Constraint-handling techniques for generative product design systems in the mass customization context

Nordin, Axel; Motte, Damien; Hopf, Andreas; Bjärnemo, Robert; Eckhardt, Claus-Christian

Published in:
Artificial Intelligence for Engineering Design, Analysis and Manufacturing

DOI:
10.1017/S0890060413000383

Published: 2013-01-01

Document Version:
Peer reviewed version (aka post-print)

Link to publication

Citation for published version (APA):
Constraint-handling techniques for generative product design systems in the mass customization context

Axel Nordin, Damien Motte, Andreas Hopf, Robert Bjärnemo and Claus-Christian Eckhardt

Artificial Intelligence for Engineering Design, Analysis and Manufacturing / Volume 27 / Issue 04 / November 2013, pp 387 - 399
DOI: 10.1017/S0890060413000383, Published online: 18 October 2013

Link to this article: http://journals.cambridge.org/abstract_S0890060413000383

How to cite this article:

Request Permissions : Click here
Constraint-handling techniques for generative product design systems in the mass customization context

AXEL NORDIN, DAMIEN MOTTE, ANDREAS HOPF, ROBERT BJÄRNEMO, AND CLAUS-CHRISTIAN ECKHARDT
Department of Design Sciences, Faculty of Engineering LTH, Lund University, Lund, Sweden
(RECEIVED June 15, 2012; ACCEPTED May 1, 2013)

Abstract
Generative product design systems used in the context of mass customization are required to generate diverse solutions quickly and reliably without necessitating modification or tuning during use. When such systems are employed to allow for the mass customization of product form, they must be able to handle mass production and engineering constraints that can be time-consuming to evaluate and difficult to fulfill. These issues are related to how the constraints are handled in the generative design system. This article evaluates two promising sequential constraint-handling techniques and the often used weighted sum technique with regard to convergence time, convergence rate, and diversity of the design solutions. The application used for this purpose was a design system aimed at generating a table with an advanced form: a Voronoi diagram based structure. The design problem was constrained in terms of production as well as stability, requiring a time-consuming finite element evaluation. Regarding convergence time and rate, one of the sequential constraint-handling techniques performed significantly better than the weighted sum technique. Nevertheless, the weighted sum technique presented respectable results and therefore remains a relevant technique. Regarding diversity, none of the techniques could generate diverse solutions in a single search run. In contrast, the solutions from different searches were always diverse. Solution diversity is thus gained at the cost of more runs, but no evaluation of the diversity of the solutions is needed. This result is important, because a diversity evaluation function would otherwise have to be developed for every new type of design. Efficient handling of complex constraints is an important step toward mass customization of nontrivial product forms.

Keywords: Complex Morphologies; Constraint-Handling Techniques; Evolutionary Computing; Generative Design; Genetic Algorithms

1. INTRODUCTION
One can sense an evolution of the largely static relationship between the consumer and the product. There is an increasing desire to participate in the designing of products and the potential experiences consumers will share with them. As put forward by Friebe and Ramge (2008), the upsurge of independent fashion labels, “crowdsourcing” initiatives, and co-working spaces indicates consumer’s demand for empowerment. This need for cocreation, implemented already in the textile (Lakhani & Kanji, 2009) and food industries (Kraft Foods, 2006) but also in more advanced consumer goods businesses like sportswear (Moser et al., 2006; Bouché, 2009), goes well beyond mere material and color choices: the future “prosumer” (Toffler, 1971) desires control over product form as well. This challenge poses major difficulties. First, consumers are not always knowledgable enough to evaluate how their design preferences may affect the functionality and manufacturability of the product. Second, if the desired product form is complex, such as in nature-inspired forms or shape grammars, even the manipulation of the form can be too cumbersome for consumers not skilled in three-dimensional modeling. The consumer must therefore be supported in form manipulation in some way. Finally, if mass customization is understood as the mass production of customized goods (Kaplan & Haenlein, 2006; Trubridge, 2010, p. 169), the product form is often severely constrained by the production system.

A possible solution to these difficulties is to implement a generative design system (GDS) that generates product designs that fulfill mass production and engineering constraints, along with consumer requirements (such as size, contour, and materials), while leaving the consumer in control of the final
design selection. A GDS intended for product design is basically structured around a graphical user interface with which the user can evaluate, select and influence the generation of product forms. A GDS is often based on an interactive optimization system or constraint satisfaction system that handles user preferences and technical constraints. A GDS is frequently used to handle complex forms usually intractable to the user. Most GDSs have been intended for professional designers, for example, to help the designer preserve the “form identity” of a brand (Pugliese & Cagan, 2002; Chau et al., 2004; McCormack et al., 2004), but they have not been specifically designed for use by consumers. Letting the consumers control their own design adds a number of requirements to the GDS.

First, such a GDS is to be used repeatedly; therefore, the solutions must be generated quickly (how fast the system is able to converge to viable solutions, i.e., convergence time, is important) and in a reliable manner (how often the system is able to converge to viable solutions, i.e., convergence rate, is important), and the system must be applicable to a wide range of problems without requiring extensive modification of the algorithm by the consumer or a programmer.

Second, consumers must be able to choose from a set of solutions, because the decision to choose one design solution over another is often not based on pure performance metrics but rather on criteria that are subjective and difficult to quantify. At the same time, in order to give the consumer a meaningful choice, the generated shapes need to fulfill all technical constraints, which may be time-consuming to evaluate and hard to satisfy. For an analysis related to structural problems, for instance, finite element techniques may be required. It is therefore necessary that the GDS ensure an adequate diversity among the proposed solutions so that the waiting time and the need to relaunch the generation process are minimized.

Diversity, convergence rate, and convergence time are interrelated: they depend upon how the solutions are generated, that is, how the constraints are handled and the viable solutions are optimized. Of these two activities, the satisfaction of constraints represents the main challenge. The time spent on the optimization can be controlled by the user (optimization can be stopped if deemed to be too time-consuming or if one is satisfied with the result), but the constraint-handling step cannot. Regarding diversity, the constraints can be very hard to satisfy, and the space of feasible solutions in those cases is sparse. Diversity is therefore unlikely to arise during the subsequent optimization step if it has not during the constraint-handling step. Finally, the convergence rate also depends on the constraint-handling step, because solutions are viable only if they fulfill all constraints. Therefore, in the following discussion, these issues are considered only under the constraint-handling aspect.

Enabling the efficient handling of such types of constraints is a step toward showing that mass customization of product forms is technically possible. In this paper, three different techniques for handling technical constraints are therefore evaluated in terms of convergence time, convergence rate, and the diversity of the generated solutions using a real design problem. The design problem is to find feasible solutions to a table support structure based on a complex form (a so-called Voronoi diagram) that is subject to technical constraints and user evaluation.

Section 2 reviews related works on GDSs and on constraint-handling techniques (CHTs). The study of diversity, convergence rate, and solution-generation time for a real design problem with selected CHTs is treated in Section 3.

Although this research addresses primarily the use of GDSs in the mass customization context, some aspects of it, especially that pertaining to the diversity issue, should also be useful for GDSs where the user is a professional industrial designer.

2. GDSs AND CHTs

The first part of this section reviews related works on GDSs and reports how constraints are handled in these systems. The second part reviews CHTs and their relevance for consumer-oriented GDSs.

2.1. Related works on GDSs

Few GDSs focus on product forms within the mass customization context. Current GDSs are mainly industrial applications in the form of online product configuration websites offering many diverse forms of mass customization, a large bandwidth of personalization options, navigation techniques, and visual quality. A collection of these websites can be found at MilkorSugar (http://www.milkorsugar.com/). One example is the Kram/Weisshaar Breeding Tables Project, which generates variations of a table design using a genetic algorithm (GA) that modifies a set of parameters ruling the support structure (Kram & Weisshaar, 2003). Despite the steady upsurge in online product configuration, there is no major market player that makes customization of the actual product form and structure available to its customers. It should also be noted that none of these configurators includes evaluation of manufacturing or structural constraints.

Within industrial design research, generative design has been investigated primarily for stylistic purposes. In the seminal work of Knight (1980), a parametric shape grammar was developed for the generation of Hepplewhite-style chairbacks. Orsborn et al. (2006) employed a shape grammar to define the boundaries between different automotive vehicle typologies. Recent works have focused on branding-related issues. With the help of shape grammars, new designs based on the Buick (McCormack et al., 2004), Harley-Davidson (Pugliese & Cagan, 2002), and Coca-Cola and Head & Shoulders (Chau et al., 2004) brands were developed. Further research is being undertaken to develop rules that link form and brand: for example, Cluzel et al. (2012) for systems based on GAs and Orsborn et al. (2008) for shape grammars. Within the mass customization context, Johnson (2012) created a graphical interface for customizing shelves while taking
into account functional aspects such as compartment size, and Piascek and Hanna (2010) used a graphical interface based on numerical sliders to control a shape, as well as a GA to aid in the design of the shape, to investigate the influence of the amount of control on the user’s satisfaction with the system.

Technical constraints and objectives have been studied more extensively within engineering design. An early example is Frazer’s application of a GA to the design of sailing yachts, taking into account constraints such as stability, center of buoyancy, and wetted surface area, as well as less well-defined criteria such as aesthetics, by combining a computational evaluation with a subjective user-based evaluation (Frazer, 1996). Agarwal and colleagues (Agarwal & Cagan, 1998; Agarwal et al., 1999) have associated the shape grammar technique with parametric cost and applied it to the design of coffee makers (see Cagan, 2001, for a review on the use of shape grammars in engineering design). Numerous efforts have also been made to take into account the engineering and production constraints early in the development process, using knowledge-based engineering systems (see El-Sayed & Chassapis, 2005; Sandberg & Larsson, 2006; Lin et al., 2009; Johansson, 2011) or a combination of knowledge-based engineering and optimization systems, as in Petersson et al. (2012), where the lightweight gripper constraint satisfaction and optimization system is based on the weighted sum technique. These works, even if they present interesting design systems, are not primarily concerned with diversity and choice.

Some works are crossing the boundaries between engineering and industrial design, taking into account functional or technical constraints and aesthetics. Shea and Cagan (1999) used a combination of shape grammar and simulated annealing for both functional and aesthetic purposes and applied it to truss structures. The shape grammar technique was used to generate new designs, and the simulated annealing technique to direct the generation toward an optimum. The evaluation was based on a weighted sum of constraint violations and objective values. The design objectives were functional (minimize weight, enclosure space, and surface area), economic, and aesthetic (minimize variations between lengths in order to get uniformity, make the proportions respect the golden ratio). Shea and Cagan’s model was reused by Lee and Tang (2009), with a combination of shape grammar and GA, to develop stylistically consistent forms and it was applied to the designing of a camera. The designs generated took into account the constraints linked to the spatial component configuration. The constraints were handled by minimizing a weighted sum of the constraint violations. A designer was in charge of the aesthetic evaluation, following the interactive GA paradigm. Ang et al. (2006) used shape grammars and GAs to develop the Coca-Cola bottle example of Chau et al. (2004) and added functional considerations (the volume of the bottle) that were constrained to approach the classic Coca-Cola bottle shape. EZCT Architecture & Design Research et al. (2004), within the interactive GA paradigm, developed a set of chairs optimized for weight and stiffness. The designer could define how loads would be applied to the structure before the optimization but could not interact with the system during the optimization. Finally, Wenli (2008) developed a system that, through adaptive mechanisms, allowed it to learn the designer’s intent faster; that system was implemented as a plug-in for a computer-aided design system and applied to boat hull design.

The handling of the constraints in the reviewed works is summarized in Table 1. Of the CHTs reviewed in the previous section, the weighted sum technique is always used if more than one constraint or objective is present.

### 2.2. CHTs

CHTs represent a field of evolutionary computing that is increasing at a fast pace. There are several techniques (for extended reviews, see Michalewicz et al., 1996; Coello Coello, 2002; Mezura-Montes, 2004; Yeniay, 2005). As mentioned in the Introduction, GDSs for mass customization will be used repeatedly. It is therefore necessary to have CHTs that are sufficiently generic for addressing different design problems and that do not require the user to modify the algorithm during use. Many of the common types of CHTs are therefore not applicable, as discussed below.

The most common approach to handling constraints is to use methods based on penalty functions. The concept behind those methods is “to transform a constrained-optimization problem into an unconstrained one by adding (or subtracting) a certain value to/from the objective function based on the amount of constraint violation present in a certain solution” (Coello Coello, 2002). The penalty factors/values must be determined by the user and is problem dependent (Mezura-Montes & Coello Coello, 2006, p. 2). The weighted sum can be seen as one specific penalty technique: the constraints are incorporated into the objective function and the given weights that penalize the fitness value. Another type of CHT consists of trying to maintain feasibility of the solutions (Michalewicz & Janikow, 1991; Schoenauer & Michalewicz, 1996); it requires a feasible starting point that may be computationally costly to find or that must be set by the user (Coello Coello, 2002, p. 1259) and/or necessitates the use of problem-specific operators (Schoenauer & Michalewicz, 1996, p. 245). Another method is based on the search for feasible solutions. One possibility is “repairing” infeasible individuals (see details in Coello Coello, 2002, section 4), which has been proved an efficient method if the individuals can be easily transformed; this unfortunately is rarely the case in real-world engineering problems. Hybrid methods also exist that combine techniques from the different categories above and/or with techniques from other domains, such as fuzzy logic (Van Le, 1996) or constraint satisfaction problems (Paridis, 1994; see also Michalewicz & Schoenauer, 1996; and Coello Coello, 2002). They require supplementary knowledge from the user for their implementation; they have therefore not been investigated further.
The types of CHTs that seem to fit the above-mentioned requirements are the lexicographic, or sequential, CHTs (SCHTs) and the multiobjective optimization techniques. Coming from the domain of multicriteria decision models (e.g., see Bouyssou et al., 2006, pp. 188–191), the lexicographic method consists in considering each constraint separately, in a specific order. When the first constraint is fulfilled, the next constraint is considered. When all constraints are fulfilled, the objective function is optimized. Although these methods do not require extensive tuning of parameters while they are running, it is necessary to select in advance a sequencing of all constraints. However, this choice of sequence needs to be done only once, before the GDS will be used. This aspect is crucial, because the ordering of constraints significantly influences the results in terms of running time and precision (Michalewicz & Schoenauer, 1996). How to choose an optimal sequence has been described elsewhere (Motte et al., 2011). The multiobjective optimization techniques are based on transforming the constraints into objectives to fulfill. This is also a promising technique for engineering optimization problems (for reviews, see Coello Coello, 2002; Mezura-Montes, 2004; Mezura-Montes & Coello Coello, 2006).

In a preceding study (Motte et al., 2011), two SCHTs were investigated against the classic weighted sum scheme using the well-known 10-bar truss benchmark problem (Haug & Arora, 1979); the behavioral memory (BM) method (Schoenauer & Xanthakis, 1993; Michalewicz et al., 1996) and the SCHT (Lexcoht; Nordin et al., 2011). Regarding convergence time, Lexcoht was more often superior to BM, and both were far superior to the weighted sum technique. It is interesting that no significant differences between different weighting schemes were found: the unweighted sum scheme (UWS), a linearly weighted scheme, and an exponentially weighted sum scheme were tested. Finding a relevant weighting scheme therefore does not seem crucial for an efficient use of the weighted sum technique, making it a potentially interesting generic CHT. SCHTs and the weighted sum technique are both therefore candidates for consumer-oriented GDSs. Although less efficient than Lexcoht in terms of convergence time, BM is interesting because its structure (presented below) is based on a diversity measure in order to allow for a greater diversity in solutions.

Therefore, it was decided to compare these three CHTs in terms of diversity, convergence rate, and convergence time. The weighted sum technique is implemented using the UWS. The implementations of Lexcoht and BM are presented in the following sections.

### 2.2.1. Lexcoht

Lexcoht (Nordin et al., 2011) can be described for each constraint by performing the following:

- Evaluate the constraint violation.
- If the constraint is satisfied: evaluate the next constraint.
- If the constraint is not satisfied: stop the evaluation and score the individual according to

\[ p = \frac{m + (1 - a)}{c}, \]

where

\[ m = \sum_{i=1}^{c} \left( \frac{1}{a_i} \right) \]

\[ c = \sum_{i=1}^{c} a_i \]

\[ a_i \] is the weight of constraint \( i \).

The number of constraints listed are those which are handled by the constraint-handling technique (CHT). Constraints handled by knowledge base systems were not taken into account. In the case of interactive genetic algorithms (GAs), the objectives handled by users were not included. NA, not applicable; ND, no data; SA, simulated annealing; WS, weighted sum.

### Table 1. Comparison of the constraint handling of the reviewed works

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Knight, 1980</td>
<td>No</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Orsborn et al., 2006</td>
<td>No</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>McCormack et al., 2004</td>
<td>No</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Pugliese &amp; Cagan, 2002</td>
<td>No</td>
<td>GA</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>Yes*</td>
<td>No</td>
</tr>
<tr>
<td>Chuzel et al., 2012</td>
<td>No</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Orsborn et al., 2008</td>
<td>No</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Agarwal &amp; Cagan, 1998</td>
<td>No</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Lin et al., 2009</td>
<td>Yes</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Johansson, 2011</td>
<td>Yes</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Sandberg &amp; Larsson, 2006</td>
<td>Yes</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>El-Sayed &amp; Chassapis, 2005</td>
<td>Yes</td>
<td>ND</td>
<td>ND</td>
<td>3</td>
<td>2</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Petersson et al., 2012</td>
<td>Yes</td>
<td>SA</td>
<td>WS</td>
<td>1</td>
<td>1</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Shea &amp; Cagan, 1999</td>
<td>No</td>
<td>SA</td>
<td>WS</td>
<td>4</td>
<td>5</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Lee &amp; Tang, 2009</td>
<td>No</td>
<td>GA</td>
<td>WS</td>
<td>1</td>
<td>3</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Ang et al., 2006</td>
<td>No</td>
<td>GA</td>
<td>–</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Chau et al., 2004</td>
<td>No</td>
<td>–</td>
<td>–</td>
<td>NA</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>EZCT Architecture &amp; Design Research et al., 2004</td>
<td>No</td>
<td>GA</td>
<td>WS</td>
<td>1</td>
<td>1</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Wenli, 2008</td>
<td>No</td>
<td>GA</td>
<td>–</td>
<td>0</td>
<td>NA</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Note: The number of constraints listed are those which are handled by the constraint-handling technique (CHT). Constraints handled by knowledge base systems were not taken into account. In the case of interactive genetic algorithms (GAs), the objectives handled by users were not included. NA, not applicable; ND, no data; SA, simulated annealing; WS, weighted sum.

*Chuzel et al. (2012) develops a similarity measure to test the performance of their interactive GA, but it is not used in the generative design system itself.
where \( p \) is the individual’s score, \( m \) is the number of constraints the individual satisfied up until the last constraint evaluated, \( a \) is the constraint violation of the last evaluated constraint, and \( c \) is the total number of constraints. Constraint violation \( a \) is normalized (e.g., \( a = \text{minimal allowed value/observed value} \)), which means that \( p \) also ranges from 0 to 1.

As a result of Eq. (1), an individual satisfying \( m \) constraints is certain to get a higher score than an individual satisfying \( m - k \) constraints, \( k \in [1, m] \). The score \( p \) is then used as fitness in the GA (see Section 3.2).

### 2.2.2. The BM technique

Schoenauer and Xanthakis (1993) describe another sequential approach: the BM technique. It is based on the BM paradigm (de Garis, 1990), in which several techniques have been implemented to increase the diversity of the population to avoid premature convergence around certain constraints. The algorithm is summarized below.

A randomly initialized population is optimized in regard to the first constraint. This continues until a certain percentage, the flip-threshold \( \phi \) of the population, satisfies the constraint. The population is then optimized in regard to the next constraint, until \( \phi \) percent satisfies the second constraint. Any individual not satisfying the prior constraints is given the score zero. This process continues until all constraints have been satisfied.

To maintain population diversity, a sharing scheme is used as described in Holland (1975) and Goldberg and Richardson (1987). This method reduces the fitness of individuals that are similar to each other to promote diversity. The user-defined parameter-sharing factor \( \sigma_{sh} \) is used to decide whether two individuals are similar; it is also used to calculate the sharing score \( s_{sh} \), which is used to penalize individuals that are similar (described below). The score \( p \) for each individual can be described as \( p = (M_t - C_i)/s_{sh} \), where \( C_i \) is the constraint violation and \( M_t \) is an arbitrarily large positive number equal to or greater than the largest constraint violation.

Furthermore, a restricted mating scheme as described by Deb and Goldberg (1989) is used which promotes mating of similar individuals to create fitter offspring. The parameter \( \sigma_{sh} \) is also used here to decide whether two individuals are similar.

This method thus requires the user to determine the flip-threshold \( \phi \) and the sharing factor \( \sigma_{sh} \). However, recommendations for tuning the last two are given in Schoenauer and Xanthakis (1993): “the order of magnitude of \( \sigma_{sh} \) can be approximated from below using large \( \phi \) and increasing \( \sigma_{sh} \) until the required percentage of feasible points cannot be reached anymore. Slightly decreasing \( \sigma_{sh} \) should then allow to find good values for both \( \sigma_{sh} \) and \( \phi \).”

### 3. THE STUDY

#### 3.1. Objectives of the study

The first objective is to compare the ability of the three CHTs to generate sufficient diversity among the proposed solutions. The second objective is to compare their relative convergence times. The third objective is to compare their relative convergence rates. In this study, the comparison is based on the table generation problem, presented next.

### 3.2. The table generation problem

The design problem is to generate Voronoi diagram based table structures based on that satisfy three production and structural constraints. A Voronoi diagram is a structure that is often found in light and strong structures in nature (Pearce, 1978; Beukers & van Hinte, 2005), such as the wing of a dragonfly or the structure of bone marrow. The manufacturing processes used are laser cutting and computer numerical control sheet-metal bending. The geometry of the bending machine limits the flange lengths of the cells to be manufactured to no shorter than 30 mm, which we call constraint \( l \), and the bending angles a minimum of 35°, which we call constraint \( a \). The structural requirements limit the maximum vertical displacement of any part of the table to 2.5 mm, which we call constraint \( f \).

The design problem is described in depth in Nordin et al. (2011). A GDS based on this design problem would allow the consumer to determine the contour of the tabletop (see Fig. 1), to choose the height of the table, and to select the table’s structure material. The GDS would then generate design proposals for the consumer to choose from. In this setup, the contour is chosen to be a square one. Note that prototypes have been built based on the computer-generated proposals and presented at several design fairs (see Fig. 2). This application can also be considered as a “real” design problem.

### 3.3. Implementation of the whole GDS

The table structure is represented as joined-beam elements, which are analyzed using the finite element method, using a finite element package called CALFEM developed at Lund University (Austrell et al., 2004). This package allows for defining a number of degrees of freedom for the cells, their positions and interconnections, as well as applicable loads and boundary conditions.

The GA used is the standard Matlab implementation. The scaling method used to assign probabilities for selection to the individuals is a simple ranking scheme where the individuals are ordered after their fitness; this approach avoids giving individuals with high fitness an unfair advantage in selection, which can result in premature convergence on local optima. The selection method chooses parents based on the individuals’ scaled fitness, in this setup Matlab’s built-in selection method stochastic uniform has been chosen. The stochastic uniform method represents the population as a line, with each individual representing a line segment whose length is proportional to the individual’s scaled fitness. The method then walks down the line in fixed-length steps, adding the individual whose line segment it lands onto the pool of parents. The top two individuals are guaranteed to survive to the next generation in order to not lose the best so-
olutions. The fraction of the children created by crossover, rather than mutation, is set to 0.8. The GA is run with a population of 50 individuals, during a maximum of 500 generations. The run is stopped when the maximum number of generations is reached or an individual satisfying all constraints has been found. Each original population was generated by randomly generating 70 Voronoi points for each of the individuals in the population.

Sharing score: Diversity measure of BM. The measure for diversity is based on the calculation of the sharing score for the BM method. The diversity of an individual in a population is calculated by comparing its genome, in this case the coordinates of its Voronoi points, to all the other genomes of the rest of the individuals in the population. This is achieved by the following pseudocode:

For each individual $i$ in the population:
   For each individual $j$ in the population:
      For each point $a$ in individual $i$’s genome:
         Find the point $b$ in individual $j$’s genome that has the smallest Euclidian distance $d_{a}$ to point $a$.
      The sum of all the distances $d_{i,j} = \sum_{a=1}^{70} d_{a}$ is individual $i$’s diversity to individual $j$.

3.4. Experimental setup and procedure

Because there are only three constraints, all six possible sequences are investigated for the Lexcoht and BM techniques. In this paper, each sequence is named after the order in which the constraints are evaluated ($laf$, $lfa$, $alf$, $afl$, $fla$, and $fal$, respectively). The sequencing has no effect on the UWS, because all constraints are evaluated simultaneously. The investigation of the three CHTs therefore amounts to 13 “treatments” to investigate. The parameters for the BM techniques were set according to the recommendations from Schoenauer and Xanthakis (1993), with $\varphi = 0.6$ and $\sigma_a = 0.05$ for all sequences. Lexcoht did not have any parameters requiring tuning.

The developed GDS is expected to be used repeatedly. Regarding convergence time, it is therefore appropriate to consider the frequency with which one wants the best technique to be faster than the others. In this test, it was decided that the best technique should generate faster solutions at least twice as often as the second best technique. In other words, 25% of the time the convergence time of the second-best technique should be below the median of the first technique (obviously, 50% of the time the convergence time of the first technique is below its median, i.e., twice as often as the second one). If the computing times of the techniques are normally distributed with the same standard deviation, then the mean is confounded with the median. In that case, the second-best mean should be at least 0.68 SD away from the best mean [$N(-0.68,0,1) = 0.25$]. The desired effect size is therefore $d = (d_{m}/\sigma) = 0.68$. In Motte et al. (2011) the distributions were positively skewed; the chosen effect size is therefore

![Fig. 1. (Color online) Three user-defined table contours.](image1)

![Fig. 2. (Color online) An image of the generated table (Nordin et al., 2011).](image2)
quite conservative. With 13 treatments to compare against each other using the Tukey test, and with \( d = 0.68 \), the minimum number of runs for each treatment is 48 (Nicewander & Price, 1997). To control for nonconvergence (estimated originally at 10%), the chosen number of runs was set at 60. Finally, a repeated-measure design was used, allowing for studying whether the original populations had an effect on diversity for the different techniques.

The performed simulation presented low convergence rates for the BM techniques (40%–66%). This was unexpected, because the BM technique had always converged in the previous study (Motte et al., 2011) and had high convergence rates in an unpublished prestudy. As the previous simulation was based on a repeated-measures design, it was not possible to exploit it for convergence rate and convergence time, because of the large number of missing data. Therefore, the original simulation based on repeated-measures design was used only for investigating diversity, and a new simulation was performed under the same conditions, with independent samples for the convergence time and the convergence rate. The number of runs in each treatment was set at 150 in order to ensure a sufficient power.

The convergence times of the treatments were obtained using the CPU time of one core of an Intel Xeon E5620 2.40 GHz processor. The total simulation time amounted to 22 days, 13 h (because three CPU cores were used simultaneously, the simulations took 254 h).

3.5. Results

3.5.1. Diversity

The diversity within a population (or “intrapopulation diversity”) is calculated by the sum of all individuals’ diversities.

The intrapopulation diversity among all treatments was 1.24 × 10^{-2} (SD = 4.48 × 10^{-2}). The intrapopulation diversity for each method and technique is reported in Table 2.

The alternatives offered by the methods did not present an appreciable variety until the diversity reached a value of 8 × 10^{-2} (e.g., Fig. 3). The alternatives with a diversity between 8 × 10^{-2} and 9 × 10^{-2} are in a gray area (see Fig. 4), while the alternatives above 9 × 10^{-2} are clearly dissimilar (Fig. 5). Unfortunately, only four pairs of different individuals for all the methods and sequences had a diversity value between 8 × 10^{-2} and 9 × 10^{-2} (two in population 27 of the BM alf sequence, two in population 30 of the same sequence, two in population 27 of the BM alf sequence, and two in population 27 of the UWS method). The different variants with a diversity above 9 × 10^{-2} were also few. For Lexcoht, one population in each of the alf and alf sequences presented 2 alternatives that could be judged as diverse. For the BM method, one population of the fal sequence presented 2 alternatives, 3 populations of the fla sequence presented 2 alternatives, 2 populations of the alf sequence presented 2 and 6 alternatives, and 2 populations of the alf sequence presented 2 and 7 alternatives. The UWS method did not have any variant above 9 × 10^{-2}. These outcomes are summarized Table 3. The probability of getting dissimilar individuals in one population is therefore not only very low; the number of dissimilar alternatives per population is also generally low: most of the time, the user is not expected to obtain more than 2 alternatives. Moreover, most of the dissimilar alternatives originate from the same original populations (populations 27, 30, 43; see Table 3). The diversity seems to depend more on the good characteristic of the original population than on the method itself. The BM method did get most of the dissimilar groups, but not as much as was expected. The sharing scheme of the BM method probably does not create diversity among individuals but seems to maintain it if it is present in the original population.

The intrapopulation diversity did not provide satisfying results. However, the diversity between populations (or “interpopulation diversity”) was much larger: 1.63 × 10^{-1} (SD = 6.85 × 10^{-2}). The minimal diversity value was 0.802 × 10^{-2}. These outcomes are summarized Table 3. The probability of getting dissimilar individuals in one population is therefore not only very low; the number of dissimilar alternatives per population is also generally low: most of the time, the user is not expected to obtain more than 2 alternatives. Moreover, most of the dissimilar alternatives originate from the same original populations (populations 27, 30, 43; see Table 3). The diversity seems to depend more on the good characteristic of the original population than on the method itself. The BM method did get most of the dissimilar groups, but not as much as was expected. The sharing scheme of the BM method probably does not create diversity among individuals but seems to maintain it if it is present in the original population.

The intrapopulation diversity did not provide satisfying results. However, the diversity between populations (or “interpopulation diversity”) was much larger: 1.63 × 10^{-1} (SD = 6.85 × 10^{-2}). The minimal diversity value was 0.802 × 10^{-2}. These outcomes are summarized Table 3. The probability of getting dissimilar individuals in one population is therefore not only very low; the number of dissimilar alternatives per population is also generally low: most of the time, the user is not expected to obtain more than 2 alternatives. Moreover, most of the dissimilar alternatives originate from the same original populations (populations 27, 30, 43; see Table 3). The diversity seems to depend more on the good characteristic of the original population than on the method itself. The BM method did get most of the dissimilar groups, but not as much as was expected. The sharing scheme of the BM method probably does not create diversity among individuals but seems to maintain it if it is present in the original population.

![Fig. 3.](image-url) (Color online) Two table support variants of population 23 for the Lexcoht alf sequence, with diversity 2.96 × 10^{-2}.
For Lexcoht, there were only two populations of the alf sequence that contained individuals with a diversity below $9 \times 10^{-2}$ and two populations in the fla sequence that contained individuals with a diversity below $9 \times 10^{-2}$. All the remaining individuals were quite dissimilar. Running several simulations with different original populations therefore ensures diversity. Figure 6 illustrates the large difference between intra- and interpopulation diversities. It is very important that there is no need, at least in this particular example, to even measure diversity, because virtually all interpopulation individuals are dissimilar. Running several simulations with different original populations therefore ensures diversity.

### 3.5.2. Convergence time

The smallest convergence rate observed in the second simulation was 28%, which amounted to 42 successful runs. This is less than the required minimum number of runs for each treatment (48; see Section 3.4), which implies a loss of power but also a decrease in type I error, which means that the multiple-comparison test is rather conservative. It was therefore decided to go on with the obtained data. The exploratory data analysis revealed that the distributions of the convergence time for each combination were markedly positively skewed, as is illustrated in Figure 7. The standard deviations were found proportional to the means; thus a logarithmic transformation was applied to the data (Howell, 2007, pp. 319–321). The log-transformed populations were mostly normally distributed; the Jarque–Bera test for normality (Jarque & Bera, 1987) failed to show a significant deviation from a normal distribution for most of the combinations (five treatments had $p_{JB} < 0.01$). With the largest variance ratio being 1:4, the heteroscedasticity was within the limit on heterogeneity of variance (i.e., less than or equal to a factor of 4) for which the analysis of variance is still robust (Wilcox, 1987; Howell, 2007, p. 317).

A one-way analysis of variance revealed that there were significant differences among the means of the 13 treatments [$F (12, 1286) = 17.98, p < 0.001$]. A Tukey test at $\alpha = 0.05$ was subsequently performed upon the 13 treatments. Figure 8 presents the log-transformed means for each method and sequence.

The Lexcoht method with the alf sequence was significantly better than the UWS method. It was not significantly better than the best BM method result, with the laf sequence, which itself was not significantly better than the UWS method.

The Lexcoht method with the alf sequence was significantly better than the worst Lexcoht sequence fla. The BM method with the best sequence (laf) was also significantly better than the worst BM sequence (fal). This confirms that the choice of the right sequence is important.
3.5.3. Convergence rate

The convergence rates of the different treatments are presented Table 4. A chi-square test for proportions produced $\chi^2 (12) = 464.02$, which is significant at $p < 0.001$. A pairwise comparison following the Tukey–Kramer procedure for proportions (Hochberg & Tamhane, 1987, p. 275) was subsequently performed. The convergence rates of Lexcoht with sequences alf and fal were significantly larger than the other treatments. The complete results are presented Table 4 and Figure 9. Almost all Lexcoht treatments have a significantly higher convergence rate than the BM treatments. The BM treatments with a computing time similar to the best Lexcoht results as well as UWS are therefore performing significantly worse in terms of convergence rate.

3.6. Discussion

In this paper, a number of different techniques for handling technical constraints have been evaluated in terms of convergence time, convergence rate, and the diversity of the generated solutions using a real product design problem. The aim has been to investigate generative product design systems used in the context of mass customization, which are required to quickly and reliably generate diverse solutions without requiring modification or tuning during use. When such systems are designed to allow for the customization of product form, they must be able to handle production and engineering constraints that can be time-consuming to evaluate and difficult to fulfill. These issues are related to how the constraints are handled in the GDS, and because of this, two promising SCHTs and the often used weighted sum technique have been investigated.

Concerning diversity, the investigation revealed that the intrapopulation diversity was not high enough to be used for presenting several alternatives to the user. In contrast, the interpopulation diversity was always high. Diversity is thus gained at the cost of more runs, but in that case there is no need to check for diversity (as all interpopulation solutions are sufficiently different). This result is also important because, if generalized, it would imply that it is not even necessary to define a diversity measure, whatever the type of complex form. It could also be shown that the specific mating scheme that is built in in the BM method did not ensure enough intrapopulation diversity.

The treatments that were most frequently the fastest were, for Lexcoht, the alf sequence and, for BM, the laf sequence. Lexcoht with the best sequence outperformed UWS by a factor of two. Although this confirms that the SCHTs are promising for the kind of problem presented here, it does not completely rule out the UWS, which performed well for the investigated design problem and requires no tuning or
sequence selection. In the case of SCHTs the different sequences need to be tested first, but the gain is substantial if the GDS is used frequently. It is important to recall that the convergence time distributions are highly positively skewed, so a good CHT not only allows for a quicker convergence on average but also avoids very lengthy runs. The parameters for the BM techniques that were set according to the recommendations from Schoenauer and Xanthakis (1993) yielded good results.

**Table 4.** Number of successful runs (out of 150), rate of convergence, and 95% Clopper–Pearson CI

<table>
<thead>
<tr>
<th>Treatment</th>
<th>N</th>
<th>Converg. Rate</th>
<th>95% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Lower Bound</td>
</tr>
<tr>
<td>L-aff</td>
<td>150</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>L-alf</td>
<td>123</td>
<td>0.82</td>
<td>0.75</td>
</tr>
<tr>
<td>L-fal</td>
<td>149</td>
<td>0.99</td>
<td>0.96</td>
</tr>
<tr>
<td>L-fla</td>
<td>129</td>
<td>0.86</td>
<td>0.79</td>
</tr>
<tr>
<td>L-laf</td>
<td>103</td>
<td>0.69</td>
<td>0.61</td>
</tr>
<tr>
<td>L-lfa</td>
<td>111</td>
<td>0.74</td>
<td>0.66</td>
</tr>
<tr>
<td>BM-afl</td>
<td>88</td>
<td>0.59</td>
<td>0.50</td>
</tr>
<tr>
<td>BM-alf</td>
<td>42</td>
<td>0.28</td>
<td>0.21</td>
</tr>
<tr>
<td>BM-fal</td>
<td>78</td>
<td>0.52</td>
<td>0.44</td>
</tr>
<tr>
<td>BM-fla</td>
<td>57</td>
<td>0.38</td>
<td>0.30</td>
</tr>
<tr>
<td>BM-laf</td>
<td>73</td>
<td>0.49</td>
<td>0.40</td>
</tr>
<tr>
<td>BM-lfa</td>
<td>61</td>
<td>0.41</td>
<td>0.33</td>
</tr>
<tr>
<td>UWS</td>
<td>132</td>
<td>0.88</td>
<td>0.82</td>
</tr>
</tbody>
</table>

**Note:** CI, confidence interval; L, Lexicoht; lfa, lfa, alf, afl, fla, fal, the order in which the constraints are evaluated; BM, behavioral memory; UWS, unweighted sum scheme.
The treatments that had the best convergence rates were Lexcoht with the *afl* and *fal* sequence. The convergence rates were poor for the BM method in this setup but were excellent in Motte et al. (2011) at 100%. Note that convergence rate and time are not correlated (compare Figs. 8 and 9). Therefore, the choice of an adequate sequence must take into account convergence rate and time, as well as computing resources (the calculations can be made in parallel with multicore or cluster setups).

One should nevertheless remember that in order to use SCHTs, a good constraint sequence has to be found. This is a time-consuming task that requires a careful experimental design. As mentioned earlier, the presented comparison took around 10 days. This comparison is interesting only if the GDS is to be used frequently; otherwise the weighted sum is the best default technique.

4. CONCLUSION

The perspective of enabling consumers to use potentially complex forms, coupled to functional, engineering, and production constraints, is appealing. Several obstacles to such an approach have been dealt with in this article. Although much research has been done in the area of GDSSs, few take into account constraints that are time-consuming to evaluate and difficult to fulfill, such as structural stability and manufacturability, a necessity for many products based on mass production systems. The ones that achieve that are focused on finding the best solution in regard to the objectives, rather than user preferences, and are not targeted at consumers. In order to give the consumer meaningful choice among the generated solutions, they must all fulfill the constraints and should be generated quickly and reliably to avoid frustration. These issues are all related to how the constraints are handled, and our aim has been to investigate how CHTs in a GDS intended to be used in the context of mass customization of product form should handle difficult constraints. In terms of CHTs, virtually all GDS applications dealing with more than one constraint or objective are applying the weighted sum technique. We have therefore evaluated three promising CHTs, two SCHTs and the UWS. The results show that the Lexcoht SCHT outperformed the UWS in terms of both convergence time and rate and that diversity can be guaranteed by launching many design generations in parallel. Enabling the efficient handling of such types of constraints is a step toward showing that form mass customization is technically possible, and beyond that a step toward total mass customization. Algorithmic form generation, coupled to an interactive compelling online experience as well as purchase, logistics, and production back-end, allows for various entrepreneurial opportunities for companies and consumers alike, as well as for the designers.

The scope of application of new digital means of interaction, designing, and fabrication is fully scalable and in that sense constitutes a unique enabler that, if consistently implemented, could potentially cut across a very large number of industries, ranging from small manufacturers to large producers of consumer products. The research presented in this paper addresses primarily the use of GDSSs in the mass customization context, but some aspects of it, especially the diversity issue, should also be useful in a GDS intended for professional industrial designers.

ACKNOWLEDGMENTS

This work was supported by the Swedish Governmental Agency for Innovation Systems, VINNOVA, through the Production Strategies and Models for Product Development program (Project Renaissance 1.5, 2009-04057).

REFERENCES


---

**Fig. 9.** A representation of the homogeneous populations (treatments ranked) for \( p < 0.05 \).
Axel Nordin is a PhD student at the Division of Machine Design, Lund University. He received his MS in industrial design engineering from Lund University and has participated in two government-funded research projects during the last 2 years. His work has been concerned mainly with studying aspects of integrating complex morphologies into the design of bespoke products, such as computational, manufacturing, structural, and usability challenges.

Damien Motte currently holds a postdoctoral position at the Division of Machine Design, Lund University. He received a PhD from the same division, a research master’s from the Industrial Engineering Laboratory at the École Centrale Paris, and an MS in industrial engineering from the École des Mines d’Albi, France. Dr. Motte is currently working on alternative engineering design and product development methodologies.

Andreas Hopf is a Design Consultant and Senior Lecturer in the School of Industrial Design at Lund University. He received a BA in industrial design from Art Center College of Design (Europe), Switzerland. At Lund University he runs seminars in industrial design and computer-aided design in two-dimensional/three-dimensional and rapid prototyping, supervises diploma projects, and participates in the development of new bachelor’s and master’s programs.

Robert Bjärnemo is a Professor of machine design at Lund University. He obtained his MS and PhD from the same university. Dr. Bjärnemo’s research interests are in engineering design methodology and product development methodology, especially integrated product development, as well as predictive design analysis.

Claus-Christian Eckhardt has been a Professor of industrial design at Lund University since 2001. He worked as an Interior Designer for Silvestrin Design and was in charge of designing consumer electronics and communication products at Blaupunkt, where he was also responsible for the design of the Bosch Telecom product series and Bosch mobile phones. Mr. Eckhardt later became Chief Designer and Head of Global Product Design with Bosch and then Head of Design at Tenovis and Avaya. He has also worked as a Design Consultant since 2000. Claus-Christian is the recipient of several national and international awards and honors. His research areas are in design management and design implementation.