Ant Colony Optimization for Continuous Domains: Application to Reservoir Operation Problems

Conference Paper · October 2008

DOI: 10.1109/HIS.2008.121 · Source: IEEE Xplore

2 authors:

Abbas Afshar
Iran University of Science and Technology
201 PUBLICATIONS **4,894** CITATIONS
SEE PROFILE

Shahrbanou Madadgar
University of California, Irvine
21 PUBLICATIONS **1,044** CITATIONS
SEE PROFILE

Eighth International Conference on Hybrid Intelligent Systems

Ant Colony Optimization for Continuous Domains: Application to Reservoir Operation Problems

A. Afshar[1], S. Madadgar[2]

[1] *Professor, Department of Civil Engineering and Center of Excellence for Fundamental Studies in Structural Mechanics, Iran University of Science and Technology, Tehran, Iran*
a_afshar@iust.ac.ir

[2] *Research Assistant, Hydroinformatic Research Center, Department of Civil Engineering , Iran University of Science and Technology, Tehran, Iran*
sh_madadgar@civileng.iust.ac.ir

Abstract

Recent attempts for direct extension of ant algorithms to continuous decision space led to a new approach called ACO_R. The authors of proposed method tested their algorithm on some well-known benchmark problems presented in the literature and concluded the strong performance of ACO_R in comparison with other ant and non-ant based methods. In this paper, we briefly review the central idea and mathematical representation of ACO_R and then present the method's application to a real-world reservoir operation optimization problem. This is the first application of ACO_R algorithm to water resource problems. The comparison of results obtained by present method with which obtained by other ant-based algorithms used by different researchers, emphasizes on robust performance of ACO_R in the reservoir operation optimization problem as locating the closest, among other ant methods, to the global optimal solution.

1. Introduction

Evolutionary and meta-heuristic algorithms have been extensively applied to reservoir operation optimization problems during last decade ([6], [7], [8]). Labadie (2004) presented a state-of-the-art review of the optimal operation of multi-reservoir systems with mathematical and heuristic optimization algorithms such as genetic algorithms, artificial neural networks, and fuzzy-based approach [4]. Bozorg Haddad and Afshar (2004) applied Honey Bees Mating Optimization $(HBMO)$ meta-heuristic algorithm to a 60-period of operating a reservoir [5].

Ant Colony Optimization (ACO) algorithm which is inspired from real-ant colonies' foraging behavior, was first introduced in the early 90's to solve combinatorial optimization problems $(COPs)$ such as the traveling salesman and the quadratic assignment problems ([9], [10]). $ACOs$ were initially proposed for discrete search spaces while in continuous domains, discretization of the search space has provided the problem to be solved by $ACOs$.

Early attempts for direct extension of ant algorithms to continuous decision space led to algorithms such as Continuous ACO $(CACO)$ [11], API algorithm [13] and Continuous Interacting Ant Colony $(CIAC)$ [12]. Conceptually, however, all these approaches are far from the original spirit of ant colony optimization. The latest approach which at the same time benefits from the original concepts of ACO, is a new algorithm called ACO_R proposed by Socha and Dorigo (2006) [1]. They tested the algorithm with standard benchmark problems and concluded the better performance of ACO_R when comparing with other ant-related algorithms in continuous domains. This paper presents the first application of ACO_R algorithm to a reservoir operation optimization problem with 60 and 120 operating periods. It is shown that for present case study, ACO_R performs better than the other ant-based algorithms employed by different researchers.

2. Ant Colony Optimization: Original concepts

The inspiring source of ACO algorithms is the real-ants foraging behavior. Ants start searching for food by random exploring the area around the nest. After finding the food source, they return back to the nest depositing some chemical liquid called pheromone on the ground the amount of which may depend on the quantity and quality of the food. Other ants prefer to search for food source following the trails with more amount of pheromone. Iterating the process finally leads to find the shortest path between the nest and the food source. Inspiration of this behavior of real ant colonies forms the initial definition of artificial ant colonies to find approximate solutions of hard combinatorial optimization problems.

At each construction step in any ACO algorithms, the solution components are selected through a probability rule, which in the original Ant System (AS) is defined as follows [10]:

$$p\left(c_{ij} \mid s^p, t\right) = \frac{\left[\tau_{ij}(t)\right]^{\alpha} . \left[\eta(c_{ij})\right]^{\beta}}{\sum_{j=i}^{J} \left[\tau_{ij}(t)\right]^{\alpha} . \left[\eta(c_{ij})\right]^{\beta}} \quad (1)$$

$$, \forall c_{ij} \in allowable\, set$$

In which $p\left(c_{ij} \middle| s^p, t\right)$ is the probability of choosing the solution component c_{ij} as the current partial solution s^p at iteration t; $\tau_{ij}(t)$ is the pheromone value associated with component c_{ij} at iteration t; $\eta(.)$ assigns the heuristic value representing the cost of choosing the solution component c_{ij}; α and β are two parameters indicating the relative importance of the pheromone trail and heuristic value; and i is the current construction step including J component solutions in the allowable set.

Once all ants have constructed their solutions, the pheromone values are updated by increasing the pheromone levels associated with chosen good solution, s_{ch}, and decreasing all the pheromone values through pheromone evaporation.

$$\tau_{ij}(t+1) = \begin{cases} (1-\rho)\tau_{ij}(t) + \rho\,\Delta\tau & if\ \tau_{ij}(t) \in s_{ch} \\ (1-\rho)\tau_{ij}(t) & Otherwise \end{cases} \quad (2)$$

Where $\Delta\tau$ is a certain value adding to pheromone levels associated with s_{ch} and $0 < \rho \leq 1$ is the coefficient of evaporation. Pheromone evaporation is in favor of decreasing the chance of pre-mature convergence of the algorithm to sub-optimal solutions. Different ACO algorithms differ in the aspect of using either the iteration-best solution or the global-best solution to accomplish pheromone updating.

3. Ant Colony Algorithms for Continuous Optimization

As mentioned earlier, the most recent ant-based method to continuous optimization problems is ACO_R algorithm [1]. This section follows the mathematical presentation of ACO_R, the authors of which claim the capability of proposed algorithm to tackle both continuous and mixed-variable optimization problems.

In discrete ACO algorithms, ants make a probabilistic choice to add c_{ij} from the allowable set of solution components to the current partial solution s^p, according to Eq. (1). This means sampling a discrete probability distribution. The central idea underying ACO_R is to sample a continuous probability distribution, that is, a probability density function (pdf) instead of the discrete one.

A Gaussian kernel pdf has been suggested to model the multi-promising area of search space. Gaussian kernel pdf prepares a more flexible sampling shape compared to a single Gaussian function (Fig. 1). It has been defined as weighted sum of several one-dimensional Gaussian functions $g_l^i(x)$ as $G^i(x)$ [1]:

$$G^i(x) = \sum_{l=1}^{k} \omega_l\, g_l^i(x) = \sum_{l=1}^{k} \omega_l \frac{1}{\sigma_l^i \sqrt{2\pi}} e^{-\frac{\left(x-\mu_l^i\right)^2}{2\sigma_l^{i\,2}}} \quad (3)$$

Where k is the number of single pdfs included in Gaussian kernel pdf at i th construction step; ω, μ^i and σ^i are the vectors of size k and respectively defines the weights, means and standard deviations associated with the individual Gaussian functions at i th construction step.

ACO_R stores a certain number (k) of the solutions in a solution archive T. In an n-dimensional problem, the archive keeps the values of n variables associated with any selected solution s_l. Fig. 2 presents the structure of the solution archive. s_l^i denotes the i th component of the l th solution in the archive. Pheromone is updated by adding the set of new superior solutions to the solution archive T and in contrary, removing the same number of inferior solutions in order to keep the total size of the archive unchanged. Therefore, only the best solutions are stored in the archive and will guide the ants in the search process. The solutions in the archive are used to determine the vectors ω , μ^i , and σ^i which describe the final shape of the Gaussian kernel pdf.

At each construction step $i = 1,...,n$, for any Gaussian kernel pdf G^i , the vector μ^i is defined as:

$$\mu_l^i = s_l^i \tag{4}$$

Where the vector μ^i is made of the values of the i th variable of current archived solutions.

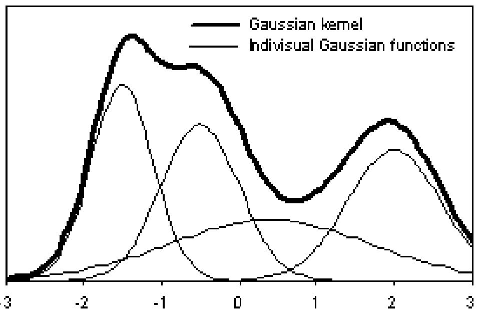

Fig. 1. Example of a Gaussian kernel pdf containing 4 Gaussian functions (illustration is limited to the range $x \in [-3,3]$)

s_1	s_1^1	s_1^2	$\ldots$	s_1^i	$\ldots$	s_1^n	ω_1
	$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$
s_l	s_l^1	s_l^2	$\ldots$	s_l^i	$\ldots$	s_l^n	ω_l
	$\vdots$			$\vdots$		$\vdots$	$\vdots$
s_k	s_k^1	s_k^2	$\ldots$	s_k^i	$\ldots$	s_k^n	ω_k
	G^1	G^2		G^i		G^n	

Fig. 2. The schematic of solution archive used in ACO_R .

After a complete iteration, all the solutions (either the solutions in the archive or the current constructed solutions) are ranked according to their fitness value, then superior k solutions are used to fill up the archive and the remained solutions will be omitted (i.e., solution s_l has rank l). The weight ω_l of the solution s_l , and consequent single pdf, is calculated by a Gaussian function with argument l, mean 1.0, and standard deviation qk , as [1]:

$$\omega_l = \frac{1}{qk\sqrt{2\pi}} e^{-\frac{(l-1)^2}{2q^2k^2}} \tag{5}$$

$q\ (>0)$ is a tuning parameter which effects on the convergence rate of the algorithm. When it is small, ants prefer searching the area around the best-ranked solutions and a pre-mature convergence may happen. But when it is large, the probability becomes more uniform and the algorithm samples the search space based on a larger number of reasonably good solutions. In other words, the search is more diversified and consequently the final optimal solution is more reliable. Unfortunately, this higher robustness is a result of slower convergence speed which usually means lower efficiency.

In the following, the procedure of sampling the Gaussian kernel pdf as defined in Equation 3 is described. Before an ant starts a solution construction, it chooses one solution in the archive and then uses the Gaussian functions associated with the chosen solution for all n construction steps. This allows exploiting the (possibly existing) correlation between the decision variables [1].

The choice of the ranked solution l (and consequently Gaussian functions l) is done probabilistically as:

$$p_l = \frac{\omega_l}{\sum_{j=1}^{k} \omega_l}, \forall l = 1,...,k \tag{6}$$

ω_l is the weight of Gaussian function l as defined in Equation 5.

In the case of finding the final shape of each Gaussian function at any construction step, computation of the standard deviation vector σ^i is the most complex issue.

It should be noticed that at each construction step different Gaussian functions, but all have the rank of l , will be sampled. For step i , $\mu_l^i = s_l^i$, and σ_l^i is calculated as [1]:

$$\sigma_l^i = \xi \sum_{e=1}^{k} \frac{\left|s_e^i - s_l^i\right|}{k-1} \tag{7}$$

which means, at each step i , the average distance from the chosen solution s_l to other solutions in the archive is determined and multiplied by the parameter ξ . The parameter $\xi > 0$, behaves similar to the coefficient of pheromone evaporation in ACO . Appropriate selection of ξ will cause the algorithm to be less biased towards the areas that have been already explored and stored in the archive. The lower values of ξ, will speed up the convergence of the algorithm.

In the first iteration, specifically in the problems with no prior information, the kernel pdfs may be initialized by selecting a more or less uniform distribution over the search domain (a,b). A set of k normal distributions $G^i(x)$ with uniformly distributed means are described as:

$$G^i(x) = \sum_{l=1}^{k} \frac{1}{k} g_l^i\left(x, a + (2l-1)\frac{b-a}{2k}, \frac{b-a}{2k} \right) \qquad (8)$$

In which ω_j is equal to $\frac{1}{k}$ and the second and third terms define the mean and standard deviation of i th pdf, respectively. At the end of the first iteration, each solution is evaluated and the first k solutions are transferred to the archive T.

4. Applications to the reservoir operation optimization problem

This section presents the initial application of ACO_R algorithm [1] to water resources problem, the reservoir optimization problem with 60 and 120 operating periods. In order to compare the results, the global optimum of the problem obtained using a powerful nonlinear programming (NLP) solver Lingo-9 (Lindo Systems Inc., 2004) with multi-start and global solver options and extensive CPU time.

The problem is to find the optimal operation of a surface reservoir in south of Iran, Dez, with minimum and maximum storage capacities of 830 MCM and 3340 MCM, respectively. The range of monthly release volume is $[0,1000]$ in MCM. Demand volumes are constant from year to year. The objective function is to minimize the sum of squared deviation from target demands over all operating periods and it is expressed as:

$$O.F. = Min \sum_{t=1}^{T} \left(\frac{D_t - R_t}{Max(D)} \right)^2 \qquad (9)$$

Where D_t and R_t are the water demand and reservoir release at time period t, respectively; $Max(D)$ is the maximum demand in entire operating periods; and T is the total number of time periods.

To adapt the problem with ACO algorithm, each operating period is considered as a solution construction step with reservoir releases as decision variables. To construct the solutions, all ants consider a specified initial storage for the reservoir and calculate the reservoir storages through continuity equations described as:

$$S_{t+1} = S_t + I_t - R_t \qquad (10)$$

Where S_t and I_t are the reservoir storage and reservoir inflow in time period t, respectively. It is assumed that no loss occurs via evaporation or seepage.

The constraints on reservoir releases and storages are:

$$\begin{aligned} S_{\min} &< S_t < S_{\max} \\ R_{\min} &< R_t < R_{\max} \end{aligned} \qquad (11)$$

Which define the allowable range of the reservoir storage and the reservoir release at time period t, respectively. If the value of S_t calculated from Equation 10 does not satisfy the storage constraint, the associated ant will be penalized by the penalty function expressed as:

$$Penalty_t = \begin{cases} C \times \left(1 - \frac{S_t}{S_{\min}}\right) & if\ S_t < S_{\min} \\ C \times \left(\frac{S_t}{S_{\max}} - 1\right) & if\ S_t > S_{\max} \end{cases} \qquad (12)$$

In which, $Penalty_t$ is the penalty imposed on the solution for violating storage constraint at time t; and $C(>1)$ is the penalty factor which describes the severity of storage constraint violation. The penalty factor $C = 100$ is used in this case. The total penalty value for an ant is the sum of all imposed penalties at any time periods.

The reservoir optimization problem was solved over 60 and 120 operating periods. Table1 presents the summary of the parameters used by ACO_R for both 60 and 120 operating periods.

The results of 10 independent runs are presented in tables 2 and 4. Reported are the statistical parameters of the objective value. The algorithm stopped after specific number of iteration.

Table 1. Summary of the parameters used in ACO_R for reservoir operation problem

k	20
q	0.3
ξ	0.8

Table 2: Results obtained by ACO_R after 1000 iterations for the problem of 60 operating periods

	Number of ants		
	20	50	100
The best	0.8112	0.7589	0.7433
Mean	0.9178	0.7763	0.7543
The worst	1.1813	0.8189	0.7919
Stan. Dev.	0.120	0.019	0.014

A nonlinear programming (NLP) solver Lingo-9 (Lindo Systems Inc., 2004) found the global optimums and reported 0.731 and 1.98 for respectively 60 and 120 operating periods. For 60-

period optimization problem, the best solution obtained by ACO_R is 0.7433 which is approximated as %98.3 of the known global best solution. It was found with 100 agents within 1000 iterations that means 100,000 number of function evaluations. Afshar et al. (2006) found the objective value of 0.7418 for the best solution (%98.5 of the global best) using Elitist $CACO$ algorithm with 150 ants within 2000 iterations [3]. It means 300,000 number of function evaluations (i.e. 3 times greater than which ACO_R achieved). Jalali et al. (2006) employed ACS_{gb} algorithm and reported the value of 0.949 as the best found solution (%77 of the global best) [2]. The result was for the performance of 150 ants within 500 iterations which equals to 75,000 number of function evaluations. To avoid any doubt about the robustness of ACO_R, the performance of the algorithm was also tested within 2000 iterations and the results of 10 independent runs are presented in table 3. The best solution with 100 agents within 2000 iterations (i.e. 200,000 number of function evaluations) found the objective value of 0.7373 (%99.1 of the global best). The number of function evaluations was until 2/3 times less than what Elitist $CACO$ achieved. The results emphasize on clear better performance of ACO_R in comparison with Elitist $CACO$ or ACS_{gb} algorithms.

Figures 3 and 4 present the rate of convergence of the best runs for the problems of 60 and 120 operating periods, respectively.

Table 3: Results obtained by ACO_R after 2000 iterations for the problem of 60 operating periods

	Number of ants		
	20	50	100
The best	0.7623	0.7467	0.7373
Mean	0.8083	0.7579	0.7443
The worst	0.9155	0.7986	0.7605
Stan. Dev.	0.043	0.016	0.007

Table 4: Results obtained by ACO_R after 2000 iterations for the problem of 120 operating periods

	Number of ants		
	40	100	200
The best	3.1839	2.3747	2.2142
Mean	3.5925	2.6228	2.4165
The worst	4.4336	2.9959	2.6845
Stan. Dev.	0.359	0.245	0.172

The best solution obtained by ACO_R for 120-period optimization problem is 2.214, which is approximated as %89.4 of the known global best solution. As the problem is high dimensional, greater number of function evaluations may improve the results.

When comparing the result of the best solution obtained by ACO_R with those of global optimal solution, the satisfactory performance of ACO_R is proven (Fig. 5 and 6). As shown, the storage values at different time periods followed almost exactly the associated values of global optimum.

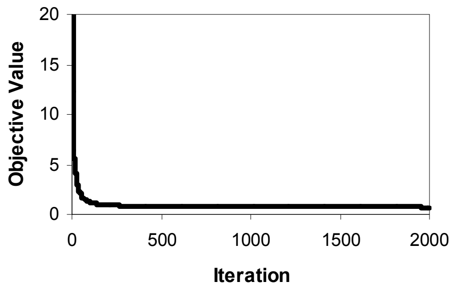

Fig. 3. Objective value evaluation of the 60-period problem within 2000 iterations for the best of 10 runs.

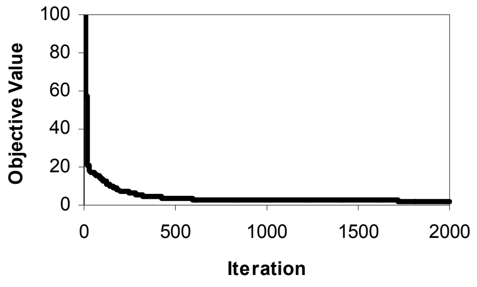

Fig. 4. Objective value evaluation of the 120-period problem for the best of 10 runs.

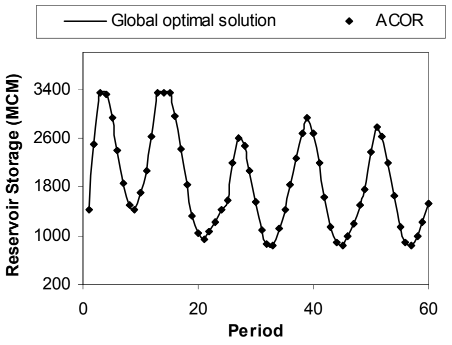

Fig. 5. Comparison between the best result of ACO_R within 2000 iterations and those of global optimum (60-period problem).

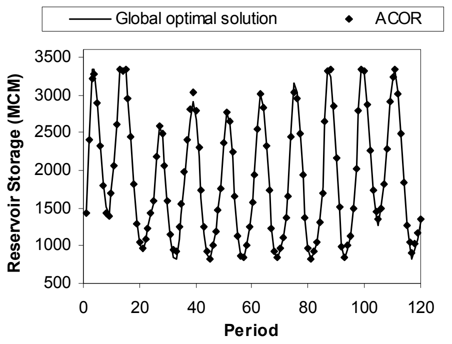

Fig. 6. Comparison between the best result of ACO_R within 2000 iterations and those of global optimum (120-period problem).

5. Conclusion

We have presented the recent proposed ant colony algorithm for continuous domain, ACO_R, to the water resource problems. The performance of the algorithm had been tested on some well-known test functions by own authors, previously. This paper has applied ACO_R to a real-world reservoir operation optimization problem and compared the obtained results to those reported by previous researchers using other ant-related algorithms. It has been observed that ACO_R has firmly outperformed other ant colony algorithms presented in the literature for the water resource problem under consideration. It can be concluded that ACO_R algorithm might be a very useful method for optimization problems in civil engineering, especially in water resource management.

References:

[1] K. Socha, and M. Dorigo, "Ant Colony Optimization for Continuous Domains", *European Journal of Operational Research*, Vol. 185, No. 3, 2006, pp. 1155-1173.

[2] M.R. Jalali, A. Afshar, and M. A. Marino, "Reservoir Operation by Ant Colony Optimization Algorithms", *Iran Journal of Science and Technology,* 30(B1), 2006, pp. 107-117.

[3] M. H. Afshar, H. Ketabchi, and E. Rasa, "Elitist Continuous Ant Colony Optimization Algorithm: Application to Reservoir Operation Problems", *International Journal of Civil Engineering*, Vol. 4, No. 4, 2006, pp. 274-285.

[4] J.W. Labadie, "Optimal Operation of Multireservoir Systems: State-of-the-art Review", *Journal of Water Resource Planning and Management*, ASCE, Vol. 130, No. 2, 2004, pp. 93-111.

[5] O. Bozorg Haddad, and A. Afshar, "MBO (Marriage Bees Optimization), A New Heuristic Approach in Hydrosystems Design and Operation", *In: Proceedings of the 1st International Conference on Managing Rivers in the 21st Century: Issues and Challenges*, Penang, Malaysia, 2004, pp. 499-504.

[6] V. Esat, and M.J. Hall, "Water Resources System Optimization Using Genetic Algorithms", *In: Hydroinformatics'94, Proceedings of the 1st International Conference on Hydroinformatics*, Balkema, Rotterdam, the Netherlands, 1994, pp. 225–231.

[7] R. Oliveira, and D. Loucks, "Operating Rules for Multireservoir Systems", *Water Resource Research*, Vol. 33, No. 4, 1997, pp. 839-852.

[8] R. Wardlaw, and M. Sharif, "Evaluation of Genetic Algorithms for Optimal Reservoir System Operation", *Journal of Water Resource Planning and Management*, ASCE, Vol. 125, No. 1, 1999, pp. 25-33.

[9] M. Dorigo, "Optimization, Learning and Natural Algorithms", *PhD thesis*, Politecnico di Milano, Milan, Italy, 1992.

[10] M. Dorigo, V. Maniezzo, and A. Colorni, "The Ant System: Optimization by A Colony of Cooperating Ants", *IEEE Transaction on Systems*, Man and Cybernetics, Vol. 26, 1996, pp. 29-42.

[11] G. Bilchev, and I.C. Parmee, "The Ant Colony Metaphor for Searching Continuous Design Spaces", *In: Fogarty, T.C. (Ed.), Proceedings of the AISB Workshop on Evolutionary Computation*, Vol. 993 of LNCS. Springer, Berlin Heidelberg New York, 1995, pp. 25–39.

[12] J. Dreo, and P. Siarry, "A New Ant Colony Algorithm Using the Hierarchical Concept Aimed at Optimization of Multiminima Continuous Functions", *In: M. Dorigo, G.D. Caro, M. Sampels (Eds.), Proceedings of the 3rd international Workshop on Ant Algorithms (ANTS 2002)*, Vol. 2463, of LNCS. Springer, Berlin Heidelberg New York, 2002, pp. 216–221.

[13] N. Monmarche, G. Venturini, and M. Slimane, "On How Pachycondyla Apicalis Ants Suggest A New Search Algorithm", *Future Generation Computer Systems*, 16, 2000, pp. 937-946.