IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 20, Issue 1, Ver. III (Jan.- Feb. 2018), PP 38-45
www.iosrjournals.org

Computationally Testing the Efficacy of a Modelling-to-Generate-Alternatives Procedure for Simultaneously Creating Solutions

Julian Scott Yeomans[1]
[1](*OMIS Area, Schulich School of Business, York University, Canada*)

Abstract: *"Real world" applications tend to contain complex performance specifications riddled with contradictory performance elements. This state arises because policymaking naturally involves multifaceted problems that are riddled with competing performance objectives and contain incompatible design requirements which are very problematic – if not impossible – to capture at the time that the requisite decision models are constructed. There are invariably unmodelled components, not readily apparent during model formulation, which could greatly impact the suitability of the model's solutions. Consequently, it proves preferable to generate a number of dissimilar alternatives that provide multiple, distinct perspectives to the problem. These different options should all possess close-to-optimal measures with respect to the specified objective(s), but be maximally different from each other in the decision space. These maximally different solution construction approaches have been referred to as modelling-to-generate-alternatives (MGA). This study provides a procedure that simultaneously generates multiple, maximally different alternatives by employing the metaheuristic, Firefly Algorithm. The efficacy of this efficient algorithmic optimization approach is demonstrated on a commonly-tested engineering benchmark problem.*

Keywords- *Firefly Algorithm, Modelling-to-generate-alternatives, Nature-inspired Metaheuristic Algorithms*

Date of Submission: 02-02-2018 Date of acceptance: 19-02-2018

I. Introduction

"Real world" decision-making tends to characterized by complex problems with design requirements that are very difficult to integrate into a corresponding mathematical programming formulation and are often inundated by numerous unquantifiable aspects components [1], [2], [3], [4], [5]. While "optimal" solutions can be determined for the mathematically modelled formulations, whether these are the best solutions to the underlying real problems can be called into question, as there are invariably unmodelled aspects not apparent during the construction of the mathematical model [1], [2], [6]. In general, it is considered more desirable to be able to produce a judicious number of distinct options that provide multiple, contrasting perspectives to the specified problem [3], [7]. These different options should all possess close-to-optimal measures with respect to the specified objective(s), but be maximally different from each other in the decision space. Several techniques collectively referred to as *modelling-to-generate-alternatives* (MGA) have been established in response to this multi-solution requirement [6], [7], [8].

The prime motivation behind MGA is the creation of a small set of alternatives that are good as measured by all of the objective(s), but are fundamentally different from each other within the decision domain. The set of resulting solutions should provide numerous alternatives that all perform similarly with regard to the modelled objectives, yet very differently with regard to any unmodelled issues [5]. The decision-makers must subsequently conduct a comprehensive evaluation of these alternatives to establish which alternative(s) would most closely satisfy their specific circumstances. Consequently, MGA methods should necessarily be classified as a decision support processes rather than as explicit solution determination procedures assumed in optimization.

Prior MGA procedures have used direct, iterative methods for generating alternatives by incrementally re-running their solution algorithms whenever new alternatives must be constructed [6], [7], [8], [9], [10]. These iterative methods mimic the seminal MGA approach of Brill *et al.* [8] where, once the initial mathematical formulation has been optimized, all supplementary alternatives are produced one-at-a-time. Consequently, these incremental approaches all require n+1 iterations of their respective algorithms – first to optimize the initial problem, then to create the subsequent n alternatives [7], [11], [12], [13].

For optimization, Yang [14], [15] has shown that the biologically-inspired Firefly Algorithm (FA) is more efficient computationally than other commonly-used metaheuristic procedures such as simulated annealing, genetic algorithms, and enhanced particle swarm optimization [16], [17]. However, what differentiates the FA from other population-based metaheuristics for functional optimization purposes, is that it has been designed to converge simultaneously into a pre-established number of local (including global) optima in vastly non-linear mathematical programming problems. Imanirad & Yeomans [12] have demonstrated how an FA's functional optimization proclivities for finding multiple local optima can be adapted for the concurrent creation of the *n* maximally different alternatives required in an MGA approach, after the initial problem optimization.

In this study, it is demonstrated sets of maximally different solution alternatives can be generated *simultaneously* by modifying the biologically-inspired FA of Yang [14], [15] and by extending the previous concurrent MGA approaches of Yeomans [18], Imanirad & Yeomans [12] and Imanirad *et al*. (([13], [19], [20], [21], [22]). This new MGA algorithm extends the earlier procedures of Imanirad *et al*. (([13], [19], [20], [21], [22]) to now permit the simultaneous generation of the overall best solution together with *n* locally optimal, maximally different alternatives in a single computational run. Specifically, to generate the additional *n* maximally different solution alternatives, the MGA algorithm would need to run exactly the same number of times that the FA would need to be run for function optimization alone (i.e. once) irrespective of the value of *n* [23], [24]. Thus, this simultaneous, FA-directed method is extremely computationally efficient for MGA. Finally, this research demonstrates the effectiveness of the FA approach for simultaneously producing multiple, good-but-very-different solution sets on a commonly evaluated benchmark engineering optimization test [23], [25].

II. Firefly Algorithm for Function Optimization

While this section provides only a relatively brief synopsis of the FA procedure, more detailed explanations can be accessed in [12], [13], [14], [15], and [17]. The FA is a biologically-inspired, population-based metaheuristic. Each firefly in the population represents one potential solution to a problem and the initial population of fireflies should be distributed uniformly and randomly throughout the solution space. The solution approach employs the following three idealized rules: (i) All fireflies within the population are considered essentially unisex, so that any one firefly could potentially be attracted to any other firefly irrespective of their sex; (ii) The relative attractiveness between any two fireflies is directly proportional to their respective brightness. This implies that for any two flashing fireflies, the less bright firefly will always be inclined to move towards the brighter one. However, attractiveness and brightness both decrease as the relative distance between the fireflies increases. If there is no brighter firefly within its visible neighborhood, then the particular firefly will move about randomly; and, (iii) The brightness of a firefly is determined by the overall landscape of the objective function. Namely, for a maximization problem, the brightness can simply be considered proportional to the value of the objective function. Based upon these three rules, the basic operational steps of the FA can be summarized within the following pseudo-code.

```
Objective Function F(X), X = (x1, x2,… xd)
Generate the initial population of n fireflies, Xi, i = 1, 2,…, n
Light intensity Ii at Xi is determined by F(Xi)
Define the light absorption coefficient γ
while (t < MaxGeneration)
        for i = 1: n, all n fireflies
                for j = 1: n, all n fireflies (inner loop)
                        if (Ii < Ij), Move firefly i towards j; end if
                        Vary attractiveness with distance r via e^(-γr)
                end for j
        end for i
        Rank the fireflies and find the current global best solution G*
end while
Postprocess the results
```

In the FA, there are two important issues to resolve: the formulation of attractiveness and the variation of light intensity. For simplicity, it can always be assumed that the attractiveness of a firefly is determined by its brightness which in turn is associated with its encoded objective function value. In the simplest case, the brightness of a firefly at a particular location $\boldsymbol{X}$ would be its calculated objective value $F(\boldsymbol{X})$. However, the attractiveness, β, between fireflies is relative and will vary with the distance r_{ij} between firefly *i* and firefly *j*. In addition, light intensity decreases with the distance from its source, and light is also absorbed in the media, so the attractiveness needs to vary with the degree of absorption. Consequently, the overall attractiveness of a firefly can be defined as

$$\beta = \beta_0 \exp(-\gamma r^2) \qquad (1)$$

where β_0 is the attractiveness at distance r = 0 and γ is the fixed light absorption coefficient for the specific medium. If the distance r_{ij} between any two fireflies *i* and *j* located at $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$, respectively, is calculated using the Euclidean norm, then the movement of a firefly *i* that is attracted to another more attractive (i.e. brighter) firefly *j* is determined by

$$\boldsymbol{X}_i = \boldsymbol{X}_i + \beta_0 \exp(-\gamma(r_{ij})^2)(\boldsymbol{X}_i - \boldsymbol{X}_j) + \alpha\boldsymbol{\varepsilon}_i \,. \tag{2}$$

In this expression of movement, the second term is due to the relative attraction and the third term is a randomization component. Yang [15] indicates that α is a randomization parameter normally selected within the range [0,1] and $\boldsymbol{\varepsilon}_i$ is a vector of random numbers drawn from either a Gaussian or uniform (generally [-0.5,0.5]) distribution. It should be explicitly noted that this expression represents a random walk biased toward brighter fireflies and if β_0 = 0, it becomes a simple random walk. The parameter γ characterizes the variation of the attractiveness and its value determines the speed of the algorithm's convergence. For most applications, γ is typically set between 0.1 to 10 ([15], [17]). In any given optimization problem, for a very large number of fireflies $n >> k$, where *k* is the number of local optima, the initial locations of the *n* fireflies should be distributed relatively uniformly throughout the entire search space. As the FA proceeds, the fireflies begin to converge into all of these local optima (including the global ones). Hence, by comparing the best solutions among all these optima, the global optima can easily be determined. Yang (2010) proves that the FA will approach the global optima when $n \rightarrow \infty$ and the number of iterations *t*, is set so that $t >>1$. In reality, the FA has been found to converge extremely quickly with *n* set in the range 20 to 50 ([14], [15], [17]).

Two important limiting or asymptotic cases occur when $\gamma \rightarrow 0$ and when $\gamma \rightarrow \infty$. For $\gamma \rightarrow 0$, the attractiveness is constant $\beta = \beta_0$, which is equivalent to having a light intensity that does not decrease. Thus, a firefly would be visible to every other firefly anywhere within the solution domain. Hence, a single (usually global) optima can easily be reached. If the inner loop for *j* in the pseudo-code is removed and $\boldsymbol{X}_j$ is replaced by the current global best ***G****, then this implies that the FA reverts to a special case of the accelerated particle swarm optimization (PSO) algorithm. Subsequently, the computational efficiency of this special FA case is equivalent to that of enhanced PSO. Conversely, when $\gamma \rightarrow \infty$, the attractiveness is essentially zero along the sightline of all other fireflies. This is equivalent to the case where the fireflies randomly roam throughout a very thick foggy region with no other fireflies are visible and each firefly roams in a completely random fashion. This case corresponds to a completely random search method. As the FA operates between these two asymptotic extremes, it is possible to adjust the parameters α and γ so that the FA can outperform both a random search and the enhanced PSO algorithms [17].

The computational efficiencies of the FA will be exploited in the subsequent MGA solution approach. As noted, within the two asymptotic extremes, the population in the FA can determine both the global optima as well as the local optima concurrently. The concurrency of population-based solution procedures holds huge computational and efficiency advantages for MGA purposes [7]. An additional advantage of the FA for MGA implementation is that the different fireflies essentially work independently of each other, implying that FA procedures are better than genetic algorithms and PSO for MGA because the fireflies will tend to aggregate more closely around each local optimum [15], [17]. Consequently, with a judicious selection of parameter settings, the FA will simultaneously converge extremely quickly into both local and global optima [14], [15], [17].

III. Modelling to Generate Alternatives

Most mathematical programming approaches arising in the optimization literature have concentrated almost exclusively upon producing single optimal solutions to single-objective problem instances or, equivalently, generating noninferior solution sets to multi-objective formulations [2], [5], [8]. While such algorithms may efficiently generate solutions to the derived complex mathematical models, whether these outputs actually establish "best" approaches to the underlying real problems is certainly questionable [1], [2], [6], [8]. In most "real world" decision environments, there are innumerable system objectives and requirements that are never explicitly apparent or included in the decision formulation stage [1], [5]. Furthermore, it may never be possible to explicitly express all of the subjective components because there are frequently numerous incompatible, competing, design requirements and, perhaps, adversarial stakeholder groups involved. Therefore, most subjective aspects of a problem necessarily must remain unquantified and unmodelled in the construction of the resultant decision models. This is a common occurrence in situations where the final decisions are constructed based not only upon clearly stated and modelled objectives, but also upon fundamentally subjective socio-political-economic goals and stakeholder preferences [7]. Numerous "real world" examples describing these types of incongruent modelling dualities appear in [6], [8], [9], and [10].

When unquantified issues and unmodelled objectives exist, non-conventional approaches are required that not only search the decision space for noninferior sets of solutions, but must also explore the decision space for discernibly *inferior* alternatives to the modelled problem. In particular, any search for good alternatives to problems known or suspected to contain unmodelled objectives must focus not only on the non-inferior solution set, but also necessarily on an explicit exploration of the problem's inferior region.

To illustrate the implications of an unmodelled objective on a decision search, assume that the optimal solution for a quantified, single-objective, maximization decision problem is $\boldsymbol{X}^*$ with corresponding objective value $Z1^*$. Now suppose that there exists a second, unmodelled, maximization objective $Z2$ that subjectively reflects some unquantifiable "environmental/political acceptability" component. Let the solution $\boldsymbol{X}^a$, belonging to the noninferior, 2-objective set, represent a potential best compromise solution if both objectives could somehow have been simultaneously evaluated by the decision-maker. While $\boldsymbol{X}^a$ might be viewed as the best compromise solution to the real problem, it would appear inferior to the solution $\boldsymbol{X}^*$ in the quantified mathematical model, since it evidently must be the case that $Z1^a \leq Z1^*$. Consequently, when unmodelled objectives are factored into the decision-making process, mathematically inferior solutions for the modelled problem can prove optimal to the underlying real problem. Therefore, when unmodelled objectives and unquantified issues might exist, different solution approaches are needed in order to not only search the decision space for the noninferior set of solutions, but also to simultaneously explore the decision space for inferior alternative solutions to the modelled problem. Population-based methods such as the FA permit concurrent searches throughout a feasible region and thus prove to be particularly adept solution procedures for searching through such a problem's decision space.

The primary motivation behind MGA is to produce a manageably small set of alternatives that are quantifiably good with respect to modelled objectives yet are as different as possible from each other in the decision space. In doing this, the resulting alternative solution set is likely to provide truly different choices that all perform somewhat similarly with respect to the known modelled objective(s) yet very differently with respect to any unmodelled issues. By generating these good-but-different solutions, the decision-makers can explore desirable qualities within the alternatives that may prove to satisfactorily address the various unmodelled objectives to varying degrees of stakeholder acceptability.

In order to properly motivate an MGA search procedure, it is necessary to apply a more mathematically formal definition to the goals of the MGA process [6], [7]. Suppose the optimal solution to an original mathematical model is $\boldsymbol{X^*}$ with objective value $\boldsymbol{Z^*} = F(\boldsymbol{X^*})$. The following maximal difference model can then be solved to generate an alternative solution, $\boldsymbol{X}$, that is maximally different from $\boldsymbol{X}^*$:

Maximize $$\Delta(\boldsymbol{X}, \boldsymbol{X^*}) = \sum_i |X_i - X_i^*| \qquad (3)$$

Subject to: $$\boldsymbol{X} \in D \qquad (4)$$

$$|F(\boldsymbol{X}) - \boldsymbol{Z^*}| \leq T \qquad (5)$$

where Δ represents some difference function (for clarity, shown as an absolute difference in this instance) and T is a targeted tolerance value specified relative to the problem's original optimal objective $\boldsymbol{Z^*}$. T is a user-supplied value that determines how much of the inferior region is to be explored in the search for acceptable alternative solutions. This difference function concept can be extended into a measure of difference between any set of alternatives by replacing $\boldsymbol{X}^*$ in the objective of the maximal difference model and calculating the overall sum (or some other function) of the differences of the pairwise comparisons between each pair of alternatives – subject to the condition that each alternative is feasible and falls within the specified tolerance constraint.

The FA-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 using the FA to solve the corresponding maximal difference problem instance. By exploiting the population structure of the FA, the Fireflies collectively evolve toward different local optima within the solution space. 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.

IV. FA-based Simultaneous MGA Computational Algorithm

The MGA method to be introduced 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 an FA to solve the corresponding, maximal difference problem. Each solution within the FA's population contains one potential set of p different alternatives. By exploiting the co-evolutionary solution structure within the population of the algorithm, the Fireflies collectively evolve each solution to-ward sets of different local optima within the solution space. In this process, each desired solution alternative undergoes the common search procedure of the FA. However, the survival of solutions depends both 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 generated in the decision space.

A direct process for generating alternatives with the FA would be to iteratively solve the maximum difference model by incrementally updating the target T whenever a new alternative needs to be produced and then re-running the algorithm. This iterative approach would parallel the original Hop, Skip, and Jump (HSJ) MGA algorithm of Brill *et al*. [8] in which, once an initial problem formulation has been optimized, supplementary alternatives are systematically created one-by-one through an incremental adjustment of the target constraint to force the sequential generation of the suboptimal solutions. While this approach is straightforward, it requires a repeated execution of the optimization algorithm [7], [12], [13].

To improve upon the stepwise alternative approach of the HSJ algorithm, a concurrent MGA technique was subsequently designed based upon the concept of co-evolution Imanirad *et al*. ([13], [19], [21]). In the co-evolutionary approach, pre-specified stratified subpopulation ranges within the algorithm's overall population were established that collectively evolved the search toward the creation of the specified number of maximally different alternatives. Each desired solution alternative was represented by each respective subpopulation and each subpopulation underwent the common processing operations of the FA. The survival of solutions in each subpopulation depended 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].

By employing this co-evolutionary concept, it becomes possible to implement an FA-based MGA procedure that concurrently produces alternatives which possess objective function bounds that are somewhat analogous to those created by the sequential, iterative HSJ-styled solution generation approach. While each alternative produced by an HSJ procedure is maximally different only from the overall optimal solution (together with its bound on the objective value which is at least x% different from the best objective (i.e. x = 1%, 2%, etc.)), a concurrent procedure is able to generate alternatives that are no more than x% different from the overall optimal solution but with each one of these solutions being as maximally different as possible from every other generated alternative that was produced. Co-evolution is also much more efficient than the sequential HSJ-style approach in that it exploits the inherent population-based searches of FA procedures to concurrently generate the entire set of maximally different solutions using only a single population [12], [21].

While a concurrent approach exploits the population-based nature of the FA's solution approach, the co-evolution process occurs within each of the stratified subpopulations. The maximal differences between solutions in different subpopulations is 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 FA 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 circumvented.

The steps in the co-evolutionary alternative generation algorithm are as follows ([18], [23], [24]):

Initialization Step. In this preliminary step, solve the original optimization problem to determine the optimal solution, $\boldsymbol{X^*}$. As with prior solution approaches Imanirad *et al*. ([13], [19], [20], [21], [22]) and without loss of generality, it is entirely possible to forego this step and construct the algorithm to find $\boldsymbol{X^*}$ as part of its solution processing. However, such a requirement increases the number of computational iterations of the overall procedure and the initial stages of the processing focus upon finding $\boldsymbol{X^*}$ while the other elements of each population solution remain essentially "computational overhead". 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.

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. $\boldsymbol{A}_p$ represents the p^{th} alternative, $p = 1,\ldots,P$, in each solution.

Step 2. In each of the K solutions, evaluate each $\boldsymbol{A}_p$, $p = 1,\ldots,P$, 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 measure is defined in Step 5). Note: Because the best solution to date is always retained in the population throughout each iteration of the FA, at least one solution will always be feasible. A feasible solution for the first step can always consists of *P* repetitions of ***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 k = 1,…, K, calculate D_k, a distance measure between all of the alternatives contained within solution k.

As an illustrative example for determining a distance measure, calculate

$$D_k = \sum_{i=1toP} \sum_{j=1toP} \Delta(A_i, A_j). \quad (6)$$

This represents the total distance between all of the alternatives contained within solution k. Alternatively, the distance measure could be calculated by some other appropriately defined function.

Step 6. 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 FA "change operations" to the each of the solutions and return to Step.

It should be apparent that the stratification approach outlined in this algorithm could be readily modified to accommodate any of the population-based solution procedures. However, as noted in Section1, the FA is an algorithm specifically designed to simultaneously converge into numerous local optima which provides distinct computational advantages over other population-based metaheuristics. A disadvantage to the FA-based procedure in comparison to iterative MGA approaches lies in the extra computational overhead required to store the expanded population size for all of the alternatives and the additional solution time required to generate both the overall optimal solution together with the set of maximally different alternatives. Based upon preliminary testing and significant experimentation in the subsequent section, it seems that the additional storage requirements can be considered essentially negligible for all practical purposes and that the extra computational effort is virtually undetectable. However, these disadvantages could potentially become more pronounced for other problem instances that might be identified in subsequent extensions to this paper.

V. Computational Testing of Simultaneous MGA Algorithm

As outlined above, "real world" decision-makers frequently prefer being able to choose from a set of close-to-optimal options that are dissimilar from each other in terms of the structures of their decision variables. The effectiveness of the MGA procedure introduced in the previous section to simultaneously produce maximally different alternatives will be illustrated using the optimization problem taken from [23] and [25], which has frequently been employed as a benchmark test for constrained, non-linear engineering optimization algorithms.

The mathematical formulation for this non-linear optimization problem can be expressed as:

$$\text{Min F(X)} = (x_1-10)^2+5(x_2-12)^2+x_3^4+3(x_4-11)^2+10x_5^6+7x_6^2+x_7^4-4x_6x_7-10x_6-8x_7 \quad (7)$$

Subject to:

$$g_1(\boldsymbol{X}) = 2x_1^2+3x_2^4+x_3+4x_4^2+5x_5-127 \le 0 \quad (8)$$

$$g_2(\boldsymbol{X}) = 7x_1+3x_2+10x_3^2+x_4-x_5-282 \le 0 \quad (9)$$

$$g_3(\boldsymbol{X}) = 23x_1+x_2^2+6x_6^2-8x_7-196 \le 0 \quad (10)$$

$$g_4(\boldsymbol{X}) = 4x_1^2+x_2^2-3x_1x_2+2x_3^2+5x_6-12x_7 \le 0 \quad (11)$$

$$-10 \le x_i \le 10, \quad i = 1, 2, 3, 4, 5, 6, 7 \quad (12)$$

The optimal solution for the specific design parameters employed within this formulation F(***X****) = 680.6300573 with decision variable values of ***X**** = (2.330499, 1.951372, -0.4775414, 4.365726, 0.6244870, 1.038131, 1.594227) (see [23], [25]).

In order to create the set of different alternatives, extra target constraints that varied the value of T by up to 1.5% between successive alternatives were placed into the original formulation in order to force the generation of solutions maximally different from the initial optimal solution (i.e. the values of the bound were set at 1.5%, 3%, 4.5%, etc. for the respective alternatives). The MGA procedure was used to create the optimal solution and the 10 maximally different solutions shown in Table 1.

Table 1. Objective Values and Solutions for the 11 Maximally Different Alternatives

Increment	1.5% Increment Between Alternatives							
	F(*X*)	x_1	x_2	x_3	x_4	x_5	x_6	x_7
Optimal	680.630	2.3304	1.9513	-0.4775	4.3657	0.6244	1.0381	1.5942
Alternative 1	683.917	2.3025	1.9353	-0.4881	4.3333	-0.6169	1.0355	1.5889
Alternative 2	687.022	2.3056	1.9076	-0.4245	4.3256	-0.6184	1.0388	1.6067
Alternative 3	696.899	2.2934	1.9096	-0.4397	4.3369	-0.6616	1.0331	1.6176
Alternative 4	705.926	2.3080	1.9171	-0.4724	4.3343	-0.6578	1.053	1.6078
Alternative 5	711.793	2.3174	1.9111	-0.4084	4.3668	-0.6166	1.0759	1.6116
Alternative 6	718.478	2.2904	1.9037	-0.427	4.3637	-0.5871	0.9955	1.6230
Alternative 7	725.652	2.3428	1.9158	-0.4459	4.3929	-0.6672	1.0382	1.6129
Alternative 8	741.897	2.2904	1.9037	-0.427	4.3637	-0.5871	0.9955	1.6230
Alternative 9	744.901	2.3468	1.9118	-0.4087	4.3557	-0.6283	0.9899	1.6024
Alternative 10	756.260	2.2985	1.9019	-0.4452	4.3577	-0.5927	1.0022	1.5770

As stated previously, most "real world" optimization applications are frequently plagued by conflicting performance requirements that are exceptionally hard to formulate. Under such circumstances, it becomes preferable to form sets of quantifiably good alternatives that provide dissimilar perspectives to any potentially unmodelled components that can be considered during the policy formulation stage. The unique performance features captured within these disparate options can provide very different system approaches with respect to the unmodelled issues, thereby potentially incorporating some of the unmodelled aspects into the actual solution selection process.

The computational example has demonstrated how an MGA modelling algorithm can be used to simultaneously generate multiple distinct alternatives by employing the very computationally efficient FA. The options created by the procedure satisfy the requisite system criteria within a prespecified bounds and yet remain maximally different from each other within the decision region. In addition to its alternative generating capabilities, the FA component simultaneously enables the MGA algorithm to perform extremely well with respect to function optimization. It should be noted that the overall best solution determined by the algorithm is indistinguishable from the optimal one obtained by Aragon *et al*. [25].

In summary, this section underscores several imperative observations with regard to the simultaneous MGA procedure: (i) The co-evolutionary aspects of the FA actually generate more good alternatives, in general, than planners would be able to create using other MGA approaches because of the evolving nature of the FA's population-based searches; (ii) By the design of the MGA algorithm, the alternatives generated are good for planning purposes since all of their structures will be maximally different from each other (i.e. the differences are not solely different from the overall optimal solution as would be the case under an HSJ-styled MGA approach); and, (iv) The approach is computationally very efficient since it will run only a single time in generating its entire set of alternatives (i.e. to generate *n* solution alternatives, the MGA procedure runs exactly once irrespective of the actual magnitude of *n*).

VI. Conclusions

Many "real world" problems possess complex performance specifications that are compounded by unquantifiable performance objectives and incongruent requirements. These decision-making environments frequently involve incompatible design elements that are difficult – if not impossible – to incorporate at the time that the decision supporting models are being formulated. Invariably, there are unmodelled components, not apparent during the model construction, that could significantly influence the acceptability of the model's solutions. These conflicting elements require that decision-makers integrate many uncertainties into their decision process prior to the ultimate solution resolution. Faced with such incongruencies, it becomes most unlikely for any single solution to simultaneously satisfy every ambiguous system requirement without significant tradeoffs. Thus, any decision supporting approaches employed must account for all of these aspects, in one way or another, while simultaneously being flexible enough to encapsulate the effects from the inherent planning contradictions.

In this study, a new MGA procedure demonstrated how the computationally efficient FA could simultaneously generate sets of maximally different, close-to-optimal alternatives via the evolutionary aspects of its population-based solution approach. In its MGA capacity, this simultaneous procedure efficiently generates the requisite set of dissimilar alternatives, with each generated solution providing a very different perspective to the problem. Since FA procedures can be employed in a wide variety of problem types, the practicality of this new simultaneous MGA approach can be extended into diverse spectrum of "real world" instances. These extensions will be examined in future research.

Acknowledgements

This research was supported in part by grant OGP0155871 from the Natural Sciences and Engineering Research Council.

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*, *21*, 2007, 1075-1090.X

[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*, *13(3)* , 2010, 229-238.

[3] M. Matthies, C. Giupponi, and B. Ostendorf, Environmental decision support systems: Current issues, methods and tools, *Environmental Modelling and Software*, *22(2)*, 2007, 123-127.

[4] H.T. Mowrer, Uncertainty in natural resource decision support systems: Sources, interpretation, and importance, *Computers and Electronics in Agriculture*, *27(1-3)*, 2000, 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*, *4(1)*, 2003, 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*, *33(5)*, 2001, 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*, *3(1)*, 2011, 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*, *28(3)*, 1982, 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*, *28(1)*, 1997, 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*, *133(2)*, 2007, 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, 19(2)*, 2012, 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*, (Amsterdam: Elsevier, 2013), 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*, *2(2)*, 2012, 43-47.

[14] X.S. Yang, Firefly algorithms for multimodal optimization, *Lecture Notes in Computer Science*, *5792*, 2009, 169-178.

[15] X.S. Yang, *Nature-Inspired Metaheuristic Algorithms*, 2nd ed. (Frome, UK: Luniver Press, 2010).

[16] L.C. Cagnina, C.A. Esquivel, and C.A. Coello, Solving Engineering Optimization Problems with the Simple Constrained Particle Swarm Optimizer, *Informatica*, *32*, 2008, 319-326.

[17] A.H. Gandomi, X.S. Yang, and A.H. Alavi, Mixed variable structural optimization using firefly algorithm, *Computers and Structures*, *89(23-24)*, 2011, 2325-2336.

[18] 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*. (New York: Springer, 2018) 261-273.

[19] R. Imanirad, X.S. Yang, and J.S. Yeomans, A co-evolutionary, nature-inspired algorithm for the concurrent generation of alternatives, *Journal on Computing*, *2(3)*, 2012, 101-106.

[20] R. Imanirad, X.S. Yang, and J.S. Yeomans, Modelling-to-generate-alternatives via the firefly algorithm, *Journal of Applied Operational Research*, *5(1)*, 2013, 14-21.

[21] 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*, *5(2)*, 2013, 33-45.

[22] 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*, *3(2)*, 2013, 1677-1686.

[23] J.S. Yeomans, Simultaneous Computing of Sets of Maximally Different Alternatives to Optimal Solutions, *International Journal of Engineering Research and Applications*, *7(9)*, 2017, 21-28.

[24] J.S. Yeomans, An Optimization Algorithm that Simultaneously Calculates Maximally Different Alternatives, *International Journal of Computational Engineering Research*, *7(10)*, 2017, 45-50.

[25] V.S. Aragon, S.C. Esquivel, and C.C.A. Coello, A Modified Version of a T-Cell Algorithm for Constrained Optimization Problems, *International Journal for Numerical Methods in Engineering*, *84*, 2010, 351-378.

Julian Scott Yeomans "Computationally Testing the Efficacy of a Modelling-to-Generate-Alternatives Procedure for Simultaneously Creating Solutions." IOSR Journal of Computer Engineering (IOSR-JCE) 20.1 (2018): PP 38-45.