Virtual Reality High Dimensional Objective Spaces for Multi-Objective Optimization: An Improved Representation

Valdés, Julio; Barton, Alan; Orchard, Robert

*Virtual Reality High Dimensional Objective Spaces for Multi-Objective Optimization: An Improved Representation **

Valdés, J., Barton, A., and Orchard, R.
2007

* Proceedings of the IEEE Congress on Evolutionary Computation. Singapore. September 25-28, 2007. NRC 49364.

Virtual Reality High Dimensional Objective Spaces for Multi-objective Optimization: An Improved Representation

Julio J. Valdés, Alan J. Barton and Robert Orchard

Abstract— **This paper presents an approach for constructing improved visual representations of high dimensional objective spaces using virtual reality. These spaces arise from the solution of multi-objective optimization problems with more than 3 objective functions which lead to high dimensional Pareto fronts. The 3-D representations of m-dimensional Pareto fronts, or their approximations, are constructed via similarity structure mappings between the original objective spaces and the 3-D space. Alpha shapes are introduced for the representation and compared with previous approaches based on convex hulls. In addition, the mappings minimizing a measure of the amount of dissimilarity loss are obtained via genetic programming. This approach is preliminarily investigated using both theoretically derived high dimensional Pareto fronts for a test problem (DTLZ2) and practically obtained objective spaces for the 4 dimensional knapsack problem via multi-objective evolutionary algorithms like HLGA, NSGA, and VEGA. The improved representation captures more accurately the real nature of the m-dimensional objective spaces and the quality of the mappings obtained with genetic programming is equivalent to those computed with classical optimization algorithms.**

I. Introduction

In multi-objective optimization, rather than finding a single best solution for a given problem, what is found is a set of "compromise" solutions from which the decision maker selects a particular one, based on additional domain knowledge. For up to three objectives, a scatter plot suffices for displaying the set of solutions on which the decision will be made, but this approach is no longer possible when the problem involves more than 3 objectives. This prevents the decision maker from using visual information in the decision making process and therefore to use the powerful geometric pattern processing capabilities of the human brain.

The development of suitable visualization techniques for multi-objective optimization is considered one important open problem. A first general approach addressing this issue using similarity structure preservation mappings was proposed [9] where the mapping between the two spaces can be computed with or without evolutionary computation methods.

In this paper, alpha shapes are introduced for constructing visual represents and compared with previous approaches based on convex hulls [9]. In addition, the mappings minimizing a measure of the amount of dissimilarity loss are obtained via genetic programming [8]. The approach is illustrated using: *i)* the representation of a collection of increasingly high dimensional theoretical Pareto fronts ($3, 4$ and 10) derived from a test problem (DTLZ2) (also covered by [9]), and *ii)* the comparison of approximations to the Pareto front of a real problem, obtained with different evolutionary computation-based multi-objective optimization methods.

J. J. Valdés, A. J. Barton and Robert Orchard are with the National Research Council Canada's Institute for Information Technology's Integrated Reasoning Group, 1200 Montreal Road, Ottawa, Ontario, Canada, K1A 0R6 (email: julio.valdes@, alan.barton@ and robert.orchard@nrc-cnrc.gc.ca).

II. Multi-objective Optimization

Multi-objective optimization (MOO) studies optimization problems involving more than one objective function and the goal is to find one or more optimal solutions. Most real world problems involve multiple objectives and typically different solutions lead to conflicting scenarios: a solution which is optimal in the sense of a given objective might not be from the point of view of one or more of the other objectives [17]. Therefore, the goal is to find a set of optimal solutions representing the best trade-offs, from which the user, with further higher level information about the problem can make a decision.

Most multi-objective algorithms use the concept of dominance in their formulation. A solution $\overleftarrow{x}_{(1)}$ is said to dominate [18] a solution $\overleftarrow{x}_{(2)}$ for a set of m objective functions $< f_1(\overleftarrow{x}), f_2(\overleftarrow{x}), ..., f_m(\overleftarrow{x}) >$ if

1) $\overleftarrow{x}_{(1)}$ is not worse than $\overleftarrow{x}_{(2)}$ over all objectives. For example, $f_3(\overleftarrow{x}_{(1)}) \leq f_3(\overleftarrow{x}_{(2)})$ if $f_3(\overleftarrow{x})$ is a minimization objective.
2) $\overleftarrow{x}_{(1)}$ is strictly better than $\overleftarrow{x}_{(2)}$ in at least one objective. For example, $f_6(\overleftarrow{x}_{(1)}) > f_6(\overleftarrow{x}_{(2)})$ if $f_6(\overleftarrow{x})$ is a maximization objective.

Accordingly, the goals of multi-objective optimization can be summarized [17] as: *i)* to find a set of solutions as close as possible to the Pareto-optimal front, and *ii)* to find a set of solutions as diverse as possible.

It is natural to use use evolutionary algorithms for solving multi-objective optimization problems (MOEA in general and MOGA if based on genetic algorithms), because an evolutionary algorithm constructs a population of individuals, which evolve through time until stopping criteria are satisfied. Several algorithms inspired by this principle have been proposed. Among them, VEGA [19], HLGA [20], NSGA-NSGA-II [21], [22], [23], SPEA [24] and many others.

III. Visualizing Objective Spaces

General approaches for constructing m-dimensional Pareto-optimal fronts [27] have been previously reported in the literature for the purpose of systematically investigating the properties of multi-objective evolutionary algorithms (MOEAs) [28]. Some design characteristics of the approaches for test problem construction are that they should be: *i)* simple to implement, *ii)* scalable to any number of decision variables and objectives and *iii)* lead to knowledge of the exact shape and location of the resulting Pareto-optimal front. However, once such a front has been constructed, either theoretically [27] or empirically, it becomes of interest to investigate its properties.

A. Space Visualization

There are many possible paradigms for creating visual spaces within data mining. In particular Virtual Reality (VR) is a suitable paradigm. It is *flexible* and allows the user to be an active participant and control the information consumption. Also it is very important that no special background knowledge is required of the user. A virtual reality technique for visual data mining on heterogeneous, imprecise and incomplete information systems was introduced in [25], [26].

One of the steps in the construction of a VR space for data representation is the transformation of the original set of attributes describing the objects under study, often defining a heterogeneous high dimensional space, into another space of small dimension (typically 2-3) with intuitive metric (e.g. Euclidean). The operation usually involves a non-linear transformation; implying some information loss.

B. Space Taxonomy

From the point of view of the property(s) which the objects in the space must satisfy, several paradigms can be considered [31]:

- *Unsupervised*: The location of the objects in the space should preserve some structural property of the data, dependent only on the set of descriptor attributes.
- *Supervised*: The goal is to produce a space where the objects are maximally discriminated w.r.t. a class distribution.
- *Mixed*: A space compromising the two goals is sought. Very often these two goals are conflicting.

In this study, unsupervised spaces are constructed which preserves distance relationships between the elements of the objective space [29], [26]. The mapping φ can be constructed to maximize some metric/non-metric structure preservation criteria for example by minimizing some error measure of information loss [30]. If δ_{ij} is a dissimilarity measure between any two objects i, j in the high dimensional space and $\zeta_{i^v j^v}$ is another dissimilarity measure defined on objects i^v, j^v (the images of i, j in the target space). A frequently used error measure is:

$$\text{Sammon error} \quad = \quad \frac{1}{\sum_{i<j} \delta_{ij}} \frac{\sum_{i<j} \left(\delta_{ij} - \zeta_{i^v j^v}\right)^2}{\delta_{ij}} \quad (1)$$

C. Mapping Taxonomy

From the point of view of their mathematical nature, the mappings can be:

- *Implicit*: the images of the transformed objects are computed directly and the algorithm does not provide a function representation.
- *Explicit*: the function performing the mapping is found by the procedure and the images of the objects are obtained by applying the function. Two sub-types are:
  - *analytical functions*.
  - *general function approximators*: neural networks, fuzzy systems, or others.

D. Mapping Computation

Explicit mappings can be constructed in the form of analytical functions (e.g. via genetic programming), or using general function approximators like neural networks or fuzzy systems. An explicit mapping (e.g. φ) is very useful. On one hand, in dynamic data sets (e.g. incremental data bases) an explicit transform φ will increase the update rate of the VR information system. On another hand, it can give semantics to the attributes of the VR space, thus acting as a general dimensionality reducer.

Classical algorithms have been used for directly optimizing Eq-1 and similar, like Steepest descent, conjugate gradient Fletcher-Reeves, Powell, Levenberg-Marquardt, and others. For this study, the Fletcher-Reeves method, which is a well known technique used in deterministic optimization [7] was used. It assumes that the function f is roughly approximated as a quadratic form in the neighborhood of an N dimensional point $\mathbf{P}$ and it uses the information given by the partial derivatives of the original function f. This is the conjugate gradient family of minimization methods and requires an initial approximation to the solution (typically random), which is then refined in a sequence of iterative steps. The convergence of these methods is relatively fast, but on many occasions the obtained solutions are locally optimal.

E. Object Representation using Alpha Shapes

In the ideal case of a multi-objective optimization with m objectives, the Pareto front will be at most a $m-1$ dimensional object as a locus of the non-dominated solutions within the feasible space. For $m = 2$ it is a curve, for $m = 3$ a surface and for $m > 3$, it will be a hypersurface. When high dimensional spaces are mapped to lower dimensional spaces (in particular to 3-D for visualization) information loss is inevitable. Typically this loss increases with the number of objectives. In addition mapping the solutions on the hypersurface of the m-dimensional Pareto front to a 3-D surface will not be possible in general. Therefore, in general the 3-D mapped m-D front will be a solid object. In [9] that object is modeled by the convex hull of the 3-D images of the m-D front solutions, in order to obtain a simple, unique geometric object containing all of the optimal solutions.

However, a more general approach for representing the set of mapped solutions is possible by relaxing the constraint of the geometric object to contain all of them. In this way, instead of a single object, a set of related objects spawning a sequence ranging from all to none of the solutions will be more helpful in understanding the structure and composition of the m-D Pareto fronts and in decision making. The use of *alpha shapes* for this purpose is particularly appealing.

The concept of alpha shapes formalizes the intuitive notion of *shape* for spatial point set data and it is a generalization of the convex hull [10], [11], [16]. A finite set of points in 3-dimensional space and a real parameter alpha uniquely define a so-called simplicial complex, consisting of vertices, edges, triangles, and tetrahedra (a simplex is the convex hull of a set of $n+1$ affinely independent points, where n is the dimension of the space). This is the alpha-complex of the points and the alpha-shape is the geometric object defined as the union of the elements in the complex. Thus, alpha shapes generalize the notion of convex hull of the point set and when varying the parameter alpha, it ranges from crude to fine shapes. The most crude shape is the convex hull itself, which is obtained for very large values of alpha. As alpha decreases, the shape shrinks and develops cavities that may join to form tunnels and voids. For sufficiently small alpha, the alpha shape is empty. For each alpha, the alpha-complex is a subcomplex of the 3-dimensional Delaunay triangulation [12], [14]. In alpha-shapes, the parameter alpha controls the maximum curvature of any cavity of the resulting object, which consists of a number of straight edges and polygonal faces [13]. Alpha shapes were computed for the 4-D and 10-D Pareto fronts of the DTLZ2 test Problem as well as for the 4-D approximations of the Pareto front obtained with the HLGA, NSGA and VEGA algorithms applied to the Knapsack problem.

IV. Mapping Computation using Genetic Programming

Genetic programming (GP) techniques aim at evolving computer programs. They are an extension of the Genetic Algorithm introduced in [3] and further elaborated in [4], [5] and [6]. The algorithm starts with a set of randomly created computer programs. This initial population goes through a domain-independent breeding process over a series of generations. It employs the Darwinian principle of survival of the fittest with operations similar to those occurring naturally, like crossover, occasional mutation, duplication and gene deletion. A computer program is understood as an entity that receives inputs, performs computations which transform these inputs and produces some output in a finite amount of time. The operations include arithmetic computation (possibly involving many other functions), conditionals, iterations, recursions, code reuse and other kinds of information processing organized into a hierarchy. Genetic programming combines the expressive high level symbolic representations of computer programs with the search efficiency of the genetic algorithm. For a given problem, this process often results in an exact solution or if not, at least provides a fairly good approximation. Those programs which represent functions are of particular interest and can be modeled as $y = F(x_1, \cdots, x_n)$, where $(x_1, \cdots, x_n)$ is the set of independent or predictor variables, and y the dependent or predicted variable, so that $x_1, \cdots, x_n, y \in \mathbb{R}$, where $\mathbb{R}$ are the reals. The function F is built by assembling functional subtrees using a set of predefined primitive functions (the function set), defined beforehand. In general terms, the model describing the program is given by $y = F(\vec{x})$, where $y \in \mathbb{R}$ and $\vec{x} \in \mathbb{R}^n$. Most implementations of genetic programming for modeling fall within this paradigm but for some problems vector functions are required. Recently a GP based approach for finding vector functions was presented [8]. In these cases the model associated to the evolved programs is $\vec{y} = F(\vec{x})$. Note that these are *not* multi-objective problems, but problems where the fitness function depends on vector variables. The mapping problem between vectors of two spaces of different dimension (n and m) is one of that kind. In this case a transformation like $\varphi : \mathbb{R}^n \to \mathbb{R}^m$ mapping vectors $\vec{x} \in \mathbb{R}^n$ to *vectors* $\vec{y} \in \mathbb{R}^m$ would allow a reformulation of Eq. 1 as in Eq. 2:

$$\text{Sammon error} = \frac{1}{\sum_{i<j} \delta_{ij}} \frac{\sum_{i<j} (\delta_{ij} - d(\vec{y}_i, \vec{y}_j))^2}{\delta_{ij}}, \quad (2)$$

where $\vec{y}_i = \varphi(\vec{x}_i)$, $\vec{y}_j = \varphi(\vec{x}_j)$.

The implication from the point of view of genetic programming is that instead of evolving expression trees, where there is a one-to-one correspondence between an expression tree and a fitness function evaluation (the classical case), the evolution has to consider populations of *forests* such that the evaluation of the fitness function depends on the set of trees within a forest [8]. In these cases, the cardinality of any forest within the population is equal to the dimension of the target space m.

A. Gene Expression Programming

Gene Expression Programming (GEP) [2] is one of the many variants of GP and has a simple string representation for the expression tree. Source code is available [1] and was extended to evolve programs that represent vector functions. In the GEP algorithm, the individuals are encoded as simple strings of fixed length with a head and a tail, referred to as chromosomes. Each chromosome can be composed of one or more genes which hold individual mathematical expressions that are linked together to form a larger expression.

To facilitate experimentation with the GEP algorithm, an extension to an existing Java-based Evolution Computing Research System called ECJ [1] from George Mason University has been made. ECJ has an infrastructure to support various types of evolutionary computing, including Genetic Programming (GP). The extension, ECJ-GEP, implements almost all of the features of GEP addressing four basic types of problem: function finding, classification, time series, and logical. It supports a large set of fitness functions and allows expressions to be created using many types of arithmetic functions in such a way that further design modifications are straightforward. The problem parameters that determine which functions to use in expressions, how to perform the evolutionary operations, the size of the population, the number of generations to run, etc. are defined within parameter files. The system has been designed so that for most problems no code should need to be written. However, for those other problems, code may be incorporated in order to address the special need (e.g. to generate data, do something special with the fitness calculation, etc). There is a graphical interface and in addition batch files may be constructed for execution upon distributed systems (e.g. Condor: `http://www.cs.wisc.edu/condor/`) to allow for large scale experimentation.

For the research described in this paper, a new idea led to an extension of the GEP algorithm, whereby an individual in the population may have multiple chromosomes (a vector function). The chromosomes are independent but evolve together from generation to generation. This supports the population of *forests* needed for such problems.

B. A Theoretical Case: Test Problem DTLZ2

Test problem DTLZ2 [28] was chosen for the study, whose fronts are segments of spherical surfaces with a straightforward generalization towards higher dimensional objective spaces (hyperspherical surfaces) (Eq.3). A collection of 5 fronts with the following radii $r = \{1, 5, 10, 20, 50\}$ in objective spaces of dimension $M = \{3, 4, 5, 7, 10\}$ were generated. In all cases the dimension of the decision space was kept the same as the corresponding objective space. For each decision space a mesh of sampling points was generated so that each decision variable $x_i \in [1, M-1]$ was sampled from a regular interval in the $[0, 1]$ domain. If n_j is the number of sampling points along a given decision variable in a j-dimensional decision space, then for each dimension determined by $1/n_j$ the number of points in the theoretical front is n_j^{M-1}. In the study $j \in \{3, 4, 5, 7, 10\}$ and $n_j = \{5, 5, 3, 3, 3\}$. For $M = 3$ the sequence of theoretical Pareto fronts is shown in Fig.1. The chosen design allows: *i)* the simulation of a MO-process "converging" towards the first front of the sequence (lower right) and *ii)* the reproduction of *the same type of relationship* in dimensions larger than 3 which makes the interpretation of the obtained VR spaces easier.

C. An Applied Case: The Knapsack problem

A 0/1 knapsack problem [32], [33], [34] is a real world situation that consists of *i)* a set of items, *ii)* weight and profit associated with each item, and *iii)* an upper bound for the capacity of the knapsack. The task is to find a subset of items which will be placed within the knapsack, which maximizes the total profit in the subset (the total weight must not exceed the given capacity of the knapsack). This problem is extended to a the multi-objective case by allowing a given number of knapsacks.

This problem is included in the Test Problem Suite for Multiobjective Optimizers [35]. In particular, data and runs using a collection of different MOGA algorithms are compiled for the 750 items, 4 knapsacks case. From them, the results corresponding to objective spaces generated with the HLGA [20], NSGA [21] and VEGA [19] algorithms for optimization run 1 were used in this study (files HLGA.1, NSGA.1 and VEGA.1). A broad comparative study including these and other algorithms has been made [24]. The 4-dimensional objective spaces represent approximations to the Pareto front obtained with the algorithms mentioned and they were represented as VR 3-D spaces using the approach described here. The idea was to study the behavior of the proposed MO visualization technique using non-theoretical objective spaces. In this way, a visual comparison between the results given by the HLGA, NSGA and VEGA is possible, complementing the numeric comparison already made [24].

TABLE I
EXPERIMENTAL SETTINGS FOR GP-GEP.

GEP Parameter	Experimental Values
No. Generations	100, 500 and 1000
Population Size	100, 200 and 500
No. Chromosomes / Individual	3
No. Genes / Chromosome	5
Gene Head Size	5
Linking function	Addition
No. Constants / Gene	4
Bounded Range of Constants	$[1, 10]$
Inversion Rate	0.1
Mutation Rate	0.044
is-transposition Rate	0.1
ris-transposition Rate	0.1
One-point Recomb. Rate	0.3
Two-point Recomb. Rate	0.3
Gene Recombination Rate	0.1
Gene Transposition Rate	0.1
rnc-mutation Rate	0.01
dc-mutation Rate	0.044
dc-inversion Rate	0.1
8 Functions(weight)	add(2), sub(2), mult(2), div(1), pow(1), exp(1), sin(1), cos(1)

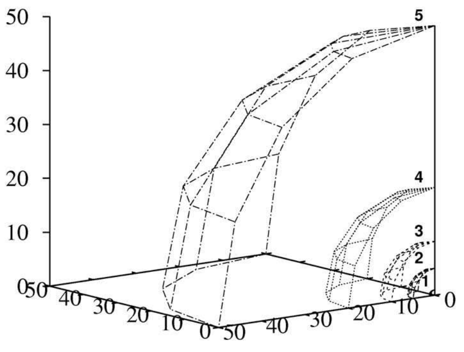


Fig. 1. An example of five theoretical 3-D Pareto fronts which correspond to fronts generated from test problem DTLZ2. Left to right: fronts with radii 50, 20, 10, 5 and 1 respectively.

V. RESULTS

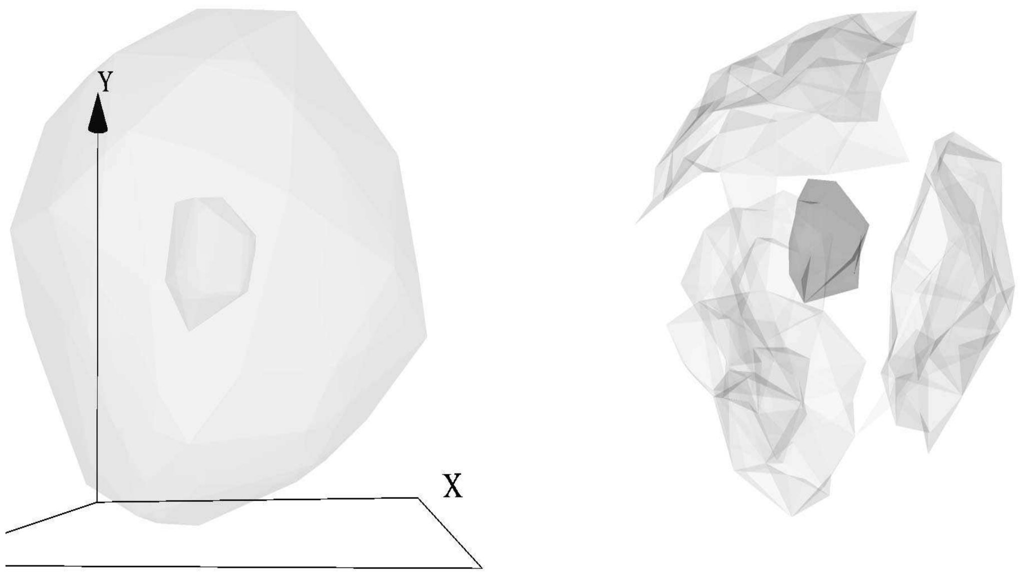
All VR spaces were computed from Eq.1 using Euclidean distance as the dissimilarity measure. Regular sampling along each dimension in the m-space leads to an exponential growth in the number of points; therefore a subset of them were used in the computation. The leader clustering algorithm [36] was used for extracting a kernel set ensuring that no similarity structure was lost beyond a specified threshold. Table II shows the thresholds and the resulting

$$
\begin{array}{ll}
\text{Minimize} & f_1(\mathbf{x}) = (1+g(\mathbf{x}_M))cos(x_1\pi/2)cos(x_2\pi/2)\cdots cos(x_{M-2}\pi/2)cos(x_{M-1}\pi/2) \\
\text{Minimize} & f_2(\mathbf{x}) = (1+g(\mathbf{x}_M))cos(x_1\pi/2)cos(x_2\pi/2)\cdots cos(x_{M-2}\pi/2)sin(x_{M-1}\pi/2) \\
\text{Minimize} & f_3(\mathbf{x}) = (1+g(\mathbf{x}_M))cos(x_1\pi/2)cos(x_2\pi/2)\cdots sin(x_{M-2}\pi/2) \\
\vdots & \vdots \\
\text{Minimize} & f_{M-1}(\mathbf{x}) = (1+g(\mathbf{x}_M))cos(x_1\pi/2)sin(x_2\pi/2) \\
\text{Minimize} & f_{M-1}(\mathbf{x}) = (1+g(\mathbf{x}_M))sin(x_1\pi/2) \\
 & 0 \le x_i \le 1, \text{for } i = 1,\ldots,n \\
 & g(\mathbf{x}_M) = \sum_{x_i \in \mathbf{x}_M} (x_i - 0.5)^2
\end{array}
\tag{3}
$$

(a) Convex hulls corresponding to the sequence of Pareto fronts.

(b) Alpha shapes corresponding to the sequence of Pareto fronts.

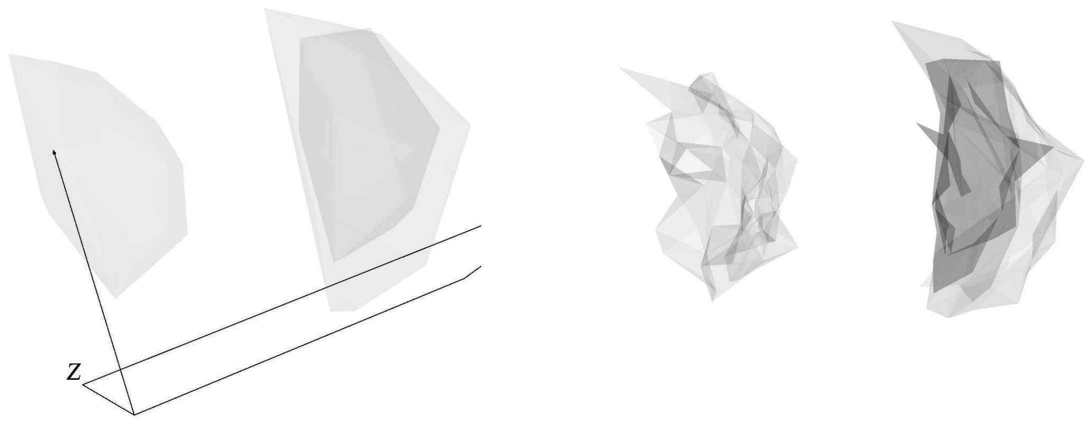
Fig. 2. Comparison of two representations (convex hulls and alpha shapes) after the 4-D test problem DTLZ2 has been mapped down to 3-D (using FRPR). Fronts constructed with radii 50, 20, 10, 5 and 1 are shown.



(a) Convex hulls corresponding to the sequence of Pareto fronts.

(b) Alpha shapes corresponding to the sequence of Pareto fronts.

Fig. 3. Comparison of two representations (convex hulls and alpha shapes) after the 10-D test problem DTLZ2 has been mapped down to 3-D (using FRPR). Fronts constructed with radii 50 and combined radii (20, 10, 5 and 1) are shown.



(a) Convex hulls corresponding to the sequence of Pareto fronts.

(b) Alpha shapes corresponding to the sequence of Pareto fronts.

Fig. 4. 4 Knapsacks Problem with 750 items. The 4-D objective space has been mapped down to 3-D (using FRPR) and shows solution sets for each of the 3 fronts obtained separately from 3 different multi-objective algorithms. Improving from left to right: H: HLGA algorithm, V: VEGA algorithm and N: NSGA algorithm.

number of points for each problem. Gower's coefficient [37] was used as a similarity measure because of its simplicity and insensitivity to scaling differences among the decision variables.

An implicit mapping for Eq.1 was computed using the classical Fletcher-Reeves-Polak-Ribiere (FRPR) conjugate gradient algorithm and the previously described GP-GEP method with experimental settings described in Table I. For the former, the best solution from 100 random initial configurations was obtained.

The mapping errors (all small) are shown in Table III. They indicate that the visible structure in the VR space should be very similar to that of the original space (objective space). With the only exception being that of the DTLZ2 (4-D) problem, where the FRPR algorithm outperformed GP-GEP by a factor of two, both algorithms produced similar results. However, GP-GEP provides explicit mapping equations for φ in Eq.2. The two computed equations (Eq.4 and Eq.5) are for the DTLZ2 (4-D) and Knapsack (4-D) problems respectively, where $f_i(\cdot)$ is the i-th objective function for the related multi-objective problem. Both equations are non-linear, with Eq.5 having higher complexity.

A comparison of two visual representations (convex hulls vs. alpha shapes) for the Pareto fronts of the DTLZ2 (4-D) problem is shown in Fig. 2 with associated alpha values for each problem shown in Table IV. The two representations in Fig. 2 clearly indicate the sizes and relative sequencing of the fronts, with the alpha shape representation being superior when both are compared with DTLZ2 (3-D) in Fig. 1, whose 4-D extension is being analyzed.

For the DTLZ2 (10-D) test problem the fronts cannot be mapped to open surfaces because of severe dimensionality reduction. However, the alpha shape still exhibits a hollow discontinuous outer surface (front radius= 50) wrapping a core composed of all other fronts (merged by the leader algorithm).

For the 4-D Knapsack problem (Fig. 4) the regions of data concentration are clearly indicated by the alpha shape, whereas outliers distort the convex hull.

The spaces obtained for DTLZ2 (4-D) with the FRPR and GP-GEP algorithms are shown in Fig. 5. Not only do they clearly exhibit the expected relationships between the individual Pareto fronts (Fig. 1) but they look very similar, despite GP-GEP having twice as large an error.

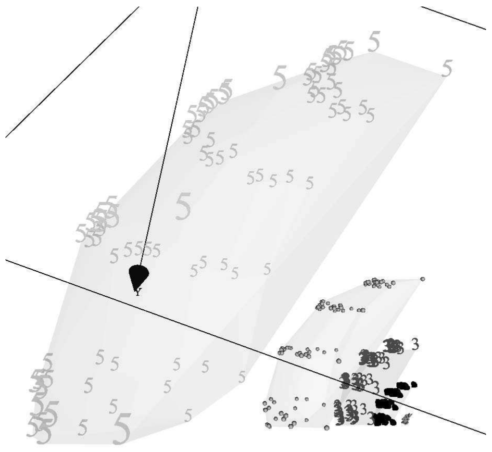
For the 4-D Knapsack problem the VR spaces for both algorithms are again very similar (Fig. 6), as expected because of their comparable error.

$$\begin{aligned}
\varphi_x &= \quad k_{x1} + f_3(\cdot) \\
\varphi_y &= \quad (f_1(\cdot) - ((f_2(\cdot) * f_1(\cdot))/(f_1(\cdot) + k_{y1}))) \\
&+ \quad k_{y2} + (k_{y3} * f_2(\cdot)) \\
\varphi_z &= \quad k_{z1} + f_4(\cdot)
\end{aligned} \tag{4}$$

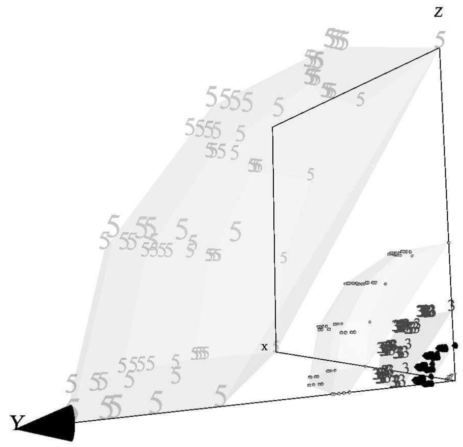
where $k_{x1} = 24.1447678$, $k_{y1} = 8.5520020$, $k_{y2} = 5.5644552$, $k_{y3} = 1.0908070$, and $k_{z1} = 12.7015751$.

$$\begin{aligned}
\varphi_x &= \quad (k_{x1} * f_4(\cdot)) + k_{x2} \\
&+ \quad cos(f_3(\cdot)) + ((k_{x3} + f_2(\cdot))/f_2(\cdot)^{f_3(\cdot)}) \\
\varphi_y &= \quad f_3(\cdot) + cos(cos((f_4(\cdot) * f_2(\cdot))^{sin(f_4(\cdot))})) \\
&+ \quad k_{y1} + cos(f_1(\cdot))^{f_1(\cdot)} + (f_3(\cdot)/k_{y2}) \\
\varphi_z &= \quad k_{z1} + sin((k_{z2} - f_2(\cdot))) + f_2(\cdot) + (f_1(\cdot)/k_{z3}) \\
&+ \quad (k_{z4} - (cos(f_1(\cdot)) + (f_4(\cdot)/f_1(\cdot))))
\end{aligned} \tag{5}$$

where $k_{x1} = -1.099411$,$k_{x2} = 7.800379$, $k_{x3} = 16446.856860$, $k_{y1} = 4.421008$,$k_{y2} = 8.823775$, $k_{z1} = 2.145460$,$k_{z2} = 8.493845$, $k_{z3} = 3.746729$, and $k_{z4} = 2.552130$.

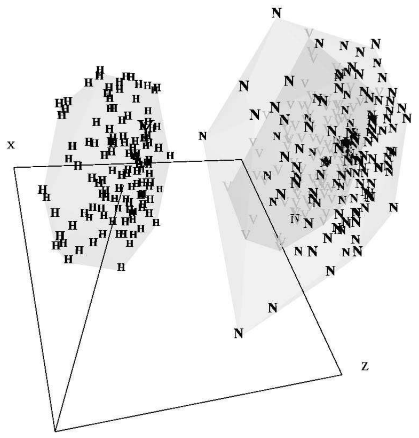


(a) Fletcher-Reeves-Polak-Ribiere classical optimization algorithm

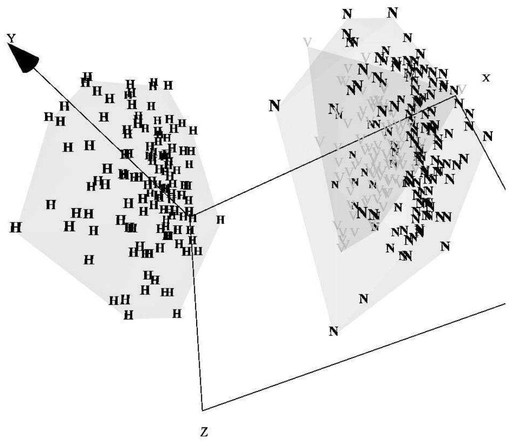


(b) Genetic Programming (Gene Expression Programming) evolutionary algorithm

Fig. 5. Comparison of two algorithms (FRPR and GP-GEP) for computing the mapping from the 4-D version of test problem DTLZ2 down to 3-D. Fronts constructed with radii 50, 20, 10, 5 and 1 respectively.



(a) Fletcher-Reeves-Polak-Ribiere classical optimization algorithm



(b) Genetic Programming (Gene Expression Programming) evolutionary algorithm

Fig. 6. Comparison of two algorithms (FRPR and GP-GEP) capable of computing a mapping for the 4 Knapsacks Problem with 750 items, which is a 4-D objective space, down to 3-D. Fronts obtained from 3 multi-objective algorithms are shown in addition to the respective solution locations. (H: HLGA algorithm, N: NSGA algorithm, and V: VEGA algorithm)

TABLE II
LEADER ALGORITHM RESULTS FOR LARGE PROBLEMS

Dimension	Number of Objects	Similarity Threshold	Number of Leaders
DTLZ2 Front's (10-D)	98415	0.94	309
Knapsack HLGA's Front (4-D)	225	0.945	123
Knapsack NSGA's Front (4-D)	314	0.945	126
Knapsack VEGA's Front (4-D)	196	0.945	96

TABLE III
RESULTS FOR THE COMPUTATION OF THE NEW SPACE USING CLASSICAL OPTIMIZATION AND GENETIC PROGRAMMING

Problem	FRPR	GP-GEP
DTLZ2 Fronts (4-D)	0.003289	0.006533
DTLZ2 Fronts (10-D)	0.056397	0.087126
Knapsack Fronts (4-D)	0.011820	0.017191

VI. CONCLUSIONS

Two multi-objective optimization problems, one theoretical (DTLZ2 4-D and 10-D) and one real (4-D Knapsack problem), have been investigated from the point of view of the construction of visual representations (using Virtual Reality) for high dimensional Pareto fronts or their approximations. Unsupervised similarity preservation mappings (using classical optimization and genetic programming separately) led to good VR spaces (of low error). Therefore, construction of convex hulls and alpha shapes exhibiting compatible properties to that of the theoretical or expected nature of the Pareto fronts was obtained. The use of a combination of uniquely defined representations (convex hulls) with that of

TABLE IV

ALPHA VALUES USED FOR COMPUTING THE SHAPES IN FIGS. 2, 3 AND 4

Problem	Critical α (α_{Cr})	Used
DTLZ2 Radius 1 Front (4-D)	3.214063	α_{Cr}
DTLZ2 Radius 5 Front (4-D)	113.515	α_{Cr}
DTLZ2 Radius 10 Front (4-D)	1862.33	α_{Cr}
DTLZ2 Radius 20 Front (4-D)	10054.7	α_{Cr}
DTLZ2 Radius 50 Front (4-D)	97113.4	75000
Knapsack HLGA's Front (4-D)	$8.42387\ 10^6$	3423870
Knapsack NSGA's Front (4-D)	$3.79343\ 10^7$	20934300
Knapsack VEGA's Front (4-D)	$1.41452\ 10^7$	7145200

subjective representations (alpha shapes) leads to alternative visual constructs for aiding decision makers when considering sets of high dimensional Pareto optimal solutions. The obtained visual representations are robust because spaces of similar spacial structure are expressed despite important deviations in mapping errors. Further, evolutionarily obtained explicit mappings (via genetic programming) provide further insight over implicit mappings because the influence of the different objectives may be studied analytically. These are preliminary results for which further experimentation would be required.

REFERENCES

[1] *ECJ, A Java-based Evolution Computing Research System*. Accessed online: http://www.cs.gmu.edu/~eclab/projects/ecj/. March, 2007.

[2] Candida Ferreira: *Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence*. Springer. 2006.

[3] J.R. Koza: *Hierarchical genetic Algorithms Operating on Populations of Computer Programs*. Proceedings of the 11-th International Joint Conference on Artificial Intelligence. 1989. 1. 768–774.

[4] J.R. Koza: *Genetic Programming: On the Programming of Computers by Means of natural Selection*. MIT Press. 1992.

[5] J.R. Koza: *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press. 1994.

[6] J.R. Koza and F.H. Bennett III and D.Andre and M.A. Keane: *Genetic Programming III: Darwinian Invention and Problem Solving*. Morgan Kaufmann. 1999.

[7] W. Pres and B.P. Flannery and S.A. Teukolsky and W.T Vetterling: *Numeric Recipes in C*. Cambridge University Press. 1992.

[8] J. J. Valdés, R. Orchard and A.J. Barton: *Exploring Medical Data using Visual Spaces with Genetic Programming and Implicit Functional Mappings*. Gecco 2007. Genetic and Evolutionary Computation Conference (accepted). 2007.

[9] J. J. Valdés, A.J. Barton: *Visualizing High Dimensional Objective Spaces for Multi-objective Optimization: A Virtual Reality Approach*. Submitted to CEC 2007. Congress on Evolutionary Computation. 2007.

[10] H. Edelsbrunner and E. P. Mucke, *Three-dimensional alpha shapes*. ACM Trans. Graphics 13. 1994. pp 43–72.

[11] H. Edelsbrunner: *The union of balls and its dual shape*. Discrete Comput. Geom. 13. 1995. pp 415–440

[12] Mark de Berg, M. van Krefeld, M. Overmars, O. Schwarzkopf: Alpha shapes: definition and software. In Proc. Internat. Comput. Geom. Software Workshop. 1995, Minneapolis.

[13] H. S. M. Coxeter: Regular Polytopes. Dover Publications, 1973.

[14] Jean-Daniel Boissonnat , Mariette Yvinec , H. Bronniman: *Algorithmic Geometry: Algorithms and Applications* Cambridge University Press. 2001.

[15] M. de Berg, M. van Krefeld, M. Overmars and O. Schwarzkopf: Computational Geometry: Algorithms and Applications. Springer Verlag 2000.

[16] N. Akkiraju, H. Edelsbrunner, M. Facello, P. Fu, E. P. Mucke, and C. Varela: *Computational Geometry: Algorithms and Applications* Proc. Int. Comput. Geom. Software Workshop. 1995.

[17] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley and Sons, 2001.

[18] E. K. Burke and G. Kendall, *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. 233 Spring Street, New York, NY 10013, USA: Springer Science and Business Media, Inc., 2005, no. 0-387-23460-8.

[19] R. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proc. First International Conference on Genetic Algorithms*, 1985, pp. 93–100.

[20] P. Hajela and C. Lin, "Genetic search strategies in multicriterion optimal design," *Structural Optimization*, vol. 4, pp. 99–107, 1992.

[21] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evol. Comput.*, vol. 2, no. 3, pp. 221–248, 1994.

[22] K. Deb, S. Agarwal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii," in *Proceedings of the Parallel Problem Solving from Nature VI Conference*, Paris, France, 16-20 September 2000, pp. 849–858.

[23] K. Deb, S. Agarwal, and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm: Nsga-ii." in *IEEE Transaction on Evolutionary Computation*, vol. 6 (2), 2002, pp. 181–197.

[24] E. Zitzler and L. Thiele, "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[25] J. J. Valdés, "Virtual reality representation of relational systems and decision rules:," in *Theory and Application of Relational Structures as Knowledge Instruments*, P. Hajek, Ed. Prague: Meeting of the COST Action 274, Nov 2002.

[26] ——, "Virtual reality representation of information systems and decision rules:," in *Lecture Notes in Artificial Intelligence*, ser. LNAI, vol. 2639. Springer-Verlag, 2003, pp. 615–618.

[27] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable Test Problems for Evolutionary Multi-Objective Optimization," Kanpur Genetic Algorithms Laboratory (KanGAL), Department of Mechanical Engineering, Indian Institute of Technology, Kanpur, Tech. Rep. KanGAL Report Number 2001001, 2001.

[28] K. Deb, "Multi-objective genetic algorithms: Problem difficulties and construction of test problems," *Evolutionary Computation*, vol. 7, no. 3, pp. 205–230, 1999.

[29] I. Borg and J. Lingoes, *Multidimensional similarity structure analysis*. Springer-Verlag, 1987.

[30] J. W. Sammon, "A non-linear mapping for data structure analysis," *IEEE Trans. Computers*, vol. C18, pp. 401–408, 1969.

[31] J. J. Valdés and A. J. Barton, "Virtual reality spaces for visual data mining with multiobjective evolutionary optimization: Implicit and explicit function representations mixing unsupervised and supervised properties," in *IEEE Congress of Evolutionary Computation (CEC 2006)*. Vancouver: IEEE, July 16-21 2006, pp. 5592–5598.

[32] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley and Sons, 1990.

[33] S. Khuri and T. Bäck and J. Heitkötter, "The zero/one multiple knapsack problem and genetic algorithms," in *Proc. 1994 ACM Symp. Applied Computing (E. Deaton, D. Oppenheim, J. Urban, and H. Berghel, Eds)*. New York: ACM-Press, 1994, pp. 188–193.

[34] R. Spillman, "Solving large knapsack problems with a genetic algorithm," in *IEEE Int. Conf. Systems, Man and Cybernetics*. Piscataway: IEEE, Oct. 22-25 1995, pp. 632–637.

[35] E. Zitzler and M. Laumanns, "Test problems and test data for multiobjective optimizers," 2006. [Online]. Available: http://www.tik.ee.ethz.ch/~zitzler/testdata.html

[36] J. A. Hartigan, *Clustering Algorithms*. New York: John Wiley & Sons, Inc., 1975.

[37] J. C. Gower, "A general coefficient of similarity and some of its properties," *Biometrics*, vol. 1, no. 27, pp. 857–871, 1973.