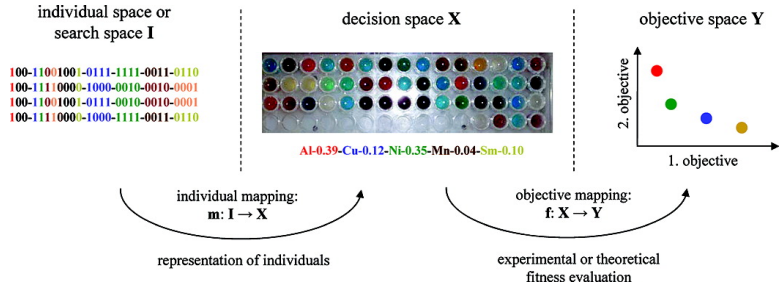
JOURNAL OF combinatorial CHEMISTRY

Article

On the Suitability of Different Representations of Solid Catalysts for Combinatorial Library Design by Genetic Algorithms

Oliver C. Gobin, and Ferdi Schüth

J. Comb. Chem., **2008**, 10 (6), 835-846• DOI: 10.1021/cc800046u • Publication Date (Web): 12 August 2008



More About This Article

Additional resources and features associated with this article are available within the HTML version:

- Supporting Information
- Access to high resolution figures
- Links to articles and content related to this article
- Copyright permission to reproduce figures and/or text from this article

View the Full Text HTML

On the Suitability of Different Representations of Solid Catalysts for Combinatorial Library Design by Genetic Algorithms

Oliver C. Gobin* and Ferdi Schüth

Max-Planck-Institut für Kohlenforschung, Kaiser-Wilhelm-Platz 1, Mülheim an der Ruhr, Germany

Received March 17, 2008

Genetic algorithms are widely used to solve and optimize combinatorial problems and are more often applied for library design in combinatorial chemistry. Because of their flexibility, however, their implementation can be challenging. In this study, the influence of the representation of solid catalysts on the performance of genetic algorithms was systematically investigated on the basis of a new, constrained, multiobjective, combinatorial test problem with properties common to problems in combinatorial materials science. Constraints were satisfied by penalty functions, repair algorithms, or special representations. The tests were performed using three state-of-the-art evolutionary multiobjective algorithms by performing 100 optimization runs for each algorithm and test case. Experimental data obtained during the optimization of a noble metal-free solid catalyst system active in the selective catalytic reduction of nitric oxide with propene was used to build up a predictive model to validate the results of the theoretical test problem. A significant influence of the representation on the optimization performance was observed. Binary encodings were found to be the preferred encoding in most of the cases, and depending on the experimental test unit, repair algorithms or penalty functions performed best.

1. Introduction

Genetic algorithms or similar evolutionary methods are increasingly becoming accepted problem solving tools in applied science.[1] Since 1990, the number of publications significantly increased, and the gap between computer and applied science diminishes more and more. In chemistry, the tutorials by Lucasius and Kateman[2,3] and Hibbert[4] had a significant impact in boosting applications, and they provide a good starting point to understand how these algorithms work. High-throughput experimentation techniques for the synthesis and screening of new compounds, such as drugs,[5] molecular catalysts,[6] or solid catalytic materials[7−11] were developed in the 1990s and made the application of genetic algorithms possible and desirable.[12,13] In the field of heterogeneous catalysis, Wolf et al.[14] were the first to use a genetic algorithm for the generation and optimization of combinatorial libraries of solid catalysts. Since then, substantial efforts were invested in improvement of the performance of genetic algorithms in this domain, as demonstrated, for instance, in studies on the influence of the genetic algorithm parameters, such as elitism, variation, or selection operators,[15−17] population size,[17,18] or on advanced algorithms that incorporate neural networks as search heuristic during the evolution.[19−21] However, up to now, no systematic study on the influence of the representation of solid catalysts on the performance was carried out. The representation of catalysts, that is, the mapping between the real catalyst formulation and the codified representation, directly influences the search itself, as the genetic algorithm only operates on the codified individuals. Figure 1 illustrates the different spaces and their relationship. Two mappings are required to obtain the fitness of each individual, and each mapping involves a change in the dimensionality and in the shape of the corresponding space. The mapping between the decision space **X**, that is, the real catalysts, and the objective space **Y** is problem dependent and can not directly be modified. In contrast, the mapping between the individual space **I** and the decision space is affected by the chosen representation. Therefore, a possible way to improve a genetic algorithm is to understand how the representation influences the search to design the best possible representation for the corresponding problem.

In this work, several encodings in combination with adequate variation operators and constraint handling techniques were systematically tested on the basis of a new multiobjective theoretical combinatorial test problem and validated by using a model of an experimental test problem, the so-called "deNOx problem".[22] We chose a multiobjective approach to test the ability not only to find the best catalyst formulation but also to find multiple best catalysts with respect to several goals. Such an approach does not only give information on the convergence of the algorithm toward the best set of catalysts but also intrinsically provides information on the ability of the representation to keep a diverse set of best catalysts and therefore to find various catalysts. In the study of this problem, several questions are encountered, including: Are binary or real valued representations more adequate for encoding of the catalyst composition? In the case of a binary vector representation, are compact encodings with fewer bits always better than encodings with more bits? Does a reduction of the decision space by chemical knowledge improve the performance?

* To whom correspondence should be addressed. E-mail: og@ogobin.org.

10.1021/cc800046u CCC: $40.75
Published on Web 08/12/2008

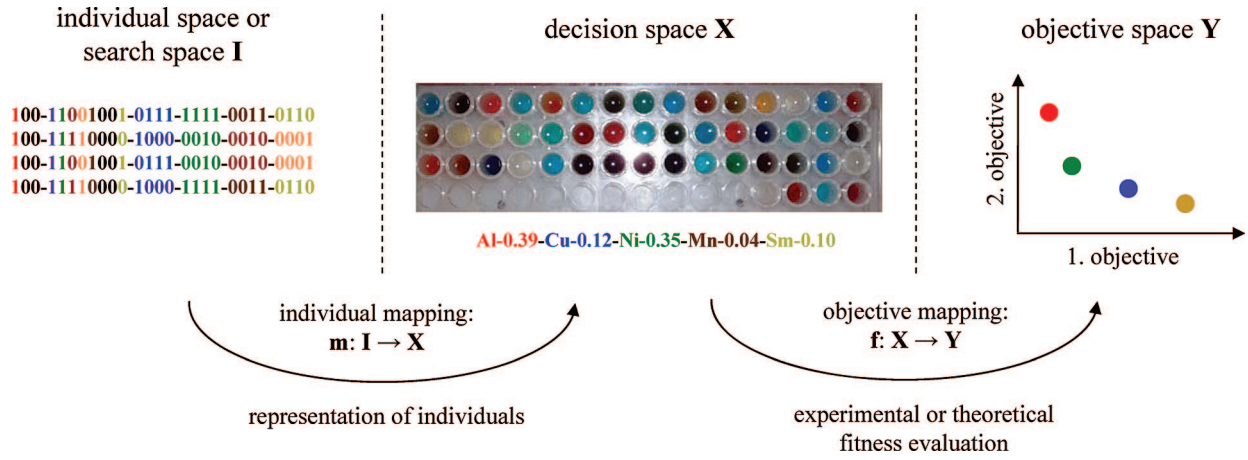


Figure 1. Schematic representation of the individual, decision, and objective space and their relationship (adapted from Zitzler[42]).

What is the best way to handle constraints in the case of an experimental optimization problem? Is it necessary to encode the existence of an element in addition to its chemical composition? We will try to address these and related questions in the following treatment of the problem.

The paper is structured in the following way: First, a very brief introduction to genetic algorithms and multiobjective optimization is given. The optimization framework, that is, the software for automation and performance assessment of the optimization process, is described and afterward a detailed specification of the test functions and of the test cases is given. In the last part, we discuss the results and on their basis give some general guidelines on the most suitable representation of solid catalysts.

2. Methods

2.1. Genetic Algorithms for Multiobjective Optimization. Genetic algorithms are stochastic global search methods based on evolutionary principles. They include heuristic strategies for searching for new and improved solutions in an intelligent way. Interactions among design variables or components are intrinsically considered. Several potential solutions, a population, are evolved in parallel, and the solutions undergo recombination, mutation, and selection steps during each iteration. One iteration loop is called a generation. After a certain number of generations, the algorithm converges, and ideally, it finds the globally optimal solution.

Especially real world problems often involve the simultaneous optimization of several, and often competing, objectives. In the case of only one objective the solution is clearly defined, and only one possible solution exists. In contrast, in the case of multiobjective optimization the situation is completely different because two solutions that optimize the various objectives in different ways and are therefore both "optimal" may be different from each other. Usually a set of optimal tradeoff surfaces composed of the optimal solutions with respect to all objectives, the so-called Pareto-optimal front, is obtained. It also includes the single objective optima. Pareto-optimality generally uses the concept of domination to decide which solutions are better than others with respect to all objectives. More precisely, a solution dominates another solution if it is not worse for all objectives and better for at least one objective. The so-called nondominated solutions, the solutions that are not dominated by any other solution, form the Pareto-optimal set.

A multiobjective optimization problem can be defined as finding a vector of decision variables, $\mathbf{x} = (x_1, x_1,..., x_m) \in \mathbf{X}$, in the feasible region of the decision space $\mathbf{X}$ that optimizes a vector function $\mathbf{f}:\mathbf{X} \rightarrow \mathbf{Y}$ by assigning the quality of a specific solution $\mathbf{x}$ to a vector of objective variables $\mathbf{y} = (y_1, y_1,..., y_n) \in \mathbf{Y}$ in the multidimensional objective space $\mathbf{Y}$.

2.2. Optimization Framework. The optimization framework was built on the basis of the platform and programming language independent interface for search algorithms (PISA).[23] The selector modules, the nondominated sorting genetic algorithm (NSGA-II),[24] the strength Pareto evolutionary algorithm (SPEA2),[25] and the indicator based evolutionary algorithm (IBEA)[26] as implemented in PISA were used as multiobjective optimization algorithms. NSGA-II and SPEA2 are so-called Pareto-based algorithms, which operate toward two goals: to minimize the distance toward the Pareto-optimal set and to maximize the diversity within the Pareto-optimal set to identify the complete set. This is achieved in both algorithms by using a combination of specific techniques, in particular Pareto-based ranking of the individuals, and refinement by additional density information in the objective space. These optimization goals are generally denoted as preference information. This preference information, however, is implemented in a different way depending on the specific algorithm. In contrast IBEA uses a novel strategy and is able to adapt to arbitrary preference information by using quality indicators, which incorporate the preference information. In subsection 2.5, two quality indicators will be described more in detail. Elitism is implemented in all three algorithms using an additional population, the so-called "external" or "archive" population. NSGA-II assesses the fitness of an individual based on the number of other individuals that dominate it, and diversity is preserved by a crowing distance algorithm. SPEA2 uses a similar fitness assessment based on the number of individuals that are dominated by or equal to a certain individual divided by the total population size plus one. Mating is performed using the individuals of both archive and regular populations. The diversity along the Pareto-optimal front is

preserved through a density estimation technique that uses a *k*-nearest neighbor clustering algorithm Selection of individuals for mating was performed by binary tournament selection. As already mentioned above, IBEA uses quality indicators that incorporate both convergence and diversity information, and thus no additional density estimation technique is needed. An ϵ-indicator was used for fitness assignment, and the IBEA parameters κ and ρ were set to 0.05 and 1.1, respectively. For detailed descriptions of the optimization algorithms, we refer to the original publications.

The variator module, which incorporates the problem specific knowledge, was implemented in Matlab as described in Gobin et al.[22,27] The population size was set to 24 for all test cases to have comparable results with our previous study. The archive population of the selector module was set to the same size. In the case of a binary vector codification, a bit-flip mutation operator (M) and a one-point recombination operator (1X) were used. Real valued elements were mutated by polynomial mutation (PM)[28] with a distribution index of $\eta = 20$ and recombined using the symmetric simulated binary crossover operator (SBX)[29] with a distribution index of $\eta = 15$. For mutation, a probability of the inverse of the vector length was used, and for recombination, a probability of 1 was used. Automation and performance assessment[30] of the optimization process was achieved by using the Monitor module.[31] Random numbers for the variator module were obtained by Random.org,[32] which offers true random numbers generated from atmospheric noise. To obtain statistically relevant data, each optimization run was repeated 100 times with different random seeds and initial populations.

The parameters of the Variator module were held constant, as a variation of these parameters did not lead to new insights with respect to the problem of representation. Studies on the influence of these parameters and on the use of dynamic probabilities to control the diversity in the population were recently performed for single-objective problems.[15,16,33] The situation of parameter settings is, however, different in the case of multiobjective optimization. It is mandatory for multiobjective optimizers to keep the diversity high because only a high diversity in the population leads to the discovery of the complete Pareto-optimal front. Therefore special operators or algorithms are usually implemented in addition to the crossover and mutation operators. We have chosen reasonable parameter settings and operator choices that are well-known and commonly used for multiobjective problems. The binary one-point crossover operator and the binary mutation operator are the classic way to implement these two variation operators. In the case of real valued representations, many implementations exist. The SBX operator and the polynomial mutation operator are well-established, are known to work together, and were successfully used in many optimization problems with similar parameter settings.[28]

2.3. Theoretical Test Problem. To simulate properties common to solid catalysts, a new combinatorial problem was designed. It is based on the definition of multiobjective test problems by Deb et al.,[34] which are easy to construct, scalable to any number of decision variables or objectives, and the shape and location of the true Pareto-optimal front is exactly known. Two modified versions of the continuous test problems DTLZ-1 and DTLZ-2 were used to design the new combinatorial problem. In the case of two objective functions, f_1 and f_2, and n decision variables x_i, the modified DTLZ-1 problem was defined as

$$\begin{aligned} \text{minimize} \quad & f_1^{\text{DTLZ1}}(\mathbf{x}) = x_{\text{M}}(1 + g(\mathbf{x}))x_1 \\ \text{minimize} \quad & f_2^{\text{DTLZ1}}(\mathbf{x}) = x_{\text{M}}(1 + g(\mathbf{x}))(1 - x_1) \\ \text{subject to} \quad & 0 \le x_i \le 1 \text{ for } i = 1, \ldots, n \end{aligned} \tag{1}$$

where

$$g(\mathbf{x}) = \sum_{i=1}^{n} (x_i - x_{\text{s}})^2 \tag{2}$$

x_{M} and x_{S} are adjustable parameters of the problem. The Pareto-optimal solution in the decision space corresponds to $\mathbf{x} = (x_{\text{S}},\ldots, x_{\text{S}})^{\text{T}}$. x_{M} defines the location of the Pareto-optimal set in the objective space. The modified DTLZ-2 problem was defined in a similar way

$$\begin{aligned} \text{minimize} \quad & f_1^{\text{DTLZ2}}(\mathbf{x}) = x_{\text{M}}(1 + g(\mathbf{x}))\cos(x_1\pi/2) \\ \text{minimize} \quad & f_2^{\text{DTLZ2}}(\mathbf{x}) = x_{\text{M}}(1 + g(\mathbf{x}))\sin(x_1\pi/2) \\ \text{subject to} \quad & 0 \le x_i \le 1, \text{for } i = 1, \ldots, n \end{aligned} \tag{3}$$

where g(**x**) is the function given in eq 2.

In the objective space, the objective functions lie on a linear hyper-plane of order 2 in the case of the DTLZ-1 problem, $(f_1 + f_2) = x_{\text{M}}$, or on a spherical hyper-surface of order 2 in the case of the DTLZ-2 problem, $(f_1 + f_2)^2 = x_{\text{M}}$. By adjustment of the x_{M} values it is possible to combine the linear and the spherical Pareto-optimal front to create a new problem, where the optimal solution is a combination of multiple problems.

In combinatorial chemistry often one goal is the search for new and improved catalysts formulations. Typically this problem consists of two parts, which are not necessarily pursued separately: first, the combinatorial problem to find the best combination of elements in a catalyst and second, the continuous part, where for a certain combination the best composition has to be found. In the case of the theoretical test problem, the combinatorial part of the problem was implemented as different combinations of the continuous test problems (DTLZ-1, DTLZ-2), of the absolute location of the global Pareto-optimal front x_{M}, and of the x_{S} values corresponding to the concentrations of the elements in the catalyst. The x_{M} value represents the optimal activity of an element. A small x_{M} value for a certain element would describe its single activity without other elements. Interactions were simulated by adding a small value Δx_{M} to the x_{M} value to influence the activities. The influence of support or promoter elements can also be simulated in this way. Assuming a linear relationship, the resulting location of the global Pareto-optimal front x_{M}^* of a combination of n elements was calculated using the following equation

$$x_{\text{M}}^* = \frac{1}{n}\sum^{n} x_{\text{M}} + \sum \Delta x_{\text{M}} \tag{4}$$

x_{M} can be chosen arbitrary in the range from 0 to 1. Δx_{M} can be positive or negative and should be small in comparison to x_{M}, which is to be minimized just like the fitness

Table 1. Parameters of the Theoretical Test Problem

type	element	x_M	Δx_M	problem
main component	Cu	0.50	−0.10	DTLZ-1
main component	Ni	0.60	0.10	DTLZ-1
main component	Co	0.60	−0.05	DTLZ-2
main component	Fe	0.50	0.10	DTLZ-1
main component	Mn	0.55	0.10	DTLZ-1
main component	La	0.60	−0.05	DTLZ-2
main component	Ce	0.70	0.00	DTLZ-2
main component	Sm	0.65	−0.05	DTLZ-2
promoter	K		−0.025	
promoter	Sr		−0.025	
with support	Al		0.00	
without support	Al		0.10	

functions. The values chosen for the theoretical problem in this work are given in table 1. For the resulting combination of elements, x_M^* was calculated according to eq 4 using the values given in Table 1. DTLZ-1 or DTLZ-2 was chosen depending on the incidences each problem occurred in the resulting combination. If the incidences of DTLZ-1 and DTLZ-2 were equal, DTLZ-1 was chosen, otherwise the problem with the highest number of incidences was chosen. To make the problem more realistic, the optimal solutions x_S^* of the continuous part of the problem in the decision space can be defined as functions of x_M^*. x_S^* was set equal to x_M^*. The global Pareto-optimal front was a combination of a Pareto-optimal front from the DTLZ-1 and DTLZ-2 problem. More precisely, from the values in Table 1, the best combinations were Al−Cu−Co−La−Sm−K−Sr with $x_{M1}^* = 0.2875$ (DTLZ-2, with support) and Al−Cu−K−Sr (DTLZ-1, with support) with $x_{M2}^* = 0.35$. Because of the different shapes of the Pareto fronts, a linear DTLZ-1 Pareto front with a higher x_M can still be better than a DTLZ-2 Pareto front with a lower x_M. In Figure 2, the Pareto-optimal fronts of DTLZ-1 and DTLZ-2 with $x_M = 0.5$ and of the theoretical test problem with x_M(DTLZ-1) = 0.35 and x_M(DTLZ-2) = 0.2875 are shown. The combined DTLZ-1 and DTLZ-2 front clearly show that this approach to construct new combinatorial problems is possible.

Expressed in more generic terms the test problem implements the following properties: the combination of a combinatorial part composed of discrete and finite objects, that is, the element combinations of a catalyst, and of a continuous part, that is, the elemental composition. The continuous part includes the following features: the shape of the problem

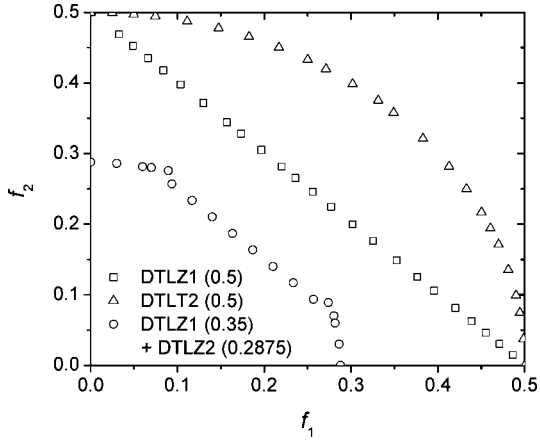


Figure 2. Pareto-optimal fronts of the multiobjective DTLZ-1 and DTLZ-2 problems and of a combination of DTLZ-1 and DTLZ-2 front. The numbers in brackets correspond to the x_M value.

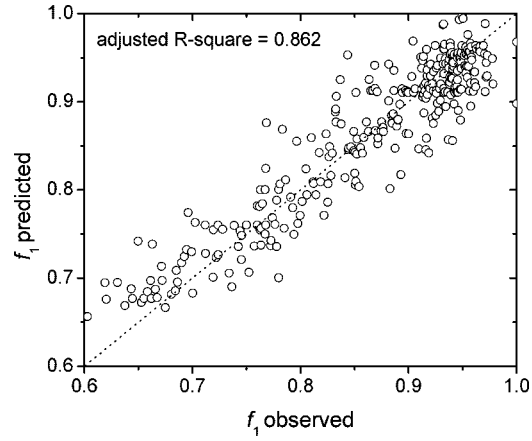


Figure 3. Predicted values of the regression model for the objective function f_1 plotted as a function of the observed values.

is linear or spherical and higher-order shapes can be easily implemented by changing the problem definitions in eqs 1−3. In this work the use of linear and spherical continuous shapes was found to yield in a complexity comparable with the one from the deNOx problem,[22] and therefore, no higher-order or periodic functions were used. Examples of more complex functions can be found in the report of Deb et al.[34] The dimensionality of the continuous part, which represents the number of element compositions n, can be set equal to the number of elements in a catalyst. In the case of the combinatorial part, the most important features are: each combination defines exactly one continuous local Pareto-optimal front of linear or spherical shape. The global Pareto-optimal front is a combination of several element combinations, and the relationship between the elements is linear as defined by eq 4. Here again higher-order interactions are possible. The resulting test problem therefore consists of many locally optimal solutions and one global Pareto-optimal front of varying shape composed of several element combinations, and is thus generating a problem with properties similar to combinatorial library design.

2.4. Experimental Test Problem. We were also interested in studying the influence of different encoding strategies for a real problem. Thus, experimental data obtained in our previous work[22] was used to build a predictive model for validation of the results of the theoretical test problem on an experimental response surface. The modeling was done using the *R* programming language and crosschecked with Statistica. Several generalized regression models were tested and multivariate adaptive regression splines (MARS)[35] were found to be simple with similar performance compared to artificial neural networks. Each objective function was treated separately. The best results were obtained considering an order of element interactions of 2. Two hundred basis functions were first tested and afterward reduced by pruning to the most significant basis functions. In Figure 3, the predicted values for the objective function f_1 are plotted as a function of the observed values. In the Supporting Information, in addition, the predicted values for f_2 can be found. The adjusted R^2 coefficient of the regression models are 0.862 and 0.631, and the generalized cross-validation (GCV) errors are 0.00136 and 0.00363 for the objective functions f_1 and f_2, respectively. One should note that less

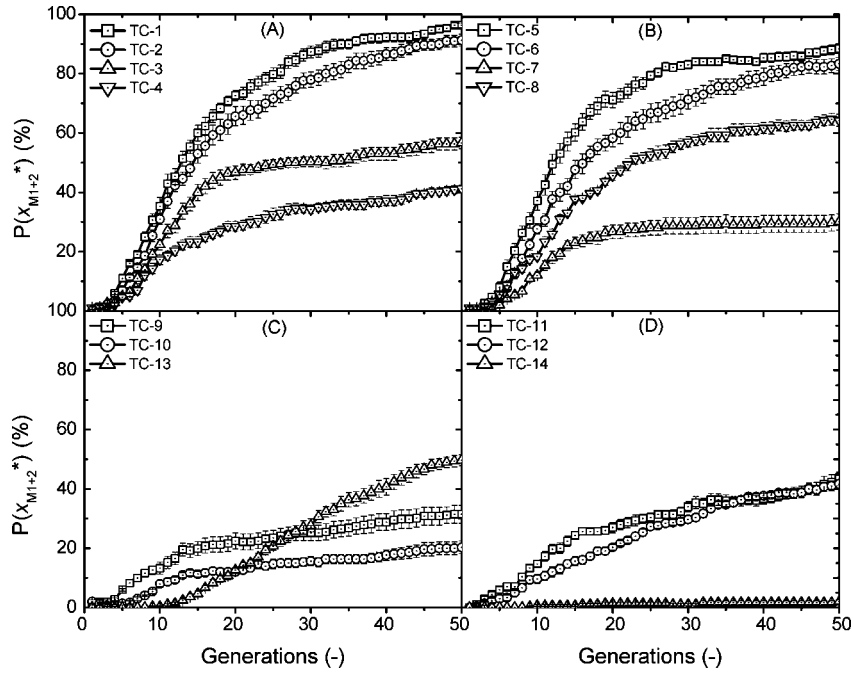


Figure 4. Probabilities $P(x^*_{M1+2})$ to find both optimal combinations of elements for the theoretical problem for all test cases.

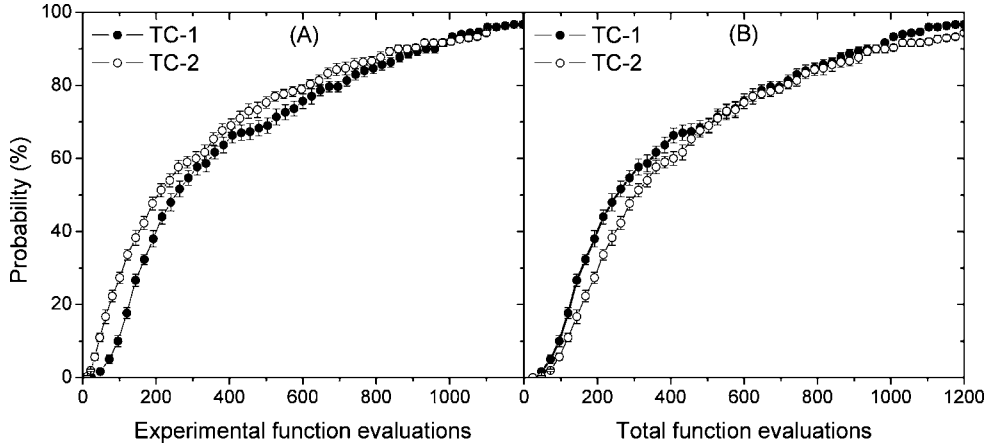


Figure 5. Probabilities $P(x^*_{M1+2})$ to find both optimal combinations of elements for the theoretical problem for the test cases TC-1 and TC-2 as a function of the experimental function evaluations (A) and as a function of the total number of fitness assignments (B).

data points are located in the region with a low fitness value corresponding to a high performance because of the optimization by the genetic algorithm, as can be seen in Figure 3. Thus the model is less accurate in predicting the performance for low fitness values, and a systematic deviation to lower performances was observed in this region.

2.5. Performance Assessment. To compare and quantify the quality of the optimization results by an evolutionary algorithm several performance metrics need to be defined. In this work, some of the metrics were specifically defined with respect to the theoretical test problem, others were metrics commonly used to assess the performance of multiobjective optimizers. Information about the optimization process can be obtained in principle in any of the three spaces, that is, individual, decision, or objective space. In the case of the theoretical test problem, the parameter x_M can be used to assess the performance of the combinatorial part of the problem in the decision space. As described in Chapter 2.3, two optimal element combinations x^*_{M1} and x^*_{M2} exists. An important performance metric can therefore be defined as the probability P to find one or both of these optimal combinations, as each optimization run was repeated 100 times for each algorithm. For instance, if in 50 runs, the algorithm was able to find both combinations, the probability $P(x^*_{M1+2})$ would be 50%. The standard deviation of the probability, corresponding to the error bars in the Figures 4 and 5, was obtained by performing a 4-fold cross validation using four different subsets of 25 runs. This procedure was repeated for each algorithm, that is, NSGA-II, SPEA2, and IBEA, and the arithmetic mean and the standard deviation $\sigma_{Algorithm}$ was calculated. The convergence of the optimization process toward the Pareto-optimal front was measured by defining two additional metrics. The first convergence metric M_1 was defined as the arithmetic mean of the x_M values of the solutions of the Pareto-optimal front. The best value for M_1 is not the average of 0.35 and 0.2875 but is nearer to 0.35 because of a nonuniform distribution of the DTLZ-1 and -2 problems along the combined Pareto-optimal front as can be seen in Figure 2, where the spherical part of the Pareto-optimal front is corresponding to the

DTLZ-2 problem with $x_M = 0.2875$, and the linear part is corresponding to the DTLZ-1 problem with $x_M = 0.35$. The best value found so far for M_1 after 50 generations was 0.33. The second convergence metric M_2 was defined as the arithmetic mean of the objective values f_1 and f_2 of the solutions of the Pareto-optimal front. The difference of M_1 and M_2 is an indication of the convergence state of the algorithm to a global or local Pareto-optimal front because M_1 defines the location of the Pareto-optimal front and M_2 defines the current state of the search. A local Pareto-optimal front is therefore reached in the case of $M_1 > 0.33 \pm 0.01$ and $M_2 - M_1 \leq 0.01$. It should be noted that the use of the arithmetic mean to calculate M_1 and M_2 results in a small error because of the nonuniform distribution of the problems along the Pareto-optimal front. However, the error is the same for both metrics and therefore does not change the information about the convergence state.

In addition, two general performance metrics were used to assess the performance to converge toward an arbitrary Pareto-optimal front and to keep the solutions along this front as diverse as possible. The so-called additive ϵ-indicator, $I_{\epsilon+}$, and the hyperspace indicator I_H were recently developed by Zitzler and Künzli[26] as flexible measures to assess convergence and diversity simultaneously. In the case of a minimization problem, the indicators are also to be minimized. Expressed in generic terms, the ϵ-indicator measures the scalar distance between two solution sets, that is, the factor by which a solution set is worse than another solution set. The hyperspace indicator measures the volume of the objective space dominated by a solution set. The hyperspace indicator especially has advantages over the epsilon indicator for problems with more than two objectives because it captures the change in volume in contrast to the change in distance. An exact definition of both indicators can be found in the original work. In the Supporting Information in Figure S1, the indicator values of three Pareto-optimal fronts are shown to get an idea of the overall sensitivity of these metrics on the convergence and diversity. In this work, these general metrics were used to assess the performance of the optimization of the theoretical problem and of the experimental response surface, that is, the deNOx problem.

3. Test Cases

3.1. Definition of the Decision Space and of the Optimization Problem. A problem typical in combinatorial materials chemistry was chosen as benchmark problem for this study. The so-called deNOx problem consists of finding the best combination and composition of noble metal free elements in a catalyst active at low temperature in the selective catalytic reduction of NO with C_3H_6. It is composed of 11 elements in different combinations and varying concentrations that can be classified into three groups: (1) elements acting as support, (2) the main elements with likely a major contribution to the catalytic reactivity of the catalyst, and (3) elements acting as promoter. Another commonly used group of elements are noble metal elements, which can be treated as elements belonging to group 2 or 3. The incorporation of synthesis conditions is possible in the same way. In our previous work,[22] Al was chosen as the element for the support; Cu, Ni, Co, Fe, Mn, La, Ce, and Sm were chosen as the main elements, and K and Sr were chosen as the promoters. Synthesis conditions were not varied. In this work, we will restrict ourselves to the treatment of this system. Without any boundary conditions, the problem is 11-dimensional. Assuming 100 concentration steps for each element, theoretically, 10^{22} possible combinations exist. In material science it is impossible to synthesize and test such a number of combinations in a reasonable time.

To reduce the decision space, in our previous work,[22] we applied several constraints mostly by incorporation of chemical knowledge to the problem definition. For instance, in the majority of the cases, it is from the chemical point of view not reasonable to screen catalysts containing all elements at the same time. Also elements acting as support or as promoter should be treated in a different manner than elements belonging to group (2). Two systems with different step sizes and concentrations ranges, denoted as system with and without support, were treated separately and the following boundary conditions (denoted as constraint C.*i*) were introduced:

C.1. The maximum number of main elements in a catalyst was four or fewer.

C.2. For systems with support, the Al concentration had to be >33.3 and <95.0 mol%. The concentration of each main element was limited to 35 mol%.

C.3. For systems without support, the maximum allowed Al concentration in a catalyst was 33.3 mol%, and the concentration for each main element was unrestricted.

C.4. The sum of the concentrations of the promoter elements was limited to 5.0 mol%; a catalyst could contain both promoter elements.

C.5. The sum of all concentrations equals 100 mol%. For both systems, Al constituted the remainder

It should be noted that both systems were not really supported because the synthesis of the catalysts was done by impregnation.[22] However, because of the concentration boundary conditions (C.2 and C.3), systems which were denoted as supported generally had a very high Al content. The Al content was not directly evolved during the search because it was obtained from the other components (C.5).

3.2. Representation in the Individual or Search Space. To the best of our knowledge, all the encodings of solid catalysts in previous studies used a representation by vectors or strings, with binary, integer, or real-valued elements or by combination of these elements into a hybrid vector. In this work, a vector representation was also chosen to encode the catalysts. The entire vector **i** consists of three parts: Vector **a** encodes special combinatorial features, such as systems with and without support and promoter existence; this part can also be used to encode additional synthesis parameters. The second vector **b** encodes the presence of main elements in a catalyst. Finally, vector **c** encodes the fraction of each element in the catalyst. Fourteen different encodings were constructed as test cases (TC-1 to TC-14). The TCs differ, for instance, in the way the composition of the catalyst is encoded (binary or real), whether the presence of elements was encoded specifically in vector **b** or only

Table 2. Definition and Overview of Test Cases

	encoding	constraint technique	recombination	mutation
TC-1	3b-8b-16b	repair function	1X+1X	M
TC-2	3b-8b-16b	penalty function	1X+1X	M
TC-3	3b-6b-16b	6-bit lookup table	1X+1X	M
TC-4	3b-8b-16b	8-bit lookup table	1X+1X	M
TC-5	3b-8b-4f	repair function	1X+SBX	M+PM
TC-6	3b-8b-4f	penalty function	1X+SBX	M+PM
TC-7	3b-6b-4f	6-bit lookup table	1X+SBX	M+PM
TC-8	3b-8b-4f	8-bit lookup table	1X+SBX	M+PM
TC-9	3b-8b-32b		1X+1X	M
TC-10	3b-8b-32b	8-bit lookup table	1X+1X	M
TC-11	3b-8b-8f		1X+SBX	M+PM
TC-12	3b-8b-8f	8-bit lookup table	1X+SBX	M+PM
TC-13	3b-0−32b		1X+1X	M
TC-14	3b-0−8f		1X+SBX	M+PM

intrinsically via the composition, in how constraints were handled, and so on. The ranges of values for the different TCs were defined as follows

$p, q, r \in \mathrm{N}^+$

$\mathbf{i} = (\mathbf{a}^\mathrm{T}, \mathbf{b}^\mathrm{T}, \mathbf{c}^\mathrm{T})$

with

$\mathbf{a} \in \mathrm{B}^p, \mathbf{b} \in \mathrm{B}^q, \mathbf{c} \in \mathrm{B}^r$ for TC-1 to TC-4, TC-9, and TC-10

$\mathbf{a} \in \mathrm{B}^p, \mathbf{b} \in \mathrm{B}^q, \mathbf{c} \in \mathrm{R}^r$ for TC-5 to TC-8, TC-11, and TC-12

$\mathbf{a} \in \mathrm{B}^p, \mathbf{b} = \{\ \}, \mathbf{c} \in \mathrm{B}^r$ for TC-13

$\mathbf{a} \in \mathrm{B}^p, \mathbf{b} = \{\ \}, \mathbf{c} \in \mathrm{R}^r$ for TC-14

$\mathrm{B} \in \{0, 1\}$

In Table 2, all the TCs are listed. According to the vector definition, the different encodings are denoted as follows: p{b,f}-q{b,f}-r{b,f}, where p, q, or r are the length of the corresponding vector, and b or f represent binary or a real valued vector elements. The number of main elements in a catalyst was limited to four (constraint C.1) in the case of the test cases TC-1 to TC-8. In contrast, for cases TC-9 to TC-14, the number of main elements was unrestricted, leading to a much larger search and decision space. TC-13 and TC-14 do not encode the element existence (vector **b**), the presence or absence of an element is only intrinsically encoded in the composition vector **c**. For all test cases, a value smaller than 0.05 in the case of real valued elements, or 0, that is, the first step, in the case of bit strings, was considered to be zero. An element is therefore not present if the element existence in vector **b** or the element composition in vector **c** is zero or both are zero. The constraints C.2 to C.5 were satisfied by using a repair algorithm; however, they were only applied to the test cases with an experimental response surface as fitness function. The only difference between the test cases TC-1, TC-2, TC-3, TC-4 on the one hand and TC-5, TC-6, TC-7, TC-8 on the other hand is the type of elements chosen for the element composition (vector **c**), that is, binary or real. If the entire vector **i** consists of different types of elements, the variation operators (recombination and mutation) can not be applied on the whole vector. This is expressed in Table 2 by "1X+SBX", which means that a binary crossover operator was used to recombine the binary part of the vector and the SBX operator for the real valued part. To be directly able to compare binary and real valued encodings, the same concept was also applied to the binary test cases, that is, the variation operators were applied on vector $\mathbf{a} + \mathbf{b}$ and on vector **c** separately. In addition, data showing the results by applying the genetic operators to the whole chromosome are summarized in the Supporting Information in Tables S1−S3. Essentially the same conclusion can be drawn and therefore we do not discuss them further.

3.3. Constraint Handling Techniques. Michalewicz et al.[36] investigated several strategies to handle constraints in the case of numerical optimization problems. Some of the most popular techniques are (a) methods based on rejection of unfeasible solutions (death penalty), (b) methods based on penalty functions, (c) methods based on special representations and genetic operators, and (d) methods based on repair algorithms. In our previous work, we chose option d as constraint handling technique because the evaluation system is used at maximal capacity, and the repair of unfeasible solutions in the deNOx problem is relatively easy. In this work, options b, c, and d will be investigated and compared. In a recent paper Holena et al.[37] described a way to satisfy constraints by using a combination of methods c and d to keep the individuals in the population valid.

The repair algorithms for the combinatorial part (vector **b**) and the continuous part (vector **c**) were defined in the same way as in our previous work.[22] Flowcharts of the algorithms are listed in the Supporting Information. In brief, these algorithms work as follows: If the number of elements is not in the allowed range, select an element randomly and change its existence to zero. Repeat this step until the number of elements is valid. In the case of the continuous part, the repair algorithm works in a similar way. If the total concentration range is too low or too high, select a random element and change its concentration by one step to get closer to the allowed concentration range. Repeat this until the individual is valid. The penalty function for the combinatorial part was defined as follows: the x_M value of an unfeasible individual, that is, an individual with more than four main elements, is penalized by the excess of elements times a constant factor of 0.2. The incorporation of the constraints into the representation was achieved in the following way: the number of combinations of exactly four elements out of eight is 70, which fits into six bits if undesired combinations, as for instance combinations containing all three elements from the lanthanides, are removed. If combinations of one up to four elements out of eight are considered, the number of combinations easily fits into eight bits. The corresponding six-bit and eight-bit look-up tables are listed in the Supporting Information.

4. Results and Discussion

The results of all test cases for the theoretical test problem are summarized in Table 3 after 10 and in Table 4 after 50 generations. Typically the number of experiments in experimental optimization is strongly limited because of time, cost, or other resources, and therefore, the results after 10 generations or 240 function evaluations are important to understand the influence of the representation on the initial search phase of the algorithm, which is especially relevant for experimental optimization. In Figure 4, the probability to find both solutions is shown as a function of the number of generations for all test cases. As can be easily seen on

Table 3. Results of the Optimization of the Theoretical Test Function after 10 Generations

	$P(x^*_{M1})$ (%)	$P(x^*_{M2})$ (%)	$P(x^*_{M12})$ (%)	$\sigma_{Algorithm}$ (%)	M_1 (−)	M_2 (−)	$M_2 - M_1$ (−)	$I_{\epsilon+}$ (−)	I_H (−)
TC-1	52	94	48	7.2	0.35 ± 0.02	0.38 ± 0.02	0.03	0.08 ± 0.02	0.04 ± 0.02
TC-2	42	91	38	10.3	0.35 ± 0.02	0.38 ± 0.02	0.03	0.07 ± 0.02	0.03 ± 0.01
TC-3	38	65	19	21.7	0.37 ± 0.03	0.39 ± 0.03	0.03	0.07 ± 0.02	0.03 ± 0.02
TC-4	21	88	18	22.2	0.36 ± 0.02	0.39 ± 0.02	0.03	0.09 ± 0.03	0.05 ± 0.02
TC-5	38	99	37	6.2	0.36 ± 0.03	0.41 ± 0.04	0.05	0.12 ± 0.04	0.08 ± 0.03
TC-6	28	98	28	5.5	0.36 ± 0.02	0.41 ± 0.03	0.05	0.12 ± 0.03	0.09 ± 0.03
TC-7	18	84	12	14.4	0.36 ± 0.02	0.38 ± 0.02	0.02	0.12 ± 0.03	0.07 ± 0.02
TC-8	21	89	18	16.7	0.37 ± 0.03	0.43 ± 0.04	0.06	0.10 ± 0.02	0.05 ± 0.02
TC-9s	14	97	13	17.9	0.35 ± 0.03	0.36 ± 0.03	0.01	0.08 ± 0.02	0.03 ± 0.01
TC-10	11	79	8	12.5	0.37 ± 0.03	0.44 ± 0.04	0.07	0.10 ± 0.03	0.06 ± 0.02
TC-11	15	98	15	7.9	0.38 ± 0.03	0.39 ± 0.04	0.01	0.08 ± 0.03	0.04 ± 0.02
TC-12	10	89	10	31.6	0.37 ± 0.02	0.51 ± 0.05	0.14	0.11 ± 0.03	0.07 ± 0.02
TC-13	15	3	0	173.2	0.45 ± 0.05	0.52 ± 0.05	0.08	0.11 ± 0.02	0.06 ± 0.02
TC-14	7	2	1	173.2	0.46 ± 0.04	0.61 ± 0.08	0.16	0.17 ± 0.05	0.12 ± 0.04

Table 4. Results of the Optimization of the Theoretical Test Function after 50 Generations

	$P(x^*_{M1})$ (%)	$P(x^*_{M2})$ (%)	$P(x^*_{M12})$ (%)	$\sigma_{Algorithm}$ (%)	M_1 (−)	M_2 (−)	$M_2 - M_1$ (−)	$I_{\epsilon+}$ (−)	I_H (−)
TC-1	97	100	97	1.6	0.33 ± 0.01	0.34 ± 0.01	0.01	0.07 ± 0.04	0.01 ± 0.01
TC-2	94	100	94	0.6	0.33 ± 0.01	0.34 ± 0.01	0.01	0.02 ± 0.01	0.01 ± 0.01
TC-3	57	93	50	14.8	0.34 ± 0.01	0.35 ± 0.01	0.01	0.07 ± 0.03	0.02 ± 0.02
TC-4	41	100	41	28.0	0.34 ± 0.01	0.35 ± 0.00	0.00	0.07 ± 0.02	0.02 ± 0.01
TC-5	88	100	88	4.5	0.34 ± 0.02	0.37 ± 0.03	0.03	0.09 ± 0.06	0.04 ± 0.04
TC-6	83	100	83	5.9	0.34 ± 0.01	0.37 ± 0.02	0.03	0.09 ± 0.05	0.04 ± 0.03
TC-7	33	95	30	15.3	0.34 ± 0.01	0.35 ± 0.01	0.01	0.09 ± 0.03	0.04 ± 0.02
TC-8	65	100	64	19.3	0.34 ± 0.01	0.37 ± 0.02	0.03	0.05 ± 0.02	0.02 ± 0.01
TC-9	31	99	31	10.5	0.34 ± 0.03	0.34 ± 0.03	0.00	0.10 ± 0.03	0.04 ± 0.02
TC-10	21	99	20	7.5	0.35 ± 0.01	0.35 ± 0.01	0.00	0.08 ± 0.05	0.03 ± 0.04
TC-11	43	100	43	13.1	0.36 ± 0.02	0.36 ± 0.03	0.01	0.09 ± 0.07	0.04 ± 0.03
TC-12	42	99	42	2.8	0.34 ± 0.01	0.41 ± 0.04	0.07	0.08 ± 0.03	0.04 ± 0.02
TC-13	89	50	50	13.7	0.34 ± 0.03	0.37 ± 0.03	0.03	0.34 ± 0.03	0.23 ± 0.04
TC-14	16	5	2	50.0	0.42 ± 0.05	0.48 ± 0.05	0.06	0.17 ± 0.05	0.15 ± 0.05

the basis of the results in Tables 3 and 4 and in Figure 4A/B, the highest probability to find both solutions after 10 generations was observed for test case TC-1. In nearly 50% of the cases, this representation was able to find both catalyst combinations. TC-2, which uses a penalty based constraint handling technique and the corresponding real valued representations TC-5 and TC-6 were in roughly 30% of the cases able to find both solutions. For all other test cases, the probability decreased to 20% or even less. This observation confirms the assumption that the representation strongly influences the search. The test cases TC-3 and TC-7, which are encoded with two bits less than the other encodings, were performing significantly worse. Thus it is obvious that representations with fewer vector elements are not always better than representations with more elements. The correspondence between the individual and the decision space is bijective in the case of TC-3 and TC-7 (3b-6b-16b and 3b-6b-4f), that is, each individual maps exactly to one composition. A so-called one-to-one correspondence between the two spaces exists. In contrast, a surjective correspondence exists for the other test cases (3b-8b-16b or 3b-8b-4f) because an element can be omitted by setting $\mathbf{b}_i$ or the element composition in vector **c** to zero or both, and in the case of less than four elements, vector **c** can be partially modified without changing the catalyst composition in the decision space for the test cases TC-1 to TC-8. It is favorable to allow various pathways in the individual space to reach a certain solution in the decision space. In the case of a 3b-8b-16b or a 3b-8b-4f vector representation, the algorithm is able to evolve a certain combination of elements by evolution of vector **b**, the combinatorial part, or vector **c**, the continuous part, or both. In addition, the lower performance of the test cases which intrinsically handle the constraints by using a look up table can be related to the improper variation operators, as will be discussed later.

Also on the basis of the convergence metrics M_1 and M_2 or on the ϵ- and hyperspace indicator in Table 3 or 4, it can be seen that the best performance to converge efficiently toward the global Pareto-optimal set and to keep this set as diverse as possible was only achieved by TC-1 and TC-2. They were able after 50 generations to reach $M_1 = 0.33$ (Table 4) and also have low ϵ- and hyperspace indicator values, an evidence for highly diverse Pareto-optimal fronts. The best diversity after 50 generations was obtained by TC-2. In contrast, independent of the multiobjective algorithm, stronger clustering was observed in all cases for the real valued representations compared to the binary representations. As can be seen in Tables 3 and 4, the best real valued representations TC-5 and TC-6 have rather large values for the ϵ- and hyperspace indicator, which is because of the cluster formation along the Pareto-optimal front. In the case of a binary representation the finite step size keeps the solutions in the decision space at a minimum distance and helps to reduce cluster formation in the objective space. In addition, the individual space is smaller in the case of the binary representation, where the step size for each concentration can be adjusted to a reasonable value. For instance, as already mentioned in our previous work, in the case of experimental optimization, the step size should be significantly larger than the experimental error of the system. Nevertheless, the choice between real valued representations and binary representations is difficult because the perfor-

mance is in both cases very good. Adaptation by building blocks, as described by the traditional schema theorem,[38] is only valid for binary representations in combination with the typical binary crossover operators. Also the binary codification offers the maximum number of schemata per bit of information of any representation.[1] However, when applied to high-precision numerical problems, a real valued encoding in combination with well-designed genetic variation operators can be favorable. If only one objective is to be optimized, the clustering is less problematic. Thus, if a local optimization, that is, an optimization with a small step size is desired in addition to a good global optimization performance a real valued representation might be preferable. However, in most of the cases, genetics algorithms for combinatorial library design are used for the first screening of large search spaces, and only a global optimization is intended; in addition a local optimization can be realized by further optimization using a local optimization technique.

The standard deviations $\sigma_{\text{Algorithm}}$ of the multiobjective algorithms, that is, IBEA, SPEA2, and NSGA-II, to find with the same probability both element combinations are also given in Tables 3 and 4. A small deviation signifies that the performance is not significantly influenced by the algorithm. The test cases TC-1/TC-2 and TC-5/TC-6, which do not directly incorporate the constraints in the representation, were almost not influenced by the algorithm. In conclusion, the overall best performance was obtained by using a binary representation with a surjective correspondence between individual and decision space, which uses either repair algorithms or penalty functions as constraint handling heuristics. The use of special representations to satisfy the constraints did not lead to an improved performance. This was because of improper variation operators, which were not able to create new individuals with properties common to their parents. It is therefore very important only to use special representations if the variation operators are adequately adapted. This was not the case for TC-3/TC-4 and TC-7/TC-8, which satisfy the constraints by using a lookup table. Infrequently representations using integer valued element were used to represent solid catalysts or additional synthesis parameters.[39] The use of integer representations has the drawback that problem specific variation operators have to be defined, and as mentioned above, special operators have to be very carefully designed to produce the desired effects. In general, the generated effects are not the same as in the case of a binary representation. In most of the cases, an integer-valued representation can be easily converted into a binary representation, which simplifies the discovery of good schemata, as more schemata are available. Hence, it is usually more convenient to use a binary instead of an integer representation.

In Figure 5, the probabilities to find both optimal combinations for the test cases TC-1 and TC-2 are plotted as a function of the experimental evaluations and as a function of the total number of fitness assignments. TC-1 uses a repair function; therefore, all individuals are valid and the number of function evaluations is equal to the number of experimental evaluations. In contrast, TC-2 uses a penalty function, and the invalid individuals are not experimentally evaluated. Especially in the initial phase up to 50% of the individuals were not valid, as shown in Figure S5 in the Supporting Information. In Figure 5, this can be seen because TC-2 performs better than TC-1 in terms of experimental functions evaluations up to 400 evaluations, whereas it performs worse in terms of the total number of fitness assignments. The reason for the lower performance of TC-1 compared to TC-2 in terms of experimental functions evaluations is the following: if a repair function is used, all individuals in a population are valid, and the information content per generation is higher than without a repair function because more individuals are tested. However, the repair operator can be considered as a third stochastic variation operator, which may destroy knowledge gained throughout the optimization process. Thus the information content per generation is not optimal. The penalty function does not destroy any knowledge, and the algorithm is able to proceed the search without any disruption. The information content in terms of experimentally evaluated individuals is optimal because each valid individual was created by the algorithm itself and not by the repair function. For practical purposes, the choice of the constraint handling technique depends on the testing equipment in the laboratory. If a parallel system is used and the time to test a full population is nearly the same compared to the time needed to test only the valid individuals, a repair algorithm would be the better choice. In contrast, if the individuals have to be tested in a sequential way, a penalty heuristic is preferable.

In Figure 4C/D and Tables 3 and 4, the results of the test cases TC-9 to TC-14 are shown. The maximum allowed number of main elements was not limited for these test cases. It can be easily seen that the mush larger search space directly led to a significant inferior overall performance. The results are consistent with the findings discussed previously: the test cases which encode the combinations in a lookup table were performing worse than the ones with a representation compatible to the recombination operator. Real valued representations have a higher trend to form clusters. TC-13 and TC-14 do not encode the existence of an element in addition to its chemical composition. An interesting new conclusion can be drawn by comparing these two test cases to the others. Up to the 10th generation, these test cases were not able to find both optimal element combinations. The whole optimization process seems to be delayed. Interestingly, after this delay, TC-13 was able to find the optimal combinations quite well. The optimal combination with four main elements (x^*_{M1}) was found much easier compared to the one with only one element (x^*_{M2}). This is because the search only occurs through evolution of the element compositions, and in such a case, it is easier for the algorithm to find the combination with four elements. Because of the much larger search space for the real-valued encoding TC-14 compared to the binary TC-13, the algorithm was nearly unable to find both element combinations in a reasonable time. This confirms the importance of encoding the element existence, especially for real-valued representations. To omit the element existence, in addition to the composition, only makes sense in cases where the search is intended to be focused on a fixed number of elements present in the catalyst.

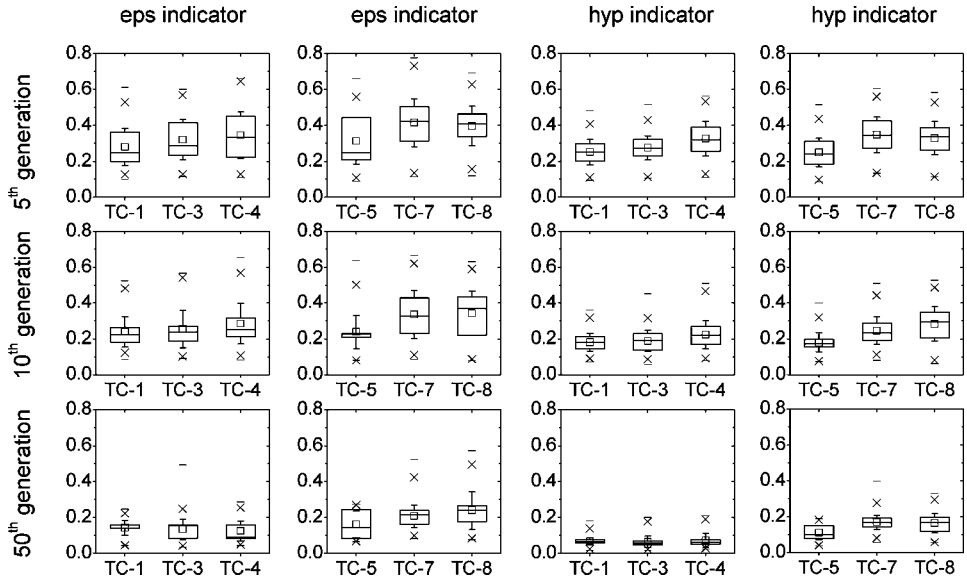


Figure 6. Results of the optimization of the experimental response surface after 5, 10, and 50 generations for several binary and real valued test cases. The box represents the lower and the upper quartile and the median. The whiskers represent the standard deviation. The minimum and maximum values are represented by a bar, the 1% and 99% marks by a cross, and the mean value by a square.

In most practical problems in combinatorial chemistry, this is not the case, and to achieve a high initial convergence rate, it is mandatory to encode the element existence in addition to the composition. Watanabe et al.[40] used a genetic algorithm to optimize Cu oxide catalysts for the methanol synthesis. The decision space consisted in combinations of five elements. The elemental composition was represented by a binary vector of 36 bits (6 bits for each element) without additionally encoding the element's existence. The optimum catalyst that was found consisted of four elements, and therefore, the genetic algorithm had to dismiss the superfluous element by changing its composition until zero. In the majority of the cases, this requires more iteration steps than the rejection of an element by changing the bit corresponding to its existence. In the case of the study by Watanabe et al.,[40] the missing element existence did not had a significant impact on the performance because the optimum combination consisted of nearly as many elements as the starting combination, and a binary representation with only few steps for each element was used. In contrast, a significant impact would be expected if the element composition is represented by a real valued vector or by a binary vector with many bits (>12 bits for each element) or if the optimum combination consists of much less elements than the starting combination. Wolf et al.[14] and other groups[41] recognized this advantage and encoded the element existence in addition to the real valued representation of the element composition.

In the following, the results of investigating the theoretical problem will be validated on the basis of the experimental response surface. The system to optimize was identical to the system we investigated experimentally in our previous work.[22] All constraints were applied, and we restricted ourselves to systems with up to four main elements. The discrete concentration steps in mol % in the case of a binary encoding can be found in our previous work or in the Supporting Information of this contribution. In Figure 6, boxplots of the ϵ- and the hyperspace indicator for all three algorithms after 5, 10, and 50 generations are shown for the binary test cases TC-1, TC-3, and TC-4 and for the real valued test cases TC-5, TC-7, and TC-8. It is reasonable to compare the results in terms of the median and the worst performing solution and not in terms of the best solution. After 5 and 10 generations, the test case TC-1 was the best performing binary representation, as expected on the basis of the results of the theoretical test function. After 50 generations, the trend was not clear anymore, and the median values of TC-1, TC-3, and TC-4 were almost the same. However, the worst solution found by TC-1 was better than the one found by TC-3 and TC-4. The direct comparison between the binary and the real valued test cases shows that the real valued cases were generally performing worse. The standard deviations were higher, which is an indication for a less stable optimization. The best real valued test case was TC-5. By comparison of the best performing binary and real valued test cases TC-1 and TC-5, it can be seen that the median value of TC-5 was sometimes slightly better than the one for TC-1. The worst performing solution of the real valued representation was always inferior compared the solution found by the binary encoding. After 50 generations, it can be seen that the distributions of the solutions found by the binary test cases were much more narrow in comparison to the distributions of the real valued test cases. The explanation for the inferior performance of the real valued test cases was already given on the basis of the theoretical test functions and is also valid for the experimental optimization problem. In conclusion, the combinatorial theoretical test problem presented in this work allows understanding the optimization process of similar problems and can be used as theoretical benchmark problem for the development of genetic algorithms for combinatorial library design of solid catalysts.

In Figure 7, a representative Pareto-optimal front obtained by optimization of the experimental response surface and the SPEA-2 Pareto-optimal front found during the experimental optimization of the same system in our previous work[22] are compared. It can be seen that the optimization

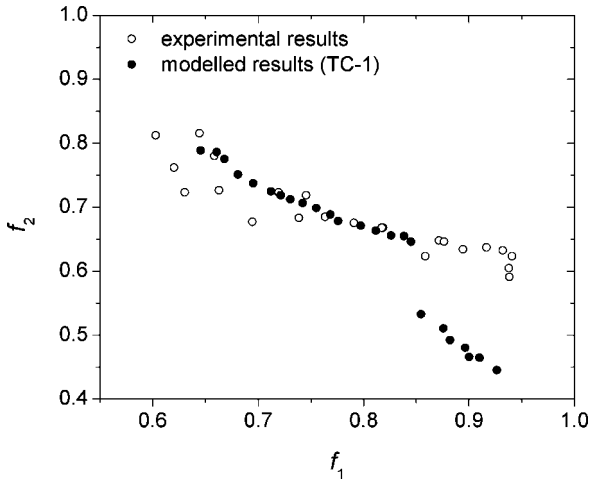


Figure 7. Comparison of a representative Pareto-optimal front obtained by optimization of the experimental response surface and the SPEA-2 Pareto-optimal front found during the experimental optimization in our previous work[22].

of the experimental response surface led to a very similar Pareto-optimal front. Both fronts were composed by Cu-, Ni-, Co-, and Al-based catalysts in varying compositions. The shapes of the Pareto-optimal fronts are comparable. Interestingly better solutions for f_1 were obtained during the experimental optimization. This is because of an erroneous modeling of the solutions for low fitness values of f_1 as already stated in section 2.4, and therefore, the regression model was not able to properly reproduce the Pareto front in this region. The predicted values for f_1 were larger than the observed. In the case of the second objective function f_2 or the lower right region of the Pareto front, the predicted values were smaller than the observed, which led to too low values in this region. This is also because of the inaccuracy of the regression model. However, we are confident, that the experimental response surface leads to a similar optimization problem as the experimental problem and can be used to validate the results of the theoretical test problem.

5. Conclusions

Genetic algorithms are highly valuable tools for combinatorial library design in high dimensional chemical spaces. Because of their flexibility, the implementation and adaptation on the specific problem can be very challenging. In this contribution, we systematically investigated the influence of the representation on the performance of the optimization in the case of a typical problem in combinatorial material science. A new theoretical test problem was defined and optimized using various types of representations. The results clearly indicate that the performance strongly depends on the representation. Several multiobjective algorithms were used to optimize the system, and no significant dependence of the performance on the optimization algorithm was observed for well defined representations. Variation operators have to be adapted to the representation to produce meaningful offsprings. Improper recombination operators lead to a significantly reduced performance. A surjective correspondence between individual and decision space was found to be favorable, in contrast to a strictly one-to-one correspondence, which narrows the possibilities to evolve through different pathways. In the case of experimental optimization, where the number of experiments is strongly limited, especially a good initial performance is required. This can only be achieved by encoding the element combinations in addition to their compositions. A binary encoding of the element compositions has several advantages compared to real valued encodings: The search space can be tailored by choosing a step size for each concentration to a reasonable value, which should be significantly larger than the experimental error of the system. In addition, because of the discrete step size and the reduced dimensionality of the search space, the individuals are kept at a minimum distance, which significantly reduces the tendency to form clusters. Finally, we observed that a repair heuristic to satisfy the constraints is a very good choice if the time to experimentally test one generation is not significantly less than the time needed to test the valid individuals in a consecutive way. If one can only test in a sequential manner and thus would loose substantial amounts of time in testing invalid individuals, a penalty function would be the better choice.

Supporting Information Available. Figures showing the predicted values of the regression model for f_2, the repair algorithms for vectors **b** and **c**, and the number of times the repair algorithms were applied, additional test cases, and tables with the 6-bit and 8-bit lookup tables and the discrete encodings of the elements concentrations. This material is available free of charge via the Internet at http://pubs.acs.org.

References and Notes

(1) Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*; Addison-Wesley Longman Publishing Co., Inc.: Boston, MA, 1989.
(2) Lucasius, C. B.; Kateman, G. *Chemom. Intell. Lab. Syst.* **1993**, *19*, 1–33.
(3) Lucasius, C. B.; Kateman, G. *Chemom. Intell. Lab. Syst.* **1994**, *25*, 99–145.
(4) Hibbert, D. B. *Chemom. Intell. Lab. Syst.* **1993**, *19*, 277–293.
(5) Broach, J. R.; Thorner, J. *Nature* **1996**, *384*, 14–16.
(6) Cole, B. M.; Shimizu, K. D.; Krueger, C. A.; Harrity, J. P. A.; Snapper, M. L.; Hoveyda, A. H. *Angew. Chem., Int. Ed.* **1996**, *35*, 1668–1671.
(7) Xiang, X. D.; Sun, X. D.; Briceno, G.; Lou, Y. L.; Wang, K. A.; Chang, H. Y.; Wallacefreedman, W. G.; Chen, S. W.; Schultz, P. G. *Science* **1995**, *268*, 1738–1740.
(8) Senkan, S. M. *Nature* **1998**, *394*, 350–353.
(9) Jandeleit, B.; Schaefer, D. J.; Powers, T. S.; Turner, H. W.; Weinberg, W. H. *Angew. Chem., Int. Ed.* **1999**, *38*, 2494–2532.
(10) Hoffmann, C.; Schmidt, H. W.; Schuth, F. *J. Catal.* **2001**, *198*, 348–354.
(11) Senkan, S. *Angew. Chem., Int. Ed.* **2001**, *40*, 312–329.
(12) Weber, L.; Wallbaum, S.; Broger, C.; Gubernator, K. *Angew. Chem., Int. Ed.* **1995**, *34*, 2280–2282.
(13) Singh, J.; Ator, M. A.; Jaeger, E. P.; Allen, M. P.; Whipple, D. A.; Soloweij, J. E.; Chowdhary, S.; Treasurywala, A. M. *J. Am. Chem. Soc.* **1996**, *118*, 1669–1676.
(14) Wolf, D.; Buyevskaya, O. V.; Baerns, M. *Appl. Catal. A: General* **2000**, *200*, 63–77.
(15) Pereira, S. R. M.; Clerc, F.; Farrusseng, D.; van der Waal, J. C.; Maschmeyer, T.; Mirodatos, C. *QSAR Comb. Sci.* **2005**, *24*, 45–57.
(16) Clerc, F.; Lengliz, M.; Farrusseng, D.; Mirodatos, C.; Pereira, S. R. M.; Rakotomalala, R. *Rev. Sci. Instrum.* **2005**, 76.

(17) Pereira, S. R. M.; Clerc, F.; Farrusseng, D.; van der Waal, J. C.; Maschmeyer, T. *Comb. Chem. High Throughput Screen.* **2007**, *10*, 149–159.
(18) Rodemerck, U.; Baerns, M.; Holena, M.; Wolf, D. *Appl. Surf. Sci.* **2004**, *223*, 168–174.
(19) Umegaki, T.; Watanabe, Y.; Nukui, N.; Omata, K.; Yamada, M. *Energy Fuels* **2003**, *17*, 850–856.
(20) Corma, A.; Serra, J. M.; Serna, P.; Valero, S.; Argente, E.; Botti, V. *J. Catal.* **2005**, *229*, 513–524.
(21) Serra, J. M.; Corma, A.; Valero, S.; Argente, E.; Botti, V. *QSAR Comb. Sci.* **2007**, *26*, 11–26.
(22) Gobin, O. C.; Martinez Joaristi, A.; Schuth, F. *J. Catal.* **2007**, *252*, 205.
(23) Bleuler, S.; Laumanns, M.; Thiele, L.; Zitzler, E. In *Evolutionary Multi-Criterion Optimization*; Proceedings of the Second International Conference, EMO 2003, Faro Portugal, April 8–11, 2003; Springer: Birmingham, U.K. 2003; Vol. 2632, pp 494−508.
(24) Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. *IEEE Trans. Evol. Comp.* **2002**, *6*, 182–197.
(25) Zitzler, E.; Laumanns, M.; Thiele, L. In *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems*; EUROGEN 2001; Giannakoglou, K., Tsahalis, D., Periaux, J., Papailiou, K., Fogarty, T., Eds.; International Center for Numerical Methods in Engineering (CIMNE): Barcelona, Spain, 2002; pp 95−100.
(26) Zitzler, E.; Künzli, S. In *Conference on Parallel Problem Solving from Nature (PPSN VIII)*; Springer: Birmingham, U.K. 2004; Vol. 3242, pp832−842.
(27) Gobin, O. C. Diploma thesis, Technical University of Munich, 2007.
(28) Deb, K.; Goyal, M. *Comp. Sci. Inf.* **1996**, *26*, 30–45.
(29) Deb, K.; Agrawal, R. B. *Complex Syst.* **1995**, *9*, 115–148.
(30) Zitzler, E.; Thiele, L.; Laumanns, M.; Fonseca, C. M.; da Fonseca, V. G. *IEEE Trans. Evol. Comp.* **2003**, *7*, 117–132.
(31) Knowles, J.; Thiele, L.; Zitzler, E., *A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers*; Computer Engineering and Networks Laboratory (TIK): ETH Zurich, Switzerland, 2006.
(32) RANDOM.ORG. True Random Number Service http://www.random.org.
(33) Farrusseng, D.; Clerc, F. *Appl. Surf. Sci.* **2007**, *254*, 772–776.
(34) Deb, K.; Thiele, L.; Laumanns, M.; Zitzler, E. Scalable multiobjective optimization test problems. In *Congress on Evolutionary Computation*; CEC 2002; IEEE: Piscataway, NJ, 2002; Vol. 1, pp 825–830.
(35) Friedman, J. H. *Ann. Stat.* **1991**, *19*, 1–67.
(36) Michalewicz, Z.; Dasgupta, D.; Leriche, R. G.; Schoenauer, M. *Comp. Ind. Eng.* **1996**, *30*, 851–870.
(37) Holena, M.; Cukic, T.; Rodemerck, U.; Linke, D. *J. Chem. Inf. Mod.* **2008**, *48*, 274–282.
(38) Holland, J. H. *Adaptation in Natural and Artificial Systems*; The University of Michigan Press: Ann Arbour, MI, 1975.
(39) Serra, J. M.; Chica, A.; Corma, A. *Appl. Catal. A: General* **2003**, *239*, 35–42.
(40) Watanabe, Y.; Umegaki, T.; Hashimoto, M.; Omata, K.; Yamada, M. *Catal. Today* **2004**, *89*, 455–464.
(41) Paul, J. S.; Janssens, R.; Denayer, J. F. M.; Baron, G. V.; Jacobs, P. A. *J. Comb. Chem.* **2005**, *7*, 407–413.
(42) Zitzler, E. PhD thesis, ETH Zurich, 1999.

CC800046U