Chapter 20
Investigation of Three Genotypes for Mixed Variable Evolutionary Optimization

Rajan Filomeno Coelho, Manyu Xiao, Aurore Guglielmetti, Manuel Herrera and Weihong Zhang

Abstract While the handling of optimization variables directly expressed by numbers (continuous, discrete, or integer) is abundantly investigated in the literature, the use of nominal variables is generally overlooked, despite its practical interest in plenty of scientific and industrial applications. For example, in civil engineering, the designers of a structure made out of beams might have to select the best cross-section shapes among a list of available geometries (square, circular, rectangular, etc.), which can be modeled by nominal data. Therefore, in the context of single- and multi-objective evolutionary optimization for mixed variables, this study investigates three genetic encodings (binary, real, and real-simplex) for the representation of mixed variables involving both continuous and nominal parameters. The comparison of the genotypes combined with the instances of crossover is performed on six analytical benchmark test functions, as well as on the multi-objective design optimization of a six-storey rigid frame, showing that for mixed variables, real (and to a lesser extent: real-simplex) coding provides the best results, especially when combined with a uniform crossover.

R. Filomeno Coelho (✉) · M. Herrera
ULB–BATir Department, Université Libre de Bruxelles, Avenue F.D. Roosevelt, 50 (CP 194/2), 1050 Brussels, Belgium
e-mail: rfilomen@ulb.ac.be

M. Herrera
e-mail: mherrera@ulb.ac.be

M. Xiao
NPU–Department of Applied Mathematics, Northwestern Polytechnical University, Shaanxi 710072, Xi'an, People's Republic of China
e-mail: manyuxiao@nwpu.edu.cn

A. Guglielmetti · W. Zhang
NPU–School of Mechanical Engineering, Northwestern Polytechnical University, Shaanxi 710072, Xi'an, People's Republic of China
e-mail: wuhong_mathnpu@hotmail.fr

W. Zhang
e-mail: zhangwh@nwpu.edu.cn

D. Greiner et al. (eds.), *Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences*, Computational Methods in Applied Sciences 36, DOI 10.1007/978-3-319-11541-2_20

Keywords Mixed variables · Evolutionary algorithms · Categorical variables · Genotype

20.1 Introduction

Real-life engineering applications are often characterized by data of versatile natures. Formally, design variables in parameterization/optimization can be divided into five classes:

- *continuous variables* are defined over an interval $I_c \subseteq \mathbb{R}$ (e.g. length, curvature radius);
- *discrete variables* are continuous variables available only within a finite set $I_d = \{d_1, \ldots, d_n\}$ where all $d_i \in \mathbb{R}$ (e.g. cross-section areas from a catalog of beam profiles). It is interesting to note that gradient-based optimizers can be adapted to discrete variables, as developed for instance by [3] through dual formulations and subgradient-based algorithms;
- *integer variables* are defined over an interval $I_i \subseteq \mathbb{N}$ or $\mathbb{Z}$ (e.g. number of holes in a plate). In engineering problems, they differ from discrete variables by the fact that no intermediate values between two integer variables can be defined (e.g. a plate can contain two or three holes, but not 2.5), which has an impact both on the simulation and optimization sides. *Binary* variables are a particular case of integer variables;
- *ordinal categorical variables*, or simply *ordinal variables*, take their values (called *attributes*) among non-numerical values endowed with a ranking, as in the set {'tiny', 'small', 'medium-sized', 'large', 'huge'};
- *nominal categorical variables*, or simply *nominal variables*, are non-numerical variables characterized by no explicit ordering, as the shape of a beam profile: { O ; □ ; ■ ; I }, the choice of a material: {'steel', 'aluminum', 'titanium'}, etc.

According to the nature of the variables, distinct fields of optimization have been developed; they can be roughly classified as *continuous* versus *combinatorial* optimization. Interestingly, due to their flexibility in data representation, evolutionary optimization algorithms are ideally suited when both types of data are involved. Additionally, since the categorical variables are characterized by non-numerical entries, they represent a challenging issue for the parameterization—and eventually for the optimization—since they require a careful investigation of the conversion procedure onto a chromosome. The optimization algorithms proposed in the literature for mixed variables have been summarized by the authors in [8], confirming that bio-inspired algorithms are an efficient option for single- and multi-objective optimization, as illustrated for example by [12]. However, a systematic examination of data structures for mixed variables is still missing.

Based on all these considerations, the goal of this paper is to investigate three genotypes (binary, real, and real-simplex) in combination with three types of

crossover (line, uniform, and two-site) in order to extract information about the mutual interaction between encoding and genetic operators.

The paper is organized as follows: first, a systematic representation of mixed variables is introduced in Sect. 20.2. Then, the genetic operators used in the evolutionary algorithms are described in Sect. 20.3. Afterwards, six analytical test cases including continuous and nominal variables are thoroughly studied (Sect. 20.4), followed by the multi-objective design optimization of a rigid frame (Sect. 20.5). Finally, the conclusions and future prospects are discussed in Sect. 20.6.

20.2 Data Structures for Mixed Variables

Historically, the initial conversion of design variables to genetic encoding has been done through *binary coding* [4, 9], as illustrated in Fig. 20.1. After conversion, the converted binary strings for all variables can seamlessly be concatenated in a chromosome, regardless of the various natures of the actual variables.

Another popular and straightforward encoding consists in modeling all variables through real values, as shown in Fig. 20.2.

While these operations are straightforward for numerical values (not only continuous, but also discrete and integer), they require an arbitrary mapping for nominal variables, as shown in Table 20.1.

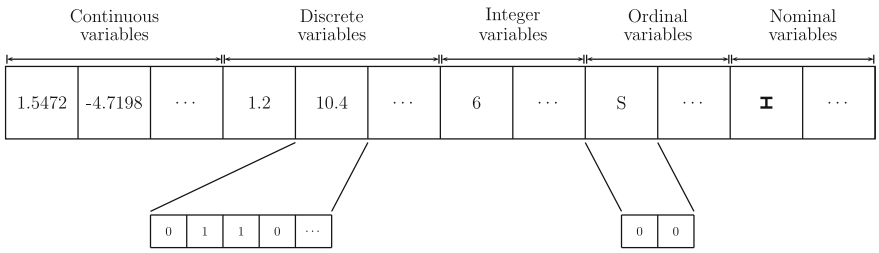

Fig. 20.1 Conversion of a mixed-variable vector to a binary chromosome

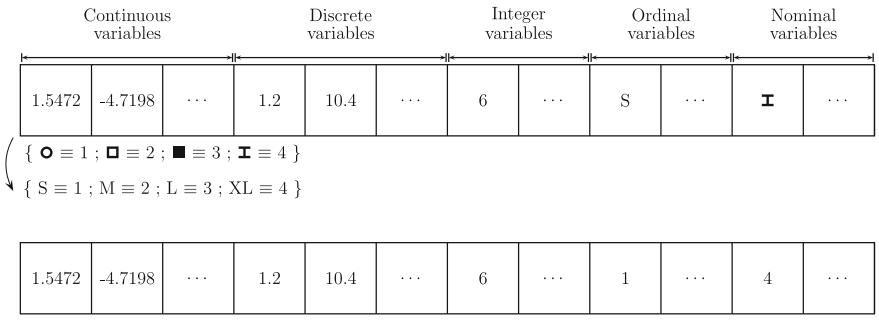

Fig. 20.2 Conversion to a real vector

Table 20.1 Conversion of a nominal variable: two legitimate mappings onto real/binary values

Nominal variable	Mapping 1		Mapping 2	
	Real	Binary	Real	Binary
○	1	(0,0)	2	(0,1)
□	2	(0,1)	4	(1,1)
■	3	(1,0)	1	(0,0)
▲	4	(1,1)	3	(1,0)

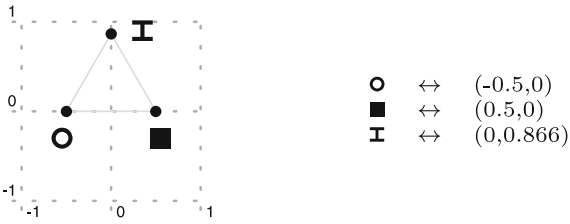

Fig. 20.3 Mapping of a three-attribute nominal variable onto a regular simplex (the Euclidean distance between each pair of attributes in the regular simplex space is always equal to 1)

Indeed, since there is no intrinsic ordering between attributes, both mappings mentioned in Table 20.1 are legitimate, but they might not exhibit the same behavior in the genetic algorithm. Therefore, another conversion is proposed here for nominal values, constraining the attributes of a nominal variable to be equidistant. This constraint can be ensured by assigning to each of the n attributes the coordinates of the vertex of a regular simplex in a $(n-1)$-space, as depicted in Fig. 20.3. Largely used in learning and classification theory [1] with categorical data, this approach was first introduced by the authors for approximation purposes [7].

To summarize, three encodings are analyzed in this paper:

1. *binary*;
2. *real*;
3. *real-simplex*, i.e. conversion to real numbers for continuous, discrete, integer, and ordered categorical variables, and regular simplex mapping for the nominal parameters.

Since the relation between coding and crossover is critical in genetic algorithms, the next section lists the types of crossovers investigated in this study.

20.3 Genetic Operators for Mixed Variables

In all three genotypes proposed here above, the conversion finally leads to a real-valued vector (the *chromosome*), on which three popular types of crossover can be applied [2]:

1. *line crossover* (LCX): for each pair of parents ($p1$, $p2$) selected among the population of the previous generation, the offspring are created as follows:

$$\begin{cases} b_i^{(c1)} = b_i^{(p1)} + \alpha_1 \left(b_i^{(p2)} - b_i^{(p1)} \right) \\ b_i^{(c2)} = b_i^{(p1)} + \alpha_2 \left(b_i^{(p2)} - b_i^{(p1)} \right) \end{cases} \tag{20.1}$$

where b_i are the components of the chromosome (a bit or a real value depending on the coding), $p1$, $p2$ refer to the parents, $c1$, $c2$ to the children, and α_1, α_2 are random numbers uniformly sampled in [0, 1], and constant for the whole chromosome. Geometrically speaking, the children are generated on the line joining the parents (in a space depending on the data structure, viz. the type of coding);

2. *uniform crossover* (UCX): each child c is generated as follows:

$$b_i^{(c)} = \begin{cases} b_i^{(p1)} \text{ if } \alpha < 0.5 \\ b_i^{(p2)} \text{ if } \alpha \geq 0.5 \end{cases} \tag{20.2}$$

where α is sampled independently for each component of the chromosome;

3. *two-site crossover* (TSX): the children are generated by swapping pieces of parental chromosomes between two sites randomly chosen within the chromosome.

After the crossover operation, if values are found that do not correspond to available values (depending on the coding and/or crossover implemented, this problem can happen for discrete, integer, or categorical variables), they are repaired in the chromosome by replacing the wrong value by the closest existing value. The remainder of the paper is devoted to an empirical analysis of the interaction between data structure and crossover on several test cases. Nevertheless, the following conditions can already be devised, and serve eventually as guidelines to analyze the results obtained:

- *Condition of exploration*: the crossover should be able to produce children different from their parents.
- *Condition of invariance*: the children generated from crossover have to be independent from the ranking of the attribute values.

20.4 Analytical Benchmark Test Cases

The first six test functions aim at showing the relation between genotypes and types of crossover for real, nominal, and mixed (real-nominal) variables. Their analytical expression is detailed in Table 20.2. For each test function, three situations are examined: (1) ten real variables, no nominal variables ($nz = 10$, $nc = 0$), (2) ten nominal variables, no real variables ($nz = 0$, $nc = 10$), and (3) five real variables, five nominal variables ($nz = 5$, $nc = 5$).

Table 20.2 Definition of the six analytical benchmarks with mixed variables

Output	
$f_{Ellipsoid,MV}$	$= \sum_{i=1}^{nz} \left(\beta^{\frac{i-1}{nz-1}} z_i\right) + \sum_{i=1}^{nc} \left(\beta^{\frac{i-1}{nc-1}} c_i\right) \qquad (\beta = 5)$
$f_{Ackley,MV}$	$= -20e^{-0.2\sqrt{\frac{1}{nz}\sum_{i=1}^{nz} z_i^2}} - e^{\frac{1}{nz}\sum_{i=1} cos(2\pi z_i)}$ $-20e^{0.2\sqrt{\frac{1}{nc}\sum_{i=1}^{nc} c_i^2}} - e^{\frac{1}{nc}\sum_{i=1} cos(2\pi c_i)} + 20 + e$
$f_{Rastrigin,MV}$	$= 10(nz + nc) + \sum_{i=1}^{nz} \left[z_i^2 - 10cos(2\pi z_i^2)\right]$ $+ \sum_{i=1}^{nc} \left[c_i^2 - 10cos(2\pi c_i^2)\right]$
$f_{Rosenbrock,MV}$	$= \sum_{i=1}^{nz-1} \left[100(z_{i+1} - z_i^2)^2 + (z_i - 1)^2\right]$ $+ \sum_{i=1}^{nc-1} \left[100(c_{i+1} - c_i^2)^2 + (c_i - 1)^2\right]$
$f_{Sphere,MV}$	$= \sum_{i=1}^{nz} z_i^2 + \sum_{i=1}^{nz} c_i^2$
$f_{Griewank,MV}$	$= \frac{1}{4000}\sum_{i=1}^{nz} z_i^2 - \prod_{i=1}^{nz} cos\left(\frac{z_i}{\sqrt{i}}\right) + \frac{1}{4000}\sum_{i=1}^{nc} c_i^2 - \prod_{i=1}^{nc} cos\left(\frac{c_i}{\sqrt{i}}\right)$
Input	
Cont. vars.	$z_i = 10^{-3} x_i^{cont}$, $x_i^{cont} \in [-300, 700]$ for $i = 1, \ldots, nz$
Categ. vars.	$c_i \in o_{permut}^{(i)}([0, 1, \ldots, 10])$ for $i = 1, \ldots, nc$

Table 20.3 Numerical errors obtained for the six analytical benchmark test functions for 10 *real* variables (the error values, averaged over 10 independent runs, are normalized between 0 and 1)

Coding	Binary			Real			Real-simplex		
Crossover	LCX	UNX	TSX	LCX	UNX	TSX	LCX	UNX	TSX
$TC1$	1	5.45e-01	4.58e-01	0	3.39e-02	6.13e-02	5.17e-02	1.81e-01	2.36e-01
$TC2$	1	3.85e-01	4.87e-01	0	2.97e-02	1.85e-02	1.17e-01	1.79e-01	1.48e-01
$TC3$	1	3.34e-01	3.71e-01	0	5.77e-02	2.42e-02	6.66e-02	1.21e-01	1.37e-01
$TC4$	1	2.20e-01	2.43e-01	2.32e-03	9.24e-03	0	4.22e-02	6.35e-02	6.58e-02
$TC5$	4.59e-01	1	9.24e-01	0	3.43e-02	2.28e-02	4.87e-02	6.39e-02	7.62e-02
$TC6$	1	3.94e-01	4.06e-01	0	3.76e-03	1.13e-03	1.40e-01	1.99e-01	1.05e-01

Since the goal is to compare the interactions between the type of crossover and the data structure (coding), the probability of mutation is set to 0 (no mutation). Moreover, a tournament selection is used; the population size is set to 100, and the number of generations to 50. 10 independent runs are performed for each configuration.

The results are collected in Tables 20.3, 20.4 and 20.5, showing the normalized error between the worst and best solutions (averaged over the ten independent runs). In other words, on each line, 0 corresponds to the best solution found (in average), while 1 refers to the worst.

Table 20.4 Numerical errors obtained for the six analytical benchmark test functions for 10 *nominal* variables (the error values, averaged over 10 independent runs, are normalized between 0 and 1)

Coding	Binary			Real			Real-simplex		
Crossover	LCX	UNX	TSX	LCX	UNX	TSX	LCX	UNX	TSX
$TC1$	1	8.43e-01	8.03e-01	6.86e-02	0	2.47e-02	1.51e-01	4.17e-02	9.79e-02
$TC2$	1	9.79e-01	7.98e-01	9.16e-02	2.95e-03	0	1.23e-01	4.28e-02	6.20e-02
$TC3$	1	9.38e-01	9.02e-01	6.78e-02	0	1.37e-02	5.67e-02	3.87e-02	9.19e-02
$TC4$	9.61e-01	9.35e-01	1	1.36e-01	0	6.78e-02	1.45e-01	1.11e-01	1.19e-01
$TC5$	9.08e-01	1	8.55e-01	1.74e-01	0	5.60e-02	1.66e-01	1.29e-01	1.39e-01
$TC6$	8.85e-01	8.22e-01	1	3.74e-02	0	9.77e-02	3.17e-01	2.73e-01	3.21e-01

Table 20.5 Numerical errors obtained for the six analytical benchmark test functions for 10 *mixed* (viz. 5 real and 5 nominal) variables (the error values, averaged over 10 independent runs, are normalized between 0 and 1)

Coding	Binary			Real			Real-simplex		
Crossover	LCX	UNX	TSX	LCX	UNX	TSX	LCX	UNX	TSX
$TC1$	1	6.05e-01	6.51e-01	7.92e-02	2.90e-03	0	2.46e-01	1.19e-01	1.08e-01
$TC2$	1	7.40e-01	6.80e-01	1.29e-01	0	1.48e-02	1.26e-01	1.16e-01	7.87e-02
$TC3$	1	3.74e-01	4.44e-01	1.81e-02	1.85e-02	0	1.63e-01	3.14e-02	3.68e-02
$TC4$	1	6.44e-01	4.77e-01	9.18e-02	0	2.77e-02	1.44e-01	7.74e-02	1.54e-01
$TC5$	6.25e-01	7.53e-01	1	0	3.39e-02	7.73e-02	4.90e-02	1.86e-01	9.40e-01
$TC6$	1	7.00e-01	8.78e-01	1.57e-03	0	7.27e-02	2.53e-01	1.95e-01	9.43e-02

In the six analytical test functions, it appears that the conversion to real numbers is the most effective for the cases with either continuous or nominal variables only. As intuition would have suggested, the line crossover allows for a better exploration of the design space for continuous variables, while the uniform crossover is more effective with nominal variables, where the swapping of variable values is performed independently and randomly for each variable. The binary coding with line crossover provides the worst results, which could be expected since the linear combination of parental chromosomes is not a meaningful operation for values converted into binary digits. Finally, the real-simplex coding generally provides reasonably good results in comparison with the real coding.

20.5 Application: Structural Design of a Rigid Frame

To analyze the efficiency of the proposed algorithmic instances on a structural design example, a 3D rigid frame is investigated [11]. The quantities of interest are the mass and the compliance, the latter being post-processed from a finite element linear

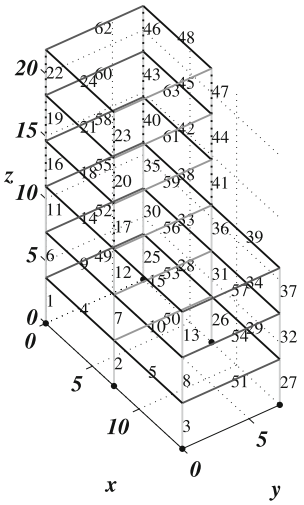

Fig. 20.4 Six-storey rigid frame (with numbering of the beam elements): the five groups of cross-sections are displayed with various shades of *gray* color. Each node connecting two (or more) elements is a rigid connection. The multi-objective optimization consists in finding the best compromise (Pareto) designs with respect to two conflicting objectives, namely the mass and the compliance for the whole structure

analysis with beam elements [6]. The loads are derived from Eurocode 3 [10], and consist in:

- the dead load of the beams and columns;
- the gravity load on the floors (19.16 kPa);
- the lateral load due to the wind (110 kN).

The beams or columns are classified in five groups of common cross-sections (as depicted in Fig. 20.4):

- group 1: {4, 5, 9, 10, 14, 15, 18, 21, 24, 28, 29, 33, 34, 38, 39, 42, 45, 48};
- group 2: {49, 51, 52, 54, 55, 57, 58, 60, 62};
- group 3: {50, 53, 56, 59, 61, 63};
- group 4: {1, 2, 3, 6, 7, 8, 11, 12, 13, 25, 26, 27, 30, 31, 32, 35, 36, 37};
- group 5: {16, 17, 19, 20, 22, 23, 40, 41, 43, 44, 46, 47}.

Ten design variables are necessary to parameterize a given structure:

- for each of the five groups of profiles, a categorical variable c defines the cross-section geometry among seven attributes: { I ; □ ; ○ ; ▯ ; ■ ; ● ; ▮ };
- for all groups of profile, one continuous bounded variable defines the maximum length l of the cross-section (either height or diameter) with $0.09\,\text{m} \leq l \leq 0.11\,\text{m}$, and with a fixed thickness (when applicable) set to 0.0025 m. For the rectangular cross-section, the width is defined as half of the height; for the I—section, the width is equal to the height.

The geometry of the cross-section is typically a nominal variable, since no ordering of the available cross-section types can be made a priori. The choice of the cross-section has a direct impact on the calculation of the quantities (area, moments of inertia) necessary to get the normal efforts, shear forces, and bending moments.

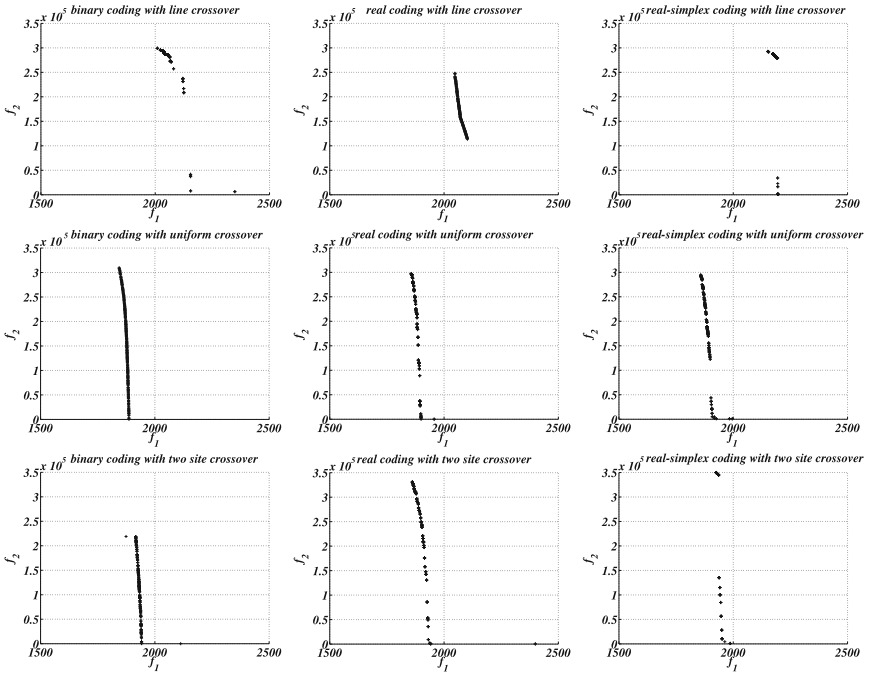

Fig. 20.5 Rigid frame: Pareto fronts obtained for the nine combinations of coding and crossover types

The multi-objective implementation of the genetic algorithm is based on the second version of the *Nondominated Sorting Genetic Algorithm*, or NSGA-II [5]. The corresponding Pareto fronts obtained for the nine possible combinations of coding and type of crossover are depicted in Fig. 20.5.

In this problem, the uniform and two-site crossovers clearly outperfom the line crossover to obtain dense and nondominated Pareto fronts. In particular, the uniform crossover provides comparable results for all codings.

In terms of designs obtained, a close examination of the Pareto set reveals that the optimal cross-sections are mostly circular (O) or I-shaped (I), and square (□) to a lesser extent. The tubular shapes constitute the best compromise between lightness and stiffness.

20.6 Conclusions and Future Prospects

In this paper, three genotypes, along with three types of crossover, are analyzed in order to extract their properties in the modeling of mixed (continuous + nominal) variables. For the analytical test functions, the real coding furnishes the best results in almost all configurations (real, nominal, or mixed variables), while the

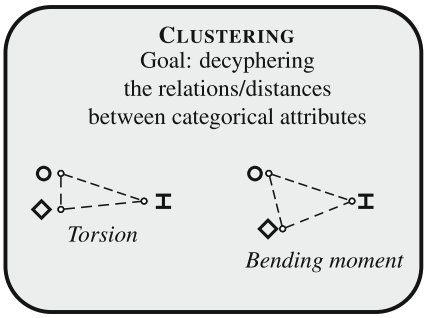

Fig. 20.6 Clustering procedure: considering for example the static structural analysis of a beam with respect to the shape of its cross-section (*square*, *circular*, or *I-shaped*), a priori calculations can build clusters of attributes depending on their impact on output quantities of interest (here: bending moment and torsion effect)

real-simplex performs fairly, and the binary coding poorly. The type of crossover also has a significant influence on the results, and this aspect is mostly visible in the multi-objective design optimization of a rigid frame, where the uniform crossover consisting in randomly swapping genes in the parental chromosome is the most effective one to find dense and widely distributed Pareto fronts.

Future studies are guided by the need for better accounting for the relations between attributes of nominal variables. In the simplex mapping representation of nominal variables, the hypothesis that all attributes are equidistant is acceptable when no information is known a priori about their correlations with the output quantities of interest (objectives and constraints). However, when knowledge about the physics is available, a non-regular simplex might be preferable, with distinct pairwise distances between attributes. For instance, the square and circular cross-section shapes of beam profiles might exhibit a closer behavior than the I-shaped profiles, which might be taken into account directly in the coding. This crucial step (under investigation) requires a *clustering phase* to identify these pairwise relationships to be performed before the optimization process (see Fig. 20.6), by using physical insight from the input-output model.

Acknowledgments This work has been supported by Innoviris (Brussels-Capital Region, Belgium) through a BB2B project entitled "Multicriteria optimization with uncertainty quantification applied to the building industry". The authors also acknowledge support by the Basic Project Foundation of Northwestern Polytechnical University (JC20120241), and by the National Natural Science Fundation of China (Grants No.11302173 and No.51275424).

References

1. Agresti A (1996) An introduction to categorical data analysis. Wiley, New York
2. Bäck T, Fogel DB, Michalewicz Z (1997) Handbook of evolutionary computation. Oxford University Press, New York

3. Beckers M (2000) Dual methods for discrete structural optimization problems. Int J Numer Meth Eng 48:1761–1784
4. De Jong KA (ed) (2006) Evolutionary computation: a unified approach. MIT Press, Massachusetts
5. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197
6. Ferreira AJM (2009) MATLAB codes for finite element analysis: Solids and structures., solid mechanics and its applications Springer Science+Business Media B.V., The Netherlands
7. Filomeno Coelho R (2012) Extending moving least squares to mixed variables for metamodel-assisted optimization. In: 6th European congress on computational methods in applied sciences and engineering (ECCOMAS 2012), Vienna, Austria, 10–14 September
8. Filomeno Coelho R (2013) Metamodels for mixed variables based on moving least squares-application to the structural analysis of a rigid frame. Optim Eng
9. Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley Longman, New York
10. Papadrakakis M, Lagaros ND, Plevris V (2005) Design optimization of steel structures considering uncertainties. Eng Struct 27:1408–1418
11. Richard Liew JY, Chen H, Shanmugam NE, Chen WF (2000) Improved nonlinear plastic hinge analysis of space frame structures. Eng Struct 22(10):1324–1338
12. Tang X, Bassir DH, Zhang W (2011) Shape, sizing optimization and material selection based on mixed variables and genetic algorithm. Optim Eng 12:111–128