# A Multicriteria Simulation-Optimization Algorithm for Generating Sets of Alternatives Using Population-Based Metaheuristics

JULIAN SCOTT YEOMANS
OMIS Area, Schulich School of Business
York University
4700 Keele Street, Toronto, ON, M3J 1P3
CANADA
syeomans@schulich.yorku.ca

*Abstract:* - Stochastic optimization problems are often overwhelmed with inconsistent performance requirements and incompatible performance specifications that can be difficult to detect during problem formulation. Therefore, it can prove beneficial to create a set of dissimilar options that provide divergent perspectives to the problem. These alternatives should be near-optimal with respect to the specified objective(s), but be maximally different from each other in the decision region. The approach for creating maximally different sets of solutions is referred to as modelling-to-generate-alternatives (MGA). Simulation-optimization approaches are commonly employed to solve computationally difficult problems containing significant stochastic uncertainties. This paper provides a new, stochastic, multicriteria MGA approach that can generate sets of maximally different alternatives for any simulation-optimization method that employs a population-based algorithm.

*Key-Words:* - Multicriteria Objectives, Population-based algorithms, Modelling-to-generate-alternatives

## 1 Introduction

Stochastic decision-making problems frequently include inconsistent and incompatible design specifications that can be difficult to formulate into mathematical decision-models [1], [2], [3], [4], [5]. Although "optimal" solutions can be determined for the mathematical models, they generally do not provide the best solution to the "real" problem as there are usually unmodeled components not apparent when the mathematical models are formulated [1], [2], [6]. Generally, it is better to construct a small number of distinct alternatives that provide complementary perspectives to the particular problem [3], [7]. These dissimilar solutions should be near-optimal with respect to the specified objective(s), but be maximally different from each other within the solution domain. The approach for creating maximally different sets of solutions is referred to as modelling-to-generate-alternatives (MGA) [6], [7], [8]. The primary impetus behind MGA is to create a set of alternatives that are "good" when measured within the modelled objective space but as different as possible from each other in the decision space. Decision-makers must undertake a subsequent evaluation of this set to determine which specific alternative(s) most closely satisfy their specific goals. Consequently, unlike the more straightforward style of explicit solution determination inherent in most "hard" optimization approaches, MGA approaches are necessarily classified into the decision support realm.

Early MGA algorithms employed direct, incremental approaches for constructing their alternatives by re-running their algorithms incrementally whenever a new solution had to be created [6], [7], [8], [9], [10]. These iterative approaches imitated the seminal MGA method of Brill *et al*. [8] where, after the initial mathematical model had been optimized, all supplementary alternatives were produced one-at-a-time. These incremental approaches all required $n$+1 iterations of their respective algorithms – initially to optimize the original problem, then to produce each of the subsequent $n$ alternatives [7], [11], [12], [13], [14], [15], [16], [17], [18]). Subsequently, it was demonstrated how a set of maximally different alternatives could be efficiently generated using *any* population-based algorithm by permitting the generation of the overall optimal solution together with $n$ distinct alternatives in a single computational run irrespective of the value of $n$ [19], [20], [21], [22], [23]. In [24] a new data structure was created that permits *simultaneous* alternatives to be constructed by population-based solution methods and this was incorporated into a bicriterion procedure in [25]. In [26] it was shown how a set of maximally different solution alternatives could be generated by extending several earlier MGA techniques into stochastic optimization.

In this study, a new multicriteria, objective is created that combines the data structure into the simultaneous solution approach to create a new stochastic MGA algorithm. The max-sum components of the objective produce a maximum distance between alternatives by ensuring that the total deviation between all of the variables in all of the alternatives is collectively large. This does not, however, preclude the occurrence of relatively small (or zero) deviations between certain individual variables within certain solutions. In contrast, max-min objectives force a maximum distance between every variable over all solutions. By considering the multiple objectives simultaneously, the alternatives created can be forced as far apart as possible for all variables in general and the closest distance in the worst case between any solutions will never be less than the value obtained for the max-min objective. This stochastic MGA algorithmic approach proves to be extremely computationally efficient.

## 2 Modelling to Generate Alternatives

Mathematical optimization has focused almost entirely on generating single optimal solutions to single-objective problems or producing sets of noninferior solutions to multi-objective constructions [2], [5], [8]. While such conventions may create solutions to the derived mathematical formulations, whether these outputs are the best solutions to the "real" problems remains debatable [1], [2], [6], [8]. With many complex, "real world" decision situations, there are numerous system specifications that can never be incorporated into the problem formulation [1], [5]. Moreover, it may not be possible to explicitly account for all of the subjective requirements as there are frequently numerous adversarial stakeholders and incompatible design components to address. Thus, most subjective features unavoidably remain unquantified and unmodelled in the mathematical decision models. This regularly occurs when conclusions are made based not only upon modelled objectives, but also upon more incongruent stakeholder preferences and socio-political-economic goals [7]. Several incongruent modelling dualities are illustrated in [6], [8], [9], and [10].

When unmodelled objectives and unquantified issues exist, non-conventional methods are needed to not only search the decision region for noninferior sets of solutions, but to also explore the decision region for alternatives that are clearly *sub-optimal* to the modelled problem. Namely, any search for alternatives to problems known or suspected to contain unmodelled components must concentrate not only on a non-inferior set of solutions, but also necessarily on an explicit exploration of the problem's inferior solution space.

To demonstrate the implications of unmodelled objectives in a decision search, assume that an optimal solution for a maximization problem is $\boldsymbol{X}^*$ with objective value $Z1^*$ [24]. Suppose a second, unquantified, maximization objective $Z2$ exists that represents some "politically acceptable" factor. Assume that the solution, $\boldsymbol{X}^a$, belonging to the 2-objective noninferior set, exists that corresponds to a best compromise solution if both objectives could have been simultaneously considered. Although $\boldsymbol{X}^a$ would be considered as the best solution to the real problem, in the actual mathematical model it would appear inferior to solution $\boldsymbol{X}^*$, since $Z1^a \leq Z1^*$. Therefore, when unquantified components are included in the decision-making process, inferior decisions to the mathematically modelled problem could be optimal to the underlying "real" problem. Thus, when unquantified issues and unmodelled objectives could exist, alternative solution procedures are required to not only explore the decision domain for noninferior solutions to the modelled problem, but also to concurrently search the decision domain for inferior solutions. Population-based algorithms permit concurrent searches throughout a decision space and prove to be particularly proficient solution methods.

The primary task of MGA is to create a workable set of options that are quantifiably good with respect to all modelled objectives, yet are as different as possible from each other within the solution space. By accomplishing this requirement, the resulting set of alternatives is able to provide truly different perspectives that perform similarly with respect to the known modelled objective(s) yet very differently with respect to various potentially unmodelled aspects. By creating these good-but-different solutions, the decision-makers are able to explore potentially desirable qualities within the alternatives that might be able to satisfy the unmodelled objectives to varying degrees of stakeholder acceptability.

To motivate the MGA process, it is necessary to more formally characterize the mathematical definition of its goals [6], [7]. Assume that the optimal solution to an original mathematical model is $\boldsymbol{X^*}$ with corresponding objective value $\boldsymbol{Z^*} = F(\boldsymbol{X^*})$. The resultant difference model can then be solved to produce an alternative solution, $\boldsymbol{X}$, that is maximally different from $\boldsymbol{X}^*$:

$$\text{Maximize} \quad \Delta(\boldsymbol{X}, \boldsymbol{X^*}) = \underset{i}{\text{Min}} \; | X_i - X_i^* | \quad (1)$$

$$\text{Subject to:} \quad \boldsymbol{X} \in D \quad (2)$$

$$| F(\boldsymbol{X}) - \boldsymbol{Z^*} | \leq T \quad (3)$$

where $\Delta$ represents an appropriate difference function (shown in (1) as an absolute difference) and $T$ is a tolerance target relative to the original optimal objective value $\boldsymbol{Z^*}$. $T$ is a user-specified limit that determines what proportion of the inferior region needs to be explored for acceptable alternatives. This difference function concept can be extended into a difference measure between any *set of alternatives* by replacing $\boldsymbol{X}^*$ in the objective of the maximal difference model and calculating the overall minimum absolute difference (or some other function) of the pairwise comparisons between corresponding variables in each pair of alternatives – subject to the condition that each alternative is feasible and falls within the specified tolerance constraint.

The population-based MGA procedure to be introduced is designed to generate a pre-determined small number of close-to-optimal, but maximally different alternatives, by adjusting the value of $T$ and solving the corresponding maximal difference problem instance by exploiting the population structure of the algorithm. The survival of solutions depends upon how well the solutions perform with respect to the problem's originally modelled objective(s) and simultaneously by how far away they are from all of the other alternatives generated in the decision space.

## 3 Simulation-Optimization for Stochastic Optimization

Finding optimal solutions to large stochastic problems proves complicated when numerous system uncertainties must be directly incorporated into the solution procedures ([27], [28], [29], [30]). *Simulation-Optimization* (SO) is a broadly defined family of stochastic solution approaches that combines simulation with an underlying optimization component for optimization ([27]). In SO, all unknown objective functions, constraints, and parameters are replaced by simulation models in which the decision variables provide the settings under which simulation is performed.

The general steps of SO can be summarized in the following fashion ([28], [31]). Suppose the mathematical model of the optimization problem contains $n$ decision variables, $X_i$, represented in the vector $\boldsymbol{X}$ = $[X_1, X_2, \ldots, X_n]$. If the objective function is expressed by $F$ and the feasible region is designated by $D$, then the mathematical programming problem is to optimize $F(\boldsymbol{X})$ subject to $\boldsymbol{X} \in D$. When stochastic conditions exist, values for the objective and constraints can be determined by simulation. Any solution comparison between two different solutions $\boldsymbol{X1}$ and $\boldsymbol{X2}$ requires the evaluation of some statistic of $F$ modelled with $\boldsymbol{X1}$ compared to the same statistic modelled with $\boldsymbol{X2}$ ([27], [32]). These statistics are calculated by simulation, in which each $\boldsymbol{X}$ provides the decision variable settings employed in the simulation. While simulation provides a means for comparing results, it does not provide the mechanism for determining optimal solutions to problems. Hence, simulation cannot be used independently for stochastic optimization.

Since all measures of system performance in SO are stochastic, every potential solution, $\boldsymbol{X}$, must be calculated through simulation. Because simulation is computationally intensive, an optimization algorithm is employed to guide the search for solutions through the problem's feasible domain in as few simulation runs as possible ([29], [32]). As stochastic system problems frequently contain numerous potential solutions, the quality of the final solution could be highly variable unless an extensive search has been performed throughout the entire feasible region. A stochastic SO approach contains two alternating computational phases; (i) an "evolutionary" module directed by some optimization (frequently a metaheuristic) method and (ii) a simulation module ([33]). Because of the stochastic components, all performance measures are necessarily statistics calculated from the responses generated in the simulation module. The quality of each solution is found by having its performance criterion, $F$, evaluated in the simulation module. After simulating each candidate solution, their respective objective values are returned to the evolutionary module to be utilized in the creation of ensuing candidate solutions. Thus, the evolutionary module aims to advance the system toward improved solutions in subsequent generations and ensures that the solution search does not become trapped in some local optima. After generating new candidate solutions in the evolutionary module, the new solution set is returned to the simulation module for comparative evaluation. This alternating, two-phase search process terminates when an appropriately stable system state (i.e. an optimal solution) has been attained. The optimal solution produced by the procedure is the single best solution found throughout the course of the entire search process ([33]).

Population-based algorithms are conducive to SO searches because the complete set of candidate solutions maintained in their populations permit searches to be undertaken throughout multiple sections of the feasible region, concurrently. For population-based optimization methods, the evolutionary phase evaluates the entire current population of solutions during each generation of the search and evolves from a current population to a subsequent one. A primary characteristic of population-based procedures is that better solutions in a current population possess a greater likelihood for survival and progression into the subsequent population.

It has been shown that SO can be used as a very computationally intensive, stochastic MGA technique ([32], [34]). However, because of the very long computational runs, several approaches to accelerate the search times and solution quality of SO have been examined subsequently [31]. The next section provides an MGA algorithm that incorporates stochastic uncertainty using SO to much more efficiently generate sets of maximally different solution alternatives.

# 4 Population-based, Multicriteria MGA Algorithm

In this section, a data structure is introduced that permits a multicriteria MGA solution approach via any population-based algorithm [24]. Suppose that it is desired to be able to produce $P$ alternatives that each possess $n$ decision variables and that the population algorithm is to possess $K$ solutions in total. That is, each solution is to contain one set of $P$ maximally different alternatives. In this representation, let $\boldsymbol{Y}_k$, $k = 1,\ldots, K$, represent the $k^{th}$ solution which is made up of one complete set of $P$ different alternatives. Namely, if $\boldsymbol{X}_{kp}$ is the $p^{th}$ alternative, $p = 1,\ldots, P$, of solution $k$, $k = 1,\ldots, K$, then $\boldsymbol{Y}_k$ can be represented as

$$\boldsymbol{Y}_k = [\boldsymbol{X}_{k1}, \boldsymbol{X}_{k2},\ldots, \boldsymbol{X}_{kP}] . \quad (4)$$

If $X_{kjq}$, $q = 1,\ldots, n$, is the $q^{th}$ variable in the $j^{th}$ alternative of solution k, then

$$\boldsymbol{X}_{kj} = (X_{kj1}, X_{kj2},\ldots, X_{kjn}) . \quad (5)$$

Consequently, an entire population, $\boldsymbol{Y}$, comprised of $K$ different sets of $P$ alternatives can be written in vector form as,

$$\boldsymbol{Y}' = [\boldsymbol{Y}_1, \boldsymbol{Y}_2,\ldots, \boldsymbol{Y}_K] . \quad (6)$$

The following population-based MGA method produces a pre-determined number of close-to-optimal, but maximally different alternatives, by modifying the value of the bound $T$ in the maximal difference model and using any population-based metaheuristic to solve the corresponding, maximal difference problem. The multicriteria MGA algorithm that follows constructs a pre-determined number of maximally different, near-optimal alternatives, by modifying the bound value $T$ in the maximal difference model and using any population-based technique to solve the corresponding maximal difference problem. Each solution in the population comprises one set of $p$ different alternatives. By exploiting the co-evolutionary aspects within the metaheuristic, the algorithm collectively evolves each solution toward sets of different local optima within the solution space. In this process, each desired solution alternative undergoes the common search procedure of the metaheuristic. However, the survival of solutions depends upon both how well the solutions perform with respect to the modelled objective(s) and by how far away they are from all of the other alternatives generated in the decision space.

A straightforward process for generating alternatives solves the maximum difference model iteratively by incrementally updating the target $T$ whenever a new alternative needs to be produced and then re-solving the resulting model [24]. This iterative approach parallels the original Hop, Skip, and Jump (HSJ) MGA algorithm of Brill *et al*. [8] in which the alternatives are created one-by-one through an incremental adjustment of the target constraint. While this approach is straightforward, it entails a repetitive execution of the optimization algorithm [7], [12], [13]. To improve upon the stepwise HSJ approach, a concurrent MGA technique was subsequently designed based upon co-evolution ([13], [15], [17]). In a co-evolutionary approach, pre-specified stratified subpopulation ranges within an algorithm's overall population are established that collectively evolve the search toward the specified number of maximally different alternatives. Each desired solution alternative is represented by each respective subpopulation and each subpopulation undergoes the common processing operations of the procedure. The survival of solutions in each subpopulation depends simultaneously upon how well the solutions perform with respect to the modelled objective(s) and by how far away they are from all of the other alternatives. Consequently, the evolution of solutions in each subpopulation toward local optima is directly influenced by those solutions contained in all of the other subpopulations, which forces the concurrent co-evolution of each subpopulation towards good but maximally distant regions within the decision space according to the maximal difference model [7]. Co-evolution is also much more efficient than a sequential HSJ-style approach in that it exploits the inherent population-based searches to concurrently generate the entire set of maximally different solutions using only a single population [12], [17].

While concurrent approaches can exploit population-based algorithms, co-evolution can only occur within each of the stratified subpopulations. Consequently, the maximal differences between solutions in different subpopulations can only be based upon aggregate subpopulation measures. Conversely, in the following simultaneous MGA algorithm, each solution in the population contains exactly one entire set of alternatives and the maximal difference is calculated only for that particular solution (i.e. the specific alternative set contained within that solution in the population). Hence, by the evolutionary nature of the population-based search procedure, in the subsequent approach, the maximal difference is simultaneously calculated for the specific set of alternatives considered within each specific solution – and the need for concurrent subpopulation aggregation measures is avoided.

Using the data structure terminology, the steps for the stochastic multicriteria MGA algorithm are as follows ([14], [19], [20], [21], [22], [23], [24]). It should be readily apparent that the stratification approach employed by this method can be easily modified for any population-based algorithm.

*Preliminary Step*. In this initialization step, solve the original optimization problem to determine the optimal solution, $\boldsymbol{X^*}$. Based upon the objective value $F(\boldsymbol{X^*})$, establish $P$ target values. $P$ represents the desired number of maximally different alternatives to be generated within prescribed target deviations from the $\boldsymbol{X^*}$. Note: The value for $P$ has to have been set *a priori* by the decision-maker.

Without loss of generality, it is possible to forego this step and to use the algorithm to find $\boldsymbol{X^*}$ as part of its solution processing in the subsequent steps. However, this significantly increases the number of iterations of the computational procedure and the initial stages of the processing become devoted to finding $\boldsymbol{X^*}$ while the other elements of each population solution are retained as essentially "computational overhead".

*Step 1*. Create the initial population of size $K$ in which each solution is divided into $P$ equally-sized partitions. The size of each partition corresponds to the number of variables for the original optimization problem. $\boldsymbol{X_{kp}}$ represents the $p^{th}$ alternative, $p = 1,\ldots,P$, in solution $\boldsymbol{Y_k}$, $k = 1,\ldots,K$.

*Step 2*. In each of the $K$ solutions, evaluate each $\boldsymbol{X_{kp}}$, $p = 1,\ldots,P$, using the simulation module with respect to the modelled objective. Alternatives meeting their target constraint and all other problem constraints are designated as *feasible*, while all other alternatives are designated as *infeasible*. A solution can only be designated as feasible if all of the alternatives contained within it are feasible.

*Step 3*. Apply an appropriate elitism operator to each solution to rank order the best individuals in the population. The best solution is the feasible solution containing the most distant set of alternatives in the decision space (the distance measures are defined in Step 5).

Note: Because the best-solution-to-date is always retained in the population throughout each iteration, at least one solution will always be feasible. Furthermore, a feasible solution based on the initialization step can be constructed using $P$ repetitions of $\boldsymbol{X^*}$.

*Step 4*. Stop the algorithm if the termination criteria (such as maximum number of iterations or some measure of solution convergence) are met. Otherwise, proceed to Step 5.

*Step 5*. For each solution $\boldsymbol{Y_k}$, $k = 1,\ldots, K$, calculate $R$ Max-Min and/or Max-Sum distance measures, $D^r_{k,}$ $r = 1,\ldots, R$, between all of the alternatives contained within the solution.

As an illustrative example for calculating the multicriteria distance measures, compute :

$$D^1_k = \Delta^1(\boldsymbol{X_{ka}}, \boldsymbol{X_{kb}}) = \min_{a,b,q} |X_{kaq} - X_{kbq}|,$$

$$a = 1,\ldots,P,\ b = 1,\ldots,P,\ q = 1,\ldots,n, \quad (7)$$

$$D^2_k = \Delta^2(\boldsymbol{X_{ka}}, \boldsymbol{X_{kb}}) = \sum_{a=1 to P} \sum_{b=1 to P} \sum_{q=1\ldots n} |X_{kaq} - X_{kbq}|. \quad (8)$$

and

$$D^3_k = \Delta^3(\boldsymbol{X_{ka}}, \boldsymbol{X_{kb}}) = \sum_{a=1 to P} \sum_{b=1 to P} \sum_{q=1\ldots n} (X_{kaq} - X_{kbq})^2. \quad (9)$$

$D^1_k$ denotes the minimum absolute distance, $D^2_k$ represents the overall absolute deviation, and $D^3_k$ determines the overall quadratic deviation between all of the alternatives contained within solution $k$.

Alternatively, the distance functions could be calculated using some other appropriately defined measures.

*Step 6*. Let $D_k = G(D^1_k, D^2_k, D^3_k,\ldots, D^R_k)$ represent the multicriteria objective for solution $k$. Rank the solutions according to the distance measure $D_k$ objective – appropriately adjusted to incorporate any constraint violation penalties for infeasible solutions. The goal of maximal difference is to force alternatives to be as far apart as possible in the decision space from the alternatives of each of the partitions within each solution This step orders the specific solutions by those solutions which contain the set of alternatives which are most distant from each other.

*Step 7*. Apply appropriate metaheuristic “change operations” to each solution within the population and return to Step 2.

## 5 Conclusion

Complex problem solving inherently involves complicated performance components that are confounded by unquantifiable requirements and incongruent performance objectives. These decision environments frequently contain incompatible design specifications that are problematic – if not impossible – to incorporate when ancillary decision support models are constructed. Invariably, there are unmodelled elements, not apparent during model formulation, that can significantly affect the adequacy of its solutions. These confounding features require the decision-makers to integrate numerous uncertainties into their solution process before an ultimate solution can be determined. Faced with such inconsistencies, it is unlikely that any single solution can simultaneously satisfy all ambiguous system requirements without significant compromises. Therefore, any decision support approach must somehow address these complicating facets in some way, while simultaneously being flexible enough to condense the potential effects within the intrinsic planning incongruities.

This paper has provided a new stochastic multicriteria approach and an updated MGA procedure that directs stochastic SO search processes. This new computationally efficient MGA method establishes how population-based algorithms can simultaneously construct entire sets of close-to-optimal, maximally different alternatives by exploiting the evolutionary characteristics of any population-based solution approach. In this MGA role, the multicriteria objective can efficiently generate the requisite set of dissimilar alternatives, with each generated solution providing an entirely different outlook to the problem. The max-sum criteria ensure that the distances between the alternatives created by this algorithm are good in general, while the max-min criteria ensure that the distances between the alternatives are good in the worst case. Since population-based procedures can be applied to a wide range of problem types, the practicality of this stochastic multicriteria MGA approach can be extended to wide range of “real world” planning situations. Such extensions will be examined in future research.

*References:*

[1] M. Brugnach, A. Tagg, F. Keil, and W.J. De Lange, Uncertainty matters: computer models at the science-policy interface, *Water Resources Management*, Vol. 21, 2007, pp. 1075-1090.

[2] J.A.E.B. Janssen, M.S. Krol, R.M.J. Schielen, and A.Y. Hoekstra, The effect of modelling quantified expert knowledge and uncertainty information on model-based decision making, *Environmental Science and Policy*, Vol. 13, No. 3, 2010, pp. 229-238.

[3] M. Matthies, C. Giupponi, and B. Ostendorf, Environmental decision support systems: Current issues, methods and tools, *Environmental Modelling and Software*, Vol. 22, No. 2, 2007, pp. 123-127.

[4] H.T. Mowrer, Uncertainty in natural resource decision support systems: Sources, interpretation, and importance, *Computers and Electronics in Agriculture*, Vol. 27, No. 1-3, 2000, pp. 139-154.

[5] W.E. Walker, P. Harremoes, J. Rotmans, J.P. Van der Sluis, M.B.A.P. Van Asselt, Janssen, and M.P. Krayer von Krauss, Defining uncertainty – a conceptual basis for uncertainty management in model-based decision support, *Integrated Assessment*, Vol. 4, No. 1, 2003, pp. 5-17.

[6] D.H. Loughlin, S.R. Ranjithan, E.D. Brill, and J.W. Baugh, Genetic algorithm approaches for addressing unmodelled objectives in optimization problems, *Engineering Optimization*, Vol. 33, No. 5, 2001, pp. 549-569.

[7] J.S. Yeomans, and Y. Gunalay, Simulation-optimization techniques for modelling to generate alternatives in waste management planning, *Journal of Applied Operational Research*, Vol. 3, No. 1, 2011, pp. 23-35.

[8] E.D. Brill, S.Y. Chang, and L.D. Hopkins, Modelling to generate alternatives: the HSJ approach and an illustration using a problem in land use planning, *Management Science*, Vol. 28, No. 3, 1982, pp. 221-235.

[9] J.W. Baugh, S.C. Caldwell, and E.D. Brill, A mathematical programming approach for generating alternatives in discrete structural optimization, *Engineering Optimization*, Vol. 28, No. 1, 1997, pp. 1-31.

[10] E.M. Zechman, and S.R. Ranjithan, Generating alternatives using evolutionary algorithms for water resources and environmental management problems, *Journal of Water*

*Resources Planning and Management*, Vol. 133, No. 2, 2007, pp. 156-165.

[11] Y. Gunalay, J.S. Yeomans, and G.H. Huang, Modelling to generate alternative policies in highly uncertain environments: An application to municipal solid waste management planning, *Journal of Environmental Informatics*, Vol. 19, No. 2, 2012, pp. 58-69.

[12] R. Imanirad, and J.S. Yeomans, Modelling to Generate Alternatives Using Biologically Inspired Algorithms, in X.S. Yang (Ed.), *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications*, Elsevier, Amsterdam, 2013, pp. 313-333.

[13] R. Imanirad, X.S. Yang, and J.S. Yeomans, A computationally efficient, biologically-inspired modelling-to-generate-alternatives method, *Journal on Computing*, Vol. 2, No. 2, 2012, pp. 43-47.

[14] J.S. Yeomans, An Efficient Computational Procedure for Simultaneously Generating Alternatives to an Optimal Solution Using the Firefly Algorithm, in X.S. Yang (Ed.), *Nature-Inspired Algorithms and Applied Optimization*, Springer, New York, 2018, pp. 261-273.

[15] R. Imanirad, X.S. Yang, and J.S. Yeomans, A co-evolutionary, nature-inspired algorithm for the concurrent generation of alternatives, *Journal on Computing*, Vol. 2, No. 3, 2012, pp. 101-106.

[16] R. Imanirad, X.S. Yang, and J.S. Yeomans, Modelling-to-generate-alternatives via the firefly algorithm, *Journal of Applied Operational Research*, Vol. 5, No. 1, 2013, pp. 14-21.

[17] R. Imanirad, X.S. Yang, and J.S. Yeomans, A Concurrent Modelling to Generate Alternatives Approach Using the Firefly Algorithm, *International Journal of Decision Support System Technology*, Vol. 5, No. 2, 2013, pp. 33-45.

[18] R. Imanirad, X.S. Yang, and J.S. Yeomans, A biologically-inspired metaheuristic procedure for modelling-to-generate-alternatives, *International Journal of Engineering Research and Applications*, Vol. 3, No. 2, 2013, pp. 1677-1686.

[19] J.S. Yeomans, Simultaneous Computing of Sets of Maximally Different Alternatives to Optimal Solutions, *International Journal of Engineering Research and Applications*, Vol. 7, No. 9, 2017, pp. 21-28.

[20] J.S. Yeomans, An Optimization Algorithm that Simultaneously Calculates Maximally Different Alternatives, *International Journal of Computational Engineering Research*, Vol. 7, No. 10, 2017, pp. 45-50.

[21] J.S. Yeomans, Computationally Testing the Efficacy of a Modelling-to-Generate-Alternatives Procedure for Simultaneously Creating Solutions, *Journal of Computer Engineering*, Vol. 20, No. 1, 2018, pp. 38-45.

[22] J.S. Yeomans, A Computational Algorithm for Creating Alternatives to Optimal Solutions, *Transactions on Machine Learning and Artificial Intelligence*, Vol. 5, No. 5, 2017, pp. 58-68.

[23] J.S. Yeomans, A Simultaneous Modelling-to-Generate-Alternatives Procedure Employing the Firefly Algorithm, in N. Dey (Ed.), *Technological Innovations in Knowledge Management and Decision Support*, IGI Global, Hershey, Pennsylvania, 2019, pp. 19-33.

[24] J.S. Yeomans, An Algorithm for Generating Sets of Maximally Different Alternatives Using Population-Based Metaheuristic Procedures, *Transactions on Machine Learning and Artificial Intelligence*, Vol. 6, No. 5, 2018, pp. 1-9.

[25] J.S. Yeomans, A Bicriterion Approach for Generating Alternatives Using Population-Based Algorithms, *WSEAS Transactions on Systems*, Vol. 18, No. 4, 2019, pp. 29-34.

[26] J.S. Yeomans, A Simulation-Optimization Algorithm for Generating Sets of Alternatives Using Population-Based Metaheuristic Procedures, *Journal of Software Engineering and Simulation*, forthcoming.

[27] M.C. Fu, Optimization for Simulation: Theory vs. Practice, *INFORMS Journal on Computing*, Vol. 14, No. 3, 2002, pp. 192-215.

[28] P. Kelly, Simulation Optimization is Evolving, *INFORMS Journal on Computing*, Vol. 14, No. 3, 2002, pp. 223-225.

[29] R. Zou, Y. Liu, J. Riverson, A. Parker, and S. Carter, A Nonlinearity Interval Mapping Scheme for Efficient Waste Allocation Simulation-Optimization Analysis, *Water Resources Research*, Vol. 46, No. 8, 2010, pp. 1-14.

[30] R. Imanirad, X.S. Yang, and J.S. Yeomans, Stochastic Decision-Making in Waste Management Using a Firefly Algorithm-Driven Simulation-Optimization Approach for Generating Alternatives, in S. Koziel, L. Leifsson, and X.S. Yang (Eds.), *Recent Advances in Simulation-Driven Modeling and Optimization*, Springer, Heidelberg, Germany, 2016, pp. 299-323.

[31] J.S. Yeomans, Waste Management Facility Expansion Planning Using Simulation-Optimization with Grey Programming and Penalty Functions, *International Journal of Environmental and Waste Management*, Vol. 10, No. 2/3, 2012, pp. 269-283.

[32] J.S. Yeomans, Applications of Simulation-Optimization Methods in Environmental Policy Planning Under Uncertainty, *Journal of Environmental Informatics*, Vol. 12, No. 2, 2008, pp. 174-186.

[33] J.S. Yeomans, and X.S. Yang, Municipal Waste Management Optimization Using a Firefly Algorithm-Driven Simulation-Optimization Approach, *International Journal of Process Management and Benchmarking*, Vol. 4, No. 4, 2014, pp. 363-375.

[34] J.D. Linton, J.S. Yeomans, and R. Yoogalingam, Policy Planning Using Genetic Algorithms Combined with Simulation: The Case of Municipal Solid Waste, *Environment and Planning B: Planning and Design*, Vol. 29, No. 5, 2002, pp. 757-778.