Style-directed Evolutionary Design

Dr B. Strug, DSc, PhD, Dr G. Ślusarczyk, DSc, PhD,
Prof. E. Grabska, Dsc, PhD.
Department of Physics, Astronomy and Applied Computer Science,
Jagiellonian University, Kraków, Poland
zpgk.fais.uj.edu.pl
e-mail: barbara.strug@uj.edu.pl

Abstract

This paper deals with an evolutionary approach to design. The aim of this paper is to present a new approach to style-oriented evolutionary design. Designs are represented by means of graphs, in which graph nodes represent components of the design while graph edges represent relations between the components. Main graph evolutionary operators, i.e., crossover and mutation are modified in this paper. The operators are based on previous graph evolutionary operators but are extended to incorporate style-preserving features. The approach is illustrated by examples of designing gardens in the Japanese style.

1. Introduction

Recently, an iterative methodology based on a cyclic process of prototyping, analyzing, testing and refining a solution or process is often used in the design process. It starts with a preliminary design, which is then analyzed to assess its feasibility and decide on changes and refinements. This process of evaluation and optimization is repeated until the quality and functionality of a design is satisfactory. In computer science such a process can be modelled by a search process, where all possible designs form a search space, thus it is possible to use search techniques such as evolutionary ones. As evolutionary search consists in evaluating and refining possible solutions, it is highly analogous to a human design iterative process of analysis, testing and optimization [1]. Similarly to the refinement step in human design, in evolutionary search designs to be modified are determined according to their evaluation (fitness). The refinement step is often performed not on actual solutions (phenotypes) but on their coded equivalents (genotypes). Yet, in human design the process is usually directed not only by the desire to obtain an optimal artefact but also such a one that meets certain requirements. Design requirements are often related to styles [2]. The aim of this paper is to present a new approach to style-directed evolutionary design.

The most popular approach in evolutionary computing is based on representing solutions as binary strings. In design problems genotypes in such a form are often insufficient, so we propose to use a graph-based representation of genotypes as it enables us not only to express geometrical properties of an object but also its attributes (like colour, material etc.) and most importantly relations between object components. Using graphs as a representation of design artefacts in an evolutionary search process requires the adaptation of traditional evolutionary operators like cross-over and mutation as well as defining an appropriate fitness function. Moreover

these operations have to take into account the desired\required style of the artefact being designed. As the graphs selected to be transformed by these operators during the evolution and their structures are not known a priori, the operators must be defined in a way which allows for a dynamic computation of resulting graphs. In our approach a cross-over operation can only exchange subgraphs, while mutation affects local and global attributes as well as the graph structure (by adding or deleting subgraphs). Moreover a mutation operator is designed in such a way that it only allows changes within a range suitable for a given style of designs. A fitness function is specified in such a way that it prefers solutions adhering to the rules of a given style. The approach is illustrated by examples of designing gardens in different styles.

2. Evolutionary Design

As mentioned in the introduction the design process can be modelled by a search process. As evolutionary search consists in evaluating and refining possible solutions it can be seen as analogous to a human design iterative process of analysis, testing and optimization [1,3,5,7]. That is the refinement step in human design, which is based on earlier analysis and testing, can be modelled in evolutionary search by transforming designs according to their evaluation calculated by the so called fitness function. In many types of evolutionary search the refinement step is often performed not on actual solutions (called *phenotypes*, which in this paper are designs) but on their coded equivalents (called *genotypes*).

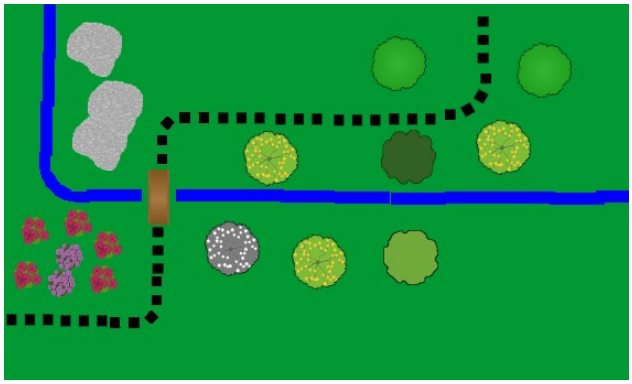

Fig.1 An example of a garden in Japanese style

In many optimization problems being solved with an evolutionary approach a binary coding is used, i.e. the actual solutions are represented by binary strings [1,3,5,7]. Yet in design problems they are usually insufficient as representing not only geometrical properties of an object, but also its other attributes (like colour, material etc.) and relations between object components is hard to achieve in a string form.

The representation of objects used in CAD problems like boundary representations, sweep-volume representations, surface representations or CSG (constructive solid geometry) [6] allow only for the "coding" of geometry of an object being designed and do not take into account the inter-related structure of many design objects, i.e. the fact that parts (or components) of an object can be related to other parts in different ways. Such a structure is usually represented as a graph [4,8].

3. Graph-based representation of designs

In CAD problems the knowledge about a design object can be expressed in a formal machine readable format. Such a format has to be on one hand well structured and machine readible, but at the same time rich enough to capture geometrical, numerical and relational properties of an object.

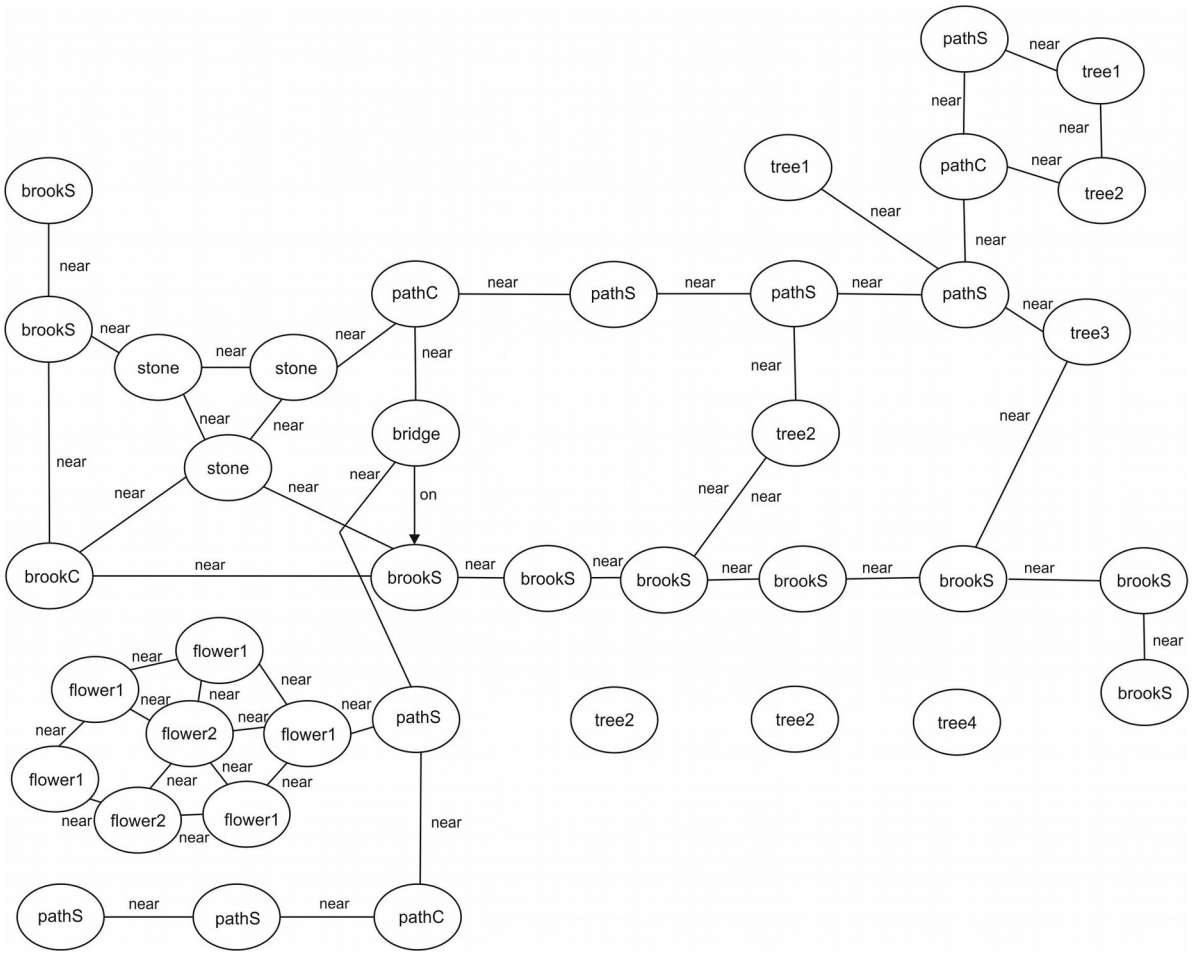

Fig.2 A graph representation of a garden from Fig.1

In this paper directed labelled graphs are used to represent design drawings. Graph nodes represent components (parts) of the object being designed, while edges express relations between them. Nodes are labelled by names of components (or types of components) and edges are labelled by names of relations between them.

In case of garden design the nodes are labelled by the names of the elements of the

garden and the relations can be both directed and undirected. The edges representing the adjacency relation are undirected and labelled *near*, while edges representing the being above relation are directed and labeled *on*.

In Fig. 2 a graph representing the garden design presented in Fig.1 is shown. Nodes labelled *tree1*, *tree2*, *tree3* and *tree4* represent different species of trees, nodes labelled *flower1* and *flower2* - flower clumps composed of two kinds of flowers. Nodes labelled *stone* represent a group of three stones and nodes labeled *pathS, pathC, brookS, brookC*, and *bridge* represent straight and winding fragments of the path, straight and winding fragments of the brook, and the footbridge. Edges labelled *near* represent the adjacency relation between garden components, and an edge labelled *on* represents the fact the footbridge is above the brook.

Style representation

As the design is represented by a labelled graph a similar representation is needed to encode the characteristic features of a style. - It can be observed that a style can be defined as a set of elements that have to be present in the design to assure this style. Moreover in many cases these elements must be arranged in some predefined way. As objects are represented by graphs also style requirements should be represented by (sub)graphs [9].

For example if the designer wants a garden to be in the Japanese style there is a number of requirements to be fulfilled: stones, water, trees and flowers should be present, paths should be winding and at least one footbridge should be present and placed over the water.

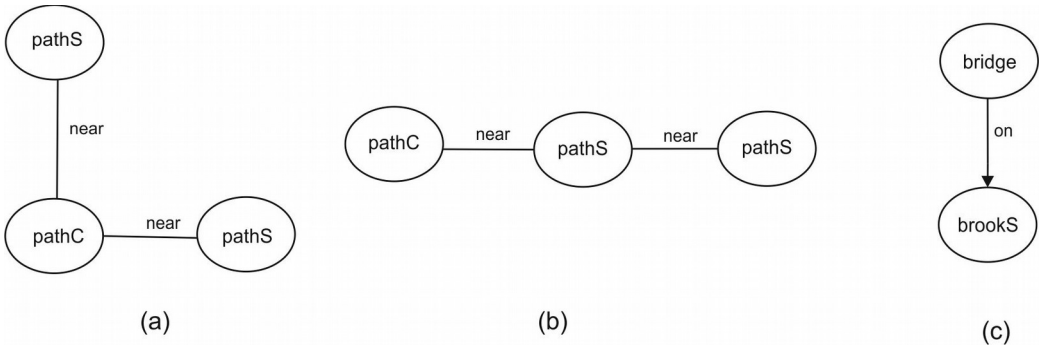

Fig.3 Examples of graphs representing style patterns

In Fig. 3 three examples of graphs representing different requirements for a Japanese-style graden are presented. In Fig.3a and b winding paths are represented while Fig. 3c depicts a graph representing the existence of a footbridge over a brook [9].

4. Evolutionary Operators on Graphs

As it has been mentioned, in this paper designed objects are represented by labelled and directed graphs. To use such a representation in an evolutionary design system a number of elements of this system must be defined. The genetic operators (usually a crossover and a mutation) constitute the fundamental element of an evolutionary algorithm. As in this paper a non-standard representation is used, also new genetic operators have to be defined.

The graph based equivalent of a standard crossover operator requires establishing subgraphs that would be exchanged during the process of evolution. When a crossover is performed on two selected graphs, G and H, the subgraphs g and h, respectively, are selected in these graphs. Then each subgraph is removed from a graph and inserted into the second one. As a result two new graphs are generated. However there may exist edges connecting nodes belonging to a chosen subgraph with nodes which do not belong to it. Such edges are called embedding of a subgraph. So removing a subgraph from a graph and placing it in another one requires a method allowing for proper re-connection of these edges. The underlying idea is that all edges should be re-connected to nodes similar to those they were connected to in the graph from which they were removed. There is probably more than one possibility of defining similarity of nodes.

In this paper a similarity-like relation is used. The definition of this relation is based upon the assumption that graphs selected for crossover code designs consisting of parts with similar or even identical functions (even if these parts have a different internal structure, material or/and geometrical properties). Thus we can define the similarity on the basis of the node and edge labels.

It is important to notice however that the graphs to be crossed over and their respective subgraphs are selected during the execution of the evolutionary algorithms so the embedding transformations cannot be defined a priori (as it is in graph grammars. The idea behind the algorithm that generates automatically such an embedding transformation is to preserve the relations between the nodes as much as possible, i.e. to connect each edge removed from one graph to a node in the second graphs that represents the same or similar object (i.e. has the same label).

In addition to dealing with the graph embedding problem in case of using the evolutionary process to generate designs adhering to a particular style an additional step must be performed. During the process of selecting subgraphs in graphs to be crossed over it is possible that the patterns representing style requirements will be broken. As the result new graphs generated by the genetic operator could not represent designs in a required style. To prevent such a situation we introduce the notion of an unbreakable subgraph. An unbreakable subgraph is a subgraph which represents a predefined requirement, for example a style component. At the outset of a design process a set of unbreakable subgraphs associated with a given style is specified. Then in each graph G representing a design all unbreakable subgraphs are found and stored together with their position in the design in a set B_G.

After selecting two graphs G and H to be crossed over its subgraphs g and h are selected. In the first step a starting node *v* is selected in graph G and a similar node *w* is selected in graph H. Then two numbers, *i* and *j*, are randomly chosen for the size of the subgraphs in both G and H. Then in graph G starting from node *v* we select adjacent nodes until the subgraph built reaches *i* nodes. Each time a node *x* is selected in G to be added to the subgraph g it is checked against the set B_G to verify if it belongs to any of the unbreakable patterns. If no, it is added to subgraph g and the selection of the subsequent node is performed. If node *x* belongs to some pattern either the whole pattern has to be added to subgraph g being generated or none of its nodes. This decision is based on the size of the subgraph. If adding the whole pattern would not exceed the selected size *i* of subgraph g it can be added, otherwise node *x* is not added to subgraph g and the selection process is continued. If the whole pattern is added to g it is also added to the set of unbreakable patterns B_g associated with g and removed from set B_G associated with G. Similarly in graph H a subgrah is built starting from node *w*. As a result we obtain two subgraphs, g and h and at the same time two sets of unbreakable patterns B_g and B_h.

Formally, a crossover operator *cx* is defined as a 6-tuple (G, H, g, h, T, U), where G, H, g, h are graphs and their subgraphs, respectively. The crucial elements of this operator are T and U that are called embedding transformations, i.e., they describe how edges of the embedding are to be re-connected. They are sets of pairs of the form (n, n'), where n denotes a node to which an edge was assigned originally and n' - the one to which it will be assigned in a new graph.

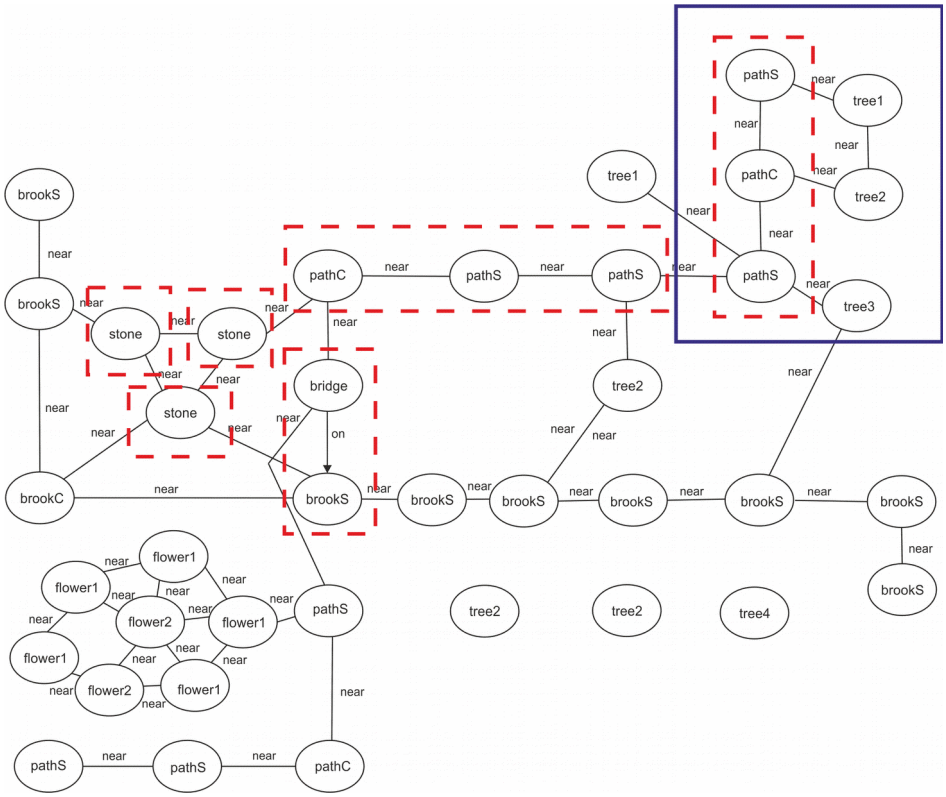

Fig.4 A graph G with depicted unbreakable subgraphs and selected subgraph g

As the result of the crossover we obtain two graphs G' and H'. Graph G' is constructed in such a way that it contains all nodes and edges remaining from G after removing g, all nodes and edges from h and edges connecting nodes from G-g and h obtained by applying transformation T. Moreover the set $B_{G'}$ is obtained by summing sets B_G and B_h. In an identical way graph H' is constructed from H-h and g and set $B_{H'}$ from sets B_H and B_g

As we have sets of patterns associated with newly generated graphs we can easily verify if each of the sets contains all required patterns. In this way we are able to evaluate the fitness of the object represented by such a graph by calculating the percentage of patterns present.

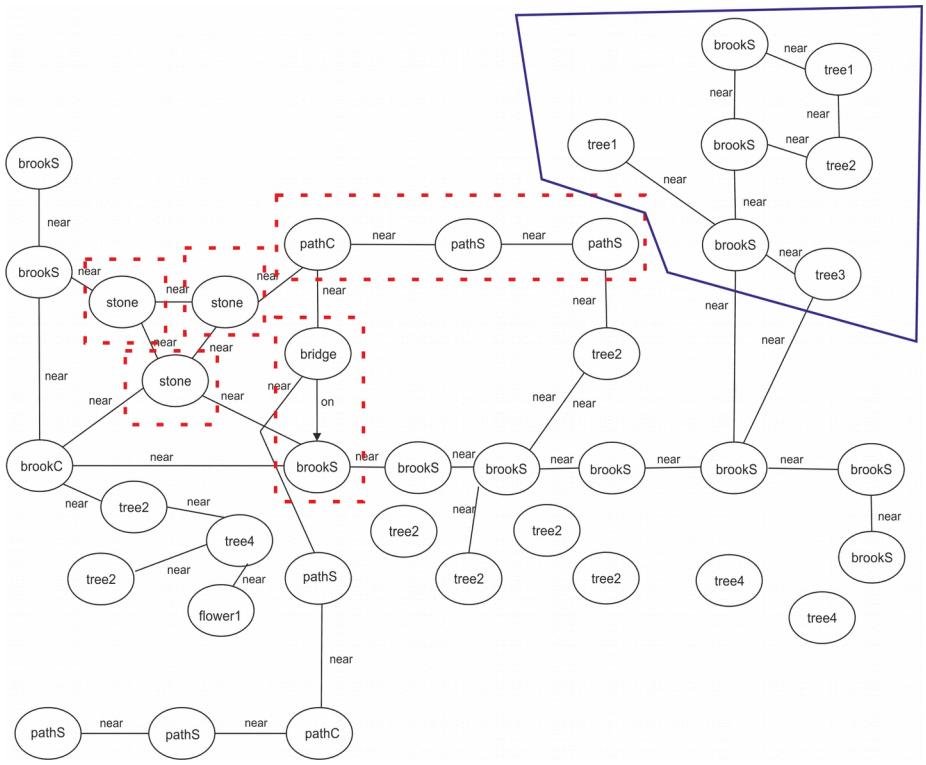

Fig.5 A graph H representing another garden with subgraph h

In Figs. 4 and 5 two graphs, G and H respectively, are depicted. On both of them some of the unbreakable subgraphs representing style patterns are marked by red dashed lines and subgraphs g and h selected for the crossover operation are marked with thick blue lines. Both graphs represent gardens designed in the Japanese style. In the first graph (Fig. 4) the selected subgraph contains one of the unbreakable patterns, while in the second one no style pattern belongs to the selected subgraph. After the crossover two new graphs are constructed according to the method described above.

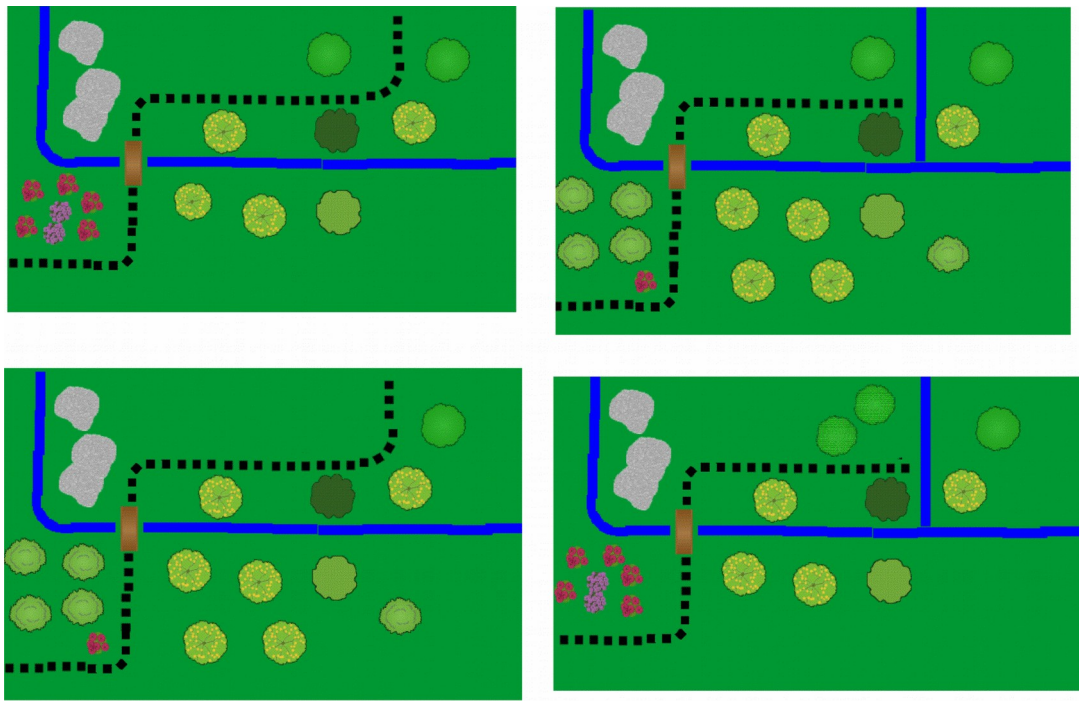

Fig.6 Gardens represented by graphs from Figs. 4 and 5 (top row) and gardens represented by graphs obtained after crossover operation (bottom row)

In Fig.6, top row gardens represented by graphs G and H are depicted, while in the bottom row gardens represented by the graphs after crossover are presented. It can be observed that as the subgraph h contains more nodes representing trees than subgraph g after the crossover one of the new gardens has a tree added to it. The path in the top right garden has also been replaced by an extension of the brook. It has to be noted that although in this case both resulting garden designs are in the Japanese style, it is possible that as the result of crossover the new designs may not follow all requirements of a given style. At the same time associating with each graph a set of style patterns makes it relatively easy to rapidly establish which patterns are missing and correct it.

5. Conclusion

In the proposed approach a graph is used as a genotype and equivalents of standard genetic operators are defined on graphs. These operators are more complex than standard binary ones but they provide us with benefits like the possibility of coding relationships between components of an artefact and ability to introduce structural changes which compensate for it. The strongest point of a graph-based representation is its ability to represent in a uniform way all types of relations and objects and to be able to preserve some required characteristics of the design.

In this paper such characteristics are related to the style of the object but in future we plan to investigate the possibility of applying such an approach to other features. It could be used to preserve some parts of the design that is considered optimal and allow only for the improvement of the remaining parts of the design. It is also a

possible ~~option~~ to use this approach to assure the presence of a predefined number of some components within a designed object.

Another direction for future research is related to defining the strength of unbreakability of patterns. In this paper none of the patterns designated as unbreakable can be broken. Yet, it can be observed that in some situations it is possible that a given pattern is present in a graph multiple times thus breaking one of the occurrences would still allow for the required style to be preserved but at the same time give possibly more freedom for creative results.

References

[1] P.J. Bentley,, Evolutionary design by Computers, Morgan Kaufmann, 1999
[2] J.Jupp, J. S. Gero, Let's look at style: Visual and spatial representation and reasoning in design. In S. Argamon, K. Burns and S. Dubnov (Eds.), The Structure of Style (pp. 159-195),Springer, 2010.
[3] D.E.Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Reading, MA, Addison-Wesley, 1989.
[4] E. Grabska, W. Palacz, Hierarchical graphs in creative design. Machine GRAPHICS and VISION, 9(1/2), 115-123. 2000.
[5] J. H. Holland, Adaptation in Natural and Artificial Systems, Ann Arbor, 1975.
[6] M. Mantyla, An Introduction To Solid Modeling, Computer Science Press, Rockville, MD,vol.87, 1988.
[7] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, Berlin Heidelberg New York, 1996.
[8] B. Strug. Hierarchical Representation and Operators in Evolutionary Design, LNCS vol 3911, pp. 447-454 Springer, 2006.
[9] B. Strug, G. Ślusarczyk, E. Grabska, Design patterns in generation of artefacts in required styles Generative Art 2016 : proceedings of XIX Generative Art Conference, 2016. - pp. 71-78, 2016.