EUROPEAN ACADEMIC RESEARCH
Vol. I, Issue 11/ February 2014

ISSN 2286-4822
www.euacademic.org

Impact Factor: 3.1 (UIF)
DRJI Value: 5.9 (B+)

Decorative Plates Drawing Using Context Free Design Grammars (CFDG)

DHAMYAA A. AL-NASRAWI[1]
ROQYIA J. YOSEF
ALI H. ABDELRASOL
NOOR A. MOSSA
HASSAN H. MUSLIM
Computer Science Department, Science College
Kerbala University, Karbala
Iraq

Abstract:

Context Free Grammar is one way to define formal languages by generating all strings belonging to these languages. Traditional applications of CFG have no relation to any artwork. In this paper, new automatic method of drawing decorative plates was proposed by using Context Free Design Grammar CFDG, which uses symbol/ picture to generate strings resulting from specific language. That means displaying picture corresponding to each terminal instead of displaying the symbol itself. The resulting plates produce beautiful and fine looking works, which can be used as patterns on clothing, textiles, and wallpapers, as well as, the possibility of setting them as background pictures for windows. The proposed system is designed in Visual stodui.Net, 2008.

Key words: Context Free Design Grammar, Formal language, String derivation.

1. Introduction

Language is a set of strings, where each string is a finite

[1] Corresponding author: dhmyaa@gmail.com

sequence of symbols. Formal languages can be specified in different ways: for languages with a finite number of strings, it would be possible to list all strings, but a more convenient way is to describe a formal language by a set of rules, which may generate the strings in the vocabulary. It is important to note that formal languages have no direct connection to the natural languages. Formal languages are mathematical, formalized concepts. They may be used when trying to describe or model the structure of natural languages (Kurth 2007).

Familiar languages include programming languages such as Java and natural languages like English, as well as unofficial "dialects" with specialized vocabularies, such as the language used in legal documents or the language of mathematics. In the case of a language like Java, a string must satisfy certain rules in order to be a legal statement, and a sequence of statements must satisfy certain rules in order to be a legal program (Martin 2011).

Roughly, there are two dual views of languages:

A: The recognition point view, typically some kind of "black box" was assumed, M, (an automaton) that takes a string, w, as input and returns two possible answers: Yes, the string w is accepted, which means that w belongs to the language, L. No, the string w is rejected, which means that w does not belong to the language, L.

B: The generation point of view, which specifies a language in terms of rules that allow the generation of "legal" strings. The most common formalism is that of a formal grammar.

Definition: An alphabet Σ is any finite set. We often write $\Sigma = \{a_1, \ldots, a_k\}$. The a_i are called the symbols of the alphabet (Gallier 2010).

In this paper, automatic drawing method of decorative plates using special kind of grammars (Context Free Design Grammar) was proposed, which puts picture corresponding to each symbols to generate legal strings.

The rest of this paper is organized as follow: section 2

explains the grammars, string derivation, and types of grammars; Context Free Design Grammar is explained in section 3, proposed method of automatic drawing with examples is explained in section 4, conclusions and future works demonstrated - in section 5 and 6.

2. Grammars

A grammar is another way to write the recursive definitions that used to define languages (Winfried 2007). Grammars are described as generators of the languages, in the sense that, given the grammar, strings in the language can be generated in a methodical way (Gallier 2010). There are the following types of grammars (called Chomsky hierarchy) and the corresponding families of languages:

1. Regular grammars (type 3-languages): formal grammars that describe regular languages.
2. Context Free grammars (type 2-languages): Here the left side of the production rules consists of single non-terminal. This class of grammars has been used to study the structure of several natural languages, plus the syntax of many computer languages.
3. Context-sensitive languages (type 1-languages): These are grammars that can be expressed using context-sensitive production rules, such as the one mentioned above: $\alpha A \beta \rightarrow \alpha X \beta$ where α and β are arbitrary symbols, and A is a non-terminal and X a non-empty symbol.
4. The recursively enumerable languages or (type 0-languages): All formal grammars.

Here there are no restrictions on the production rules. In this paper we focused on CFG.

2.1. Context Free Grammar (CFG)

A more general class of languages is the context free languages. A straightforward way to specify a context free language is via

a context free grammar. Context free grammars can be thought of as being for context free languages the analogue of regular expressions for regular languages. They provide a mechanism for generating elements on the language (Carter 2005).

A context free grammar G = (V, T, S, P) consists of two alphabets V and T (called variables and terminals, respectively), an element $S \in V$ called the start symbol, and a finite set of production rules P. Each production rule is of the form $A \rightarrow \alpha$ where $A \in V$ and $\alpha \in (V \cup T)^*$.

Example of CFG:

Here is a language over the alphabet T = {a, b},

$S \rightarrow R$

$R \rightarrow aRa \mid bRb \mid a \mid b$

2.2. Derivation of Strings

It is useful to consider how a string is generated by a CFG. A derivation may provide information about the structure of the string, and if a string has several possible derivations, one may be more appropriate than another.

A derivation of a string x in a CFG is a sequence of steps, and a single step is described by specifying the current string. It is easy to see that for each derivation in a CFG, there is exactly one derivation tree. The derivation begins with the start symbol S, which corresponds to the root node of the tree, and the children of that node are determined by the first production in the derivation. At each subsequent step, a production is applied, involving a variable occurrence corresponding to a node N in the tree; that production determines the portion of the tree consisting of N and its children. For example, the derivation of the following grammar shown in Figure 1 (Martin 2011):

$$S \Rightarrow S + S \Rightarrow a + S \Rightarrow a + (S) \Rightarrow a + (S * S) \Rightarrow a + (a * S) \Rightarrow a + (a * a)$$

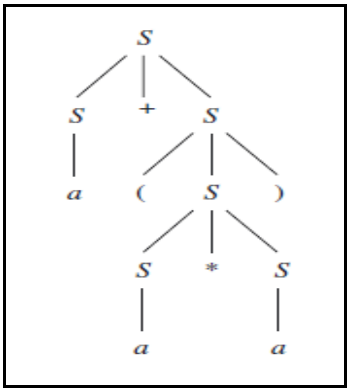

Figure 1: Derivation Tree of a+ (a*a)

3. Context Free Design Grammar (CFDG)

Chris Coyne has built a simple language for creating visual generative grammars: Context Free Design Grammar. It is a digital art program that takes a description of an image and generates the image (as a bitmap, vector image, or movie). The typical use of a context free grammar is to analyze a sentence of symbols to see if they can be reduced to some root symbol. Context Free turns this around by starting with a root symbol/shape (called the start shape) and using the grammar rules to elaborate this into a sentence of symbols/shapes (contextfreeart 2012).

Similar to formal grammars, a Context Free Design Grammar has production rules and non-terminal and terminal symbols. The terminal symbols are now geometrical primitives in Chris Coyne's original implementation circles and squares were used as the basic primitives (contextfreeart 2012).

CFDG systems offer a couple of advantages, though. The CFDG unites the substitution rules and the geometrical operators. This makes the representation slightly more intuitive and it makes it possible to implement more flexible termination rules (Christensen 2009).

4. Proposed Method of Drawing

In this paper, automatic drawing methods were proposed to produce decorative plates with fun like and beauty. For performing this purpose, Context Free Design Grammar (CFDG) was used to generate the terminals of resulting strings from specific grammars. The difference between CFG and CFDG can be shown in the following:

In Context Free Grammar CFG is 4-tuble:

G= (S, T, V, P), where

1. T is alphabet of language called terminals, such as T= {a, b}.
2. N is finite set of symbols called Variables or non-terminals, such as N= {A, B} .
3. S is start symbol.
4. P is finite set of productions.

While, in Context Free Design Grammar CFDG is 5-tuble:

G= (S, T, V, P, C), where

1. T is alphabet of language called terminals, each of them represented by specific picture, such as
   T={a, b} .

a=  b= 

2. N is finite set of symbols called Variables or non-terminals, such as N= {A, B}.
3. S is start symbol.
4. P is finite set of productions.
5. C is the color of Background of starting drawing.

For more details, if the production is

$S \rightarrow aB$

$B \rightarrow ab$

Then, the resulting drawing is:

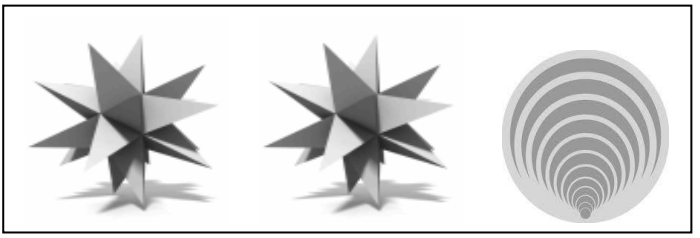

Figure 2: Drawing of aab

Now, set color to background, C = gray. Then the resulting drawing can be shown in the following:

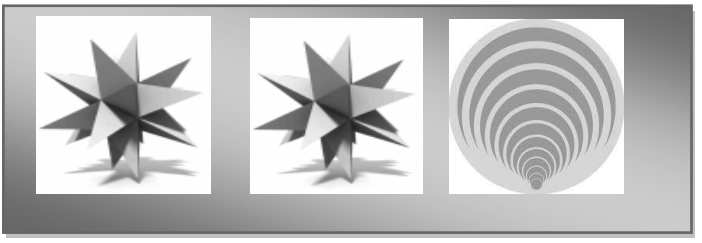

Figure 3: Drawing of aab with background color

The steps or stages of running are explained in Figure 4:

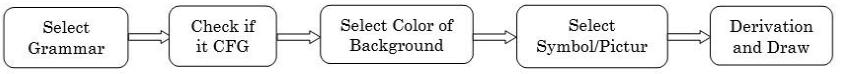

Figure 4: Diagram of Running Stages

The proposed work has more advanced features:

1. Ability to choose appropriate picture to each terminal.
2. Ability to choose color for background, represent the color of start symbol for given grammar.
3. Ability to choose grammar to generate different drawing.
4. Ability to clear previous artwork to start new one.
5. Ability to display the chosen grammar.
6. Ability to draw different artwork by the same grammar.
7. Ability to check the type of chosen grammar to show if it is CFG or not.

These characteristics are shown in GUI of proposed system - see Figure 5 that draws **aaa** string.

Figure 5: GUI of Proposed System when draw aaa string

There are many buttons in this interface, and the project starts by choosing one of listed grammars such as:

1. $S \rightarrow aS \mid \varepsilon$
2. $S \rightarrow aZcde$
   $Z \rightarrow b \mid e$
3. $S \rightarrow ABS \mid \varepsilon$
   $A \rightarrow ab$

When selecting grammar 1 as example, the resulting drawing is as in Figure 6.

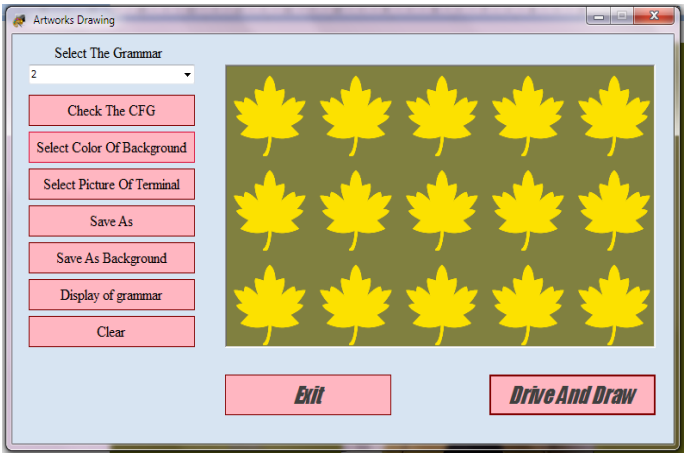

Figure 6: Drawing grammar 1

The result of choosing grammar 2 is shown in Figure 7.

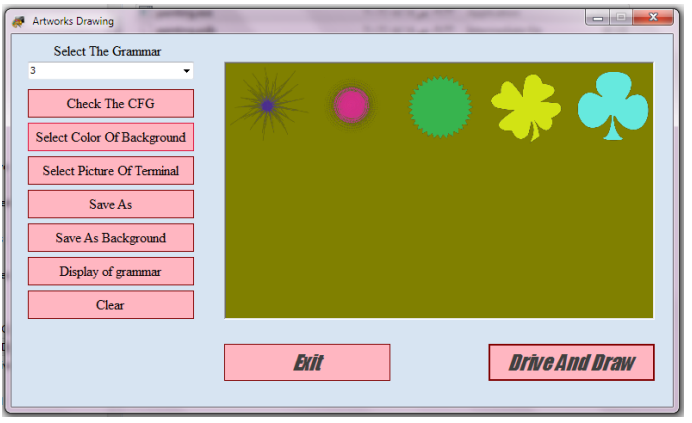

Figure 7: Drawing grammar 2

The result of drawing grammar 3 is shown in Figure 8.

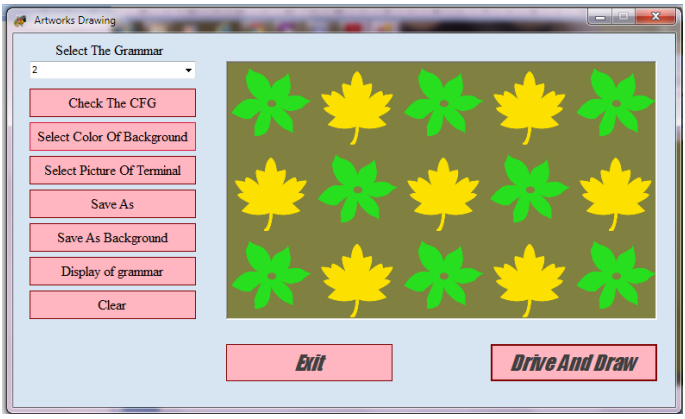

Figure 8: Drawing grammar 2

The pictures corresponding to each symbol are shown in Figure 9.

Figure 9: Display of Symbol/ Pictures

5. Conclusions

As shown from this paper, we can conclude that:

1. The Context Free Grammar can be applied in artistic works and can produce beautiful and fine looking works.
2. For the same grammar, more art works can be designed by choosing different pictures for terminals symbols.
3. The resulting drawing depends on the structure of

grammars.
4. The large number of terminals produces more diversity in pictures of resulting drawing.

6. Future Works

1. The system can be developed by passing each picture of terminal to screen saver, with simple animations.
2. Two dimension transformation (rotation, reflection …etc.) can be used to organize the derivation of terminal based on given grammars.
3. More developments can be made by associate geometrical figure for each terminal that make us control on colors of figures and produce more art works.

BIBLIOGRAPHY:

_____. "Context Free Art", available at: http://www.contextfreeart.org/mediawiki/index.php/CFDG_Introduction.

Carter, Tom. 2005. "Introduction to theory of computation." Complex Systems Summer School. Retrieved from http://csustan.csustan.edu/~tom/LectureNotes/Computation/computation.pdf.

Christensen, Mikael Hvidtfeldt. 2009. "Structural Synthesis using a Context Free Design Grammar Approach." *12th Generative Art Conference GA.*

Gallier, Jean. 2010. "Introduction to the Theory of Computation Formal Languages and Automata Models of Computation." Retrieved from http://www.cis.upenn.edu/~jean/gbooks/cis51108sl1.pdf.

Kurth, Winfried. 2007. "Specification of morphological models with L-systems and relational growth grammars." *Image – Journal of Interdisciplinary Image Science* 5. Special

Issue.

Martin, John C., 2011. *Introduction to Languages And The Theory Of Computation*. Fourth Edition. New York: McGraw-Hill.