ARTICLE TEMPLATE

A User-guided *Innovization*-based Evolutionary Algorithm Framework for Practical Multi-Objective Optimization Problems

Abhiroop Ghosh, Kalyanmoy Deb, Erik Goodman, and Ronald Averill

Michigan State University, East Lansing, MI 48824, USA
https://www.coin-lab.org

ARTICLE HISTORY
Compiled September 13, 2023

COIN Report 2023015

ABSTRACT
Knowledge of experienced users in solving real-world optimization problems can be formulated as inter-variable relationships to guide an optimization algorithm towards good solutions faster. Alternatively, such interactions can be learned algorithmically during the optimization by analyzing good solutions – a process called *innovization*. Any common pattern extracted from good solutions can be used as a repair operator to modify candidate solutions. The key aspect is to strike a balance between the relevance of the pattern and the extent of its use in the repair operator. This article proposes a multi-objective evolutionary algorithm (MOEA) framework that combines problem-specific knowledge and online innovization approaches to solve two real-world large-scale multi-objective problems: 879- and 1,479-variable truss design and 544-variable solid fuel rocket design. Four repair operators suitable for uncovering monotonic relations involving multiple decision variables are proposed. Performance variations resulting from different combinations of initial user knowledge and repair operators are also presented.

KEYWORDS
Multi-objective optimization, 'innovization', knowledge-based optimization.

1. Introduction

For practical multi-objective optimization problems (MOPs), researchers often ignore existing qualitative user knowledge in devising an algorithm, anticipating 'dilution' of the overall study. However, for a practical problems, using such information can be advantageous. Meaningful solutions to such problems usually come with physical parameters that are related in certain ways. In a billion-variable resource allocation problem (Deb and Myburgh 2017), the linearity of constraint structures allowed the development of an efficient customized evolutionary algorithm. A customized micro-genetic algorithm (GA) (Szollos, Šmíd, and Hájek 2009) combining range-adaptation and knowledge-based reinitialization is successfully applied on an airfoil optimization problem. Other knowledge-incorporation techniques exist in the literature (Landa-Becerra, Santana-Quintero, and Coello 2008). Another study used the concept of semi-independent variables (Gandomi et al. 2019), in which a redefinition of variables is proposed to handle user-specified monotonic relationships among variables.

CONTACT Abhiroop Ghosh. Author emails: {ghoshab1, kdeb, goodman, averillr}@msu.edu

Pre-specifying problem information is not the only way to provide guidance to an optimization algorithm. Cultural algorithms (Galanti 2000) attempt to learn and incorporate domain knowledge during the search process by encoding them inside a belief space (Coello and Tapia 2021). Self-organizing maps (SOMs) have been used to identify the relative importance of design variable clusters (Obayashi and Sasaki 2003). Recent *innovization* studies (Bandaru and Deb 2013; Gaur and Deb 2016) have demonstrated faster convergence using problem information extracted from high-performing solutions.

Integrating additional problem information raises several issues. How can qualitative information be quantified? How can any learned information be validated to be useful in speeding progress toward the optimal region? To what extent should such information be used to avoid premature or false convergence?

This article addresses the issues mentioned above and proposes a generic knowledge-based multi-objective evolutionary algorithm (MOEA) framework for solving practical problems. Three possible ways to use problem knowledge in the form of a repair operator are presented as an adaptive ensemble method. The possibility of imperfect learned knowledge is also taken into account and the algorithm can adjust the extent of influence of such information. The approach is demonstrated on two practical large-scale multi-objective optimization problems (LSMOPs).

2. Proposed User-guided Innovization-based MOEA Framework (MOEA/I)

The proposed framework combines initial user-provided guidance and online innovization to perform a more efficient search. It also makes provisions for verifying the correctness of each piece of learned information, adjusting its influence on the optimization process accordingly. This requires an optimization algorithm that is customizable. Classical point-based methods do not offer such flexibility, making MOEAs such as NSGA-II (Deb et al. 2002) a better choice. The major components are described below, and a figure illustrating the entire framework is provided in the supplementary document.

- Problem specification - As a first step in the optimization, the user specifies the objective functions, the decision variables, the constraints, and the model to be used. Any problem-specific information is also provided at this stage.
- Optimization strategy design - Here, the optimization algorithm designer chooses or creates a suitable algorithm to be used for the problem. In addition, if innovization is to be performed, the corresponding procedure is specified here.
- User-guided innovization-based MOEA (MOEA/I) - This is the proposed framework containing the following major components:
    - Generating new solutions - Generates new solutions from the parent population using reproduction operators such as crossover and mutation.
    - Innovization - Analyze good solutions in the parent population and extract patterns in the form of 'innovization rules'.
    - Repair - Modify new solutions according to the innovization rules learned previously.
    - Evaluation - Evaluate the objective functions for the repaired solution set.

The subsequent sections present the user knowledge specification procedure and the four repair operators used in this article, three of which are an extension of the

operators proposed by Ghosh et al. (2021). Each repair operator functions differently when explicit relations are specified compared to when only variable groups are specified without any explicit relation. In this article, the scope of the repair operators is limited to inequality relations. However, if the problem demands it, custom repair operators can be designed and used within the MOEA/I framework.

2.1. *User knowledge specification*

User knowledge can be of different types, relating two or more *comparable*[1] variables or relating some variables and objective or constraint functions in certain ways. This information is in addition to the optimization problem formulation provided by the user and perhaps acquired over many years of experience of the user in dealing with past solutions of the problem.

An upper-diagonal relationship matrix $U = [u_{ij}]$, where $i, j \in [1, n], i < j$ is proposed to store all user-provided pair-wise variable information:

$$u_{ij} = \begin{cases} 0, \text{ if } x_i \text{ and } x_j \text{ are not likely to have any relationship,} \\ 1, \text{ if } x_i \text{ and } x_j \text{ have a relationship, but unknown,} \\ 2, \text{ if } x_i < x_j, \\ 3, \text{ if } x_i > x_j, \\ 4, \text{ if } x_i \approx x_j. \end{cases} \tag{1}$$

As mentioned before, all variable pairs for which the relationship index $u_{ij} > 0$ are required to have the same scale and represent similar quantities. This is justified by some practical considerations. Two variables representing totally different quantities, such as length and weight, should not be expected to have any direct inequality relationship.

For LSMOPs, a large number of decision variables will make specifying all $\frac{n(n-1)}{2}$ relationships extremely cumbersome. However, in practical problems, variable patterns can be specified or envisioned to have relationships in groups (or variable clusters). K groups of variables can be specified (G_k, $k = 1, 2, \ldots, K$). An example of a 6-variable Umatrix and 3 groups (G_1-G_3) is given in the supplementary document.

For the unknown relationship ($u_{ij} = 1$), it is expected that the optimization task will analyze its best population members to try to discover an exact relationship between variables x_i and x_j. For $u_{ij} = 2$ to 4, the optimization task is expected to establish whether the specified relationship exists and utilize the relationship to fix any population member that violates it. Thus, the qualitative relationships provided at the start of the optimization task must first be validated and utilized to make a faster convergence. As a by-product, the user also gets a quantitative version of the relationships they provided. This repair-based quantification process is described below.

2.2. *Repair procedure*

Every variable pair (i, j) in a group G_k is cross-checked with the relationship matrix U to determine which variable pairs are designated by the user to be connected by an

[1]Comparable variables mean that they are of identical units and varying in similar range. For example, two size-related variables varying within $[a, b]$ are defined as comparable variables here.

unknown relationship ($u_{ij} = 1$). In order to learn the unknown relationships, the best feasible designs found so far are used. For MOPs, the non-dominated solution set at the end of every generation is a logical choice. A reference vector ($\mathbf{x}_{ref}$), calculated from the best solutions obtained so far, encodes the 'average' relationship between each variable pair among good designs found so far. The calculation procedure varies among different repair operators. A score p_{ij} is assigned equal to the proportion of the good solution set that follow the relationship existing among variable pair (i, j), defined using $\mathbf{x}_{ref}$. This score represents how well ($\mathbf{x}_{ref}$) reflects the set of good solutions and also acts as the probability with which a particular offspring is repaired.

For every variable group, if an explicit relation is defined ($u_{ij} > 1$), the repair operator ensures every offspring follows it. Otherwise, the relationships learned through innovization ($u_{ij} = 1$) are applied with the probability calculated during the innovization procedure.

The repair operators used in this article are outlined in Table 1. Each operator uses the knowledge available during the optimization run to different extents. For example, operator IR1 constrains the offspring the least, whereas operator IR3 constrains them the most. In addition, an ensemble operator (I-ES) is also proposed that automatically switches between IR1, IR2 and IR3 based on their individual performances. If no repair operators are used, the algorithm is referred to as 'Base MOEA' in this article.

2.2.1. MOEA/IR1 Procedure

It is assumed that the user has already provided the relationship matrix U and variable groups G. MOEA/IR1 computes the variable-wise average ($x_{i_{avg}}$) from all non-dominated (ND) solutions at the end of a generation and repairs a pair of variables of an offspring solution (x_i and x_j) based on the supplied problem information u_{ij}. The repaired variable values of x_{ir} for $u_{ij} = 1$, 2, and 3 are shown in Table 1. If $u_{ij} = 0$, no repair is performed and a free-form evolution is allowed. If $u_{ij} = 1$, meaning that a relation is expected, but is unknown, MOEA/IR1 attempts to learn the evolved relationship between x_i and x_j present among the non-dominated (ND) solutions and to enforce repair of both variable values. If $x_{i_{avg}}$ is smaller than $x_{j_{avg}}$ and x_i is also smaller than x_j, the current (x_i, x_j) pair matches the relationship among ND solutions and hence, the (x_i, x_j) pair is not modified. However, if $x_i \geq x_j$, disagreeing with the relationship found between their average ND values, the repaired (x_{ir}, x_{j_r}) pair in Table 1 are closer to their ($x_{i_{avg}}$, $x_{j_{avg}}$) values. After the repair, the difference $|x_{i_{avg}} - x_{ir}| = |x_{i_{avg}} - 0.5(x_{i_{avg}} + x_i)| = 0.5|x_{i_{avg}} - x_i|$ is smaller than the original difference $|x_{i_{avg}} - x_i|$ by 50%. The same is true for repaired variable x_{j_r}. For $u_{ij} = 2$ and 3, the x_i and x_j are repaired carefully (shown in Table 1) so that the supplied relationship among the two variable is satisfied. For $u_{ij} = 4$, $x_{i_r} = x_{j_r} = \text{random}(x_{i_{avg}}, x_{j_{avg}})$ is assigned.

2.2.2. MOEA/IR2 Procedure

This repair operator follows a similar repair process to that of MOEA/IR1, except that $x_{i_{avg}}$ is replaced with $x_{i_{ref}}$, which uses a history of change of the average x_i from the past to the current generation, as shown in Table 1. The notable difference between the IR1 and IR2 operators is the inclusion of a momentum parameter (γ), noting that $\gamma = 0$ makes both methods identical. The parameter is intended to provide a boost to the optimization algorithm toward a projected good solution. The average value $\mathbf{x}_{avg}$ of the variables under consideration reflects the approximate pattern followed by good

Table 1.: Repair operator description.

Relation	Operator	Repair equations
Unknown Relationship ($u_{ij} = 1$)	IR1	$x_{i_r} = \begin{cases} \frac{x_{i_{avg}} + x_i}{2}, & \text{for } (x_{i_{avg}} - x_{j_{avg}})(x_i - x_j) < 0, \\ x_i, & \text{otherwise.} \end{cases}$ $x_{j_r} = \begin{cases} \frac{x_j + x_{j_{avg}}}{2}, & \text{for } (x_{i_{avg}} - x_{j_{avg}})(x_i - x_j) < 0, \\ x_j, & \text{otherwise.} \end{cases}$
	IR2	$x_{i_r} = \begin{cases} \frac{x_{i_{ref}} + x_i}{2}, & \text{for } (x_{i_{ref}} - x_{j_{ref}})(x_i - x_j) < 0, \\ x_i, & \text{otherwise.} \end{cases}$ $x_{j_r} = \begin{cases} \frac{x_j + x_{j_{ref}}}{2}, & \text{for } (x_{i_{ref}} - x_{j_{ref}})(x_i - x_j) < 0, \\ x_j, & \text{otherwise.} \end{cases}$ where $x_{i_{ref}}(t) = x_{i_{avg}}(t) + \gamma \left(x_{i_{avg}}(t) - x_{i_{avg}}(t-1) \right)$
	IR3	$x_{i_r} = \mathcal{U}(x_{i_{ref}} - \sigma_i, x_{i_{ref}} + \sigma_i)$, $x_{j_r} = \mathcal{U}(x_{j_{ref}} - \sigma_j, x_{j_{ref}} + \sigma_j)$, where $\mathcal{U}(a, b) \equiv$ Uniform distribution between [a,b]
Direct Relationship ($u_{ij} = [2, 3]$)	IR1	$x_{i_r} = \begin{cases} x_{ij_{avg}} - \frac{\lvert x_{ij_{avg}} - x_{i_{avg}} \rvert}{2}, & \text{for } u_{ij} = 2, \\ x_{ij_{avg}} + \frac{\lvert x_{ij_{avg}} - x_{i_{avg}} \rvert}{2}, & \text{for } u_{ij} = 3. \end{cases}$ $x_{j_r} = \begin{cases} x_{ij_{avg}} + \frac{\lvert x_{ij_{avg}} - x_{j_{avg}} \rvert}{2}, & \text{for } u_{ij} = 2, \\ x_{ij_{avg}} - \frac{\lvert x_{ij_{avg}} - x_{j_{avg}} \rvert}{2}, & \text{for } u_{ij} = 3. \end{cases}$ where $x_{ij_{avg}} = \frac{x_{i_{avg}} + x_{j_{avg}}}{2}$.
	IR2 & IR3	$x_{i_r} = \begin{cases} x_{ij_{ref}} - \frac{\lvert x_{ij_{ref}} - x_{i_{ref}} \rvert}{2}, & \text{for } u_{ij} = 2, \\ x_{ij_{ref}} + \frac{\lvert x_{ij_{ref}} - x_{i_{ref}} \rvert}{2}, & \text{for } u_{ij} = 3. \end{cases}$ $x_{j_r} = \begin{cases} x_{ij_{ref}} + \frac{\lvert x_{ij_{ref}} - x_{j_{ref}} \rvert}{2}, & \text{for } u_{ij} = 2, \\ x_{ij_{ref}} - \frac{\lvert x_{ij_{ref}} - x_{j_{ref}} \rvert}{2}, & \text{for } u_{ij} = 3. \end{cases}$ where $x_{i_{ref}}(t) = x_{i_{avg}}(t) + \gamma \left(x_{i_{avg}}(t) - x_{i_{avg}}(t-1) \right)$

solutions in the current generation. It also allows the algorithm to avoid following $\mathbf{x}_{avg}$ too closely, thus, preserving diversity. This repair method is more trustworthy than IR1, as an attempt is made to make the variable values close to historically agreeable average values, rather than current average value alone. The update of variables for $u_{ij} = 4$ is identical to that in IR1. A sensitivity analysis of parameter γ is provided in the supplementary document.

2.2.3. MOEA/IR3 Procedure

This repair operator is similar to IR2 for $u_{ij} = 2$ and 3, but for $u_{ij} = 1$, the values are repaired to be within one-sigma (σ_i is the standard deviation of the x_i values among ND solutions of the current generation) away from the historical average point. Thus, this method trusts the observed relationships of x_i and x_j more closely than IR1 and IR2. The update of variables for $u_{ij} = 4$ is identical to that in IR1 and IR2.

2.2.4. Ensembled MOEA/I-ES Procedure

Testing all repair operators designed for MOEA/I through separate experiments might prove costly. A solution is to combine these operators into an ensemble. During the

optimization run, the performance of each repair operator is tracked, and the number of solutions allowed to be repaired by each operator is based on the historical performance of the offspring generated by each. The ensemble method also treats Base MOEA as a repair operator, which represents the case when the offspring is not repaired. This allows the optimization to remain relatively unaffected by bad repair operators. A total of four repair operators are used in the ensemble process.

The performance of each offspring generated by the i-th repair operator is based on its offspring survival rate (r_s^i). Greater the survival rate of offsprings created by an operator, greater is the probability of it being used to repair subsequent offsprings. The probability ($\widehat{p_r^i}$) update operation for the i-th operator is described below:

$$p_r^i(t+1) = \max\left(p_{min},\ \alpha\frac{r_s^i}{\sum_i r_s^i} + (1-\alpha)\widehat{p_r^i}(t)\right), \tag{2}$$

$$\widehat{p_r^i}(t+1) = \frac{p_r^i(t+1)}{\sum_i p_r^i(t+1)}, \tag{3}$$

where α is the learning rate, $r_s^i = \frac{n_s^i}{n_{\text{off}}}$ (n_s^i and n_{off} are the number of offsprings created by the i-th operator that survive in generation t and the total number of offsprings that survive in generation t, respectively). It is possible that at any point during the optimization, no solution generated by one of the repair operators survives. This might cause the corresponding selection probability to go down to zero without any possibility of recovery. To have each repair operator available at every generation, the probability update step ensures that a minimum selection probability (p_{min}) is always assigned to each repair operator.

The learning rate (α) determines the rate of change of the repair probabilities. A high α would increase the sensitivity, and can result in large changes in repair probabilities over a short period of time. A low α exerts a damping effect which causes the probability values to update slowly. Through trial and error, $\alpha = 0.5$ and $p_{min} = 0.1$ are found to be suitable to the problems of this study.

3. Truss Design Problem

A scalable 3D truss design problem (Ghosh et al. 2021) is considered, details of which are provided in the supplementary document. There are two objective function to be minimized: (a) weight, and (b) compliance. There are two types of design variables: size variables, defined by member radii ($\mathbf{r}$), and shape variables, defined by length of the vertical members ($\mathbf{L}_v$) parallel to the z-axis. The problem is scaled for 820 and 1,380 members to have different number of variables and constraints.

3.1. *User-guided information*

In this problem, the radii of members are comparable variables. The same is true for the lengths of vertical members. Thus, a pair of variables (i, j) within each of these classes (radii and lengths) can be be assigned a relationship index $u_{ij} > 0$. This, in turn, can be mathematically expressed and integrated into the MOEA/I algorithm. Two types of knowledge can be specified for this problem:

- *Symmetry*: Due to the symmetric nature of the loading and supports, it can be expected that an optimal truss design will be symmetric in terms of shape and weight distribution. For the truss under consideration, there can be two planes of symmetry: (a) a plane parallel to the y-z plane and passing through the midpoint of the truss in the x-direction, and, (b) a plane parallel to the x-z plane and passing through the midpoint of the truss in the y-direction.
- *Monotonicity*: For an optimal truss, the length of vertical members will monotonically increase while moving from the support to the middle. In addition, for a simply-supported truss like the one considered here, it is known that the bending moment monotonically increases from the support towards the middle. So the radii of the corresponding members may also monotonically increase to withstand the large bending moment.

The incorporation of this *a priori* user knowledge into the optimization process requires defining the variable groups (G) and the relationship matrix (U) defining the associated relationships. Four groups of variables are considered in this article – length of vertical members (G_1), radii of the top longitudinal members (G_2), radii of the bottom longitudinal members (G_3), and radii of vertical members (G_4). Each group (G_i) is also divided into two or more subgroups (G_{ij}, j being the subgroup number) used for specifying symmetry relationships. Each group of size variables is designed taking into account the physical location of the members corresponding to those variables. For example, the members oriented in x-direction at the top of the truss can be expected to be related to each other in some manner rather than to a member with a different location and physical orientation. Detailed information about the variable groups is provided in the supplementary document.

Based on the variable groups, eight different scenarios (S_1 to S_8) are created with varying extents of user knowledge being supplied (details provided in supplementary document). The first four scenarios from S_1 to S_4 do not enforce any symmetry, making the optimization more challenging. The scenarios S_5 to S_8 enforce symmetry ($u_{ij} = 4$) in the truss.

In scenario S_1, no knowledge is provided by the user. This is the reference scenario based on which the effectiveness of varying degrees of user knowledge coupled with different innovization-based repair operators is evaluated.

In scenario S_2, only group G_1 is used. u_{ij} is set to 1 for all variable pairs within subgroups G_{11} and G_{12}. This signifies that the length of each vertical member is expected to consistently follow a monotonically increasing or decreasing pattern, but it is not known beforehand what the exact relationship would be. It is up to the innovization process to determine the exact nature of the relationships among these two groups of variables dictated by the ND solutions.

S_3 extends S_2 and applies $u_{ij} = 1$ among all variables within each subgroup in all groups. Thus, relationships in both shape and size variables will now be obtained by the innovization process and will be enforced by the proposed procedure to various degrees dictated by the relative success of the three repair schemes.

Scenario S_4 provides more problem information to the optimization algorithm by specifying precise relationships among variables of each subgroup of the four groups. For example, within the subgroup G_{11}, variable pairs (x_i, x_{i+1}) are assigned a relationship ($u_{i,i+1} = 2$): $x_i \leq x_{i+1}$ for $i = 1$ to $|G_{11}| - 1$. For G_{12}, ($u_{i,i+1} = 3$): $x_i \geq x_{i+1}$ for $i = 1$ to $|G_{12}| - 1$ is assigned.

Scenarios S_5 to S_8 use the same grouping as scenarios S_1 to S_4, respectively. The only addition is the application of symmetry relations within the respective variables

of the subgroups in each group. Two planes of symmetry exist as mentioned earlier, and for the corresponding variable pairs (i, j), u_{ij} is set as 4.

In this article, only inequality-based relationships are considered due to the nature of the problems considered. However, it is possible to extend the scope to cover other types of relations (such as power laws) as well. In order to test the robustness of the proposed repair operators, different knowledge levels are considered. Experiments are performed to determine how the algorithm performs in the most adverse conditions, such as too little or too much information, and scenarios involving asymmetric trusses.

3.2. *Experimental Settings*

In this article, NSGA-II (Deb et al. 2002), a state-of-the-art MOEA, has been used for the MOEA/I framework. However, other MOEAs can also be used with the proposed framework. The original NSGA-II algorithm, without any modifications, will be referred to as the base optimization case or Base NSGA-II. Four types of repair methods, MOEA/IR1, MOEA/IR2, MOEA/IR3 and MOEA/I-ES, presented in Sections 2.2.1 to 2.2.4, are used, and are referred to as NSGA-II/IR1, NSGA-II/IR2, NSGA-II/IR3 and NSGA-II/I-ES, respectively. For scenarios S_4 and S_8, additional experiments are performed using semi-independent variables, referred to as NSGA-II/SIV, described in the supplementary material.The parameter settings for NSGA-II as well as for the repair operators are presented in the supplementary material. Two truss cases, one with 820 members, and another with 1,380 members, are considered. Different experiments are performed with multiple levels of user knowledge integration, which are described in Section 3.1. The number of decision variables for scenarios S_1-S_8 are 879 and 1,479 for the 820 and 1,380-member trusses, respectively. For scenarios S_5 to S_8, Base NSGA-II takes into account the symmetry relations by evolving only one of any pair of variables following a symmetry relation. Thus, the problems are run with reduced dimensions for these cases only for Base NSGA-II. For the 820 member truss, the number of decision variables in that case would be 850 for S_5-S_6 (shape symmetry) and 450 for S_7-S_8 (shape and size symmetry). For the 1380 member truss, the number of decision variables in that case would be 1,439 for S_5-S_6 and 750 for S_7-S_8. 20 runs are performed and for both truss cases, the maximum number of function evaluations available is set at 2 million.

Hypervolume (HV) (Zitzler and Thiele 1998) is used as a performance metric. The median HV of scenario S_1 achieved by Base NSGA-II is set as the target hypervolume (HV^T). Over 20 runs, for each method, the mean and standard deviation of the number of function evaluations taken to achieve HV^T are recorded. The algorithm with the lowest number of median function evaluations ($\mathrm{FE}^{\min}_{\mathrm{median}}$) required for achieving HV^T is chosen as reference. Using this data, the Wilcoxon test is performed with 95% significance level and the p-values are calculated. An algorithm with a p-value greater than 0.05 means that there is not a statistically significant performance difference with the algorithm having the lowest median function evaluations. Details about the Wilcoxon test are provided in the supplementary document.

3.3. *Results and Discussion*

The optimization results for the 820 member truss case is given in the supplementary document. The 1,380-member truss case results are presented in Table 2. Some additional details about the optimal designs found during the optimization, are included

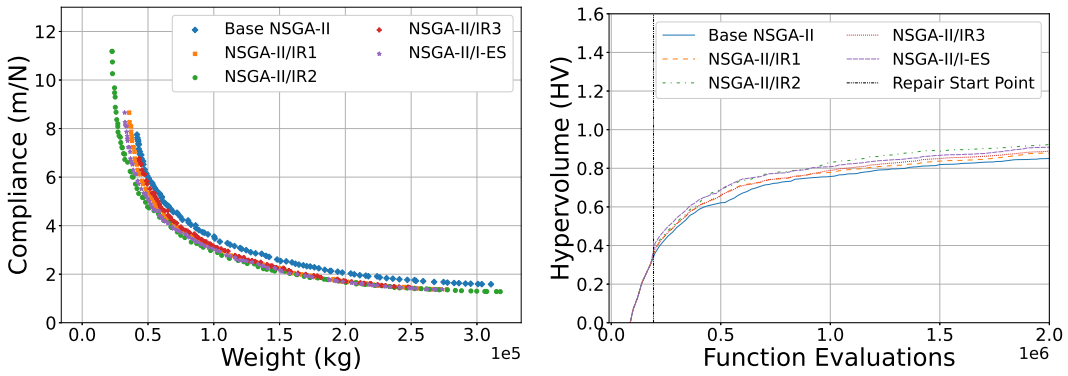
in the supplementary document.

Table 2.: FEs required to reach target $HV^T = 0.80$ for 1,380-member, 396-node truss problem. Best performing algorithm for each scenario (S_1 to S_4 does not use symmetry of any kind but more knowledge is supplied from S_1 towards S_4, and similar knowledge embedding is done for S_5 to S_8, except that symmetry knowledge is used for these latter scenarios) is marked in bold. Algorithms with performance not statistically different from the best algorithm are marked in italics. N/A indicates 'Not Applied' when the user-provided information is not used (Base NSGA-II), or used in specific instances (NSGA-II/SIV). Wilcoxon test p-values are given in braces.

Scn.	$\mathcal{K}$	Algorithm Used					
		Base NSGA-II	NSGA-II /IR1	NSGA-II /IR2	NSGA-II /IR3	NSGA-II /I-ES	NSGA-II /SIV
S_1	0	2M	N/A	N/A	N/A	N/A	N/A
S_2	3.14e-03	*2M* ($p = 0.2347$)	*1.8M*±42k ($p = 0.3212$)	**1.6M**±48k	*1.8M*±9k ($p = 0.3205$)	*1.7M*±25k ($p = 0.3153$)	N/A
S_3	4.70e-03	2M ($p = 0.0153$)	*1.6M*±20k ($p = 0.1394$)	**1.4M**±25k	1.9M±10k ($p = 0.0265$)	*1.5M*±17k ($p = 0.1277$)	N/A
S_4	6.27e-03	**2M**	2M (HV=0.31, $p = 0.0019$)	2M (HV=0.44, $p = 0.0058$)	2M (HV=0.57, $p = 0.0103$)	2M (HV=0.66), $p = 0.0216$)	2M (HV=0.30, $p = 0.0038$)
S_5	6.52e-03	**1.6M**±25k	*1.7M*±15k ($p = 0.3120$)	*1.7M*±23k ($p = 0.2961$)	*1.8M*±12k ($p = 0.1849$)	*1.7M*±11k ($p = 0.2971$)	N/A
S_6	3.26e-02	1.6M±25k ($p = 0.0431$)	*1.3M*±20k ($p = 0.0713$)	**1.0M**±30k	1.5M±10k ($p = 0.0481$)	*1.1M*±18k ($p = 0.0692$)	N/A
S_7	3.59e-02	1.4M±15k ($p = 0.0303$)	*1.2M*±19k ($p = 0.0572$)	**0.90M**±23k	*1.2M*±10k ($p = 0.0576$)	*1.0M*±15k ($p = 0.0861$)	N/A
S_8	4.83e-02	**1.4M**±15k	2M (HV=0.68, $p = 0.0422$)	2M (HV=0.73, $p = 0.0490$)	2M (HV=0.72, $p = 0.0481$)	2M (HV=0.71, $p = 0.0477$)	2M (HV=0.66, $p = 0.0302$)

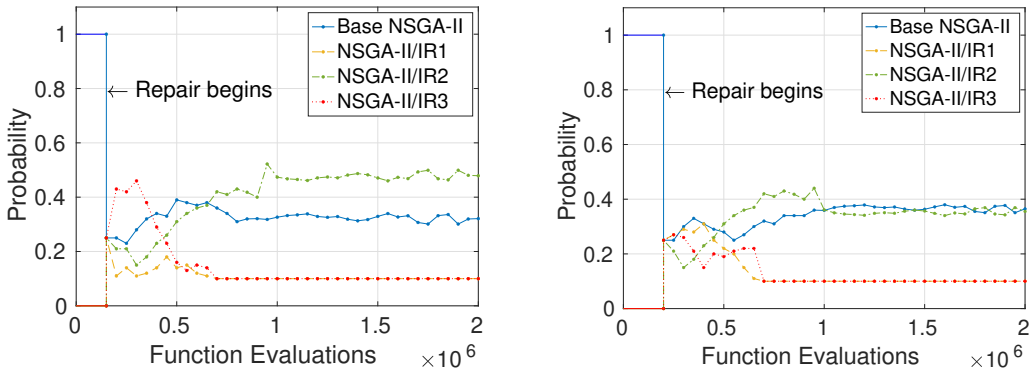
A metric ($\mathcal{K}$) is proposed here which measures the extent of user-specified knowledge, details of which are provided in the supplementary document. A higher value of $\mathcal{K}$ implies a greater amount of user-specified knowledge. Scenarios S_1 to S_8 are designed to show the results of the proposed MOEA/I approach (implemented as NSGA-II/I), and also for NSGA-II paired with SIVs. In both the truss cases, the row for scenario S_1 is marked as N/A for all the algorithms except for Base NSGA-II. This is because S_1 is the no-knowledge case where all the elements in the matrix U are set to 0. Thus, all the repair operators remain inactive, and the results are the same as Base NSGA-II. For most of the cases, it is seen that NSGA-II/IR2 performs the best (marked in bold), except for scenarios S_5 and S_8 (both cases), and S_4 (only for 1380-member truss). For S_5, even though Base NSGA-II performs the best in terms of FEs, the other algorithms give a statistically similar performance ($p > 0.05$). For S_8, all the repair operators and NSGA-II/SIV have a statistically worse performance ($p < 0.05$) compared to Base NSGA-II. Table 2 show that a moderate usage of knowledge obtained through innovization gives the optimal results. Like in the truss problems, too little or too much knowledge is detrimental to the optimization performance. The performance of the ensemble approach (NSGA-II/I-ES) is statistically similar to the best algorithm in all scenarios except for S_4 and S_8. For example, with a Wilcoxon ranksum test, it is observed that for S_7 in Table 2, NSGA-II/I-ES has a statistically similar performance (with $p = 0.0861$) to NSGA-II/IR2, the best-performing algorithm. The ensemble approach (NSGA-II/I-ES) has the advantage that the user does not need to think about which repair operator to use, since it is handled automatically.

Proceeding row-wise from top to bottom, it is seen that performance generally improves until a particular point (S_3 and S_7) and then drops (S_4 and S_8). Interestingly, S_4 and S_8 are the ones which explicitly specify the relationships instead of letting the IR operators determine it. A possible cause is that over-specification of knowledge complicates the search process and constrains the algorithm efficiency. An interesting comparison is the difference in performances between NSGA-II/IR approaches and NSGA-II/SIV (Gandomi et al. 2019) (applicable to cases with $u_{ij} = 2$ or 3 only). For the 1,380-member truss (Table 2), NSGA-II/SIV is not able to reach the desired performance within 2M function evaluations for all the scenarios. This shows that the SIV approach is still susceptible to knowledge overspecification for large-sized problems.



(a) ND fronts. (b) Median hypervolume of 20 runs.

Figure 1.: Results for 1,380-member truss for scenario S_7 (symmetry among subgroups, unknown relationships within subgroups).



(a) 820-member truss. (b) 1,380-member truss

Figure 2.: Repair probability variation for median run.

Results for scenario S_7 are shown in Figure 1. The ND front plots (Figure 1a) clearly show NSGA-II/IR2 as the better performer, providing higher quality solutions than the others, with NSGA-II/I-ES following close behind. The median HV plots (Figure 1b) show that NSGA-II/IR2 and NSGA-II/I-ES achieve a higher median HV, faster, as shown previously in Table 2.

The NSGA-II/I-ES median repair probabilities plots (Figure 2) show the selection probability of each repair operator as well as base NSGA-II for generating new solu-

tions in NSGA-II/I-ES. It is seen that in both truss cases, a high probability is assigned to IR2 compared to others. Probabilities of IR1 and IR3 are reduced to the minimum level very early on, and new solution generation is controlled mostly by IR2 and the base NSGA-II. This clearly indicates the ability of the proposed ensembled method (I-ES) to pick the most successful operator on the fly for solving a problem efficiently.

It is interesting from Table 2 that with more information ($\mathcal{K}$) being provided, the performance of all algorithms does not improve monotonically. This means that there exists an optimal amount of problem information for every problem below or above which there is under- or over-specification of information provided. When less than necessary information is provided, an algorithm needs to work its way to find relevant building blocks for solving the problem, thereby requiring more solution evaluations. On the other hand, if more than necessary information is provided, even if the information is correct, the algorithm may get restricted in searching for other required information needed to solve the problem. For both trusses, the S_7 scenario provided the right amount of additional problem information. Combining it with the NSGA-II/IR2 algorithm which uses the right amount of available knowledge, the problem can be solved with at most 22% and 45% of the median number of solution evaluations required by the base NSGA-II procedure, for the 820- and 1,380-member problems, respectively. NSGA-II/I-ES provides a statistically comparable performance, thus negating the need to know the appropriate repair operator in advance.

4. Solid Rocket Design Problem

Solid rocket motor design can be formulated as an optimization problem as proposed in Ghosh et al. (2020). The variable vector $\mathbf{x}$ specifies each type of propellant in each layer of each segment, thickness of each layer, and geometry of core (finocyl) propellant arrangement. Two objectives are minimized: (i) squared difference between obtained and target thrust profile with time and (ii) sum of mean and standard deviation of segment residues at the end of burn. The pressure inside the rocket is restricted within a lower and upper bound. Details of the rocket optimization problem are provided in the supplementary document.

4.1. *User Knowledge*

This problem is intended to demonstrate the effectiveness of the proposed approach on a new and less-analyzed practical problem. Due to the lack of previous knowledge, most of the supplied user knowledge cases analyzed here will not specify any exact relationships, but rather, leave it to the innovization process to figure them out. A logical way to define variable groups is to do it segment-wise. Groups G_{r1}-G_{r3} represent the star shape for the three star segments. G_{r4}-G_{r9} represent the propellants for each cylindrical segment. G_{r10}-G_{r15} represent the layer thicknesses for each cylindrical segment.

Five scenarios (C_1-C_5) can be designed based on different levels of user knowledge extent, detailed description of which is provided in the supplementary document.

4.2. *Results and Discussion*

The experimental settings are same as those for the truss design problem described in Section 3.2, with only the maximum generations increased to 100,000. For scenario C_5 consisting of explicitly-specified information, NSGA-II/SIV is also compared to the other repair operators (IR1, IR2, IR3 and I-ES).

The optimization results are presented Table 3. The extent of supplied information increases row-wise from scenario C_1 to C_5 as can be seen from the values of $\mathcal{K}$. For the no-knowledge case C_1, all the elements of matrix U are set as 0. Thus, all repair operators remain inactive, so apart from base NSGA-II, all the other cells are marked as N/A. It is seen that in all of the cases except for C_5, NSGA-II/IR2 is the best performing algorithm (marked in bold), with NSGA-II/I-ES showing a comparable performance. Scenario C_4 combined with the NSGA-II/IR2 operator gives the best performance overall. This shows that for both the amount of user-supplied knowledge and the extent of online knowledge usage, an optimal level exists. Interestingly, for C_5 the relationships between the propellants in the cylindrical layers are explicitly defined. But this seems to constrain the algorithm and degrades its performance to below that of Base NSGA-II. A possible cause could be the lack of a direct relation between the thrust and the individual layer propellants. The same thrust value can be produced by a lot of different propellant combinations across all segments, and simple monotonic relationships may not exist. Due to the problem not being very well-studied, the exact nature of such relationships is not known beforehand. The supplementary document shows some more results.

In terms of final HV obtained, NSGA-II/IR2 provides the fastest convergence, demonstrated by the low number of function evaluations taken to reach the target HV, with NSGA-II/I-ES following closely. Base NSGA-II always performs the worst in scenarios C_2 to C_4. For C_5, the perceived knowledge $u_{ij} = 3$ for G_{r4}-G_{r9} is found to be not correct and they harm the performance of NSGA-II/IR methods.

Table 3.: FEs required to reach target $\text{HV}^T = 0.93$ for 544-variable solid fuel rocket design.

Scn.	$\mathcal{K}$	Algorithm Used					
		Base NSGA-II	NSGA-II/IR1	NSGA-II/IR2	NSGA-II/IR3	NSGA-II/I-ES	NSGA-II/SIV
C_1	0	50.0M	N/A	N/A	N/A	N/A	N/A
C_2	1.5e-04	50.0M ($p = 0.0053$)	*32.0M*±2.0M ($p = 0.1205$)	**30.5M**±1.5M	36.0M±3.0M ($p = 0.0104$)	*31.5M*±1.0M ($p = 0.1516$)	N/A
C_3	4.0e-03	50.0M ($p = 0.0061$)	*28.0M*±3.0M ($p = 0.0998$)	**25.0M**±4.0M	32.0M±1.0M ($p = 0.0051$)	*26.0M*±1.0M ($p = 0.1336$)	N/A
C_4	7.8e-03	50.0M ($p = 0.0029$)	19.0M±5.0M ($p = 0.0350$)	**12.0M**±5.0M	22.0M±4.0M ($p = 0.0154$)	*13.0M*±2.0M ($p = 0.7710$)	N/A
C_5	9.8e-03	**50.0M**	50.0M (HV=0.74, $p = 0.0233$)	50.0M (HV=0.78, $p = 0.0135$)	50.0M (HV=0.66, $p = 0.0104$)	50.0M (HV=0.84, $p = 0.0157$)	50.0M (HV=0.69, $p = 0.0119$)

5. Summary of Results

For each scenario in all the three case studies (879- and 1,479-variable truss designs and 544-variable rocket design), the five algorithms (base NSGA-II, three innovized repair based NSGA-IIs, and the ensembled NSGA-II) are ranked based on the median

FEs (over 20 runs) needed by each to reach a target HV. More details are included in the supplementary material. The final row of Table 4 indicates that NSGA-II/IR2 performs the best, followed by NSGA-II/I-ES. The base NSGA-II and NSGA-II/IR1 are tied in third place. NSGA-II/IR3 is ranked last, showing that an aggressive use of available and potentially imperfect information performs the worst. Interestingly, even though a single balance of problem information and its use within NSGA-II (IR2) performs the best for these problems, with all repair methods being included, the proposed ensembled NSGA-II performs the second best.

Table 4.: Ranking of different NSGA-IIs on three case studies. A scenario-wise breakdown is provided in the supplementary material.

Problem	Base	IR1	IR2	IR3	I-ES
879-variable Truss Design	4	3	**1**	5	2
1,479-variable Truss Design	3	5	**1**	4	2
544-variable Rocket Design	4	3	**1**	5	2
Final Rank	3	3	**1**	4	2

6. Conclusions and future work

In this article, a framework is proposed in which guidance from experienced users in terms of additional problem information is sought to enable repair-based algorithms to focus on identified variable combinations. The following observations can be made from the systematic study presented in this article:

- Amount of additional problem information provided to an algorithm is crucial to achieve the best performance.
- Too little or too much utilization of the supplied information within an optimization algorithm is also found to be harmful.
- Since the ideal amount of problem information and its utilization are not known *a priori* for a problem, an adaptive ensemble-based method has been proposed, which is found to provide a good overall compromise.
- Adequate additional problem information and its adequate utilization within an MOEA allows a faster convergence (requiring only 22% to 45% overall function evaluations in the current tests) compared to the baseline MOEA in solving the two large-scale problems with 544 to 1,479 real and integer variables, and involving time-consuming evaluation procedures.

This study has demonstrated a path towards achieving a truly interactive optimization algorithm, but further investigation is needed to include more generic relationships (such as, power laws, decision trees, periodic relations, and combinations thereof) intermittently guided by experienced users and automatically detected and validated by machine learning based optimization algorithms.

Data availability statement

The data that support the findings of this study are available from the corresponding author upon request.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

The support for Abhiroop Ghosh is provided by Koenig Endowed Chair funding to Kalyanmoy Deb under MSU Grant RT083557.

References

Bandaru, Sunith, and Kalyanmoy Deb. 2013. "Higher and lower-level knowledge discovery from Pareto-optimal sets." In *Journal of Global Optimization*, Vol. 57, oct, 281–298. Springer.

Coello, Carlos Artemio Coello, and Ma Guadalupe Castillo Tapia. 2021. "Cultural Algorithms for Optimization." In *Handbook of AI-based Metaheuristics*, 219–238. CRC Press.

Deb, K., and C. Myburgh. 2017. "A Population-Based Fast Algorithm for a Billion-Dimensional Resource Allocation Problem with Integer Variables." *European Journal of Operational Research* 261 (2): 460–474.

Deb, Kalyanmoy, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. 2002. "A fast and elitist multiobjective genetic algorithm: NSGA-II." *IEEE Transactions on Evolutionary Computation* 6 (2): 182–197.

Galanti, Geri Ann. 2000. "An introduction to cultural differences." In *Western Journal of Medicine*, edited by Anthony V. Sebald and Lawrence J. Fogel, Vol. 172, 335–336. World Scientific Press.

Gandomi, Amir H., Kalyanmoy Deb, Ronald C. Averill, Shahryar Rahnamayan, and Mohammad Nabi Omidvar. 2019. "Using semi-independent variables to enhance optimization search." *Expert Systems with Applications* 120: 279–297.

Gaur, Abhinav, and Kalyanmoy Deb. 2016. "Adaptive use of innovization principles for a faster convergence of evolutionary multi-objective optimization algorithms." In *GECCO 2016 Companion - Proceedings of the 2016 Genetic and Evolutionary Computation Conference*, 75–76. Association for Computing Machinery, Inc.

Ghosh, Abhiroop, Kalyanmoy Deb, Ronald Averill, and Erik Goodman. 2021. "Combining User Knowledge and Online Innovization for Faster Solution to Multi-objective Design Optimization Problems." In *Evolutionary Multi-Criterion Optimization*, Cham, 102–114. Springer International Publishing.

Ghosh, Abhiroop, Erik Goodman, Kalyanmoy Deb, Ronald Averill, and Alejandro Diaz. 2020. "A Large-scale Bi-objective Optimization of Solid Rocket Motors Using Innovization." *2020 IEEE Congress on Evolutionary Computation (CEC)* 1–8.

Landa-Becerra, Ricardo, Luis V. Santana-Quintero, and Carlos A. Coello. 2008. "Knowledge incorporation in multi-objective evolutionary algorithms." *Studies in Computational Intelligence* 98: 23–46.

Obayashi, Shigeru, and Daisuke Sasaki. 2003. "Visualization and data mining of Pareto solutions using Self-Organizing Map." *Lecture Notes in Computer Science* 2632: 796–809.

Szollos, András, Miroslav Šmíd, and Jaroslav Hájek. 2009. "Aerodynamic optimization via multi-objective micro-genetic algorithm with range adaptation, knowledge-based reinitialization, crowding and ϵ-dominance." *Advances in Engineering Software* 40 (6): 419–430.

Zitzler, Eckart, and Lothar Thiele. 1998. "Multiobjective optimization using evolutionary algorithms - A comparative case study." In *Lecture Notes in Computer Science*, Vol. 1498 LNCS, 292–301. Springer Verlag.