Towards An Automated Innovization Method for Handling Discrete Search Spaces

Abhinav Gaur and Kalyanmoy Deb
Computational Optimization and Innovation Laboratory (COIN)
Department of Electrical and Computer Engineering
Michigan State University
East Lansing, MI 48824, USA
Email: {gaurabhi,kdeb}@msu.edu

COIN Report Number 2015001

***Abstract*—Following manual observation of hidden relationships present in Pareto-optimal (PO) solutions of a multi-objective optimization problem, an automated *Innovization* procedure was suggested earlier for extracting innovative design principles. The goal was to obtain closed form and simple to understand relations that exist among PO solutions in a design or other problems. The proposed automated Innovization method was developed for handling continuous variable spaces. Since, most practical design problems have discrete variables in their descriptions, the aim of this study is to extend the earlier automated Innovization procedure to handle discrete variable spaces. We discuss the difficulties posed to an automated procedure due the search space granularity and demonstrate the working of our proposed method on one numerical problem and two engineering design problems. Our study amply demonstrates that the extension of a real-parameter automated Innovization is not straightforward to discrete spaces, however such a procedure for discrete spaces raises new challenges which must be addressed for handling problems with mixed continuous-discrete search space problems.**

***Keywords*—Innovization, Design Principles, Discrete space, Knowledge Mining. Multi-objective optimization.**

I. Introduction

The term *Innovization* has its genesis in [1] and it can be summarized as the process of finding Innovative design principles stored in Pareto-optimal solutions of a multi-objective optimization problem. The task is to first obtained a set of Pareto-optimal (PO) solutions and then identify meaningful relationships common to the Pareto-optimal solutions. Automated Innovization is the way of automating this process and [2] presented a method to do so for design problems involving continuous variables.

There are some studies in the domain of post-optimality analysis using data mining techniques for achieving a similar task. However, most of these studies provide visual information and need human interaction for a complete analysis. For example, [3] used k-means clustering on the trade-off solutions to simplify the task of analyzing them. Dendograms are used to depict strongly related decision variables from PO solutions in [4]. [5] used a combination of kriging and self-organizing maps to visualize the structure of decision variables using non-dominated solutions. [6] proposed heatmap visualization inspired by micro-array analysis. [7] proposed the idea of 'Pareto shells' to visually analyze PO solutions in multi-objective optimization problems. [8] used proper orthogonal decomposition to separate the design vector into mean and fluctuating vectors for a further analysis.

There are some other studies that represent extracted knowledge with 'if-then' type of rules using association rule mining method [9] or rough set theory [10]. [11] derives useful relationships by conducting a decision tree analysis of the solution sets from multi-objective optimizations. Such rules, though very helpful in some ways, is not compact and crisp. For a multi-dimensional data set, a typical rule associating decision tree to a region of the PO front may be very deep and provide unnecessarily detailed relationships. Such knowledge may be good for prediction but not for enhancing our understanding of the underlying system.

Another class of studies used data modeling techniques, such as regression, multivariate adaptive regression splines (MARS) [12] and kriging [13]. These techniques come up with complex closed-form mathematical expressions for the relations for the entire data. Being flexible, these techniques change the mathematical function to suite the complexities of the relations, hence giving closed-form yet meaningless rules from the designer's perspective. A previous study [2] provided a clear distinction of the rules obtained from an automated Innovization technique from those obtained from the MARS approach. A recent work [14] applied the automated Innovization process to obtain many interesting design principles on three real-world engineering design problems. This study used a machine learning based optimization method and provided the first step needed to develop an automated Innovization procedure. However, the existing automated Innovization method has been developed only for continuous variable spaces and the handling of discrete variables has been postponed, due to apparent complexities involved in dealing with the granularity of the search space under discretization. However, most variables in a real-world optimization problem are discrete in nature, allowing a few discrete options or values to be set for a variable. Thus, it has always been motivating to us to develop an automated Innovization procedure for handling discrete search space. We make an initial attempt here in this direction and reveal some

interesting and challenging issues related to discrete-variable handling.

In the remainder of this paper, Section II discusses the existing automated Innovization method briefly. Then, Section III discusses the proposed automated Innovization method by illustrating it using an easy-to-understand numerically generated problem. Section IV discusses the results for two engineering design problems which were also used in the original automated Innovization study, thereby allowing us to differentiate and bring out the salient differences between handling discrete and continuous variables. Finally, Section V concludes this study and proposed some future extensions of this work.

II. Existing Automated Innovization Method

An earlier study [2] made the first attempt at automating the process of Innovization. It is important to note that this method in its present form is designed to work with continuous variables only. It is a two step process, along with a post processing step. The first step involves finding a set of high-performing solutions using any multi-objective optimization procedure. All results in the earlier study used an evolutionary multi-objective optimization (EMO) method (NSGA-II [15]) to generate a set of near-Pareto-optimal solutions in a single simulation. The second step involves analyzing the obtained trade-off solutions to extract important design principles hidden in them. The post processing step then removes redundant rules and reveals the important design principles. Next, we discuss these steps one by one.

A. Step 1: Finding Set of Pareto-optimal Solutions

In this step, a set of PO or near-PO solutions are obtained using a modified *NSGA-II* [15] (called *NSGA-II-m* here), inspired by [16]. The NSGA-II-m method is capable of handling constraints without the use of any additional parameter. If discrete variables are present in the optimization problem, they were handled using the discrete versions of SBX and polynomial mutation operators [17]. Thus, NSGA-II-m is capable of handling mixed discrete and continuous variables at the optimization step, the subsequent automated Innovization procedure was capable of handling continuous variables alone. We address this issue in this work.

B. Step 2: Extracting Novel Design Principles from Pareto-optimal Solutions

In the PO data, the existing automated Innovization method searches for rules of the following mathematical structure,

$$\prod_{j=1}^{N} \phi_j(\mathbf{x})^{a_j b_j} = c. \tag{1}$$

In (1), ϕ_j's are symbolic entities, which can be design variables, objectives, constraints or any other entity, which depends upon the design variables of the problem. They are also referred to as *basis functions*. The parameter N is the number of such symbolic entities which is decided a priori. With the Boolean variables, a_j's, and real exponents, b_j's, any rule of the power-law form can be generated. The reason for assuming such a power-law form are; (a) earlier works [1], [18] of manual Innovization found rules with such power-law structure existing in the PO set in many engineering design problems and (b) [19] argues that such structures are common in physical systems. To avoid multi-modalities because of real values of b_j's (e.g. $\phi_1\phi_2^2$ and $\phi_1^2\phi_2^4$ are identical rules), following transformation is applied to b_j's,

$$b_j = \frac{b_j}{\{b_p | p = (argmax_p |a_p b_p)\}}, \tag{2}$$

to have $b_j \in [-1, 1] \; \forall \; j = 1$ to N. For any candidate rule of the form (1), two issues need to be assessed to ascertain its importance relative to other candidate rules. (A) How many points of the PO set adhere to this rule, and (B) of the ones that do adhere to the rule, how closely those PO points adhere to the rule. Answering these questions requires a clustering approach.

1) Grid-based Clustering: The existing automated Innovization method uses a *grid-based clustering* method, which first superimposes a grid on the input data points. It then flags the grid divisions which have more number of points than some threshold value and calls them sub-clusters. It then goes on to coalesce adjacent sub-clusters and this way it finally identifies regions of high point density or clusters. We now explain the grid-based clustering using the terminology used in this work . Let,
d : be the number of grid divisions,
m : be the number of PO points supplied,
P_t :be the number of points in tth division.
For any candidate rule of the form (1), the clustering is done as follows (refer Figure 1):

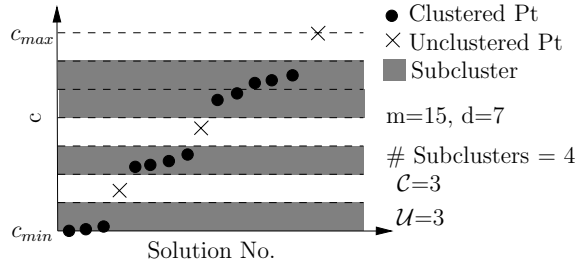

Fig. 1. Grid clustering explained. (Image Source : [20])

1) Evaluate the rule for all m PO solutions to get c-values and sort them.
2) Divide the range $[c_{min}, c_{max}]$ into d equal divisions.
3) Count the number of points in each division.
4) Label the divisions, that satisfy $P_t \geq \lfloor \frac{m}{d} \rfloor$, as *sub-clusters*. The points belonging to divisions that do not satisfy this inequality, label them as *unclustered* points.
5) Coalesce adjacent sub-clusters.
6) Count number of clusters $\mathcal{C}$ and total number of unclustered points $\mathcal{U}$.

2) Solving Optimization Problem: Now, to find the best design rules, the method solves the optimization problem given in (3) which uses the grid-based clustering for evaluation of objective value. To solve this optimization problem, the method uses NSGA-II with a *niched-tournament selection* operator

[21] to solve (3). This helps in obtaining multiple competent rules in the final population of the GA.

$$\begin{aligned}
\text{Minimize} \quad & \mathcal{C} + \sum_{k=1}^{\mathcal{C}} c_v^{(k)} \times 100 \\
\text{Subject to} \quad & 1 \leq \sum_{j=1}^{\mathcal{N}} a_j \leq \mathcal{N}, \\
& -1.0 \leq b_j \leq 1.0 \; \forall \; j : a_j = 1, \\
& |b_j| \geq 0.1 \; \forall \; j : a_j = 1, \\
& 1 \leq d \leq m, \\
& \mathcal{U} = 0, \\
& S = \frac{m - \mathcal{U}'}{m} \times 100\% \geq S_{reqd}, \\
\text{Where} \quad & \mathcal{N} : \text{Max allowed basis functions,} \\
& \text{in a rule. } \mathcal{N}(< N) \text{ is chosen a priori,} \\
& N : \text{Total no. of basis functions,} \\
& a_j's \text{ are Boolean } \forall \; j \in \{1, 2, \ldots, \mathcal{N}\}, \\
& b_j's \text{ are Real } \forall \; j \in \{1, 2, \ldots, \mathcal{N}\}, \text{ and} \\
& d\text{: \# of divisions, } m\text{: \# of PO pts,}
\end{aligned} \tag{3}$$

In (3), the objective function is a weighted combination of number of clusters ($\mathcal{C}$) and sum of coefficient of variation (c_v) in percent for all clusters. In the first constraint, the lower bound prevents trivial design rules by not allowing any design rules which have no basis functions. For, such a rule will be evaluated $= 1$ for all PO points and will have a $c_v = 0$. The upper bound in first constraint is to avoid very complex rules involving large number of basis functions. The third constraint is also to avoid trivial relationships. If for a rule, b_j's are allowed to be arbitrarily close to zero, then for such a rule all PO points will evaluate to ≈ 1. Such a rule will have artificially low c_v. The absence of unclustered points, i.e. the constraint $\mathcal{U} = 0$, ensures a high value of d so that a fine grid is possible for clustering. This fine mesh is good for accuracy as noisy data points form there own clusters. At the same time, this artificial constraint make number of clusters in the data artificially large. Hence one separate step is needed to reject clusters which have very low number of points. This is done by re-evaluating the number of actual unclustered points, $\mathcal{U}'$, by classifying a set of points within a grid division as unclustered if number of points in that cluster are $\leq \lfloor \frac{m}{d} \rfloor + \epsilon$. The last constraint lets the user define a minimum allowable percent significance (S_{reqd}), that is, a minimum number of clustered points for a rule.

C. Step 3: Post processing in Automated Innovization

Upon solving (3) for a PO dataset, we get a number of unique design rules (in the form of an *exponent matrix*) that exist in the data. An exponent matrix is a matrix which has the same number of rows as the number of different design rules obtained in the output of (3) and the number of columns equaling the number of basis functions. Each row of the exponent matrix represents a design rule and each element of a row being the exponent of corresponding basis functions in the design rule.

Now, these design rules also have some redundancies. Consider and example problem in which we have three basis functions, namely ϕ_1, ϕ_2 and ϕ_3. Further assume that the following design rules are obtained as the output of (3) for the data; $\phi_1\phi_3^{-1} = c_1$, $\phi_1\phi_2 = c_2$ and $\phi_2\phi_3 = c_3$. Then, the exponent matrix of these rules will be as shown in Table I. $\phi_1\phi_3^{-1} = c_1$ can be obtained from the rules

TABLE I. AN EXAMPLE OF AN EXPONENT MATRIX.

1	0	-1
1	1	0
0	1	1

$\phi_1\phi_2 = c_2$ and $\phi_2\phi_3 = c_3$. If all three are present in the output of the optimization algorithm, then we have to eliminate the redundant rules. The author of [20] suggests using row-reduced-echelon form in order to eliminate some of the redundant rules from the output of Innovization algorithms. In linear algebra, reduced row echelon forms (RREF) are used to identify linearly dependent rows of a matrix. When design principles of the form given in (1) are combined, the exponents undergo addition or subtraction operations. Therefore, a matrix formed by the exponents of all design principles can be reduced to a condensed set of design principles. A tolerance parameter is required for obtaining RREF. Any element of the matrix of exponents whose absolute value becomes lower than 'tolerance' during row operations is replaced with a zero. An appropriate value of the tolerance value is a matter of trial and error. To remove redundancy in the exponent matrix, we employ a method suggested by [20] with a slight modification. We first round all the elements of the exponent matrix to three decimal places. begin with a very low initial value for tolerance (0.001) and use this tolerance value to find the RREF of the exponent matrix. If the resultant matrix is of a lower rank, it means that one or more dependent rows were eliminated and a reduced set of design principles is obtained. In case, the rank remains the same, then tolerance value is increased by 0.001 and the process is repeated until, either the rank of exponent matrix reduces or the tolerance value reaches 1.0, whichever comes earlier. The condensed form of rules obtained from this strategy loose out on the accuracy of the exponents. Thus the significance-value have to be re-evaluated to confirm if the condensed set of design rules indeed have desired significance.

D. Difficulties With The Above Automated Innovization Method

Below, we list some of the difficulties encountered with the above-described automated Innovization method. The first two are particular to discrete search spaces. The third one is more general and applies to continuous search spaces as well.

1) In a rule of the form (1), if all the basis functions are of discrete nature, then the grid-based clustering wrongly captures it as a parametric form of rule. Such a rule is of no significance as we already know that natural clusters exist in discrete search space.
2) Each basis function that has discreteness in it, is captured as a rule. Due to this, not only does the method captures wrong rules, but also makes it difficult to obtain other rules by reducing exponent matrix using RREF because such rules directly provide pivot rows to the exponent matrix and coupled relations among the basis functions become difficult to come decipher.

3) The parameter ϵ in (3) indirectly controls the quality of clustering and is not very intuitive for the user to relate to the common metrics of assessing the quality of a cluster, i.e. the c_v of values over a cluster. Hence, this parameter is difficult to set initially.

III. Proposed Innovization Method for Handling Discrete Search Spaces

A. Method

Like the existing automated Innovization method, the proposed method is also a two-step process with a final post processing step. The first step (Section II-A) and post processing step (Section II-C) are the same in both methods. The second step of proposed method differs from that of existing method which we explain further below.

As in the existing method, the proposed method also searches for rules of the form given in (1). Also, the exponents b_j's are transformed as shown in (2). From this point on, the proposed method is different from the existing automated Innovization method of Section II. The proposed method uses a 'window clustering' approach, instead of grid-based clustering method. This is because the existing Innovization method assumes the existence of parametric form of rules to exist in the PO set. The Figure 1 shows an example of such a rule. When represented like this, parametric rules are characterized by disconnected regions of almost constant c value. This poses a problem when a design rule is composed of a single discrete variable, such as $x_i = c$, because the grid-based clustering method will always capture the above as a rule, as x_i naturally takes a set of different discrete values. But in reality, it is a triviality and is not a rule which we would like to rediscover.

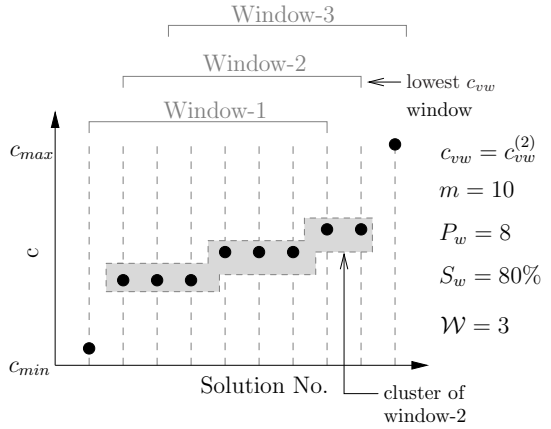

Fig. 2. Window clustering method is described.

1) Window Clustering: We now explain our proposed window-based clustering using the following terminology. Let,
m : be the number of PO points supplied,
S_w : be the % of points out of m points that should constitute a window cluster (provided by user).
P_w : be the # of points out of m points that should constitute a window.
$\mathcal{W}$: be the number of windows of contiguous P_w points possible in a total of m contiguous points.
$c_{vw}^{(k)}$: be the c_v for the for the points belonging to the k^{th} window.
c_{vw} : be the c_v for a rule.
For any candidate rule of the form (1), the clustering is done as follows (refer Figure 2):

1) Evaluate the rule for all m PO solutions to get m c-values and sort them in ascending order.
2) Evaluate P_w using $P_w = \lfloor S_w \times m \rfloor$.
3) Evaluate $\mathcal{W}$ using $\mathcal{W} = m - P_w + 1$.
4) Now, for each of the $\mathcal{W}$ windows, evaluate $c_{vw}^{(k)}$, where $k = 1, 2, \cdots, \mathcal{W}$.
5) Choose the lowest c_v among all $\mathcal{W}$ windows to be the c_v for the rule under consideration, i.e. c_{vw}.

2) Solving Optimization Problem: Now, the method solves the optimization problem given in (4) which uses the window clustering for evaluation of objective value. As in the existing Innovization method, the method uses NSGA-II with a *niched-tournament* selection operator [21] to solve (4).

$$\begin{aligned}
\text{Minimize} \quad & c_{vw} \\
\text{Subject to} \quad & 1 \leq \sum_{j=1}^{\mathcal{N}} a_j \leq \mathcal{N}, \\
& -1.0 \leq b_j \leq 1.0 \; \forall \; j : a_j = 1, \\
& |b_j| \geq 0.1 \; \forall \; j : a_j = 1, \\
& c_{vw} \leq c_{vw}^{(max)}, \\
\text{Where} \quad & c_{vw} \text{ : coeff. of var. for a rule,} \\
& m \text{ : \# of PO points,} \\
& S_w \text{ : desired window significance in } \%, \\
& \mathcal{N} \text{ : Max allowed basis functions.}
\end{aligned} \tag{4}$$

The proposed change in the Innovization method (refer Section III-A) addresses all the issues associated with existing Innovization method discussed in Section II-D. We discuss how the proposed method addresses these issues one by one.

1) The new method captures a rule only if the rule has at least $S_w\%$ contiguous c-values that together have a c_v below an acceptable limit, i.e. $c_{vw} \leq c_{vw}^{(max)}$.
2) In the form of $c_{vw}^{(max)}$, this method gives a direct control handle to the user to define what is an acceptable quality for a rule in terms of the overall c_v of the cluster window, i.e c_{vw}. Setting a $c_{vw}^{(max)}$ is very intuitive and can be known a priori by the user.
3) Since, the wrong rules are not captured (by virtue of discrete nature of variables), it is easier to reduce the exponent matrix using RREF.

B. Proof of Principle Results

We choose a modified version of the ZDT1 problem [17] to illustrate the working of our proposed discrete automated Innovization method. The variable x_1 is real and variables x_2

and x_3 are discrete.

$$\begin{aligned} \text{Minimize} \quad & x_1, \\ \text{Minimize} \quad & g\left(1-\sqrt{\frac{x_1}{g}}\right), \\ \text{Subject to} \quad & 0 \le x_i \le 1.0 \;\forall\; i \in \{1,2,3\}, \\ \text{Where} \quad & g = 1 + 9\left(x_2 x_3 - 0.5\right)^2 \text{ and,} \\ & x_1 \text{ is real, } x_2 \text{ and } x_3 \text{ are discrete.} \end{aligned} \tag{5}$$

The PO region for the above problem satisfies the following conditions:

$$0 \le x_i^* \le 1, \text{ for } i = \{1,2,3\} \text{ and } x_2^* x_3^* = 0.5, \tag{6}$$

Using this knowledge, we numerically generate 100 PO solutions for the above problem. Note that in later sections, when we apply Innovization procedure to engineering design problems, we generate the PO solutions by solving the design optimization problem using *NSGA-II-m* method.

Looking at (6), we see that value of x_1 is immaterial for the PO solutions and it can be any value in [0,1]. Hence for x_1, we randomly generate 100 values in [0,1] using random number generator. For the discrete variables x_2 and x_3, we first choose ten pairs of values that satisfy (6). These ten pairs are shown in Table II. We then take ten counts of each of these pairs which together comprise the 100 values each for x_2 and x_3. This way, we get 100 triplets, (x_1, x_2, x_3) as PO solutions to modified ZDT1 problem. This data of PO solutions is shown in Figure 3. For clarity in figure, the variables x_1 and x_2 are sorted in ascending order and variable x_3 is shown in descending order. It is clear from (6) that the only rule that exists in the PO solutions of ZDT1m is $x_2 x_3 = c = 0.5$. The following is the set of basis functions (refer Section II-B) chosen for this problem:

$$\phi_1 = x_1, \; \phi_2 = x_2, \; \phi_3 = x_3. \tag{7}$$

Subsequently, the existing and proposed automated Innovization methods are applied to the above PO data. The optimization problems, given by (3) for the existing Innovization method and (4) for the proposed Innovization method, are solved using *NSGA-II*. The parameters used for the *NSGA-II* are given in Table III and they were kept the same for both the cases. The problem specific parameters used in solving (3) and (4) are given in Table IV. (3) (for existing Innovization method) or (4) (for proposed Innovization method) is solved with this PO data as one of the inputs.

Tables V and VI show the rules obtained using the existing Innovization method and the proposed Innovization method, respectively. The last column of Table V shows percentage of PO solutions ($\geq S_{reqd}$) that adhere to the corresponding rule. In Table VI, the last column shows the percent c_{vw} ($\leq c_{vw}^{(max)}$) for a window size of $S_w = 90\%$ of PO solutions adhering to the captured rule.

Analyzing Tables V and VI, we note that the existing method wrongly captures $x_2 = 0.736$ and $x_3 = 0.714$ as rules, whereas the proposed method rejects these rules. This is because, the existing Innovization method perceives these rules to be parametric rules existing in the PO set, but in reality these rules are just a manifestation of the discrete nature of variables x_2 and x_3. Having shown this proof-of-principle results with our proposed approach, we are now ready to present an application of the approach to two engineering design problems.

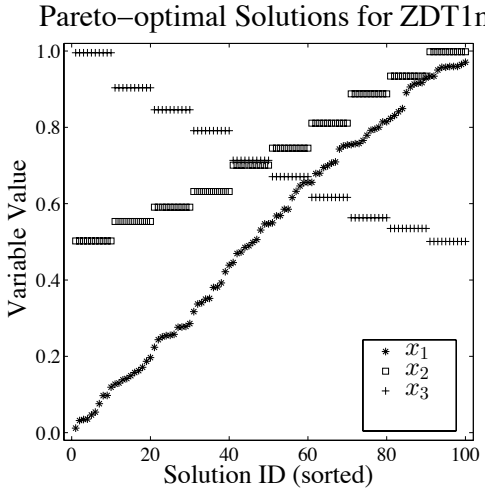

Fig. 3. Pareto-optimal solutions set for the modified ZDT1 problem.

TABLE II. VALUES OF DISCRETE VARIABLE PAIR x_2 AND x_3 IN PARETO-OPTIMAL SOLUTIONS FOR MODIFIED ZDT1 PROBLEM.

Pair #	Pair Values (x_2, x_3)	Product $x_2 \cdot x_3$
1	(0.502317112, 0.995387153)	0.5
2	(0.553326385, 0.903625805)	0.5
3	(0.590923514, 0.846133193)	0.5
4	(0.631901458, 0.791262615)	0.5
5	(0.700904017, 0.713364438)	0.5
6	(0.745432046, 0.670751952)	0.5
7	(0.811027566, 0.616501857)	0.5
8	(0.887856339, 0.563154170)	0.5
9	(0.934347353, 0.535132891)	0.5
10	(0.998067358, 0.500968192)	0.5

TABLE III. NSGA-II PARAMETERS USED IN EXISTING AND PROPOSED INNOVIZATION OF THE MODIFIED ZDT1 PROBLEM.

Parameter Name	Value
Population Size	100
# of Generations	100
Crossover Probability	0.95
Mutation Probability	0.05
SBX Distribution Index [22]	10
Polynomial Mutation Index [22]	50
Binary Variable Crossover Probability	0.85
Binary Variable Mutation Probability	0.15

TABLE IV. PROBLEM SPECIFIC PARAMETERS USED IN EXISTING AND PROPOSED INNOVIZATION OF THE MODIFIED ZDT1 PROBLEM.

Existing Innovization Method (3)		Proposed Innovization Method (4)	
$\mathcal{N}$	3	$\mathcal{N}$	3
N	3	N	3
m	100	m	100
S_{reqd}	90%	S_w	90%
ϵ	3	$c_{vw}^{(max)}$	5%

TABLE V. RULES FOR ZDT1M IDENTIFIED BY EXISTING AUTOMATED INNOVIZATION.

#	Rule	% Pts.
1	$x_2 x_3 = 0.500$	100
2	$x_2 = 0.736$	100
3	$x_3 = 0.714$	100

TABLE VI. RULES FOR ZDT1M IDENTIFIED BY PROPOSED AUTOMATED INNOVIZATION.

#	Rule	% c_{vw}
1	$x_2 x_3 = 0.500$	0.0

IV. Results

First, we consider a two-bar truss design problem.

A. Two-bar Truss Design Problem

The two-bar truss design optimization problem is described in [18] and is given as follows,

$$\begin{aligned} \text{Minimize} \quad & V = x_1\sqrt{16+y^2} + x_2\sqrt{1+y^2}, \\ \text{Minimize} \quad & S = \max(\sigma_{AC}, \sigma_{BC}), \\ \text{Subject to} \quad & \max(\sigma_{AC}, \sigma_{BC}) \leq 10^5, \\ & 0 \leq x_1, x_2 \leq 0.01, \\ & 1.0 \leq y \leq 3.0, \\ \text{where} \quad & \sigma_{AC} = \frac{20\sqrt{16+y^2}}{yx_1}, \sigma_{BC} = \frac{80\sqrt{1+y^2}}{yx_2}. \end{aligned} \tag{8}$$

1) Discretization and Obtaining Pareto-optimal Solutions: The original problem regards all three variables (x_1, x_2, y) as continuous. This paper introduces discreteness in the search space by considering variables x_1 and x_2 as discrete and variable y is kept continuous. This is because bars are manufactured with certain steps in cross-sectional areas, whereas the lengths of the bars can be cut with an increased precision. A total of *five* discrete cases along with the original continuous case are considered here. In discrete variable cases, the variables (x_1 and x_2) are allowed to take 10, 50, 100, 500 and $1,000$ equispaced values, respectively, in the bounds shown in (8). The PO solutions are obtained using NSGA-II-m. A population of 1,000 solutions with SBX recombination operator ($p_c = 0.95$ and $\eta_c = 10$) and polynomial mutation ($p_m = 0.05$ and $\eta_m = 50$) are evolved for 1,000 generations.

2) Obtaining Innovized Rules: We consider the following basis functions for the Innovization procedure:

$$\phi_1 = V,\ \phi_2 = S,\ \phi_3 = x_1,\ \phi_4 = x_2,\ \phi_5 = y. \tag{9}$$

The following design rules were discovered in [18] for this problem:

$$\frac{V}{x_2} = 2\sqrt{5},\ S.x_2 = \frac{200}{\sqrt{5}},\ \frac{x_1}{x_2} = 0.5,\ y = 2. \tag{10}$$

We apply the proposed automated Innovization method to the same PO solution set obtained in the previous section. Again, NSGA-II is used to solve (4) for obtaining the design principles. Tables VII and VIII show both the NSGA-II parameters as well as parameters specific to the optimization problems (3). The existing automated Innovization method captures all four rules shown in (10) for the continuous variable case as shown in a previous paper [20]. In the discrete version of the problem, the existing method wrongly captures the rules $x_1 = c$ and $x_2 = c$ as parametric rules for reasons detailed in Section III-B.

The results using the proposed Innovization method, when applied to the continuous and discrete cases of the truss design problem, are shown in Table IX. In each sub-table of Table IX, the condensed design principles (obtained by applying RREF (ref. Section II-C) on the exponent matrix) are shown on the left. Since, applying the RREF to obtain these condensed design principles changes the accuracy of their exponents (refer Section II-C), the c_{vw} for the desired clustering window size is re-evaluated for each condensed rule and are shown on right

TABLE VII. NSGA-II Parameters used in applying Proposed Innovization method to the Truss Design Problem.

Parameter Name	Value
Population Size	400
# of Generations	500
Crossover Probability	0.95
Mutation Probability	0.05
SBX Distribution Index	10
Polynomial Mutation Index	50
Binary Variable Crossover Probability	0.85
Binary Variable Mutation Probability	0.15

TABLE VIII. Problem Specific Parameters used in applying Existing and Proposed Innovization methods receptively to the Truss Design problem.

Existing Innovization Method (3)		Proposed Innovization Method (4)	
$\mathcal{N}$	3	$\mathcal{N}$	3
m	1000	m	1000
S_{reqd}	80%	S_w	80%
ϵ	3	$c_{vw}^{(max)}$	5%

TABLE IX. Tables for Rules in the Truss Design case. All the cases are individually presented.

A

Rules (Continuous Case)	$\%cv_w$
$V^{1.000}x_2^{-1.000} = 4.459$	0.24
$S^{1.000}x_2^{0.999} = 89.971$	0.26
$x_1^{1.000}x_2^{-1.000} = 0.500$	0.54
$y^{1.000} = 2.001$	0.85

B

Rules ($\mathcal{D}_{1000}$ case)	$\%cv_w$
$V^{1.000}x_2^{-1.000} = 4.470$	0.29
$S^{1.000}x_2^{1.000} = 89.579$	0.29
$x_1^{1.000}x_2^{-1.000} = 0.500$	0.66
$y^{1.000} = 1.998$	1.07

C

Rules ($\mathcal{D}_{500}$ Case)	$\%cv_w$
$V^{1.000}x_2^{-0.999} = 4.446$	0.37
$S^{1.000}x_2^{1.000} = 89.563$	0.37
$x_1^{1.000}x_2^{-1.002} = 0.504$	0.88
$y^{1.000} = 2.004$	1.47

D

Rules ($\mathcal{D}_{100}$ Case)	$\%cv_w$
$V^{1.000}x_2^{-1.001} = 4.481$	1.07
$S^{1.000}x_2^{1.002} = 88.969$	1.03
$x_1^{1.000}x_2^{-1.000} = 0.500$	2.58
$y^{1.000} = 1.999$	4.50

E

Rules ($\mathcal{D}_{50}$ Case)	$\%cv_w$
$V^{1.000}x_2^{-0.999} = 4.439$	1.99
$S^{1.000}x_2^{1.006} = 87.292$	1.96
$x_1^{1.000}x_2^{-1.001} = 0.502$	4.58
$y^{1.000} = 1.970$	8.35

F

Rules ($\mathcal{D}_{10}$ Case)	$\%cv_w$
$V^{1.000}x_2^{-0.994} = 4.283$	9.20
$S^{1.000}x_2^{0.999} = 94.375$	7.51
$x_1^{1.000}x_2^{-0.967} = 0.427$	11.03
$y^{1.000} = 1.850$	20.85

of each sub-table of Table IX. If the c_{vw} for any case crosses the desired upper bound of $c_{vw}^{(max)} = 5\%$, it is considered a failure to capture the rule. We conclude from these results that the proposed Innovization method

- captures all four rules in (10) with desired significance for following cases: Continuous, $\mathcal{D}_{1000}$, $\mathcal{D}_{500}$, $\mathcal{D}_{100}$,
- captures three out of four rules in (10) with desired significance for $\mathcal{D}_{50}$ case, and
- fails to capture any rules in (10) with desired significance for $\mathcal{D}_{10}$ case.

A quick comparison of the rules obtained in Tables IX-A and Table IX-F tells us that as the number of discrete options of a variable reduces in a PO set, the resulting Innovized rules may not remain the same. This is because the PO points may differ from the original continuous case, as the continuous PO solutions are now increasingly likely to be nonexistence in the discretized search space. Thus, the discretization of search space may cause the original continuous case design principles to be no more valid. To understand this matter, in Figure 4, we shows the PO front for the truss problem in two cases: (a) Continuous variable case, and (b) $\mathcal{D}_{10}$ case. Since, the PO front itself has changed significantly between the two cases,

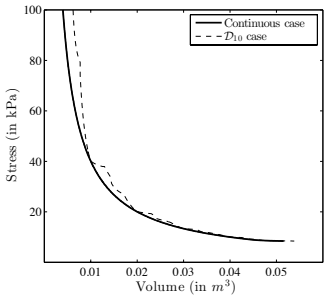

Fig. 4. Efficient fronts for the truss design problem for continuous and $\mathcal{D}_{10}$ discrete cases.

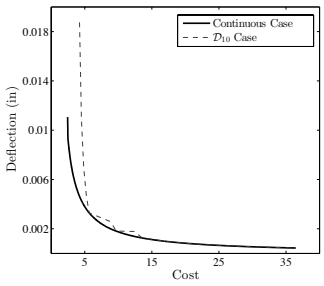

Fig. 5. Efficient fronts for the weld beam design problem for continuous and $\mathcal{D}_{10}$ discrete cases.

it is obvious that the Innovization algorithm will converge to a different set of rules. It is important to note that rules of the form (1) can change in one or more of three ways, (a) basis functions changes, (b) exponents of various terms in a design rule changes, and (c) the right side constant of the rule changes. Table IX shows the deviation of exponents and right side constant values of the rules from their original continuous case values to discrete case values.

B. Welded Beam Design Problem

Next, the well-studied welded beam design problem [18] is considered:

$$\begin{aligned}
\text{Minimize} \quad & C = 1.10471h^2l + 0.04811tb(14.0+l),\\
\text{Minimize} \quad & D = 2.1952/(t^3b),\\
\text{Subject to} \quad & g_1 = 13,600 - \tau \geq 0,\\
& g_2 = 30,000 - \sigma \geq 0,\\
& g_3 = b - h \geq 0,\\
& g_4 = P_c - 6,000 \geq 0,\\
& 0.125 \leq h, b \leq 5.0,\ 0.1 \leq l, t \leq 10.0,\\
\text{where} \quad & \hat{\tau} = 0.25(l^2 + (h+t)^2),\\
& \tau' = 6,000/\sqrt{2}hl,\\
& \tau'' = \frac{6,000(14+0.5l)\sqrt{\hat{\tau}}}{2\{0.707hl(\hat{\tau} - l^2/6)\}},\\
& \tau = \sqrt{(\tau')^2 + (\tau'')^2 + \frac{(l\tau'\tau'')}{\sqrt{\hat{\tau}}}},\\
& \sigma = \frac{504,000}{t^2b},\\
& P_c = 64,746.022(1 - 0.0282346t)tb^3.
\end{aligned} \tag{11}$$

1) Obtaining Pareto-optimal Solutions: [20] considered all four variables, namely h, l, t and b, of continuous type. To introduce discreteness in the search space, we consider the variables t, b, h as discrete and variable l as continuous. This is because, variables t and b directly controls the cross-sectional area of the beam, and beams are manufactured only in certain cross sectional areas. The variable h is considered discrete because the weld thickness can only be changed by certain amount in every weld pass and, the variable l is considered real as it can be varied arbitrarily.

As in the truss design problem, we consider $five$ discrete variable cases in which the variables h, t and b are allowed to take 10, 50, 100, 500 and 1,000 equispaced values within the box constraints mentioned in (11). We will refer to the PO solutions obtained in these cases as $\mathcal{D}_{10}$, $\mathcal{D}_{50}$, $\mathcal{D}_{100}$, $\mathcal{D}_{500}$ and $\mathcal{D}_{1000}$, respectively. The continuous variable case is also considered to show that the proposed Innovization method works in both the cases. The PO solutions are obtained using NSGA-II-m in both continuous and discrete cases. Identical parameter values to those used in the truss design problem are chosen here.

TABLE X. TABLES FOR RULES CAPTURED IN THE WELDED BEAM DESIGN PROBLEM. ALL THE CASES ARE INDIVIDUALLY PRESENTED.

A		B	
Rules (Continuous Case)	%cv$_w$	Rules ($\mathcal{D}_{1000}$ Case)	%cv$_w$
$t^{1.000} = 10.000$	0.00	$t^{1.000}D^{0.000} = 9.974$	0.04
$b^{1.000}D^{0.997} = 2.23\times10^{-3}$	0.22	$b^{1.000}D^{0.997} = 2.23\times10^{-3}$	0.17
$\sigma^{1.000}D^{-0.999} = 2.27\times10^{+6}$	0.11	$\sigma^{1.000}D^{-0.998} = 2.27\times10^{+6}$	0.09
$P_c^{1.000}D^{2.999} = 4.94\times10^{-3}$	0.08	$P_c^{1.000}D^{2.991} = 5.24\times10^{-3}$	0.53
C		D	
Rules ($\mathcal{D}_{500}$ Case)	%cv$_w$	Rules ($\mathcal{D}_{100}$ Case)	%cv$_w$
$t^{1.000}D^{-0.001} = 10.066$	0.15	$t^{1.000}D^{0.006} = 9.586$	0.30
$b^{1.000}D^{1.001} = 2.18\times10^{-3}$	0.32	$b^{1.000}D^{0.989} = 2.37\times10^{-3}$	0.86
$\sigma^{1.000}D^{-1.001} = 2.31\times10^{+6}$	0.15	$\sigma^{1.000}D^{-0.994} = 2.20\times10^{+6}$	0.30
$P_c^{1.000}D^{3.000} = 4.92\times10^{-3}$	0.76	$P_c^{1.000}D^{2.959} = 6.66\times10^{-3}$	2.46
E		F	
Rules ($\mathcal{D}_{50}$ Case)	%cv$_w$	Rules ($\mathcal{D}_{10}$ Case)	%cv$_w$
$t^{1.000}D^{0.002} = 9.868$	0.66	$t^{1.000} = 8.900$	0.00
$b^{1.000}D^{0.987} = 2.41\times10^{-3}$	1.83	$b^{1.000}D^{1.000} = 3.11\times10^{-3}$	0.00
$\sigma^{1.000}D^{-0.998} = 2.26\times10^{+6}$	0.66	$\sigma^{1.000}D^{-1.000} = 2.05\times10^{+6}$	0.01
$P_c^{1.000}D^{2.976} = 5.95\times10^{-3}$	5.49	$P_c^{1.000}D^{2.998} = 1.31\times10^{-2}$	0.05

2) Obtaining Innovized Rules: We consider the following basis functions for the Innovization procedures:

$$\phi_1 = D,\ \phi_2 = \sigma,\ \phi_3 = P_c,\ \phi_4 = t,\ \phi_5 = b. \tag{12}$$

In the truss design problem, the expected rules given in (10) are known analytically. This is not the case in the welded beam design problem. Instead, [20] obtains a set of rules for the problem by applying the existing Innovization method to the welded beam design problem of (11), for all the variables considered continuous. These rules are given below:

$$\begin{aligned}
t = 10, \qquad & bD^{0.999} = 2.20\times10^{-3},\\
\frac{\sigma^{0.999}}{D} = 2.30\times10^{6}, \quad & P_cD^{2.999} = 4.92\times10^{-3}.
\end{aligned} \tag{13}$$

After obtaining the PO solutions in the previous set, we apply the automated Innovization procedure to get the important design rules commonly present in them. The parameters for NSGA-II and the problem parameters for (4) are kept same as shown in Tables VII and VIII. The existing automated Innovization method captures all four rules for the continuous variable case as shown in [20]. Table X shows the results of the proposed Innovization method applied to the same PO solutions of welded beam design problem. In each sub-table of Table X, the obtained design principles on the exponent matrix obtained from Innovization algorithm are shown on the left. The c$_{vw}$ for the desired clustering window size is reevaluated for each condensed rule obtained and are shown against respective rule on the right of each subtable. If the $\text{c}_{vw}^{(max)}$ for any case crosses the desired upper bound of $\text{c}_{vw}^{(max)} = 5\%$,

it is considered a failure to capture the desired rule. As is clear from these results, the proposed Innovization method

- captures all four rules in (10) with desired significance for following cases : Continuous, $\mathcal{D}_{1000}$, $\mathcal{D}_{500}$, $\mathcal{D}_{100}$, $\mathcal{D}_{10}$,
- captures three out of four rules in (10) with desired significance for $\mathcal{D}_{50}$ case.
- fails to capture any of the four rules in $\mathcal{D}_{10}$ PO set. Reason being, the PO set of the $\mathcal{D}_{10}$ is no longer able to reach near the true Pareto front. This is shown in Figure 5. Hence, in this new front, no the design rules loose there correlation and hence the algorithm fails to find any rule with $S_w = 80\%$ window size that has $c_{vw} \leq c_{vw}^{(max)} = 5\%$.

V. Conclusions and Future Extensions

Recent works [2], [14], [18], [20] have shown that the Innovization can improve our understanding of engineering design problems by extracting common principles that bind together high performing solutions in engineering design problems. In this work, we extended an existing automated Innovization method developed in [2] to be applied to problems with discrete variables involved in a design task. As an improved procedure, the proposed Innovization method has replaced a non-intuitive clustering parameter, which was difficult to guess for a user, with a more intuitive clustering parameter. The paper has demonstrated that as the number of points for a discrete variable is reduced, the hidden rules starts changing in many ways compared to the rules obtained for the same problem when the same variables are of continuous nature. A few ways in which this change can occur are in (a) basis functions, (b) the exponents of the basis functions, and (c) the constant value of the rule. This is a consequence of the discrete nature of the variables. The exact nature of change in rules as the granularity of the search space increases and their understanding are interesting aspects which would another level of knowledge which the designers would be interested in.

This study raises certain interesting features of discrete search spaces and proposes an automated Innovization procedure. As a future research, we plan to apply the method and study other forms of Innovization methods, such as *higher-level and lower-level Innovizations* [23] and *temporal Innovization* [24] applied to discrete search spaces. Variables can be sorted according to their largest effect (or sensitivity) in changing the Innovized principles. Finally, the proposed method must now be applied to more challenging discrete and mixed-integer problems. It will also be interesting to investigate how automated Innovization methods can be designed for solving combinatorial optimization problems.

References

[1] K. Deb, "Unveiling innovative design principles by means of multiple conflicting objectives," *Engineering Optimization*, vol. 35, no. 5, pp. 445–470, 2003.

[2] S. Bandaru and K. Deb, "Towards automating the discovery of certain innovative design principles through a clustering-based optimization technique," *Engineering optimization*, vol. 43, no. 9, pp. 911–941, 2011.

[3] H. Taboada and D. Coit, "Data mining techniques to facilitate the analysis of the pareto-optimal set for multiple objective problems," in *Proceedings of the 2006 Industrial Engineering Research Conference (CD-ROM)*, 2006.

[4] T. Ulrich, D. Brockhoff, and E. Zitzler, "Pattern identification in pareto-set approximations," in *Proceedings of the 10th annual conference on Genetic and evolutionary computation*. ACM, 2008, pp. 737–744.

[5] S. Obayashi, S. Jeong, and K. Chiba, "Multi-objective design exploration for aerodynamic configurations," *AIAA Paper*, vol. 4666, p. 2005, 2005.

[6] A. Pryke, S. Mostaghim, and A. Nazemi, "Heatmap visualization of population based multi objective algorithms," in *Evolutionary multi-criterion optimization*. Springer, 2007, pp. 361–375.

[7] D. J. Walker, R. M. Everson, and J. E. Fieldsend, "Visualisation and ordering of many-objective populations," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*. IEEE, 2010, pp. 1–8.

[8] A. Oyama, T. Nonomura, and K. Fujii, "Data mining of pareto-optimal transonic airfoil shapes using proper orthogonal decomposition," *Journal of Aircraft*, vol. 47, no. 5, pp. 1756–1762, 2010.

[9] K. Sugimura, S. Obayashi, and S. Jeong, "Multi-objective design exploration of a centrifugal impeller accompanied with a vaned diffuser," in *ASME/JSME 2007 5th Joint Fluids Engineering Conference*. American Society of Mechanical Engineers, 2007, pp. 939–946.

[10] ——, "Multi-objective optimization and design rule mining for an aerodynamically efficient and stable centrifugal impeller with a vaned diffuser," *Engineering Optimization*, vol. 42, no. 3, pp. 271–293, 2010.

[11] C. Dudas, A. H. Ng, L. Pehrsson, and H. Boström, "Integration of data mining and multi-objective optimisation for decision support in production systems development," *International Journal of Computer Integrated Manufacturing*, no. ahead-of-print, pp. 1–16, 2013.

[12] J. H. Friedman, "Multivariate adaptive regression splines," *The annals of statistics*, pp. 1–67, 1991.

[13] T. W. Simpson, J. Poplinski, P. N. Koch, and J. K. Allen, "Metamodels for computer-based engineering design: survey and recommendations," *Engineering with computers*, vol. 17, no. 2, pp. 129–150, 2001.

[14] K. Deb, S. Bandaru, D. Greiner, A. Gaspar-Cunha, and C. C. Tutum, "An integrated approach to automated innovization for discovering useful design principles: Case studies from engineering," *Applied Soft Computing*, vol. 15, pp. 42–56, 2014.

[15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, 2002.

[16] K. Deb and M. Goyal, "A flexible optimization procedure for mechanical component design based on genetic adaptive search," *Journal of Mechanical Design*, vol. 120, no. 2, pp. 162–164, 1998.

[17] K. Deb, *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, 2001, vol. 16.

[18] K. Deb and A. Srinivasan, "Innovization: Discovery of innovative design principles through multiobjective evolutionary optimization," in *Multiobjective Problem Solving from Nature*. Springer, 2008, pp. 243–262.

[19] M. E. Newman, "Power laws, pareto distributions and zipf's law," *Contemporary physics*, vol. 46, no. 5, pp. 323–351, 2005.

[20] S. Bandaru, "Automated innovization: Knowledge discovery through multi-objective optimization," Ph.D. dissertation, Indian Institute of Technology-Kanpur, Dept. of Mechanical Engineering, India, 2012.

[21] C. K. Oei, D. E. Goldberg, and S.-J. Chang, "Tournament selection, niching, and the preservation of diversity," *Urbana*, vol. 51, p. 61801, 1991.

[22] K. Deb, A. Anand, and D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter optimization," *Evolutionary computation*, vol. 10, no. 4, pp. 371–395, 2002.

[23] S. Bandaru and K. Deb, "Higher and lower-level knowledge discovery from pareto-optimal sets," *Journal of Global Optimization*, vol. 57, no. 2, pp. 281–298, 2013.

[24] K. Deb, S. Bandaru, and C. Celal Tutum, "Temporal evolution of design principles in engineering systems: Analogies with human evolution," in *Parallel Problem Solving from Nature - PPSN XII*, ser. Lecture Notes in Computer Science, C. Coello, V. Cutello,

K. Deb, S. Forrest, G. Nicosia, and M. Pavone, Eds. Springer Berlin Heidelberg, 2012, vol. 7492, pp. 1–10. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-32964-7_1