Evolutionary Systems for Industrial Design

Ewa Grabska

Institute of Computer Science, Jagiellonian University
Nawojki 11, 30-072 Cracow
uigrabsk@cyf-kr.edu.pl

Abstract

The aim of this paper is to propose a new approach to creative design. The process of designing simulates the evolutionary process at the genotype level. Structures of designed objects are represented by means of graphs as genotypes. This representation of genotypes forces new interesting extensions of genetic operators. The modified crossover operator is the major computational engine of genetic algorithms.
Possibilities of application of evolutionary methods and graph transformations in industrial design are presented. Our approach is illustrated by examples from the case study of chair design with the use of an evolutionary graph-based design-system.

Keywords: creative design, evolutionary techniques, genotype, genetic operators, and hierarchical graph.

1 Introduction

Designing is one of the basic activities that humans carry out. Therefore the development of software assisting the designer attracts many researchers. The kind of novel software provides the potential for the production of novel design objects. The notion of novelty in designing is associated with the notion of creativity.
The paper deals with *creative* designing, i.e., the kind of designing which characterizes the variability of design structure. By design *structure* we understand the basic components and their relationships of a design object (Gero, 1999).
Recently evolutionary techniques like genetic algorithms have drawn much attention as applicable in creative design (Bentley, 1999). Using the techniques of simulated evolution, a computer is able to create novel design objects (Holland, 1975; Michalewicz, 1996). However, there exists a serious disadvantage of application of evolutionary methods in designing. Genetic algorithms are usually associated with binary encoding, i.e., strings lacking in the capability to represent complex design objects.
In this paper we propose a new approach to design, which removes the drawback. Structures of designed objects are represented by means of graphs as genotypes. This representation of genotypes forces new interesting extensions of genetic operators. Moreover, the graph genotypes can be simply transformed into phenotype (patterns, images, 3D-structures). Applying a genetic algorithm makes the computer responsible

for all operations on the underlying representation (i.e. producing graphs, transforming them into phenotypes and displaying them), while the designer can concentrate on evaluating the obtained results. A user evaluating designs generated by the system can replace the fitness function in this application. It is however possible to define a number of different fitness functions depending on the type of structures being design.
The presented new approach is illustrated by examples from the case study of chair design with the use of an evolutionary graph-based design system.

2 Evolutionary Algorithms in Designing

Evolutionary methods have been used successfully as an analog of designing methodologies. Fundamental to this analogy are the following aspects:

- the process of designing simulates the evolutionary process at the genotype level,
- the design description corresponds to the phenotype, and
- design evaluation replaces the fitness function.

In genetic algorithms all designs are represented in two forms: in an encoded form of genotypes and in the decoded form of phenotypes (Gero, 1999).
Usually, binary strings represent genotypes. As it has been considered binary strings lack the capability to represent complex design objects.
In this paper we describe a new approach that combine the graph-based representation and evolutionary methods in creative design.

2.1 Hierarchical Graphs

There are many methods used to represent design objects in computer aided design systems. One of them is a graph-based representation. Since several years we have developed graph-based tools that should assist the designer in taking innovative decisions (Grabska, 1994; Grabska & Borkowski, 1996). At present hierarchical graphs are the key construction of the tools (Grabska & Palacz, 2000).
Designers need the same view of the artifact that is being designed but at different levels of abstraction. For instance, given a component layout, a designer might want a detailed view dawn to the level of sub-components together with relation between them. On the other hand, the designer, walking through some use case scenarios, will want a much-elided view of these same components. But, he/she is still working at the same artifact's model, irrespective of a level of abstraction.
In other words, hierarchical graphs (HGs) should be seen as a hierarchical tree of the design object together with its graph structure describing relations between object's components. Fig. 1 shows the hierarchical graph describing a chair and gathering information both on its hierarchy levels and its structure.
Hierarchical graphs (HGs) consist of nodes and edges. Nodes and edges represent components of the design object and relations between them, respectively. Nodes of

HGs can contain internal nodes. These nodes called children, can in turn contain other internal nodes and can be connected to other nodes with the exception of its ancestor. Nodes and edges of HGs can be labeled and attributed. Fig. 1 shows an example of a hierarchical graph with labeled nodes. For instance, *chair* is a label of the node, which represents the whole chair (level 0), whereas all nodes at level 2 are labeled with the same label *leg*.

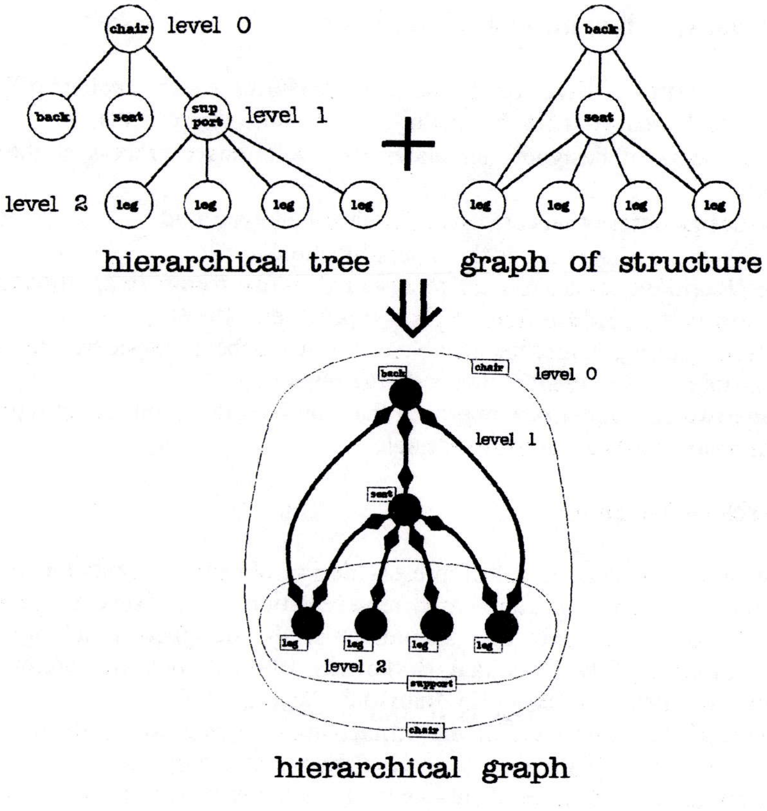

Fig.1: The hierarchical graph of a chair.

One of the most essential concepts for a HG is its sub-graph. We define the sub-graph in a specific way. Each node of a sub-graph of HG belongs to the sub-graph together with its descendants and all edges of HG connecting nodes of the sub-graph are also elements of this sub-graph.
Hierarchical graphs describe only structures of design objects. Their interpretation allows one to visualize the objects. The graph interpretation is defined as follows.

Definition 1.
Let G be a HG. An interpretation of G is a pair (fv, fe) of functions, where

- *fv assigns components of design objects to the nodes of G, and*
- *fe establishes a correspondence between relations determining the object structure and edges of G.*

In our approach hierarchical graphs allows one to represent design objects in an encoded form of genotypes. Genotypes are modified by means of genetic operators. In designing the modifications consist in changing structures of design objects. The way genetic operators are defined strongly depends on the type of genotypes used in a given application. In our approach to creative design we need completely new definitions of crossover and mutation operators.

2.2 Genetic Operators

The crossover operator is called the major computational engine of genetic algorithms (Gero, 1999). This operator for binary strings divides parental genotypes at a given position and exchanges corresponding sub-strings.
Applying the crossover operator to the nonstandard pair of genotypes in the form of hierarchical graphs requires establishing firstly, their sub-graphs that would be exchanged and secondly, rules of embedding each of these sub-graphs in another parental genotype.
In a formal way:

Definition 2.
Let G and G' be a pair of HGs.
A crossover operator (CROSS) defined on G and G' is a 4-tuple (g, g', T, T') defined as follows:

- *g and g' are HGs such that there exists a bijection between their nodes at the highest level (pairs nodes determined by the bijection has identical labels),*
- *there exist a sub-graph of G and another sub-graph of G' isomorphic with g and g', respectively,*
- *T and T' are rules of embedding for the sub-graph isomorphic with g' in G and the sub-graph isomorphic with g in G', respectively.*

Let us consider two HGs describing chairs together with its interpretations presented in Fig. 2. Then we consider the sub-graphs of the HGs determined by label *back*. The sub-graph *back* in Fig. 2a contains 4 inner nodes and 4 edges between them and its equivalent in Fig. 2b has 5 inner nodes connected by 4 edges. We exchange these sub-graphs and create new two HGs that are presented together with their interpretations in Fig.4 a and in Fig. 4b. In case of interpretations shown in Fig. 4 this operator causes changes representation of backs.

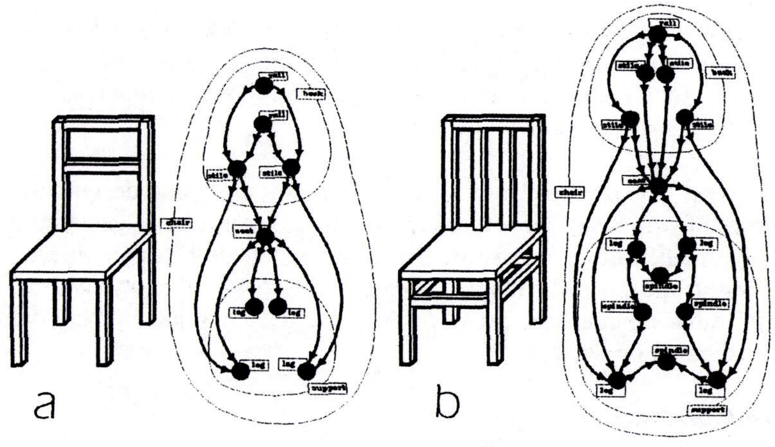

Fig. 2: Two HGs and their interpretations.

The second genetic operator for a binary string allows flipping bit at a given location of the string. The two following types of mutations can be applied to HGs: structural mutation which allows to modify graph structures (deleting and adding nodes), and attribute mutation for modifying values of attributes.

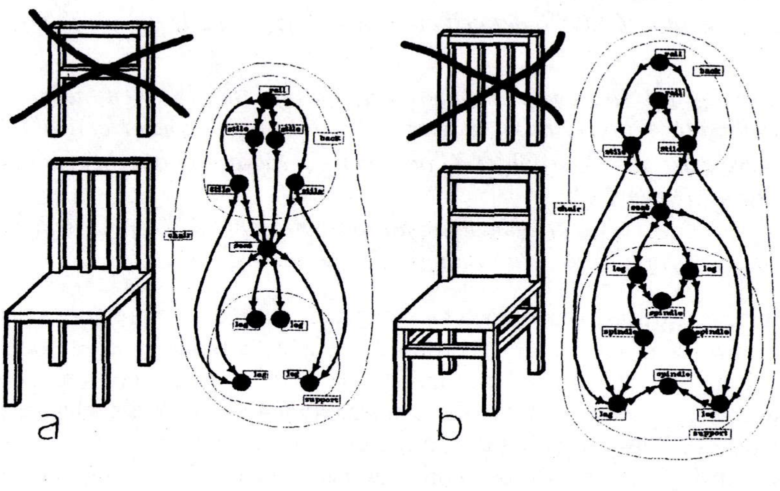

Fig. 3: The result of applying the crossover operator to the HGs shown in Fig. 2.

3 Implementation

Genetic algorithms were implemented in the code DESIGN&EVOLUTION developed at the Jagiellonian University last year. The program is written in Java language (JDK 1.2 of Sun Microsystems corporation). Data structures are based on package iiuj.zhzi.graph. This package contains a full class framework for representing hierarchical graphs. The representation contains graph nodes and edges, and attributes related to graph interpretations. Moreover, basic graph operators are provided, such as adding and deleting nodes or edges. All genetic operators were implemented with the use of hierarchical graph structures. User interface and graphical part are based on Java 3D package (Java 3D 1.2 of Sun Microsystems corporation) in OpenGL version.
Industrial design was the first domain of application of the system DESIGN&EVOLUTION.
The program "CHAIRS" is a tool allowing the users to generate chairs with the use of genetic algorithms (Nikodem, 2001). The design system starts with an initial population of chairs. Each element of the population is represented in two forms: in encoded form of genotypes and in the decoded form of phenotypes. Fig. 4a presents two chairs (phenotypes) of the initial population. Chair genotypes are represented by a hierarchical graph.

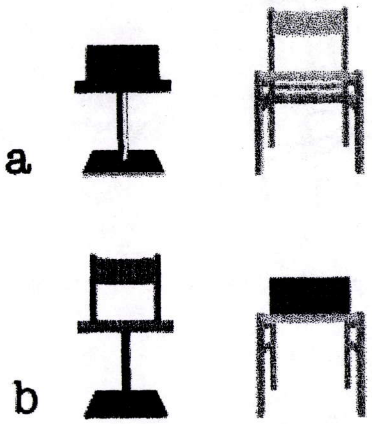

Fig.4: Applying the crossover and mutation operators: a) two chairs of the initial population; b) the result of applying the operators to genotypes of the two chairs.

Nodes of HGs represent chair components that are made up of transformed geometrical primitives. Attributes of nodes can describe geometry of components and materials from which the components are made.
A chair genotype contains three nodes labelled by *back*, *seat* and *support*. Each of these nodes can include any number of children (or descendants). For instance, the node *support* can contain four children representing chair's legs. Graph edges represent adjacency relation between chair components.

The important task an evolutionary techniques is an evaluation method. The implemented genetic algorithm of the system CHAIRS contains the fitness function which evaluates aesthetic aspects of chairs, for instance fitting different materials and/or different shapes of components. A fitness aesthetic value is assigned to each individual (chair) of populations. Fitness values associated with individuals are essential in the selection process of parents for crossover operator.
In the program CHAIR the result of applying the crossover operator can be mutated with the probability defined by the user. Both crossover and mutation operators allow one to generate new design objects. Fig. 4b presents the result of applying the crossover operator to genotypes of the two chairs shown in Fig. 4a.
Finally, we present four chairs generated by the aid of the program CHAIRS (30 iterated steps).

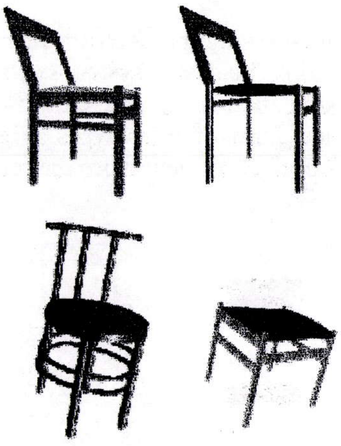

Fig. 5 Chairs generated by the aid of the program CHAIRS.

4 Conclusions

The aim of this paper is to present a new approach to creative design. The special type of knowledge representation – the hierarchical graphs – is useful for solving design problems in the industrial design. The genetic search that we apply does not impose the restrictions typical for conventional designing.
The method of design representation and genetic operators presented here can be used independently from the application domain.
The layout optimisation of trusses is concerned as the next domain of application of genetic algorithms. We propose to start with generating an initial layout using appropriate graph transformations and then to perform the following three iterative tasks: optimization of geometry and members, genetic search, and picking up the best layout.

References

John Gero (1999). Recent Development in Evolutionary Systems for Design, In: A.Borkowski (eds), *Artificial Intelligence in Structural Engineering*, WNT.

P. Bentley (1999). Evolutionary design by computer, Morgan Kaufman Publisher.

J.H. Holland (1975). Adaptation in Natural and Artificial Systems, Ann Arbour Univ. of Michigan Press.

Z. Michalewicz (1996). Genetic Algorithm + Data Structures = Evolutionary Programs, New York, Springer-Verlag.

Ewa Grabska (1994). Graphs and designing, In: H. J. Schneider and H. Ehrig (eds.), *Graph Transformations in Computer Science*. Lecture Notes in Computer Science, 776, Springer-Verlag.

Ewa Grabska and Adam Borkowski (1996). Assisting Creativity by Composite Representation, in: J. S. Gero and F. Sudweeks (eds.), *Artificial Intelligence in Design'96*, Kluwer Academic Publishers.

Ewa Grabska and Wojciech Palacz (2000). Hierarchical graphs in creative design. Machine GRAPHICS & VISION, 9(1/2), 2000.

Piotr Nikodem (2001). Applying Genetic Algorithm in Industrial Designing. In: M. Kurzyński and E. Puchała (eds.), Computer Recognition Systems, Wroclaw.