WATER RESOURCES RESEARCH, VOL. 33, NO. 4, PAGES 839–852, APRIL 1997

Operating rules for multireservoir systems

Rodrigo Oliveira
Departamento de Hidráulica, Laboratório Nacional de Engenharia Civil, Lisbon, Portugal

Daniel P. Loucks
School of Civil and Environmental Engineering, Cornell University, Ithaca, New York

Abstract. Multireservoir operating policies are usually defined by rules that specify either individual reservoir desired (target) storage volumes or desired (target) releases based on the time of year and the existing total storage volume in all reservoirs. This paper focuses on the use of genetic search algorithms to derive these multireservoir operating policies. The genetic algorithms use real-valued vectors containing information needed to define both system release and individual reservoir storage volume targets as functions of total storage in each of multiple within-year periods. Elitism, arithmetic crossover, mutation, and "en bloc" replacement are used in the algorithms to generate successive sets of possible operating policies. Each policy is then evaluated using simulation to compute a performance index for a given flow series. The better performing policies are then used as a basis for generating new sets of possible policies. The process of improved policy generation and evaluation is repeated until no further improvement in performance is obtained. The proposed algorithm is applied to example reservoir systems used for water supply and hydropower.

Introduction

The coordinated operation of multiple-reservoir systems is typically a complex decision-making process involving many variables, many objectives, and considerable risk and uncertainty. System operators are challenged to meet often conflicting objectives while complying with all legal contracts, agreements, and traditions affecting water allocations and use.

Multiple-reservoir system operation has been quite extensively studied, often with the aid of various optimization as well as simulation models. Clearly, the use of optimization models in identifying policies for the real-time operation of reservoir systems can be quite beneficial [see, e.g., *Crawley and Dandy*, 1993; *Georgakakos*, 1989; *Martin*, 1995; *McLaughlin and Velasco*, 1990; *Mizyed et al.*, 1992; *Yang et al.*, 1995]. For real-time management, a multiperiod optimization model (containing the current system state and the expected current and future inflows and system objectives) is run each time a decision needs to be made to identify or suggest the release decisions now and over a number of future periods that maximize the expected future system performance (however measured) subject to operating constraints. Although these models identify the release or storage decisions for a number of periods into the future, of primary interest are the immediate decisions that must be made at each reservoir in the current period. Once these current decisions are made and the current inflows are observed, the model's input data, including future inflow forecasts, operating objectives, and constraints, are then updated and the model is rerun to obtain the suggested release and storage decisions for the next decision period and similarly for each subsequent decision period.

Despite the potential for the use of optimization in real-time multiple-reservoir operation, with some exceptions, optimization models still play a minor role in identifying possible reservoir releases. In our view, this is more due to institutional limitations than to technological or mathematical ones. Until those who are responsible for establishing policies, policies that are now normally published on paper for all to see, to debate, to negotiate, and to understand in advance of their implementation, can convince the public and the courts that such models can be used to meet more efficiently any specified operation objectives and constraints, it is unlikely real-time operation will benefit very much from the sequential use of adaptive optimization models. But such models can be used for planning to help identify and evaluate alternative policies for fixed or predefined goals or objectives.

The need for a comprehensive negotiations and subsequent agreements on how to operate a reservoir system seems to be a main reason why most reservoir systems are still managed based on fixed predefined rules. These predefined rules indicate with varying detail the actions to be taken by the system operators as a function of a relatively small number of variables, such as the time of the year, state of the system, and expected future hydrological conditions. These rules are usually presented in the form of graphs or tables [see, e.g., *Wurbs*, 1996; *Yeh*, 1985; *Loucks and Sigvaldason*, 1982]. They provide guidance to system operators. Operator judgment is still required. It is widely acknowledged that system operators often deviate from these rules to adapt to specific conditions, objectives, or constraints that may exist at various times.

Predefined operating rules are often evaluated using simulation models, but before they can be simulated, these rules must be defined. Optimization models can help define these predefined rules, rules that satisfy various constraints on system operation while minimizing future spills or maximizing energy production or minimizing expected future undesired deviations from various water release, storage volume and/or energy production targets.

Paper number 96WR03745.
0043-1397/97/96WR-03745$09.00

Defining effective predefined operating rules is a challenging task, especially those that apply to multiple reservoirs serving multiple purposes and objectives. Many operating rules in use have been derived from experience or from trial-and-error simulation studies. Most of these have become very efficient over time, but as demands and objectives change, or as new facilities such as hydropower ones are installed, existing policies may need to be modified. There is a need for improved ways of doing this in spite of considerable past research on multiple-reservoir operation.

In this paper an approach to identifying reservoir-operating rules on the basis of genetic search algorithms is proposed. We believe that the approach overcomes some of the limitations of many techniques based on more traditional mathematical programming (constrained optimization) models. At the same time, we admit there is still room for improvement.

Predefined Operating Rules

Operating policies for multireservoir systems must specify not only the total release from the system but also the amounts, if any, to be released from each reservoir. Such operating rules usually take into account the probability of spillage, the evaporation losses when and where significant, and the impacts on the various users of water flows and storage volumes. For multireservoir systems, policies that define the individual reservoir releases as a function of the existing total system storage volume as well as the individual reservoir storage volumes clearly define the actions to be taken at any time and for any state of the system. System release rules typically indicate the total release to be made from the reservoir system as a function of the water available in the system and time of the year. A comparison of the individual reservoir storage targets to the actual storage volumes in each reservoir identifies which reservoirs should release water and which should not release water to meet the total system release target. Having both system-wide release functions as well as individual reservoir storage volume target functions defines a multiple-reservoir operating policy that permits the coordinated operation of the entire system.

Release rule and storage balancing functions are used in many simulation models (e.g., the HEC-3 and HEC-5 models developed by the U.S. Corps of Engineers [*Hydrologic Engineering Center* (*HEC*), 1981, 1989] and others [*Loucks et al.*, 1995]). The methods presented here define the operating policy of multireservoir systems in terms of a system-wide release rule and individual reservoir storage volume target functions. The system-wide release rule specifies what to release from the system, and individual reservoir storage target (or balancing) functions identify the desired storage volumes in each reservoir, all as functions of total storage volume and time of year.

Other types of predefined rules include allocation functions for systems with more than one demand site per reservoir and pumpage/recharge rules for systems that rely on aquifers to satisfy water demands. Allocation functions establish a relationship between the amount of water released from a reservoir and the amount of water allocated to a specific use, while withdraw/recharge rules specify the withdraw and/or recharge pumping rates as functions of water table levels. The algorithm discussed in this paper also could be applied to identify these type of rules.

Various mathematical optimization models have been proposed to derive the properties of efficient fixed and predefined operating rules. These models include those based on various fields of mathematics, such as ordinary calculus, Markov processes, inventory theory, and dynamic programming. They are used to determine the properties an "optimal" operating policy should satisfy. These properties may include statements pertaining to the types and derivatives of the policy's independent variables [*Massé*, 1946; *Gessford and Karlin*, 1958; *Bather*, 1962; *Amir*, 1967; *Arunkmar and Yeh*, 1973; *Sobel*, 1975; *Sand*, 1984]. Most of these models are valid only for simplified system configurations, inflow distributions, objectives, and constraints. Nevertheless, these simplified results can often be used to (1) improve one's understanding of system operation, (2) help identify more efficient heuristic policies, or (3) reduce the computational effort required by reducing the number of reasonable alternatives that need to be simulated. Simulation modeling is essential for a more complete evaluation of any proposed operation policy.

Most optimization models yield numerical values of releases and storage volumes in each of the reservoirs over time resulting from a given a set of input data. These input data include the definition of the system configuration and storage capacities, a scenario of its inflows over time, in some cases its initial storage volumes, and the objectives and constraints that the operating policy must satisfy. From the resulting time series outputs of releases and storage volumes one attempts to estimate improved operating policies. While some modeling approaches make it easier than others, it is, in general, not always a trivial task to derive the operating rules, that is, the policy itself, from the set of numerical storage and release values generated by the models. Furthermore, many numerical optimization models of multiple-reservoir systems require quite large computer resources. Hence this continues to be an active area of research. Extensive reviews of past research in multireservoir operating policy optimization and simulation are available [see, e.g., *Yakowitz*, 1982; *Yeh*, 1985; *Wurbs*, 1995].

Existing Rules for Multireservoir Systems

Several rules of thumb that provide some guidelines to the operation of multireservoir systems have been proposed. *Bower et al.* [1962] suggested two rules for determining releases over time. The pack rule specifies that whenever there is an excess of water, and releases beyond specified targets have some value, water available in excess should be released to realize those benefits. By freeing reservoir space these releases also may reduce the likelihood of future spills. This rule applies more to water supply than to hydropower reservoirs, unless of course spills are certain and heads are a maximum. The hedging rule specifies that whenever there is a shortage of water, and the marginal value of water is a decreasing function of the amount of water supplied, it is advantageous to accept a small current deficit in order to decrease the probability of a more severe water or energy shortage in the future.

For single purpose water supply systems having reservoirs in series, common sense suggests that the water in downstream reservoirs should be used before using water stored in upstream reservoirs to meet water demands downstream of all reservoirs. This procedure minimizes unnecessary spilling at the most downstream reservoir in the event of high lateral flows, that is, flows that do not enter the system through upstream reservoirs. For water supply systems having parallel reservoirs supplying joint demands, that is, downstream demands that can be satisfied by any one or more of the multiple reservoirs, two rules are usually used. The space rule [*Bower et al.*, 1962] attempts to equalize the ratio of available space in each of parallel reservoirs at the end of a period to the expected inflow into each reservoir during the remainder of the refill season, while the NYC rule [*Clark*, 1956] attempts to equalize the probability of filling of each reservoir. Both the space and the NYC rules attempt to avoid the situation of having some reservoirs spilling while the others remain unfilled. *Sand* [1984] showed that for some reasonable assumptions concerning the inflows to these reservoirs, the space rule and the NYC rule yield identical storage target or balancing functions. Both rules were shown to minimize expected shortages in a parallel system having a single downstream demand. For arbitrary inflows the NYC rule was shown to minimize spills.

For water supply systems with side demands, that is, where each demand can be satisfied by only one reservoir, one must balance the likelihood of spills in each reservoir with the likelihood of shortages at each side demand site. *Johnson* [1981] and D. P. Sheer (unpublished report, 1986) suggested the definition of a reserve or safety zone in reservoirs serving side demands. When storage volumes are within those zones, the stored water should be used only to satisfy those side demands. However, they did not provide any guideline for defining these reserve or safety zones.

For systems having distinct refill and drawdown seasons and having side demands, *Wu* [1988] and *Johnson et al.* [1991] defined different policies for each of the two seasons that were shown to be efficient. For the refill season the space rule was modified to consider the existence of side demands. The modification attempts to make the empty and available active storage space in each reservoir proportional to its cumulative expected inflow minus side demands, from the beginning of each period to the end of the refill season.

For the drawdown season, *Wu* [1988] described a rule he called the storage rule. This rule attempts to keep the storage volume of each reservoir proportional to the expected net side demand during the remainder of the drawdown season; that is, it tends to reserve more water in reservoirs having greater side demands, to avoid the risk of some reservoirs being unable to supply their side demands while others that cannot satisfy those demands are still holding extra water.

Johnson et al. [1991] proposed the definition of storage guidelines for the drawdown season that leave enough water in each reservoir to meet the forecast demand minus expected inflow, and balance the remaining water among reservoirs proportionally to their desired active storage. These desired active storage levels were based on flood control limits.

For hydropower systems, *Johnson et al.* [1991] defined an energy production space rule, which is similar to the water supply space rule. These rules state that if potential energy stored in each reservoir is a linear function of storage over the range of storage values of interest, that is, the reservoir has a constant head, then the water supply space rule is equivalent to the energy production space rule. In that case the combination of the water supply space rule with the appropriate energy and water constraints yields a policy that minimizes water and energy spills.

The appendix of this paper defines in more mathematical terms some of the rules just discussed. Each of these rules, however, is useful for the operation of single-purpose reservoir systems. When the reservoir system serves several purposes and its operation is heavily constrained, these rules cannot be applied directly and do not provide clear indications on how to operate the system efficiently. A procedure that considers all system objectives and operational constraints is then needed to produce guidelines for these more complex systems. The technique based on genetic algorithms described in the next sections seems, in our view, a promising one and worthy of additional research. The technique can deal with any system configuration and multiple system objectives, including those involving hydropower. It can consider side demands of energy and water in addition to joint demands and it can accommodate any operational constraints. Moreover, it can provide system operators with rules expressed in forms commonly used in practice.

Genetic Algorithms

Genetic algorithms (GA) are heuristic techniques for searching over the solution space of a given problem in an attempt to find the best solution or set of solutions [*Forrest*, 1993]. The use of GA was first proposed, in 1975, by *Holland* [1992], who based his research on Darwin's principle of evolution. Inspired by the natural mechanisms of selection and reproduction, Holland created an "intelligent" form of a random search that explores the solution space to find the more promising regions and searches for solutions more intensely in those more promising regions. Since then, genetic algorithms have been applied to a variety of problems. *Goldberg* [1989] and *Davis* [1991] review many important applications of genetic algorithms. In the water resources field, genetic algorithms have been applied to a variety of water resource systems management problems [*Verwey et al.*, 1994], to groundwater management models [*McKinney and Lin*, 1993], to pipe network optimization problems [*Goldberg*, 1989; *Simpson et al.*, 1994], and to the calibration of rainfall-runoff models [*Wang*, 1991] and hydrologic streamflow routing models [*Loucks et al.*, 1995]. Hence in this paper we do not review in detail the basic procedures of GA but rather focus on how they are modified to best address this problem.

The power of genetic algorithms arises from a simple assumption: the best solutions are more likely to be found in the regions of the solution space containing high proportions of good solutions. These regions can be located by sampling the entire solution space randomly. When a promising region is found, further exploration in that region is carried on, but at the same time the genetic algorithm maintains the search for other promising regions. Genetic algorithms attempt to balance the exploration of solutions from new areas of the solution space with the more detailed exploitation of solutions in the regions already identified as promising. Users of genetic algorithms set the values of various algorithm parameters that determine this balance.

Genetic algorithms are distinct from hill-climbing methods based on gradients, that is, methods that begin at a randomly generated point and proceed over the response surface in the direction of maximum improvement until they can find no further improvement. Genetic algorithms deal with sets of multiple possible discrete solutions, rather than just one solution, and a differentiable response surface about that solution. Furthermore, genetic algorithms do not guarantee that each new solution will be better than the ones from which it is derived. They only guarantee that the probability of them being better is higher. These two characteristics reduce the chances of getting trapped at some local optimum.

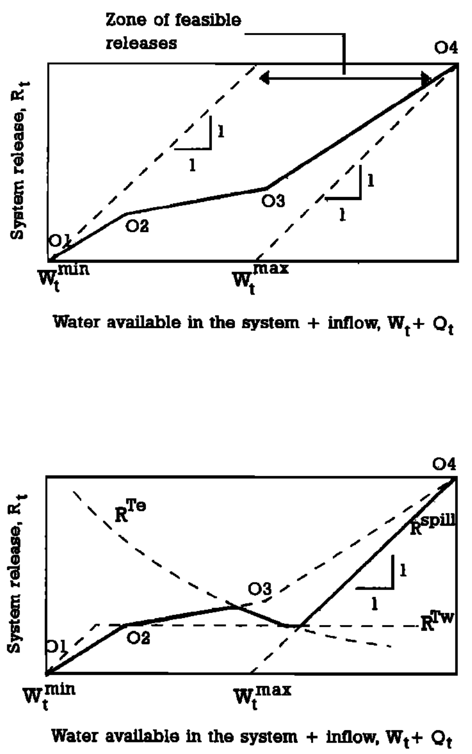


Figure 1. System release rule for a period t. Top figure shows maximum release, which is defined by four coordinates O1 to O4. The actual release (solid line in bottom figure) may be less than the maximum allowable if the energy, R^{T_e}, and water, R^{T_w}, release targets can be met with a smaller release.

A main attraction of genetic search algorithms is that they are independent of the particular problem or system being analyzed. Required are measures of system performance, and these measures may be nonlinear, nondifferentiable, and even discontinuous. The algorithm requires only that any proposed solution can be evaluated on the basis of some measure of performance, and that its performance can be quantified and compared to the performances of other solutions. Indeed, for this application to multiple-reservoir operation, the performance "function" that generates values of system performance is a computer simulation program with intricate loops and complex logic structures. Other attractions of genetic algorithms are their domain independence and robustness, that is, their capacity to deal with a diversity of problems using essentially the same computer code, and the ease of modification, which allows the incorporation of problem specific knowledge to enhance the algorithm efficiency. Finally, the parallel nature of genetic algorithms (many independent solutions must be obtained many times) makes them appropriate for parallel processing. This may be especially attractive if evaluating the performance of any single solution takes considerable computer time, as the use of genetic algorithms require a large number of such evaluations.

Among the drawbacks of genetic algorithms, one might cite their inability to clearly identify the very best single solution, the solution that will give the maximum system performance. This is not of practical concern in this application for several reasons. One reason is that reservoir operation is relatively robust: there are conditions in which different reservoir policies can result in the same overall system performance. In addition, many multipurpose multiplereservoir operations take place in a conflicting multiobjective environment. In such situations it is never clear in practice just what is total maximum system performance. While genetic algorithms may not find the solution that maximizes (or minimizes) some particular objective or performance measure, they can yield very good solutions usually at reasonable computational costs. In some applications, better solutions have been found using genetic algorithms than from using any alternative technique [see, e.g., *Simpson et al.*, 1994].

Another possible problem may occur when there exist many nontrivial constraints, that is, inequalities involving complex combinations of multiple decision variables, as is often the case in multiple-reservoir operation. The problem occurs during the process of generating new candidate solutions, each of which must be feasible; that is, they must satisfy these complex constraints. Our modifications of the basic GA approach were made in part to overcome this problem.

The next section explains how the GA procedure was used to estimate the total system release rule and the individual reservoir storage target balancing functions.

Applying the Genetic Algorithm

The formulation of a genetic algorithm to solve a particular problem requires (1) the choice of a representation scheme that groups the decision variables (variables whose values are unknown) together in a way that permits their definition using genetic operators, (2) the creation of the initial multiple sets of feasible values for each of the decision variables, (3) the definition of an evaluation module that assigns an overall performance (or "fitness") value to each set of decision variable values, (4) the specification of a selection mechanism for generating multiple sets of new solutions from the existing ones in a manner that increases the likelihood of improved solutions, and (5) the choice of values for the genetic algorithm parameters. This section describes how these steps were accomplished in this application.

Genetic algorithms require encoding schemes that transform the vectors of decision variable values that define feasible solutions to a structure that permits genetic operations, for example, reproduction, crossovers, and mutation. These genetic operations will end up creating new (and hopefully improved) sets of values of these decision variables. The most common encoding schemes use binary strings (called "chromosomes"), that is, vectors of binary 0 and 1 bits (called "genes") [*Holland*, 1992; *Goldberg*, 1989]. The use of real-valued (decimal) strings in genetic algorithms has been controversial, with "theoreticians wondering why practitioners have paid so little heed to the theory, and practitioners wondering why the theory seems so unable to come to terms with their findings" [*Goldberg*, 1991, pp. 139–140]. Papers by *Antonisse* [1989], *Janikow and Michalewicz* [1991], *Radcliffe* [1991], *Wright* [1991], and *Eshelman and Schaffer* [1993] have helped to settle some differences over this issue. In this study, real-valued chromosomes were used since they provided a more straightforward way of representing the solutions and permitted the design and use of efficient genetic operators that guaranteed the feasibility of the generated solutions.

In this research the system release rule and the reservoir balancing functions were assumed to be piecewise linear functions (as illustrated in Figures 1 and 2). The coordinates of their inflection points are the unknowns of the optimization problem. The problem is to find the coordinate values that define the operating policy that maximizes system performance.

The release rule defines the release in each within-year period as a function of the existing total system storage and any release targets. Let $R_t^{\max}$ be the maximum release (to be determined) in period t, which is a function of the water available, W_t. The rule to be used expects the operator to meet the period's water supply target demand of T_w, or energy target demand of T_e, if those targets require releases smaller than $R_t^{\max}$. Otherwise, the operator should release $R_t^{\max}$. Let $R_t^{T_w}$ and $R_t^{T_e}$ be the system releases that meet the water supply target, T_w, and the energy target, T_e, respectively, and let R_t^{spill} be the release needed to avoid violating the maximum active storage constraint. The release, R_t, from the system in each period t can be computed using

$$R_t = \max\{R_t^{\text{spill}}, \min[\max(R_t^{T_w}, R_t^{T_e}), R_t^{\max}]\} \qquad (1)$$

The maximum release function, $R_t^{\max}$, for each period t is defined as a piecewise linear function of total system storage consisting of up to NR − 1 connected linear segments (Figure 1). To ensure feasibility and to restrict the solution space of the optimization problem, the coordinates of first point, O_1, are $(W^{\min}, 0)$, and coordinates of the last, or NRth, point O_{NR}, are the coordinates of an arbitrarily assigned point far from the origin. This point is $(2W^{\max}, \text{ER}_t^{\max})$ with $\text{ER}_t^{\max}$ being an estimate of the maximum reasonable release. For water supply systems, $\text{ER}_t^{\max}$ is simply the target demand. For hydropower systems, this maximum may be estimated by the ratio of the energy (release times head) target divided by the minimum head of the most downstream reservoir. Thus, although there are NR segment end points, only NR − 2 need to be estimated. Since each endpoint consists of two coordinates, 2(NR − 2) parameters per period are needed to define the release rule. In this study, NR was set to 4.

The release rules generated by the genetic algorithm must satisfy various constraints, such as those specifying release values cannot exceed the water available in the system. Other constraints could be added to ensure that the release rule is a nondecreasing function of total system storage volume.

These overall system release rules define only a part of a multireservoir operating policy. We also need to know how much to release, if any, from each reservoir. We can obtain this information from reservoir storage target (or balancing) functions that also must be defined.

The balancing functions for reservoir r in each period t are defined by connected piecewise linear functions having end points G_i, $i = 1, \cdots,$ NB, with coordinates (u_{it}, v_{irt}) as illustrated in Figure 1. The abscissa has no index r since at any system storage level the sum of slopes of each reservoir's balancing function must equal 1, and therefore all balancing functions in a period t must change slope at the same value of total system storage. In other words, each reservoir balancing function for any particular period t must have the same number of linear segments. In other words, the endpoints of the each ith segment must have the same abscissa values.

The definition of all the balancing functions requires the estimation of a large number of endpoint coordinates. Fortunately, there are several constraints that reduce this number. First, the first and last point of each balancing function are known. For reservoir r and period t they are $(W_{t+1}^{\min}, S_{r,t+1}^{\min})$ and $(W_{t+1}^{\max}, S_{r,t+1}^{\max})$, where $W_{t+1}^{\min}$ and $W_{t+1}^{\max}$ are the minimum and maximum allowable active system storage volumes at the

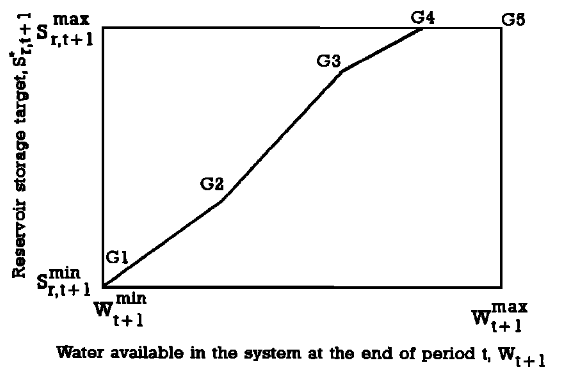


Figure 2. A storage target (balancing) function for a reservoir r in a period t. This function is defined by five coordinates G1 to G5.

end of the period t, and $S_{r,t+1}^{\min}$ and $S_{r,t+1}^{\max}$ are the corresponding minimum and maximum active storage volumes for reservoir r. Second, since the sum of storage volumes in all reservoirs must equal the total system storage volume, in a system with R reservoirs, only $R - 1$ balancing functions have to be estimated in each within-year period t. Thus, for a system with R reservoirs, if NB points (each defined by two parameters, one of which is common for all r) are chosen to define the balancing functions, (NB − 2)(R − 1) + NB − 2 or (NB − 2)R parameters need to be estimated each period. In this research NB was set to 5, as shown in Figure 2.

The balancing functions generated by the genetic algorithm have to satisfy the following sets of constraints. One set of constraints requires that the balancing functions do not assign to a given reservoir less water than its minimum admissible storage or more water than its maximum allowable storage. Another set of constraints requires that all balancing functions are increasing functions of the system storage with slopes between 0 and 1. Note that it is conceivable to have balancing functions with slopes smaller than 0 or greater than 1, although this situation seldom, if ever, arises in practice.

Using the described encoding scheme, the operating policy of a system with R reservoirs required to satisfy joint demands of water and energy may be defined by a set of [2(NR − 2) + R(NB − 2)]T parameters, where again NR and NB are the number of points defining the overall system release rule and the each of the storage volume balancing functions for the R reservoirs, respectively, in each of the T within-year periods.

The genetic algorithm starts by randomly generating the initial set of solutions. At this step two issues are of concern. First, each generated solution must be feasible. This condition precludes the option of randomly generating each parameter independently of the others. Second, the probability of generating a solution anywhere in the feasible solution space has to be fairly uniform to avoid any bias in the search process. Thus the generation of coordinate values defining part of a release or balancing function is constrained by the points that have been generated previously. To avoid bias, the order of point generation in our approach is random. If the generation process produced infeasible rules, these were rejected and the process was repeated until a feasible solution was obtained.

After generating the initial set of operating policy solutions, a reservoir system simulation model can be used to evaluate the performance of each solution. In our study the performance of each operating policy was evaluated using the same flow series. A function of the average period's water or energy deficit was used as a measure of performance. A normalization scheme was used to assign a relative fitness value to each solution. Both windowing and ranking schemes [see *Davis*, 1991] were tested. The ranking scheme, which proved to be better, sorts the policies by their performances. It assigns a performance index of M, the number of solutions in each generated population set, to the best policy, $M - 1$ to the second best policy, and so forth to 1 for the worst performing policy.

A roulette-wheel parent-selection scheme [*Goldberg*, 1989] was used for reproducing a new population of solutions from the existing population. This roulette-wheel scheme assures a higher probability of reproducing solutions that have better performances. (The probability of selecting any particular solution having a performance index or fitness ranking of m, within the range 1 through M, is m divided by the sum of integers 1 though M.) Next this new population of reproduced solutions are paired, assuring that each solution of a pair is not identical to the other solution in that same pair. These pairs of solutions make up what are called parents. These "parents" will produce "children": other solutions having many but not all of the characteristics of their parents. The second feature employed, usually called elitism, copied without any change the best solution of a population to the next population.

Children solutions are produced from parent pairs of solutions using crossover and mutation operations. Among the several crossover operators tested during this research, two of them proved to yield the best results: the uniform crossover operator and the quadratic crossover operator [*Oliveira*, 1994]. To ensure the feasibility of the operating rules, these operators deal with groups of parameters, each defining one operating rule, rather than each parameter individually.

Let $\mathbf{P}_1$ and $\mathbf{P}_2$ be vectors containing the coordinates of all the points that define the operating policies of the two parents. These points may be grouped in vectors that define either a release rule or a set of storage volume balancing functions. Let $\mathbf{RR}_t^{\mathbf{P}_1}$ and $\mathbf{RR}_t^{\mathbf{P}_2}$ be the vectors containing the parameters of the system release rule for period t defined by the two parent solutions, and let $\mathbf{BF}_t^{\mathbf{P}_1}$ and $\mathbf{BF}_t^{\mathbf{P}_2}$ be the vectors containing the parameters of all the balancing functions for period t defined by the two parent solutions. In addition, let $\mathbf{RU}$ be a vector of binary random variables.

The child solution is to be derived from a series of uniform crossover operations. The uniform crossover operator is a modification of the uniform crossover for bit strings described in the literature [*Syswerda*, 1989]. Each child operating rule solution is set equal to the correspondent rule defined by either parent, depending on the outcome of a binary random variable. If the ith element of the random vector $\mathbf{RU}$ is equal to 1, then the ith rule of the child solution is equal to the ith rule of solution $\mathbf{P}_1$. Otherwise, it is equal to the ith rule of solution $\mathbf{P}_2$. The operation is illustrated below.

$$\begin{array}{llllllllll}
\mathbf{P}_1 = [\mathbf{RR}_1^{\mathbf{P}_1} & \mathbf{BF}_1^{\mathbf{P}_1} & \mathbf{RR}_2^{\mathbf{P}_1} & \mathbf{BF}_2^{\mathbf{P}_1} & \mathbf{RR}_3^{\mathbf{P}_1} & \mathbf{BF}_3^{\mathbf{P}_1} & \cdots & \mathbf{RR}_t^{\mathbf{P}_1} & \mathbf{BF}_t^{\mathbf{P}_1}] \\
\mathbf{P}_2 = [\mathbf{RR}_1^{\mathbf{P}_2} & \mathbf{BF}_1^{\mathbf{P}_2} & \mathbf{RR}_2^{\mathbf{P}_2} & \mathbf{BF}_2^{\mathbf{P}_2} & \mathbf{RR}_3^{\mathbf{P}_2} & \mathbf{BF}_3^{\mathbf{P}_2} & \cdots & \mathbf{RR}_t^{\mathbf{P}_2} & \mathbf{BF}_t^{\mathbf{P}_2}] \\
\mathbf{RU} = [1 & 1 & 0 & 0 & 1 & 0 & \cdots & 1 & 0] \\
\mathbf{C} = [\mathbf{RR}_1^{\mathbf{P}_1} & \mathbf{BF}_1^{\mathbf{P}_1} & \mathbf{RR}_2^{\mathbf{P}_2} & \mathbf{BF}_2^{\mathbf{P}_2} & \mathbf{RR}_3^{\mathbf{P}_1} & \mathbf{BF}_3^{\mathbf{P}_2} & \cdots & \mathbf{RR}_t^{\mathbf{P}_1} & \mathbf{BF}_t^{\mathbf{P}_2}]
\end{array}$$

Alternatively, the quadratic crossover operator with extrapolation is an arithmetic operator that can be represented by (2) and (3). Let w_t and z_t be random weights. If parent chromosome $\mathbf{P}_1$ is fitter than parent $\mathbf{P}_2$, the random weights are initially set to $1.25 - q^2$, with q being a random variable uniformly distributed over 0 and 1.23. Otherwise, the weights are set to q^2. The values of each pair of weights w_t and z_t are independently determined. The random weights are thus random variables distributed between -0.25 and 1.25, but they are biased towards the most fitted parent. The child chromosome, $\mathbf{RR}_t^{\mathbf{C}}$, may then be computed using the following crossover operations:

$$\mathbf{RR}_t^{\mathbf{C}} = w_t \mathbf{RR}_t^{\mathbf{P}_1} + (1 - w_t)\mathbf{RR}_t^{\mathbf{P}_2} \tag{2}$$

$$\mathbf{BF}_t^{\mathbf{C}} = z_t \mathbf{BF}_t^{\mathbf{P}_1} + (1 - z_t)\mathbf{BF}_t^{\mathbf{P}_2} \tag{3}$$

Finally, let $\mathbf{C}$ be a vector containing the coordinates of all the points that define the operating policy of the child solution that is to be derived from its two parents. The vector $\mathbf{C}$ combines the vectors $\mathbf{RR}_t^{\mathbf{C}}$ and $\mathbf{BT}_t^{\mathbf{C}}$ for all periods t.

Figure 3 illustrates (3) ((2) is very similar). Figure 3 shows two balancing functions for a given reservoir and season that have been selected for crossover. The child balancing function is defined by points located on the dashed lines connecting the points defining the parent balancing functions. The exact location of these points is determined by the values the weights assume, but there is a higher probability of being closer to the correspondent point of the more fitted parent. If the generated policy is infeasible, the random weights w_t and z_t are reduced until feasibility is reached.

Both these crossover operators satisfy the *Radcliffe* [1991] principles of respect and proper assortment if we perceive the operating policy functions, either the release rule of a given period or the set of balancing functions for a given period, as the characteristics to be maintained from the parent solutions. The principle of respect states that if the parent operating policies have the same release rule or a set of balancing functions for a particular period, the child operating policy will also have that release rule or set of balancing functions for that same period. The principle of proper assortment asserts that if each parent's operating policy has a particular feature, and these two different features do not conflict with each other, it is at least possible that the child operating policy will inherit both of those features.

Finally, each generated child solution resulting from these reproduction, pairing, and crossover operations can be mutated. The mutation operators evaluated in our research lead us to the use of a modified floating-point nonuniform operator

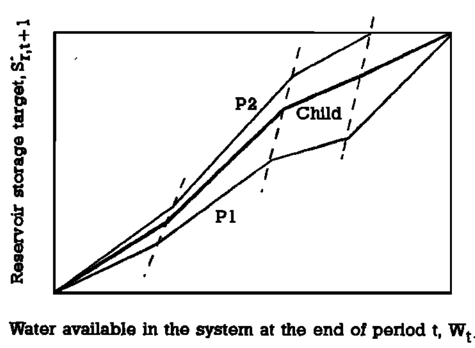

Reservoir storage target, $S^*_{t,t+1}$
P2
Child
P1
Water available in the system at the end of period t, W_{t+1}

Figure 3. Arithmetic crossover operation of two storage target balancing functions, $\mathbf{P}_1$ and $\mathbf{P}_2$, to produce a child, $\mathbf{C}$. The child is a linear combination of the parent functions.

[*Davis*, 1991; *Michalewicz*, 1991, 1992]. The mutation of an operating policy is obtained by changing the coordinates of the points that define the policy, if a probability test is passed. The probability test is performed by generating a random number uniformly distributed between 0 and 1. If this number is smaller than a user set parameter p_{mut}, a mutation is performed. Hence there is a probability p_{mut} of performing a mutation.

To ensure that the outcome of the mutation is feasible, the mutation of release rules is slightly different than the mutation of balancing functions. For total system release rules the probability test is applied to every segment endpoint of the piecewise linear release rule. A displacement vector $[D_{xt}, D_{yt}]$ is applied to each coordinate x, y point randomly selected for mutation. This displacement can differ in each period t. The amount of each x (storage) and y (release) displacement, $[D_{xt}, D_{yt}]$, is normally distributed with 0 mean and standard deviations, SD_{xt} and SD_{yt}, as computed by (4). The parameters x_{mut} and y_{mut} are user-defined and range from 0 to 1. Figure 4 illustrates the mutation of a coordinate point.

$$SD_{it} = i_{\text{mut}}(W_t^{\max} - W_t^{\min}) \qquad i = x, y \tag{4}$$

For balancing function mutations, one probability test is applied to each set of points that have the same abscissa. If a set of points is selected for mutation, the mutation is applied in two steps. First, as shown in Figure 5, the abscissa value (the same for all reservoirs) is changed randomly, and the ordinates of all points for all reservoirs are modified to maintain feasibility. Then, one reservoir is randomly selected, and its ordinate is randomly changed. The ordinates of the other reservoirs are again modified to maintain feasibility. The displacements, D_{xt} and D_{yt}, of the coordinates x and y randomly selected for mutation are normally distributed about their current values. The standard deviations of those displacements again vary for each period t and are computed using (4) for D_{xt} and (5) for the D_{yt} corresponding to the selected reservoir r.

$$SD_{yt} = y_{\text{mut}}(S_{rt}^{\max} - S_{rt}^{\min}) \tag{5}$$

If through mutation an operating policy becomes infeasible, the coordinates of the mutated point are corrected to the nearest point that makes the rule feasible. Finally, since the change of location of any point modifies the interval over

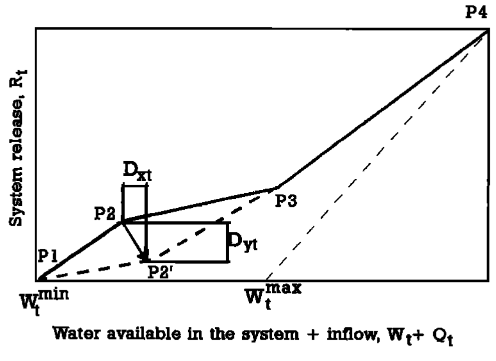
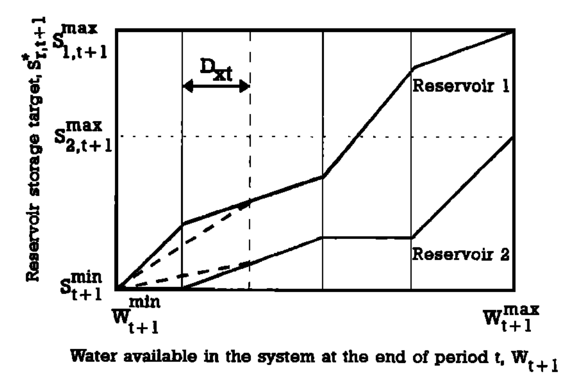


Figure 4. An example of the mutation of the system release rule. Point P2 is displaced to point P2′. The displacement coordinates D_{xt} and D_{yt} are random values that can come from either normal or uniform distributions.



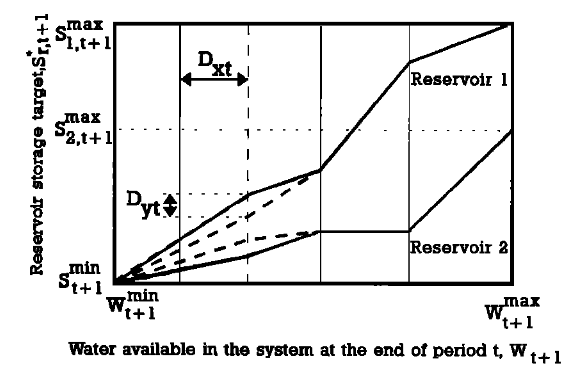


Figure 5. An example of the mutation operation of the storage balancing functions for two reservoirs. In this example, the second set of points is selected for mutation. First, the abscissa is displaced by a random D_{xt} and the ordinates are altered to maintain feasibility. The result is shown in dashed lines in the top figure. Next a random displacement of D_{yt} is applied to the reservoir 1 ordinate, and the reservoir 2 ordinate is corrected to maintain feasibility. The end result is shown by dashed lines in the bottom figure.

which the other points may change, the order of each point or set of points tested for mutation is random to avoid any bias.

The values of the mutation parameters x_{mut} and y_{mut} vary throughout the solution generating, modifying and evaluating (evolutionary) processes. We have found that relatively high values (close to 1) work well in the beginning. As the process proceeds through successive generations of solutions, the parameter values can be decreased and new and relatively low values gradually reduced as generations pass. When p_{mut} is increased and x_{mut} and y_{mut} are reduced as generations pass, the genetic algorithm search moves gradually from exploring the solution space to refining good solutions that have already been found.

Algorithm Parameter Setting

To use the described algorithm, values of each of the GA parameters must be defined. These parameters include the sample size of each population of solutions and the probabilities of crossover and mutation. Determining the best set of values for the parameters is not a trivial task, owing to the number of possible combinations of parameter values. Several authors [*De Jong*, 1975; *Schaffer et al.*, 1989] have tried to find parameter values that work well across a wide variety of function optimization problems. They have, however, assumed a traditional genetic algorithm, and thus the derived parameter

Table 1. Suggested Set of Parameter Values for the Genetic Algorithm

Parameter	Value
Population size, M	>40
Crossover rate, p_c	1.0
Mutation rate, p_{mut}	0.05–0.2
Standard deviation of mutation, i_{mut}	0.2–0.05

sets, if applied to our model, would probably lead to poor results. *Grefenstette* [1986] used a metagenetic algorithm to find good parameter values for a lower level genetic algorithm. Although his parameter set outperformed De Jong's hand-derived parameters, the improvement was small. Given these circumstances, we decided to use a trial-and-error procedure to try to find a good set of values for the algorithm parameters and to identify the most sensitive ones.

The genetic algorithm was applied to three reservoir systems, each having a different objective function. Several parameter sets were tested with each example problem. The parameter values that consistently lead to the best results are presented in Table 1.

The suggested set of parameter values indicates that the probability of crossover should be 1.0, the probability of mutation should vary from 0.05 to 0.2, and the mutation standard deviation parameter should vary from 0.2 to 0.05. Table 1 does not recommend a specific value for the population size. The experience with the algorithm has shown that the results may be very sensitive to the size of the population employed. Small populations do not permit a sufficient sampling of the solution space, whereas a large population requires substantial computational resources without necessarily leading to significant improvements in the results. Also, GA applied to different systems seems to prefer different population sizes. To derive the operating policy of a large system, it is preferable to use a genetic algorithm with a large population because it provides a better sampling of the vast solution space.

The optimal population size also depends on the number of simulations the algorithm will perform. If this number is small, due to computer resource limitations, it is preferable to select a small population size and allow it to evolve for more generations than to select a larger population and let it evolve for fewer generations. Figure 6 shows this trade-off by plotting the average evolution of the objective function value of a four-reservoir hydropower system obtained from five genetic algorithms, each with a different population size. The genetic algorithm with a population of one corresponds to a random hill-climbing algorithm. Each average evolution curve was obtained from 50 runs of the genetic algorithm. As Figure 6 shows, the population size that yields the smallest deficit (highest performance) varies with the number of policies simulated. Large population sizes yield good policies if the algorithm is run for many generations. Conversely, small population sizes perform well at the initial stages of the evolution, but their relative performance decreases if the algorithm is run for many generations. This is similar to the random hill-climbing algorithm that performs well at the initial stages of the evolution but poorly at later stages of the evolution.

Figure 6 presents the average results of several runs of the genetic algorithm. In practice, we are not interested in the average deficit of several runs but rather in the best policy obtained from a number of runs. Typically, when we use a genetic algorithm to derive an operating policy, we perform several runs and select the policy that yields the best results. Our experience has shown that although the random hill-climbing algorithm is competitive with the genetic algorithm, the genetic algorithm yields better results if a large population, say, one greater than 100, is allowed to evolve for many generations. This comparative advantage increases as the number of reservoirs in the system increases.

A final parameter that needs to be defined is the length of the flow series to be used by the simulation model to obtain system performance values for each proposed solution. Ideally, we would like to estimate the performance of each operating policy over a long sequence of flows, but this increases the computational burden. Note that most of the computer resources consumed by a genetic algorithm (70 to 90%) are typically used to evaluate each solution of each successive population.

Results

This section reports the results obtained from two hypothetical systems showing that genetic algorithms are a practical and robust way to estimate efficient operating policies for multireservoir systems. The simple example systems presented here have only two reservoirs that are either in series or in parallel. Only two within-year seasons are considered: winter and summer. Inflows for these seasons have been generated from a stochastic flow generator. The reservoir systems are required to satisfy a single purpose, either water supply or hydropower. The reservoirs were assigned the same capacity, 40 volume units, and dead storage zones and flood storage zones were not considered. Finally, it was assumed that both reservoirs have the same storage-head functions that were represented by second-order polynomials. These simple models are used here only to be able to judge the reasonableness of the results of the application of GA. More complex examples, having policies that are much less intuitive, are given by *Oliveira* [1994].

The genetic algorithm was applied to the example problems with the set of parameter values suggested in Table 1. A population of 40 was allowed to evolve for 60 generations. A flow sequence of 1000 periods was used. The genetic algorithm was run 10 times, that is, 10 generations of solutions were obtained. Although each run of the genetic algorithm produces a set of M (the population size) operating policies, only the one that lead to the best objective function (smallest sum of deficits)

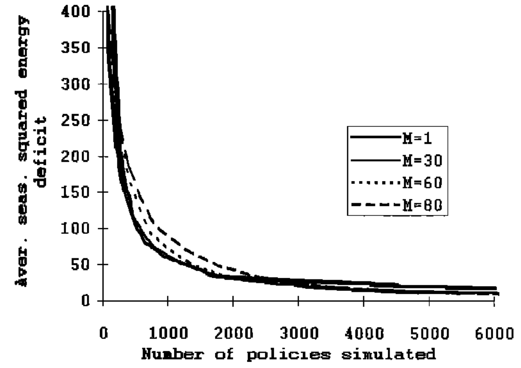


Figure 6. Evolution of the average seasonal squared energy deficit for various population sizes. The averages are based on 50 independent simulations.

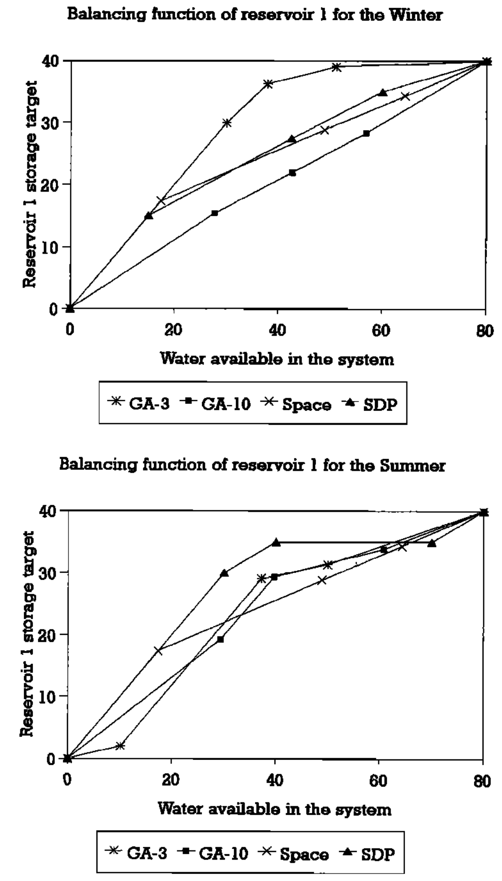


Figure 7. Comparisons among storage balancing functions derived for the example problem PWS.

value was considered. To assess the quality of the derived policies, these policies were then compared with policies derived by other techniques.

Example problem PWS considers a system with two reservoirs in parallel that are required to satisfy a joint water demand of 13 units of water in the winter refill period and 35 units in the summer drawdown period. The release rules derived by the genetic algorithm coincide with the "standard operating rule" that minimizes sum of water deficits, that is, a policy that meets the target whenever it is possible or releases all the available water whenever the targets are not achievable. Figure 7 shows the two balancing functions obtained with the genetic algorithm, one derived from the space rule and one obtained from a stochastic dynamic programming (SDP) code [*Perera*, 1986; *Wu*, 1988]. As can clearly be seen, all balancing functions are very similar, especially the ones obtained for the summer season. There are differences in the policies derived for the winter season because there is some leeway in the water distribution at the end of the refill season. For water supply systems exclusively with joint water demands, the only operating concern is avoiding spillage, and therefore the critical water distribution is at the beginning of the refill season.

The operating policies shown in Figure 7 were simulated with 10 independent but equally likely flow sequences, each 4000 periods long. Table 2 presents the results obtained. Table 2 shows that the policy derived by the space rule is the one that yields the lowest average seasonal deficit, although there is no statistical difference between this policy and the policies derived by the genetic algorithm or with the policy derived by stochastic dynamic programming.

Table 2. Results From the Simulation of Different Operating Policies for Example Problem PWS With 10 Independent but Equally Likely Flow Sequences of 4000 Periods Each

Average Seasonal Deficit	GA-1	GA-2	Space	SDP
Mean	0.877	0.876	0.866	0.869
Standard deviation	0.104	0.103	0.102	0.104

SDP, stochastic dynamic programming.

Example problem PHP2 again considers a system with two reservoirs in parallel. The system is required to satisfy a joint energy requirement of 500 every season. The operational objective is the minimization of squared deficits of energy. The results obtained for example problem PHP2 are presented in Figures 8 and 9 and in Table 3, where some of the policies obtained with the genetic algorithm are compared with a heuristic policy that includes a release rule derived by a greedy hill-climbing algorithm and balancing functions derived from (A10), in the appendix. The greedy hill-climbing method attempts to derive the release rule that maximizes the system objectives for a set of balancing functions computed by (A11), in the appendix. Given an initial release rule, several other release rules in its neighborhood are simulated. If a better rule is found, that release rule replaces the existing one and the

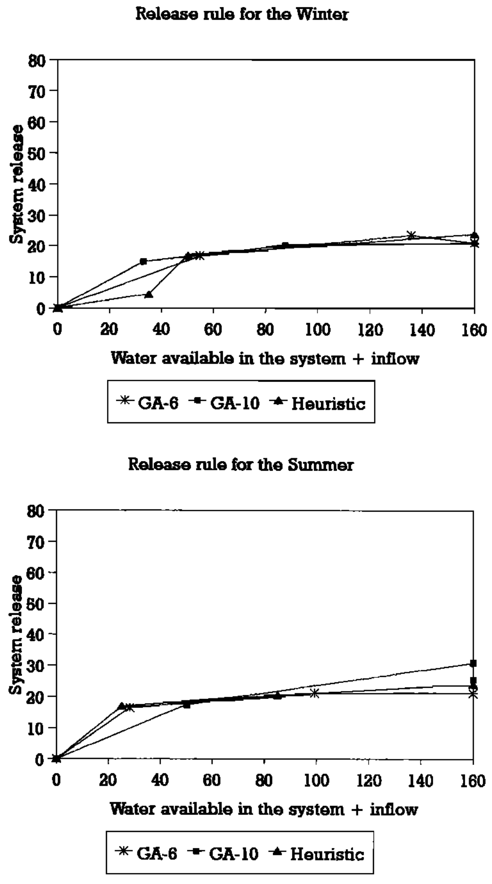


Figure 8. Comparisons among release rules derived for the example problem PHP2.

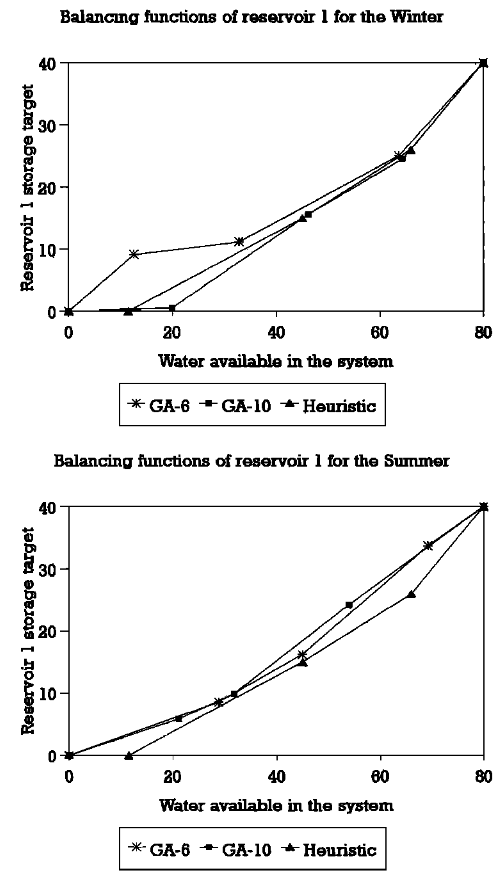


Figure 9. Comparisons among storage balancing functions derived for the example problem PHP2.

procedure starts again. If a better release rule is not found, the existing release rule is considered to be the best option. This algorithm is not random. To mitigate the problem of local optima, the hill-climbing method was applied several times with different starting release rules to derive several local optima. The release rule that performed best was chosen from this set.

Figures 8 and 9 show that all policies are very similar. Table 3 indicates that the operating policies derived by the genetic algorithm yield a lower average seasonal deficit than the policies derived by (A11).

Computational Requirements

We have explored the relationship between the system size and the computational requirements needed to achieve efficient operating policies. To do this, the genetic algorithm was applied to four systems with an increasing number of reservoirs: 2, 4, 6, and 8. The number of seasons was kept constant to 4.

A normalized loss index was defined to compare the performance of the genetic algorithm for different example problems. This normalized loss is the average over all genetic algorithm runs of the ratio between the amount the algorithm output exceeds the minimum loss and a normalization factor. This normalization factor is the average loss of the initial population for all runs minus the minimum loss achievable with the flow sequence used by the genetic algorithm. Let L_{kg} be the loss returned by the best policy in the population at the gth generation of the kth run of the algorithm. Define L_0 as the average loss of the initial population for all runs, which is an estimate of the average loss of policies in the solution space. Finally, let $L^{\min}$ be the minimum loss achievable for the flow series used by the genetic algorithm. If the algorithm is run N_{run} times, the normalized loss in the gth generation can be computed by

$$\text{Normalized loss at generation } g = \sum_{k=1}^{N_{\text{run}}} \frac{L_{kg} - L^{\min}}{L_0 - L^{\min}} \tag{6}$$

This normalized loss (deficit) varies from 1 to 0. A value of 1 corresponds to the estimate of the average loss of the policies in the solution space, whereas a value of 0 corresponds to the minimum achievable loss.

The number of simulations required to achieve a given normalized deficit is a function of the number of operating policy variables to be estimated. The number of simulations required to achieve a given normalized deficit increases linearly with the number of policy variables to be estimated, but since the computational time required to perform each simulation increases with the system complexity, the computational time increases nonlinearly.

Potential Improvements

The genetic algorithm described in this paper is only one way to derive the operating policy of multiple-reservoir systems. It may not be the best way, but it is one of the best ways we have found so far, especially for hydropower systems that cannot be operated using real-time optimization. One of the difficulties facing potential users of genetic algorithms is the lack of standards in the field. Each user exploits the genetic algorithm concepts in a different way, and it is hard to perceive which are the best implementations for particular applications. Users of GA seem to be forced to try different alternatives, and certainly different GA parameter values, and to choose those that perform best for their particular application. On the other hand, this need for experimentation and judgment is not unique to GA. Modeling is in part an art, which makes it interesting!

Algorithm improvement will result from any increase in the efficiency and speed of the simulation model used for performance evaluations, since it consumes the most computer time. An additional possible improvement would be to change, as the successive generations of solutions are defined, the length of the flow series used to evaluate each policy. In the early stages of the search, a short flow sequence could be used to locate the most promising regions, whereas in the final stages a longer flow sequence could be used to distinguish between close policies. An alternative would be to evaluate each policy only at critical periods, when the inflows are low and/or de-

Table 3. Results From the Simulation of Different Operating Policies for Example Problem PHP2 With 10 Independent but Equally Likely Flow Sequences of 4000 Periods Each

Average Seasonal Loss	GA-6	GA-10	Heuristic
Mean	3588.8	3578.1	3847.3
Standard deviation	273.7	274.6	273.8

mands are high. The performance index of each policy could therefore be the average of a number of short-period performance indices. This would save computer simulation time devoted to evaluating each policy during high flow periods, when many policies perform equally well.

It may be possible to improve the best solutions identified by genetic algorithms by completing the search process with some hill-climbing technique. The genetic algorithm can be used to identify the most promising region of the solution space, and the hill-climbing method can complete the search by identifying the local optima of that region. Some authors have reported good results using this combined genetic algorithms–hill-climbing method [*De Jong*, 1993a; *McKinney and Lin*, 1994].

Some additional possible improvements include (1) the seeding of the initial population with better-than-average solutions computed by heuristic rules, (2) the use of an incremental replacement of the population instead of en bloc replacement [*Whitley*, 1989; *Davis*, 1991; *De Jong*, 1974, 1993b], and (3) the dynamic modification of the crossover and mutation rates based on the average performance increase each operator is producing or based on some characteristics of the existing population, like diversity indices [*Wilson*, 1986; *Booker*, 1987; *Davis*, 1989, 1991].

There is considerable research being performed in the genetic algorithm field. Several journals routinely publish papers where the authors put forward different formulations and many refinements over the traditional algorithm. In this section we have just mentioned a few. Since it is hard to estimate which proposals might be most effective without implementing them, there is considerable experimental work remaining to be done.

Conclusions

The research reported in this paper focused on the estimation of effective operating policies for multipurpose multireservoir systems using genetic algorithms. Particularly, we have studied ways to derive predefined operating policies that indicate the total release from the system of reservoirs and the individual reservoir storage targets as functions of time of the year and the existing total storage volume in the system. The release from each reservoir is determined by attempting to minimize deviations of actual storage volumes in each reservoir from their desired target storage volumes while making the specified total system release. These operating policies define medium-term to long-term target storage levels and target releases. They can be used to guide the system operators in their decision-making process.

Alternatively, optimization models can be designed to compute the short-term releases that satisfy all system requirements and leave the system in a state, as close as possible, to the one defined by these long-term targets. These optimization models can then be used as a decision-aid tool in the day-to-day operation of reservoirs systems. Both the predefined policies and the real-time optimization models can be incorporated into simulation models of reservoir system operation to study the impact of those policies based on different hydrological scenarios.

For complex reservoir systems that are to meet multiple objectives and have side demands or constraints on releases or power production, the general rules of thumb for reservoir operation cannot result in efficient system-wide operation. In these situations, another approach is needed. This research suggests that genetic algorithms may be a practical and robust way of estimating operating policies for such systems.

The genetic algorithm used here to derive and verify these operating rules employed real-valued chromosomes containing the coordinates of the points that define piecewise linear operating rule functions. The algorithm was applied to several hypothetical example problems with different system configurations and objectives. In all cases effective operating policies were identified and verified using simulation.

Since genetic algorithms use a simulation model to evaluate each generated operating policy, no restrictions on how to define the operating policy and how to evaluate them are imposed. Thus the algorithm can generate policies in the formats desired and used by the system operators. These policies can be simulated, and their performance evaluated, using any simulation model applicable to the system being simulated and acceptable to the system operators. Finally, the computational requirements of the GA-simulation approach for systems of moderate size and complexity do not appear to be excessive.

Some improvements over the algorithm used in this study are possible. Among those most promising include seeding the initial population with operating policies generated by heuristic rules, concluding the search procedure by a hill-climbing method, and including a mechanism that dynamically adapts the mutation and crossover rates. Also, to reduce the computational requirements of the genetic algorithm, it may be beneficial to change, as the population evolves, the length of the flow sequence used to evaluate each policy.

Appendix: Some Simple Operating Principles

The space rule with side demands can be expressed mathematically. Consider a system of R reservoirs. Let $S^*_{r,\tau}$ be the storage target level in reservoir r at the beginning of period τ. Define $\mathrm{CI}^{Rf}_{r,\tau}$ to be the total expected inflow to reservoir r and $\mathrm{CD}^{Rf}_{r,\tau}$ be the total expected demand from reservoir r, both from the beginning of refill period τ to the end of the refill season. The expected inflow may include releases from upstream reservoirs if they exist. Also, let $K_{r,\tau}$ be the active storage capacity of reservoir r at the beginning of period τ, and W_τ be the total system storage volume at the beginning of period τ. Then,

$$\frac{K_{r,\tau} - S^*_{r,\tau}}{\mathrm{CI}^{Rf}_{r,\tau} - \mathrm{CD}^{Rf}_{r,\tau}} = \frac{\sum_{r=1}^{R} K_{r,\tau} - W_\tau}{\sum_{r=1}^{R} (\mathrm{CI}^{Rf}_{r,\tau} - \mathrm{CD}^{Rf}_{r,\tau})} \qquad \text{(A1)}$$

for all reservoirs $r = 1, \cdots, R$ and all periods τ of the refill season. Note that in the refill season the expected cumulative inflow $\mathrm{CI}^{Rf}_{r,\tau}$ must be greater than the expected cumulative demand $\mathrm{CD}^{Rf}_{r,\tau}$ otherwise refill will not occur.

The storage rule can also be defined mathematically. Let $\mathrm{CI}^{Dd}_{r,\tau}$ be the total expected inflow to reservoir r from the beginning of a drawdown period τ to the end of the drawdown season, and let $\mathrm{CD}^{Dd}_{r,\tau}$ be the total demand to be met from reservoir r from the beginning of period τ to the end of the drawdown season. Then,

$$\frac{S^*_{r,\tau}}{(\mathrm{CD}^{Dd}_{r,\tau}-\mathrm{CI}^{Dd}_{r,\tau})}=\frac{W_t}{\sum_{r=1}^{R}(\mathrm{CD}^{Dd}_{r,\tau}-\mathrm{CI}^{Dd}_{r,\tau})} \quad \text{(A2)}$$

for all $r = 1, \cdots, R$ and all τ of the drawdown season. Again, note that in the drawdown season the expected cumulative demand $\mathrm{CD}^{Dd}_{r,\tau}$ will be greater than the expected cumulative inflow $\mathrm{CI}^{Dd}_{r,\tau}$, otherwise drawdown will not occur.

For hydropower systems with variable-head reservoirs, some general rules that maximize the system-expected power production and satisfy the system requirements of water and energy can be derived. These rules indicate what releases should be made during each period that maximize the sum of expected power production from the current period on.

Consider the system operator decision problem at the beginning of a period t. Assume that the system is required to meet a known energy target, E, during the next immediate period. In addition to the symbols already defined, let $R_{r,\tau}$ be the release from reservoir r to the hydropower plant at reservoir r during period τ, $Q_{r,\tau}$ be the expected inflow to reservoir r in period τ, and $h_r(S_{r,\tau})$ be the storage head of reservoir r associated with the storage volume $S_{r\tau}$. Ignoring the risk of spillage (releases in excess of maximum power plant flows) and assuming that at the beginning of the decision period the operator knows the actual reservoir storage levels and the estimated inflows to system for the immediate period, the mathematical formulation of the operator's optimization problem for period t can be stated by (A3)–(A6). The unknowns of the optimization problem are the future storage volumes, $S_{r,\tau}$, for all periods $\tau > t$ and all reservoir releases, $R_{r,\tau}$. The known parameter N is the number of periods of the optimization horizon.

For maximum energy production over this N number of time periods,

$$\max \sum_{\tau=t+1}^{t+N-1}\sum_{r=1}^{R}\frac{h_r(S_{r,\tau})+h_r(S_{r,\tau+1})}{2}R_{r,\tau} \quad \text{(A3)}$$

is subject to meeting an "energy" (storage head times release) target, E, this period ($\tau = t$) only:

$$E=\frac{h_r(S_{r,t})+h_r(S_{r,t+1})}{2}R_{r,t} \quad \text{(A4)}$$

and mass balance and nonnegativity constraints:

$$R_{r,\tau}=S_{r,\tau}+Q_{r,\tau}-S_{r,\tau+1} \quad \text{(A5)}$$

$$\tau=t,\cdots,t+N,\ r=1,\cdots,R$$

$$S_{r,t+1}\geq 0,\ R_{r,\tau}\geq 0 \qquad \tau=t,\cdots,t+N,\ r=1,\cdots,R \quad \text{(A6)}$$

The above formulation ignores a number of constraints on the system operation, such as reservoir and power plants capacities, that were not included here only to keep the problem simple. In addition we will from now on drop the nonbinding nonnegativity constraints (A6) thus eliminating all inequality constraints.

We can use the Lagrange-multiplier method to compute the future storage volumes and heads that maximize the energy production while insuring an amount of energy (expressed as simply average storage head times release) in the immediate period t. This will allow us to estimate what releases from each reservoir would increase our energy production in the current period most effectively, that is, insuring a minimum loss of energy production in the future. Our purpose is to compute the appropriate reservoir releases from each reservoir in the period t given known initial storage volumes and expected current inflows.

The Lagrangian of this optimization problem is

$$L=\sum_{\tau=t+1}^{t+n-1}\sum_{r=1}^{R}\frac{h_r(S_{r,\tau})+h_r(S_{r,\tau+1})}{2}(S_{r,\tau}+Q_{r,\tau}-S_{r,\tau+1})$$
$$+\lambda\left(E-\sum r\,\frac{h_r(S_{r,t})+h_r(S_{r,t+1})}{2}(S_{r,t}+Q_{r,t}-S_{r,t+1})\right) \quad \text{(A7)}$$

Equating the derivatives of L in respect to the unknown $S_{r,t+1}$, we get for the current period t:

$$\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}\frac{R_{r,t+1}}{2}+\frac{h_r(S_{r,t+1})+h_r(S_{r,t+2})}{2}=\lambda\left(\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}\frac{R_{r,t}}{2}-\frac{h_r(S_{r,t})+h_r(S_{r,t+1})}{2}\right) \quad \text{(A8)}$$

The first parts of both sides of (A8) can be approximated by

$$\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}R_{r,t}\approx\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}(Q_{r,t}-\Delta S_{r,t})$$
$$\approx\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}Q_{r,t}-[h_r(S_{r,t+1})-h_r(S_{r,t})] \quad \text{(A9)}$$

$$\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}R_{r,t+1}\approx\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}(Q_{r,t+1}-\Delta S_{r,t+1})$$
$$\approx\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}Q_{r,t+1}-[h_r(S_{r,t+2})-h_r(S_{r,t+1})] \quad \text{(A10)}$$

By introducing (A9) and (A10) in (A8), we obtain the following expression for λ, the change in the energy produced in the future due to a unit increase in the amount of energy that must be produced in this current period t:

$$\left(\frac{1}{2}\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}Q_{r,t+1}+h_r(S_{r,t+1})\right)\Bigg/\left(\frac{1}{2}\frac{\partial h_r(S_{r,t+1})}{\partial S_{r,t+1}}Q_{r,t}-h_r(S_{r,t+1})\right)=\lambda \quad \text{(A11)}$$

Equation (A11) can be used to define balancing functions for reservoir systems producing hydroelectric power. Equation (A11) shows that the inflows should, if possible, be allocated to those reservoirs r in such a way as to keep a constant λ, which increases as heads, and hence storage volumes, increase and is smaller for reservoirs having lower heads and higher marginal values of their head-storage functions. Hence when increasing system storage for maximum future energy production, one will want to store water first in the reservoir having the smallest λ to increase it until its value equals the value of the next higher λ and then continue filling those two reservoirs keeping their λ values equal until they equal the value of the next higher λ, and so on. Thus, if there is a choice, inflows will be allocated to reservoirs having relatively lower heads and steeper marginal values of their storage-head functions. To meet an energy target, operators should release water so as to decrease the highest values of lambda. This would be water from reservoirs having relatively low marginal values of their storage-head functions at their current storage volumes, relatively low expected inflows, and relatively high heads.

If the system of reservoirs producing hydropower is required to release additional water to satisfy a joint water supply demand in the current period, a similar calculation to the one just described shows that to minimize the loss of future energy production, releases should be made from a reservoir having relatively low heads and relatively low marginal values of their storage head functions.

The target end-of-period storage levels may be infeasible for a given set of beginning-of-period storage levels and reservoir inflows. In that situation, a good heuristic is to try to leave the system as close as possible to the desired conditions. This heuristic usually yields end-of-period storage levels close to the preferred values, although in some situations it may fail to do so.

Acknowledgments. We would like to acknowledge the financial support of Junta Nacional de Investigacao Cientifica, Fundacao Calouste Gulbenkian, and the Fundacao Luso-Arriericana para o Desenvolvimento of Portugal and the comments of G. Kuczera and two anonymous reviewers on an earlier draft of this paper.

References

Amir, R., Optimum operation of a multi-reservoir water supply system, Ph.D. thesis, Standford Univ., Stanford, Calif., 1967.

Antonisse, J., A new interpretation of schema notation that overturns the binary encoding constraint, in *Proceedings of the Third International Conference on Genetic Algorithms*, edited by J. D. Schaffer, Morgan Kaufmann, Los Altos, Calif., 1989.

Arunkumar, S., and W. W.-G. Yeh, Probabilistic models in the design and operation of multi-purpose reservoir system, *Contrib. 144*, Calif. Water Resour. Cent., Univ. of Calif., Davis, 1973.

Bather, J. A., Optimal regulation policies for finite dams, *J. Soc. Ind. Appl. Math.*, *10*(3), 395–423, 1962.

Booker, L. B., Improving search in genetic algorithms, in *Genetic Algorithms and Simulation Annealing*, edited by L. Davis, Morgan Kaufmann, San Francisco, Calif., 1987.

Bower, B. T., M. M. Hufschmidt, and W. W. Reedy, Operating procedures: Their role in the design of water-resource systems by simulation analysis, in *Design of Water-Resource Systems*, edited by A. Maass et al., Harvard Univ. Press, Cambridge, Mass., 1962.

Clark, E. J., Impounding reservoirs, *J. Am. Water Works Assoc.*, *48*(4), 349–354, 1956.

Crawley, P. D., and G. C. Dandy, Optimal operation of multiple-reservoir system, *J. Water Resour. Plann. Manage.*, *119*(1), 1–17, 1993.

Davis, L., Adapting operator probabilities in genetic algorithms, in *Proceedings of the Third International Conference on Genetic Algorithms*, edited by J. D. Schaffer, Morgan Kaufmann, San Francisco, Calif., 1989.

Davis, L. (Ed.), *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1991.

De Jong, K. A., *An Analysis of the Behavior of a Class of Genetic Adaptative Systems*, Ph.D. dissertation, Univ. of Mich., Ann Arbor, 1975.

De Jong, K. A., Genetic algorithms are not functional optimizers, in *Foundations of Genetic Algorithms*, vol. 2, edited by L. D. Whitely, Morgan Kaufmann, San Francisco, Calif., 1993a.

De Jong, K. A., Generations gap revisited, in *Foundations of Genetic Algorithms*, vol. 2, edited by L. D. Whitely, Morgan Kaufmann, San Francisco, Calif., 1993b.

Eshelman, L. J., and J. D. Schaffer, Real-coded genetic algorithms and interval-schemata, in *Foundations of Genetic Algorithms*, vol. 2, edited by L. D. Whitley, Morgan Kaufmann, San Mateo, Calif., 1993.

Forrest, S., Genetic algorithms: Principles of natural selection applied to computation, *Science*, *261*(13), 872–878, 1993.

Georgakakos, A. P., Extended linear quadratic gaussian control: Further extensions, *Water Resour. Res.*, *25*(2), 191–201, 1989.

Gessford, J., and S. Karlin, Optimal policy for hydroelectric operations, in *Studies in the Mathematical Theory of Inventory and Production*, edited by K. J. Arrow, S. Karlin, and H. Scarf, Standford Univ. Press, Standford, Calif., 1958.

Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, 2nd ed., Addison-Wesley, Reading, Mass., 1989.

Goldberg, D. E., Real-coded genetic algorithms, virtual alphabets and blocking, *Complex Syst.*, *5*(1991), 139–167, 1991.

Grefenstette, J. J., Optimization of control parameters for genetic algorithms, *IEEE Trans. Syst. Man Cybern.*, *16*(1), 122–128, 1986.

Holland, J. H., *Adaptation in Natural and Artificial Systems*, 2nd ed., Mass. Inst. of Technol., Cambridge, 1992.

Hydrologic Engineering Center, HEC-3 Reservoir system analysis for conservation, Users manual, U.S. Army Corps of Eng., Davis, Calif., 1981.

Hydrologic Engineering Center, HEC-5 Simulation of flood control and conservation systems, Users manual, U.S. Army Corps of Eng., Davis, Calif., 1989.

Janikow, C. Z., and Z. Michalewicz, An experimental comparison of binary and floating point representations in genetic algorithms, in *Proceedings of the Forth International Conference on Genetic Algorithms*, edited by R. K. Belew and L. Booker, Morgan Kaufmann, San Mateo, Calif., 1991.

Johnson, L. E., An interactive method for development and evaluating of reservoir operating policies, Ph.D. dissertation, Cornell Univ., Ithaca, N. Y., 1981.

Johnson, S. A., J. R. Stedinger, and K. Staschus, Heuristic operating policies for reservoir system simulation, *Water Resour. Res.*, *27*(5), 673–685, 1991.

Loucks, D. P., and O. T. Sigvaldason, Multiple-reservoir operation in North America, in *Operation of Multiple Reservoir Systems, IIASA Collab. Proc. Ser. CP-82-53*, edited by Z. Kaczmarck and J. Kinder, Int. Inst. for Appl. Syst. Anal., Laxenburg, Austria, 1982.

Loucks, D. P., J. R. Stedinger, and D. Haith, *Water Resource Systems Planning and Management*, Prentice-Hall, Englewood Cliffs, N. J., 1981.

Loucks, D. P., P. French, and M. R. Taylor, IRAS—Interactive river-aquifer simulation: Program description and operation, Civ. and Environ. Eng., Cornell Univ., Ithaca, N. Y., 1995.

Martin, Q. W., Optimal reservoir control for hydropower on Colorado River, Texas, *J. Water Resour. Plann. Manage.*, *121*(6), 438–446, 1995.

Massé, P., *Les réserves et la régulation de l'avenir dans la vie economique*, vol. 2, Hermann, Paris, 1946.

McKinney, D. C., and M.-D. Lin, Groundwater optimization using genetic algorithms, *Water Resour. Res.*, *30*(6), 1897–1906, 1994.

McLaughlin, D., and H. L. Velasco, Real-time control of a system of large hydropower reservoirs, *Water Resour. Res.*, *26*(4), 623–635, 1990.

Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolutionary Programs*, Springer-Verlag, New York, 1992.

Michalewicz, Z., and C. Z. Janikow, Handling constraints in genetic algorithms, in *Proceedings of the Fourth International Conference on Genetic Algorithms*, edited by R. K. Belew and L. Booker, Morgan Kaufmann, San Francisco, Calif., 1991.

Mizyed, N. R., J. C. Loftis, and D. G. Fontane, Operation of large multireservoir systems using optimal-control theory, *J. Water Resour. Plann. Manage.*, *118*(4), 371–387, 1992.

Oliveira, R., Operating rules for multi-reservoir systems, Ph.D. dissertation, 327 pp., Cornell Univ., Ithaca, N. Y., 1994.

Perera, B. J. C., and G. P. Codner, Derivation of optimal target storage curves for an urban water supply reservoir system, paper presented at Hydrology and Water Resources Symposium, Inst. of Hydrol. and Water Resour. Grifith Univ., Brisbane, Australia, Nov. 25–27, 1986.

Radcliffe, N. J., Equivalence class analysis of genetic algorithms, *Complex Syst.*, *5*(1991), 183–205, 1991.

Sand, M. G., An analytical investigation of operating policies for water supply reservoirs in parallel, Ph.D. thesis, Cornell Univ., Ithaca, N. Y., 1984.

Schaffer, J. D., R. A. Caruana, L. J. Eshelman, and R. Das, A study of control parameters affecting online performance of genetic algorithms for function optimization, in *Proceedings of the Third International Conference on Genetic Algorithms*, edited by J. D. Schaffer, Morgan Kaufmann, San Mateo, Calif., 1989.

Simpson, A. R., G. C. Dandy, and L. J. Murphy, Genetic algorithms compared with other techniques for pipe optimization, *J. Water Resour. Plann. Manage.*, *120*(4), 1994.

Sobel, M. J., Reservoir management models, *Water Resour. Res.*, *11*(6), 767–776, 1975.

Syswerda, G., Uniform crossover in genetic algorithms, in *Proceedings of the Third International Conference on Genetic Algorithms*, edited by J. D. Schaffer, Morgan Kaufmann, San Mateo, Calif., 1989.

Verwey, A., et al., *Hydroinformatics '94*, A. A. Balkema, Rotterdam, Netherlands, 1994.

Wang, Q. J., The genetic algorithm and its application to calibrating conceptual rainfall-runoff models, *Water Resour. Res.*, *27*(9), 2467–2471, 1991.

Wilson, S. W., Classifier systems learning of a boolean function, *Res. Memo. RIS-26r*, Rowland Inst. for Sci., Cambridge, Mass., 1986.

Witley, D., The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best, in *Proceedings of the Third International Conference on Genetic Algorithms*, edited by J. D. Schaffer, Morgan Kaufmann, San Francisco, Calif., 1989.

Wright, A., Genetic algorithms for real parameter optimization, in *Foundations of Genetic Algorithms*, edited by G. J. E. Rawlings, Morgan Kaufmann, San Mateo, Calif., 1991.

Wu, R.-H., Derivation of balancing curves for multiple reservoir operation, M.Sc. thesis, Cornell Univ., Ithaca, N. Y., 1988.

Wurbs, R. A., *Water Management Models: A Guide to Software*, Prentice Hall, Englewood Cliffs, N. J., 1995.

Wurbs, R. A., *Modeling and Analysis of Reservoir Systems Operations*, Prentice Hall, Englewood Cliffs, N. J., 1996.

Yakowitz, S., Dynamic programming applications in water resources, *Water Resour. Res.*, *18*(4), 673–696, 1982.

Yang, X., et al., Comparison of real-time reservoir operation techniques, *J. Water Resour. Plann. Manage.*, *121*(5), 345–351, 1995.

Yeh, W. W.-G., Reservoir management and operation models: A state-of-the-art review, *Water Resour. Res.*, *21*(12), 1797–1818, 1985.

D. P. Loucks, School of Civil and Environmental Engineering, Cornell University, Hollister Hall, Ithaca, NY 14853. (e-mail: DPL3@cornell.edu)

R. Oliveira, Departamento de Hidráulica, Laboratório Nacional de Engenharia Civil, P-1799 Lisboa Codex, Portugal.

(Received September 5, 1996; revised November 23, 1996; accepted December 3, 1996.)