from TS are used than just that of a Tabu List. This paper does not compare the MSATS with other heuristics other than the SA, and the reader is referenced to Reca et al. (2008) for a comparison with methods such as GA and TS. The purpose of this work is to described, in details, how the SA and the MSATS can be applied to pipe networks. Besides, a few modifications from the original algorithm from Gil et al. (2002) are proposed.

Finally, it is important to note that the optimization problem used in almost every work on the optimization of pipe networks are only simplifications of the real problem of design a pipe network. The paper by Walski (2001) already pointed out these important differences, and even some considerations on the simplifications made during the analysis of such networks were discussed in Todini (2003). For a more detailed modeling of such problems the reader can consult Formiga (2005).

2. MIXED SIMULATED ANNEALING AND TABU SEARCH

2.1 Simulated Annealing

The Simulated Annealing algorithm is based on a real process called annealing, which is used to rearrange the molecules inside a solid (Kirkpatrick et al., 1983; Dréo et al., 2006). In the real annealing technique, a solid is heated, thus imparting high energy to it. This energy allows the molecules to move, and consequently the molecules start to rearrange themselves inside the body. During this process, the molecules will likely move to an arrangement with less potential energy than the current arrangement, since this state will be more stable from the physical point of view. The material is then cooled slowly, thus reducing the total energy available in the system. This gradual reduction of the energy leads to a reduction in the movement of the molecules, until the material solidifies itself again and the molecules stop moving. At the end of the process, the solid will probably reach a state of minimal potential energy, since rearrangements which reduce the potential energy prevail over other rearrangements. This state of minimal energy corresponds to a more stable state, represented by an ordered arrangement of the molecules (a crystal, for example). Besides, the energy given to the system allows the molecules to sometimes move randomly, thus allowing them to skip local minima of potential energy.

Applying these concepts to optimization problems leads to the Simulated Annealing algorithm (SA), first described by Kirkpatrick et al. (1983). In this approach, the design vector (the vector composed of the design variables) is analogue to the arrangement of the molecules of the solid in the real annealing technique. Thus, the design variables are analogue to the position of these molecules. In order to simulate a random move of the molecules, a perturbation is applied to the current design vector. This is made by moving the design vector to a neighbor state by randomly modifying it. This neighbor solution will, in general, have energy (the value of the objective function) different from the energy of the current design vector. If this energy is lower, the new design vector is taken as the new current solution of the problem. However, if the energy is higher there are two possibilities. First, if the temperature of the system is high, random moves to states of higher energy may be allowed. However, if the temperature is low, moves to states of higher energy will probably be discarded. This condition for accepting or not perturbations which increase the energy of the solution (a worsening in the objective function) is called acceptance rule.

The acceptance rule can be checked, based on concepts from statistical physics (Dréo et al., 2006), by drawing a random number p in the interval [0,1] and accepting the new solution if

$$p \leq e^{\frac{-\Delta E}{T}}, \tag{1}$$

where $-\Delta E$ is the change in energy (change in the objective function) and T is the current temperature of the system.

Note that according to Eq. (1), worse solutions will be accepted more easily when the temperature is high, and will practically be inadmissible when the temperature is close to zero. Consequently, it is necessary to define the temperature at each stage and described how the temperature will change during the SA procedure. For this purpose, the geometrical law of decrease can be used, which is defined as (Dréo et al., 2006)

$$T_{k+1} = \alpha . T_k \quad , \alpha < 1 , \tag{2}$$

where T_{k+1} is the temperature at the next stage, T_k is the temperature at the current stage and α is a constant.

From Eq. (2) it can be seen that an important aspect of the algorithm is the definition of an initial temperature for the procedure. It is desirable that at the initial stages of the algorithm, moves toward states with higher energy (worsening of the solution) be allowed. Consequently, the initial temperature must be defined accordingly. An adequate value for this parameter can be obtained empirically for some cases, but the procedure described by Dréo et al. (2006) may be recommended. In this procedure, an initial rate of acceptance τ_0 is defined. Then, a number of disturbances are randomly applied to the initial solution, and the mean value of $|\Delta E|$ is obtained from the corresponding values of ΔE. The initial temperature T_0 is then deduced from Eq. (1), where p is substituted by τ_0 and $-\Delta E$ is substituted by the mean value $-|\Delta E|$.

In the real annealing procedure, the temperature is decreased only after the system reaches thermodynamic equilibrium, for the current temperature. However, thermodynamic equilibrium is an abstract concept in the SA algorithm. One common way of assuming thermodynamic equilibrium in SA is when a given number of moves are accepted or when a given number of moves are attempted. Even if such number may be different for different problems, Dréo et al. (2006) suggest the following conditions for assuming thermodynamic equilibrium, considering the problem has N design variables: $12.N$ perturbations are accepted or $100.N$ perturbations are attempted. Note that different conditions may be defined for each problem, and the conditions given previously are only suggestions.

Finally, it is necessary to define a termination criterion for the procedure. The procedure can be stopped when a given number of temperature stages (3, for example) are done without accepting a perturbation. This is analogue to reaching a state where the molecules do not move anymore.

Considering the previous paragraphs, the basic SA algorithm can be summarized as:

1. Start the algorithm with an initial solution; an initial temperature; a decrease rule for the temperature; and an acceptance rule;
2. For the current temperature repeat until thermodynamic equilibrium is assumed:
    a. Apply a perturbation to the current solution, obtaining a trial solution;
    b. If the trial solution is better than the current one, take it as the new current solution. Else, take it as the new current solution only if it satisfies the acceptance rule from Eq. (1) for a random number p between in the interval [0,1];
3. Evaluate the new temperature according to its decrease rule from Eq. (2);
4. If the termination criterion is satisfied, stop the algorithm. Else, return to step 2.

2.2 Mixed Simualted Annealing and Tabu Search - MSATS

The Tabu Search (TS) algorithm was first present by Glover (1986), and was originally conceived for solving combinatorial optimization problems. The concepts behind TS are extensive and this paper does not try to make full use of them. Instead, in the context of this paper, two main concepts are necessary: that of short-term memory and that of aspiration conditions.

In the SA algorithm, it can happen that a perturbation brings the solution back to a state previously visited. This may be desirable at long term, since this allows the algorithm to return from regions with unpromising solutions by the same way it entered the region. In fact, important aspects of the SA theory are related to the possibility of returning to previous states (Dréo et al., 2006). However, at short term, allowing the algorithm to move to states already visited may be disadvantageous, since this allows the algorithm to move in circles and prevents an adequate exploration of the feasible domain.

It is then desirable to avoid moving back during the search procedure, but without prohibiting the algorithm from moving to an already visited state when strictly necessary (when moving away from a local minimum, for example). The answer to this need is the concept of short-term memory, borrowed from TS. According to this concept, moves made in the N past steps are "remembered" and the corresponding reverse moves prohibited (stored in a Tabu List). However, the reverse moves are prohibited only during N steps, and after this time they are allowed again. Thus, the list changes continually, since one new tabu move is added to the list when a move is taken. Besides, at each step, prohibited moves older than N are deleted from the Tabu List. Incorporating a Tabu List into an SA algorithm was first

proposed by Gil et al. (2002), and leads to the development of the MSATS. This approach was also applied to the problem of optimization of pipe networks by Reca et al. (2008).

The concept of aspiration conditions can be better explained by the following reasoning. Sometimes, the algorithm may find itself close to an optimum solution of the problem, but the Tabu List may prevent it from reaching this solution. This may happen since the necessary move may be currently prohibited. In order to avoid this undesirable effect, the concept of aspiration conditions (Dréo et al., 2006) can be used. In these cases, the algorithm may accept a move even if it is currently a tabu move, if this move leads the algorithm to a solution better than all those visited previously, which is aspirated by the search procedure. Consequently, the algorithm will not lose opportunities to improve the solution, even if the move needed is currently a tabu move. The use of this concept, which was not discussed in the original paper by Gil et al. (2002), may lead to important improvements in the MSATS algorithm.

Considering the previous paragraphs, the Mixed Simulated Annealing and Tabu Search algorithm as here proposed can be summarized as:

1. Start the algorithm with an initial solution; an initial temperature; a decrease rule for the temperature; an acceptance rule, and a size for the Tabu List;
2. For the current temperature stage, repeat until thermodynamic equilibrium is assumed:
    a. Apply a perturbation to the current solution obtaining a trial solution;
    b. Check if the move that leads to the trial solution is a Tabu Move;
    c. If the move is a currently prohibited, accepts the trial solution only if it is better than the best solution obtained so far. Else, accept it or not according to an acceptance rule;
    d. If the trial solution is accepted and it improves the solution, put the reverse move in the Tabu List;
    e. If the Tabu List gets bigger than the maximum size allowed, delete the bottom entries of the list until its size becomes the admissible one;
    f. Store the best solution obtained so far;
3. Update the temperature according to its decrease rule;
4. If the termination criterion is satisfied, stop the algorithm and take the best solution obtained so far as the solution of the problem. Else, return to step 2.

Note that the algorithm here proposed presents a significant modification from the one proposed by Gil et al. (2002), the use of aspiration conditions. Even if this is a simple modification, this may lead to significant improvement of the algorithm, as shown in a following example.

In the MSATS algorithm, a reverse move is only included in the Tabu List if the corresponding move improves the solution. The reason for this is that the SA may accept random moves, which may not represent a promising search direction. Consequently, creating a Tabu List based on every accept move may give a Tabu List which does not "guide" adequately the search procedure. However, if only moves which really improve the solution are considered, the Tabu List will be composed of really pertinent moves. Therefore, the list will "guide" the search procedure adequately.

3. THE PROBLEM OF PIPE NETWORK DESIGN

3.1 Statement of the problem

The problem of pipe networks optimization can be stated as:

find **d**

which gives

$$\min C(\mathbf{d}) = \sum_{i=1}^{n} c_i . l_i \ , \tag{3}$$

subjected to the constraints

$$g(\mathbf{d})_j = p(\mathbf{d})_j - p_j^{\min} \geq 0 \quad (j = 1,..,m) \tag{4}$$

and

$$d_k \geq 0 \quad (k = 1,..,n) \ , \tag{5}$$

where **d** is the design vector composed of the pipe diameters, C is the total cost of the network, c_i is the cost per length of the pipe i, l_i is the length of the pipe i, n is the number of pipes, g_j is a pressure constraint, p_j is the pressure at a given node, p_j^{min} is the minimum allowable pressure at this node, m is the number of pressure constraints, d_k is the diameter of a given pipe and n is the number of pipes to be designed.

Equation (3) represents the fact that a minimum cost of the network is sought, by changing the diameters of the pipes. Equation (4), however, states that the pipes diameters must give the minimum pressures required to the use of the network. Finally, Eq. (5) just states that the diameters cannot be negative, since this is physically impossible.

Optimization problems that are stated in the form of the previous equations can be solved by several optimization methods. However, some aspects of this problem restrict the range of available methods that can be efficiently applied. First, note that in order to evaluate the nodal pressures, it is necessary to solve a system of nonlinear equations. Consequently, using methods that need to evaluate the gradient of the objective function and of the constraints may lead to numerical difficulties. Second, the diameters available for the design come from the standard diameters produced by industry, and consequently the design variables of this problem are discrete values. Finally, this problem may contain several local minima, and convex optimization algorithms will likely give unsatisfactory solutions for the problem. As explained in the previous section, metaheuristics present answers to these difficulties, and consequently are frequently used for the solution of the problem of pipe networks design.

However, if metaheuristics is used for this problem, it is convenient to transform Eq. (3) and Eq. (4) into a single equation. This new objective function will then be used to compare different solutions by a single scalar. Equation (3) and Eq. (4) can be transformed to an optimization problem without constraints by a penalization method (Nocedal and Wright, 1999), giving

$$\min F = w_c . \sum_{i=1}^{n} c_i . l_i + w_g . \sum_{j=1}^{m} \max\left[p_j^{\min} - p_j, 0\right], \tag{6}$$

where F is the new objective function of the problem, w_c is a weighting factor for the cost and w_g is a weighting factor for the constraints. The weighting factors are defined in order to allow the tuning of the importance of the cost and of respecting the constraints. The first part of F is the cost of the network, and consequently the new objective function will still decrease when the cost decreases. However, note that the second part of F is now composed of a function that will give positive values when the constraints are violated and zero when they are respected. Consequently, not respecting the pressure constraints will increase the value of the objective function. If the values of w_c and w_g are chosen appropriately, the minimization of F will give the network with minimum cost which respects the pressure constraints. Besides, this expression can be handled easily by metaheuristics since it gives a scalar. Note, however, that solutions which do not respect the pressure constraints are not forbidden, just avoided.

The constraints defined in Eq. (5) were dropped since the design variables (pipe diameters) will now be chosen from an array of available diameters given by the engineer. Thus, it is not necessary to define bounds on the allowable values of these parameters.

3.2 The design vector and perturbations

As described previously, the design vector of the problem is composed of the pipe diameters. However, these diameters are to be chosen from an array of standard diameters defined by

$$\mathbf{D} = \{D_1, D_2, ..., D_{ND}\}, \tag{7}$$

where D_i are the available diameters and ND is the number of available diameters for the design. The pipe diameters of the network can then assume the values given in **D**. However, note that defining too many available diameters leads to more complex optimization problems, and may consequently burden the optimization algorithm. Besides, each diameter D_i has a cost per length c_i, which is used to evaluate the total cost of the network.

A given design vector of the problem will be

$$\mathbf{d} = \{d_1, d_2, ..., d_n\} \quad , d_i \in \mathbf{D}, \tag{8}$$

where n is the number of pipes in the network. The design vector **d** will be composed of the diameters available in **D**.

In SA, the algorithm moves between neighbor solutions in order to reach an optimum design vector. Thus, given an arbitrary design vector **d**, in order to obtain a new design vector in its neighborhood N(**d**) it is necessary to apply a perturbation to **d**. A perturbation can be applied to **d** by the following procedure:

1. Start the procedure with a design vector **d** and a vector of available diameters **D**;
2. Drawn an integer random number p between 1 and n, randomly choosing a pipe from **d** to be modified;

3. If the pipe p has a diameter d_p equal to D_1, assign it the new diameter D_2. Else, if it has a diameter equal to D_{ND}, assign it the new diameter $D_{ND\text{-}1}$. Else, the diameter will be equal to D_i, so chose randomly between assigning it D_{i+1} or $D_{i\text{-}1}$;
4. Terminate the procedure with the design vector $\mathbf{d}_t \in N(\mathbf{d})$.

From the previous algorithm, note that the perturbed vector $\mathbf{d}_t$ will be in the neighborhood of $\mathbf{d}$. This is an important property to be respected when using the SA, since it may not be desirable to get $\mathbf{d}_t$ too far from $\mathbf{d}$ when applying a perturbation. Allowing movements to states too far from the current solution would lead to a non systematic exploration of the solution space.

3.3 The Tabu List

As discussed previously, one concept from TS used in MSATS is that of short-term memory (Gil et al., 2002). Short-term memory can be included in the search algorithm by the use of a list of prohibited moves (in this case, perturbations), which can be defined by

$$\mathbf{T} = \begin{bmatrix} \mathbf{M}_1 \\ \vdots \\ \mathbf{M}_N \end{bmatrix}, \tag{9}$$

where $\mathbf{T}$ is the Tabu List, $\mathbf{M}$ are the vectors representing the prohibited moves and N is the number of moves to be stored in $\mathbf{T}$ (the memory size).

When a perturbation is applied to the design vector $\mathbf{d}$, the difference between $\mathbf{d}_t$ and the $\mathbf{d}$ defines a move, which can be evaluated by

$$\mathbf{M}_t = \mathbf{d}_t - \mathbf{d}, \tag{10}$$

where the subscript t indicates that the move is related to the perturbed vector $\mathbf{d}_t$.

A new design vector $\mathbf{d}_t$ will then be accepted only if its corresponding move $\mathbf{M}_t$ does not belong to the Tabu List $\mathbf{T}$. Note that the acceptance rule from the SA must still be checked, as described previously.

The construction of the Tabu List $\mathbf{T}$ proceeds as follows. If the new design vector $\mathbf{d}_t$ is accepted according to the criterion of the short-term memory (the move $\mathbf{M}_t$ does not belongs to $\mathbf{T}$), the algorithm proceeds to the check in the acceptance rule from the SA. If $\mathbf{d}_t$ also pass this last check, it is then taken as the new current design vector $\mathbf{d}$ of the problem. However, the move $-\mathbf{M}_t$ is added to the top of the Tabu List $\mathbf{T}$ only if this move improves the solution of the problem. Note that the reverse move $-\mathbf{M}_t$ is added to $\mathbf{T}$ (not $\mathbf{M}_t$), which means the algorithm will be prohibited to move back for some time. Moving in the same direction will still be allowed in the subsequent steps. When the number of tabu moves stored in $\mathbf{T}$ becomes bigger than N, the oldest entries of $\mathbf{T}$ are deleted. Consequently, the reverse of moves taken N steps before (or more) will be again allowed, and, if necessary, the algorithm will be allowed to move back.

3.4 The analysis of the pipe networks

Since the analysis of steady state flows in pipe networks leads to a system of nonlinear equations, solving this problem is many times demanding. Among the several methods currently used for solving this problem, the Gradient Method (Todini and Pilati, 1987) distinguishes itself since it is implemented in most analysis packages. However, in order to allow the development of a more efficient optimization routine, the authors decided to use an alternative method for the analysis of the pipe networks. This method is described by Kutas and Čiupailaitë (1997) and is sometimes called Finite Element Approach for the analysis of pipe networks, but the authors prefer to call it Fixed Point Method because of its clear resemblance to fixed point methods commonly used in literature. The Fixed Point Method is used in this work since according to the authors it is easier to translate into computational routines, gives satisfactory results and can be easily incorporated into an optimization routine.

3.5 General algorithm

The general algorithm for the design of pipe networks using the MSATS as here described is defined as follows:

1. Start the procedure with an initial solution $\mathbf{d}_1$, an initial value for the objective function F_1, an initial temperature T_1, a size for the tabu list N, a constant α for the decrease of the temperature, and a counter $s = 0$;
2. Update the temperature stages counter to $s = s + 1$;
3. Do while the system does not reach thermodynamic equilibrium ($i = 1,2,3,...$):

a. Apply a perturbation to the current solution $\mathbf{d}_i$ giving a new trial solution $\mathbf{d}_t$
b. Check if the move $\mathbf{M} = \mathbf{d}_t - \mathbf{d}_i$ is a prohibited move currently in $\mathbf{T}$;
c. Evaluate the objective function F_t of the trial solution $\mathbf{d}_t$ by Eq (6), and evaluate $\Delta F = F_t - F_i$;
d. If $\mathbf{M}$ is currently in $\mathbf{T}$, accept $\mathbf{d}_t$ only if F_t is better than the best solution obtained so far. Else, accept it or not according to the acceptance rule from Eq. (1);
e. If the move is accepted and improves the current solution, put the move $-\mathbf{M}_t$ in the top of the Tabu List $\mathbf{T}$;
f. If the length of the list $\mathbf{T}$ is bigger than N, delete the bottom entries until the size of $\mathbf{T}$ reaches N;
g. Store the best solution obtained so far;

4. Update the temperature according to Eq. (2);
5. If the solutions stop to change or the maximum number of temperature stages to be used is achieved stop the algorithm.

4. NUMERICAL EXAMPLES

The example from Fig. 1 is used for comparisons in this section. All pipes have a length of 50m, except the one from node 8 to node 16, which has a length of 100m. Node 1 has a prescribed pressure of 10m, while all other nodes have a demand of $5.10^{-3}m^3/s$ (5 L/s). The minimum allowable pressure in each node is of 5m. The available pipe diameters are 0.001m, 0.1m, 0.2m and 0.3m with costs per length respectively of 0, 5, 10 and 15 monetary units. The fluid is water at 15°C. The cost scale factor is $1.8182x10^{-4}$, the constraints scale factor is 50, the size of the Tabu List is 21 and the initial temperature is 0.5. The other parameters necessary are defined according to the suggestions given by Dréo et al. (2006).

Note that in this example, the pipe with diameter of 0.001m simulates a pipe with diameter equal to 0m, which is the same as not using a pipe in that position. Since defining a pipe with diameter equal to 0m leads to numerical difficulties during the analysis of the networks, this value is replaced by 0.001. Consequently, the cost per length of this diameter is taken as zero. Besides, in Figs. 2, 3, 4 5 and 8 the mean values and standard deviations are evaluated for 20 runs of the algorithm.

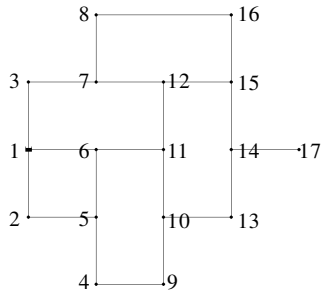


Figure 1. Pipe network with its nodes numbered.

The first comparison is made for the MSATS considering and not considering aspiration conditions. The results of both approaches are shown in Fig. 2, from where it can be seen that considering aspiration conditions leads, for this case, to better results. This can be explained by the fact that when not using aspiration conditions, the Tabu List may prevent the algorithm from moving to a better solution than all those previously visited, and the algorithm may skip a promising solution of the problem. Besides, aspirate conditions can be easily incorporated in the MSATS and do not lead to increase in computational efforts.

The second comparison is on the MSATS including the reverse of every move taken in the Tabu List; and including only the reverse of moves which improves the solution in the Tabu List. These results are shown in Fig. 3, and it is important to note that the second approach is the one proposed in this work, and in the work of Gil et al. (2002). It is interesting to note how a simple modification in the way the Tabu List is assembled may lead to significant changes in the performance of the MSATS. In this case, assembling the Tabu List by including the reverse of every move taken clearly leads to a degenerate algorithm. Consequently, it can be concluded that the MSATS presents a significant dependence on how the Tabu List is assembled.

The third comparison is made between the SA and the MSATS as here proposed. Results for the mean values are shown in Fig. 4 and standard deviations are shown in Fig. 5. From Fig. 4 it can be seen that the MSTAS presented better results, mainly when few function evaluations were used. The convergence of the results for a higher number of function evaluations is expected, since both methods have the ability to find good solutions at the long range. Consequently, when allowing a high number of function evaluations, both methods will converge to the similar solutions.

The standard deviation of both algorithms behaved as expected, as show in Fig. 5. For a higher number of objective function evaluations this parameter is expected to decrease, since the algorithms start to obtain the same solution over and over again. Consequently the standard deviation decreases. Besides, the standard deviations for both the SA and the MSATS are very similar.

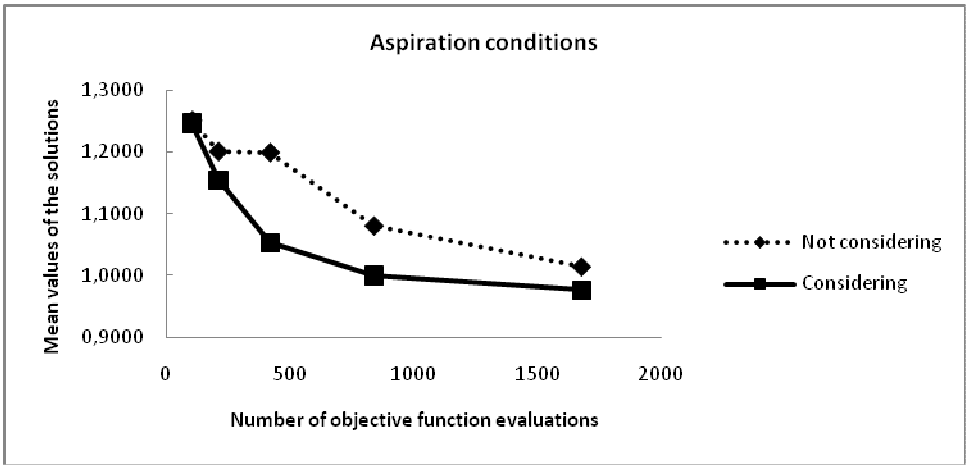


Figure 2. Comparison of the results given by the MSTAS considering and not considering aspiration conditions, for the case of Fig. 1.

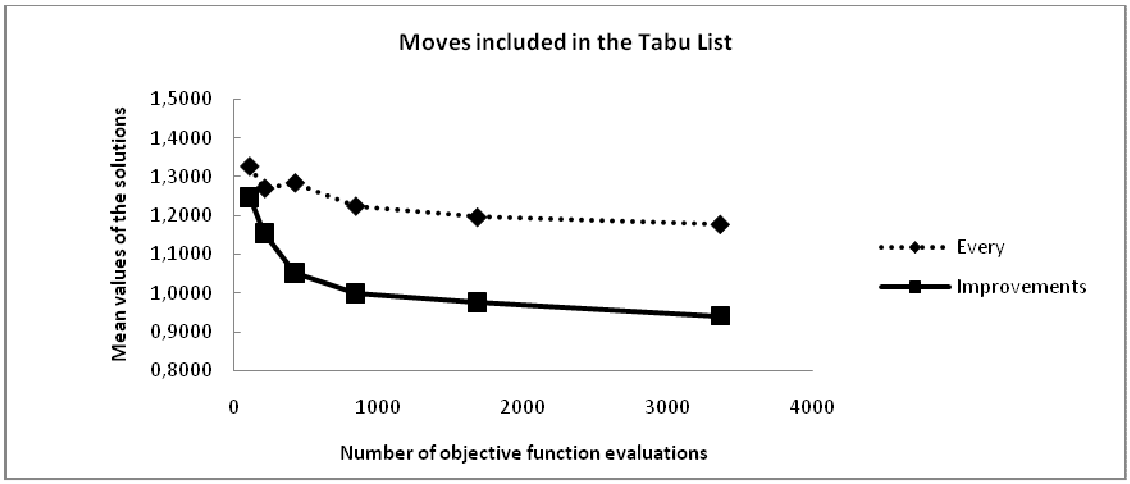


Figure 3. Comparison of the results given by the MSTAS including the reverse of every move in the Tabu List, and including only the reverse of moves which improve the solution, for the case of Fig. 1.

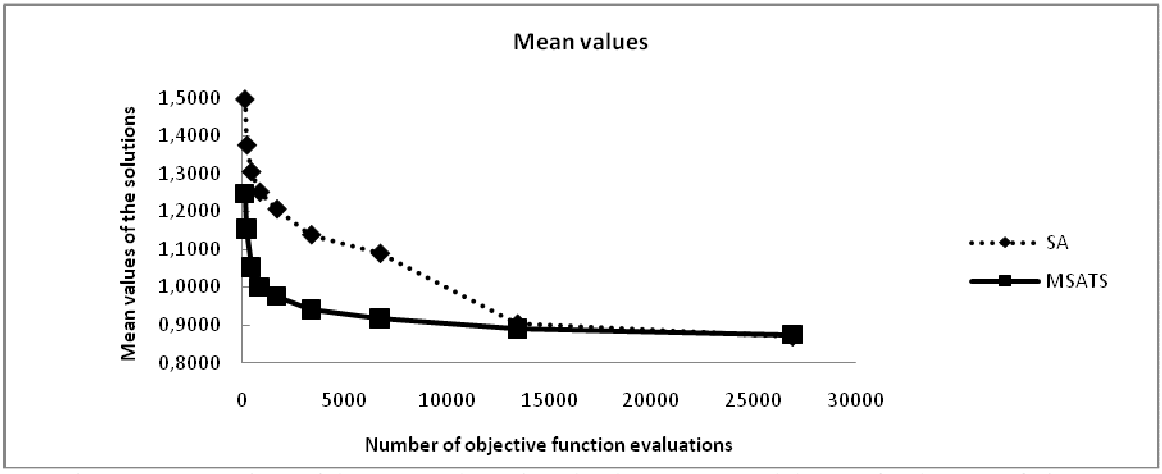


Figure 4. Comparison of the mean values given by the MSTAS and the SA, for the case of Fig. 1.

The second example is that from Fig. 7. All pipes have a length of 50m. Node 1 has a prescribed pressure of 10m and node 9 has a demand of $10.10^{-3}m^3/s$ (10 L/s). The minimum allowable pressure in node 9 is of 5m, while there is no

minimum pressure for the other nodes. The available pipe diameters are 0.001m, 0.1m, 0.2m and 0.3m with costs per length respectively of 0, 5, 10 and 15 monetary units. The fluid is water at 15°C. The cost scale factor is 3.33×10^{-4}, the constraints scale factor is 50, the size of the Tabu List is 12 and the initial temperature is 0.5. The other parameters necessary are taken equal to the suggestions given by Dréo et al. (2006).

Figure 8 shows that in this case the difference between the MSATS and the SA is even more significant, being the results presented by the MSATS much better than the ones provided by the SA. This is an interesting example since the true minimum solution is known, and is composed of every combination of four pipes with diameter equal to 0.1m linking node 1 and node 9.

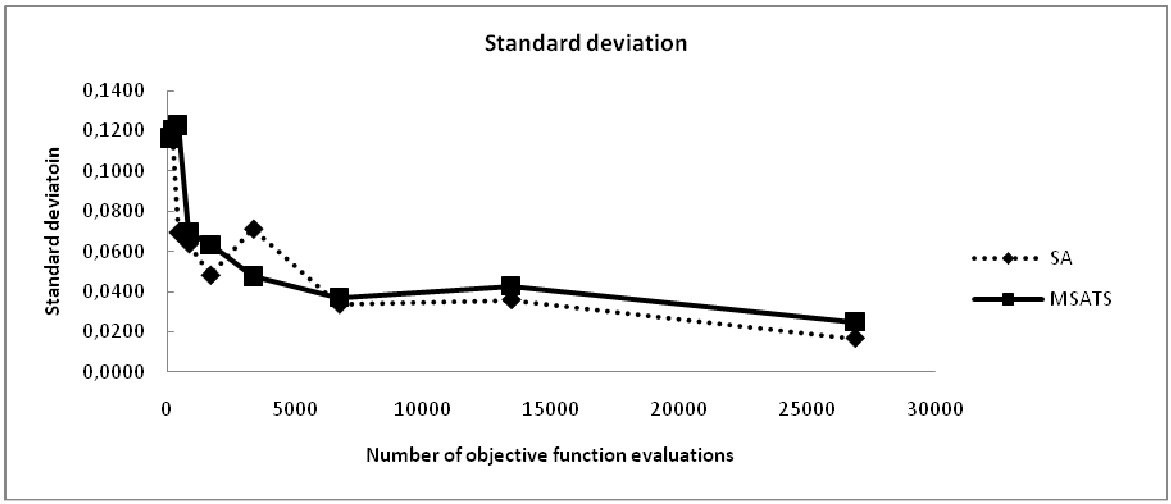
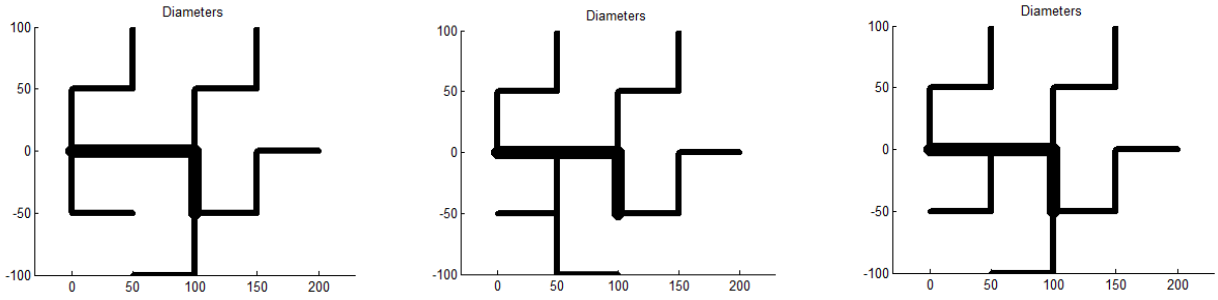


Figure 5. Comparison of the standard deviation given by the MSTAS and the SA, for the case of Fig. 1.



a) b) c)

Figure 6. Three pipe network designs obtained with the MSTAS technique, considering the case from Fig. 1. Bigger diameters are of 0.2m, while smaller diameters are of 0.1m. The value of the objective function for all three solutions is 0.8636.

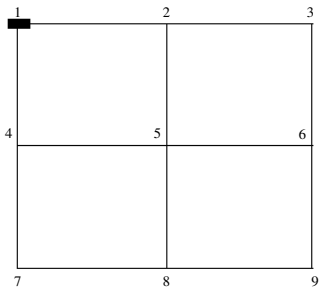


Figure 7. Pipe network with its nodes numbered.

5. CONCLUSIONS

For the cases here studied, the MSATS presented better results than the SA. The main reason for this improvement in performance may be explained by the fact that a Tabu List may help preventing the algorithm from moving in circles.

Besides, aspiration conditions can be easily implemented to the MSATS, allowing further improvement of the algorithm.

The MSATS appears to be much dependent on how the Tabu List is assembled. It was shown that a simple modification on this procedure may lead to very different results. In the case here studied, including the reverse of every move taken to the Tabu List appears to degenerate the algorithm. Since the management of the Tabu List appears to present significant effects to the MSATS, it is expected that improvements may be obtained by using more sophisticated concepts on short term memory. The use of a Tabu List with variable size, for example, may allow further improvement of the MSATS. These improvements may enhance the ability of the MSATS to carry an extensive search for a solution in a given region of the feasible domain, when this region presents promising solutions.

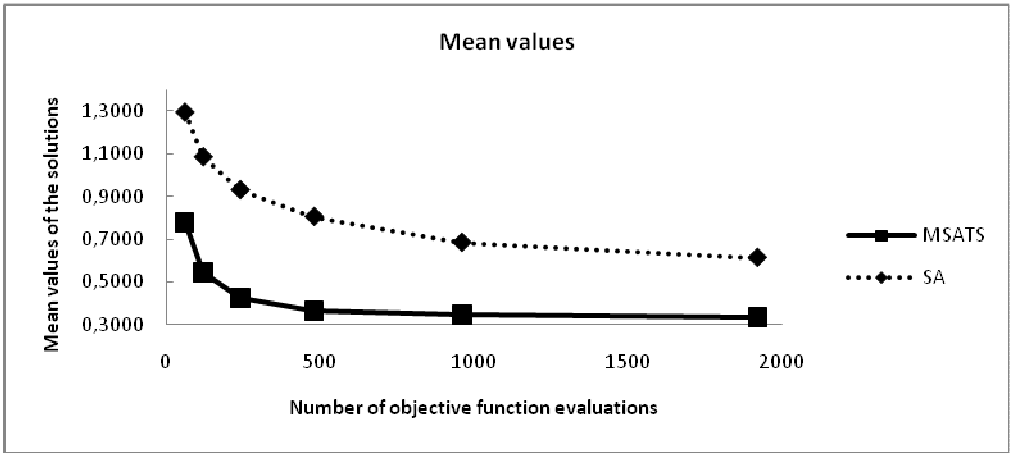


Figure 8. Comparison of the mean values given by the MSTAS and the SA, for the case of Fig. 7.

6. REFERENCES

Arora, J.S., 2004, "Introduction to optimum design", 2nd Edition, Elsevier, San Diego, 728p.

Dréo J., Pétrowski A., Siarry P., Taillard E., 2006, "Metaheuristics for Hard Optimization: Simulated Annealing, Tabu Search, Evolutionary and Genetic Algorithms, Ant Colonies,…", Springer, Berlin, 369 p.

Formiga, K.T.M, 2005, "Otimização multiobjetivo de projeto de redes de distribuição de água", PhD. Thesis, Escola de Engenharia de São Carlos, Universidade de São Paulo.

Formiga, K.T.M, Chaudhry, F.H., 2008, "Modelos de análise hidráulica de redes de distribuição de água considerando demanda dirigida pela pressão e vazamentos", Engenharia sanitária e ambiental, Vol. 13, No. 2, pp. 153-162.

Gil, C., Ortega, J., Montoya, M.G., Baños, R., 2002, "A mixed heuristic for circuit partitioning", Computational Optimization and Applications, Vol.23, pp.321–340.

Goldberg, D.E., ASCE, M., Kuo, C.H., 1987, "Genetic algorithms in pipeline optimization", Journal of Computing in Civil Engineering, Vol. 1, No. 2, pp. 128-141.

Gupta, I., Gupta, A., Khanna, P., 1999, "Genetic algorithm for optimization of water distribution systems", Enviromental Modelling and Software, Vol. 14, pp. 437-447.

Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. "Optimization by simulated annealing", Science, 220, 671-680.

Nocedal, J., Wright, S.J., 1999, "Numerical optimization", Springer-Verlag, New York, 634p.

Reca, J., Martínez, J., Gil, C., Baños, R., 2008, "Aplication of several meta-heuristic techniques to the optimization of real looped water distribution networks", Water Resources Management, Vol. 22, pp.1367-1379.

Savic, D.A., Walters, G.A., 1997, "Genetic Algorithms for least-cost design of water distribution networks", Journal of Water Resources Planning and Management, Vol. 123, No. 2, pp. 67-77.

Todini, E., Pilati, S, 1987, "A gradient method for the analysis of pipe networks", International Conference on Computer Applications for Water Supply and Distribution, Leicester Polytechnic, UK.

Todini, E., 2003, "A more realistic approach to the "extended period simulation" of water distribution networks", In: C. Maksimovic, D. Butler, et al (Ed.), Advances in Water Supply Management. Lisse A. A. Balkema Publishers, p.732.

Walski, T.M., 2001, "The Wrong Paradigm—Why Water Distribution Optimization Doesn't Work", Journal of Water Resources Planning and Management, Vol. 127, No. 4, pp. 203-205.

7. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.