Fractal Image Generation Based on Petri Net Theory

Hussein Karam Aboul Ella Hassanien Masayuki NAKAJIMA
Graduate School of Information Science & Engineering, Tokyo Institute of Technology
2-12-1 O-okayama, Meguro-ku, Tokyo, 152-8552 Japan
{huss,abo,nakajima}@cs.titech.ac.jp

Abstract

A novel algorithm for fractal image generation applied to the dynamics of petri net is presented. It translates the evolution of Petri net into graphic output via marking reachability. The idea is derived from the concept of fractal iteration principles in the escape-time algorithm and chaos game. The approach uses a Petri net as a powerful abstract modeling tool for the generation of self-similar fractal images via its duality, deadlocks, inhibitor arcs, firing sequence, and marking reachability. Generating fractal images via the dynamics of a petri nets allows an easy and direct proof for the similarity and correspondence between the dynamics of complex quadratic fractals via a recursive procedure and the state of a petri nets via a reachability problem. Reachability problem will be adopted in terms of dynamics of the fractal in order to generate images via two proposed methods. Validation of our approach is given in terms of experimental results. An investigation of the relationships between the generated images and their corresponding petri nets is also discussed.

Keywords: fractals, chaos game, escape-time algorithm, petri nets, iterated function system, reachability problem, PN duality

1 Introduction

Recently, the beauty of fractals has attracted wide interest among mathematicians, computer scientists and artists. Computer graphics made it possible to recognize the beauty of fractals and turned them into an art form [24]. Mandelbrot [21] formally defines a fractal to be a set whose fractal dimension exceeds its topological dimension. In fact, fractals and deterministic chaos are mathematical tools to modelise different kinds of natural phenomena or objects. Fractals are most applicable in a graphic representation of chaos, the apparently unpredictable behavior arising in a dynamic system because of great sensitivity to initial conditions [12]. Fractal methods are quite popular for the modeling of natural phenomena in computer graphics ranging from random fractal models of terrain [24], to deterministic botanical models such as L-systems, iterated function system (IFS) and recurrent iterated function system (RIFS)[1,3,4].

On the other hand, petri nets (PN) [2,20,23] has been proved to be a powerful graphical and mathematical modeling tool for the formal description of systems whose dynamics are characterized by concurrency, synchronization, mutual exclusion and conflict. More details on petri nets can be found in [22,25]. Reachability analysis is fundamental and central to the study of petri nets since many problems in petri nets can be reduced to the reachability problem [13]. The reachability problem in case the of fractal image generation can be viewed in some forms as an escape-time algorithm [14,15,16,27] and other as a "chaos game"[5,19] described by an IFS. Chaos game in fractal image generation is similar to the nondeterminism characteristic of the petri net execution. If at any time more than one transition is enabled, then any of the several enabled transitions may fire. The choice as to which transition fires is made in a nondeterministic manner, i.e., randomly. Several techniques for generating fractal shapes were developed and used to produce fascinating images [8,9,10,11,17,26].

In this paper we'll present a new technique for reachability problem solving in order to generate fractal images via two proposed methods one of them is called PN-escape-time which modify the escape-time algorithm using the petri net properties to generate fractal images. The other one is called PN-chaos which is based on chaos game. Compared with the previous methods, our approach is suggested to use because it is independent of geometry specification, easy to use

due to graphics visualization, simple for image generation, and has the ability to generate unlimited class of fractal patterns.

The paper is organized as follows. Section 2, give an overview of a petri nets reachability problem, definition, execution and outlines our idea for using a petri nets as a filter for image generation. Section 3, points out the relationship between fractal image generation methods and their corresponding nets by introducing our proposed methods. Finally, some experimental results with discussion and conclusion are given in sections 4 and 5 respectively.

2 PN Execution and Reachability Problem

In this section, we'll derive an equation which form the main idea for reachability problem and marking execution with some illustrated example. The two proposed methods depend mainly on these derived equation and the properties of a petri nets such as deadlock, duality, bounded, reversible and live. The derived equation exhibit a certain analogy for the iterative procedure of the fractal image generation via the escape-time algorithm and chaos game for manipulating the reachability problem in order to generate images. A basis definition and execution of petri net necessary for the present paper are given. A petri nets $PN = (N, M_0)$ consists of a structure N and initial marking M_0, where:

$N = (P, T, F, W)$ is a petri net structure,
$P = \{p_1, p_2, \cdots, p_m\}$ is a finite set of m places,
$T = \{t_1, t_2, \cdots, t_n\}$ is a finite set of n transitions,
$F \subseteq (P \times T) \cup (T \times P)$ is set of arcs (flow relation),
$W : F \rightarrow \{1, 2, 3, \cdots\}$ is a mapping which associates to each arc (edge) of the net its weight.
$M_0 : P \rightarrow \{0, 1, 2, 3, \cdots\}$ is the initial marking representing the initial state of PN. $P \cap T = \phi$ and $P \cup T \neq \phi$.

A special petri nets in which place capacities and arc weights are equal to one is called a condition/event net (C/E net). A transition without any input place is called a source transition and one without any output place is called a sink. Labels for unitary weight are usually omitted. In modeling using petri nets, we regard the places as conditions and the transitions as events. The dual of a petri net $N = (P, T, F, W)$ is the petri nets $\bar{N} = (T, P, F, W)$ which results from interchanging places and transitions. On the other hand, the state of a petri nets is described by means of the concept of marking. A marking is a function that assigns to each place a nonnegative integer called token. A token is a primitive concept of a petri nets (like places and transitions). From a graphic point of view places are usually represented by circles, transitions by rectangles and marks by black dots into places. The dynamics of the net is described by moving tokens among places according to the following transition firing rules [25]:

1) A transition t is said to be enabled to "fire" if each input place p of t is marked with at least $W(p,t)$ tokens, where $W(p,t)$ is the weight of the arc from p to t.
2) An enabled transition may or may not fire depending on whether or not the event actually takes place.
3) A firing of an enabled transition t removes $W(p,t)$ tokens from each input place p of t and adds $W(t,p)$ tokens to each output place p of t, where $W(t,p)$ is the weight of the arc from t to p.
4) The marking of the other places which are neither input nor output of t remains unchanged.

Reachability is a fundamental basis for studying the dynamic properties of any system. The firing of an enabled transition will change the tokens distribution in a net according to the transition firing rule described before. In general, firing a transition will change the state marking M to a new marking $\acute{M}$. The state space of a petri nets with n places is the set of all markings, that is N^n. We'll formulate such state change caused by a transition firing using a partial function δ which we call the next-state function, where $\delta : N^n \times T \rightarrow N^n$. The function δ when applied to a marking M and a transition t_j yields the new marking $\acute{M}$ which results from firing the enabled transition t_j in marking M that is,

$$\delta(M, t_j) = \begin{cases} \text{undefined} & \text{if } t_j \text{ is not enabled} \\ \acute{M} & \text{if } t_j \text{ is enabled} \end{cases} \quad (1)$$

where, $\acute{M}$ is the marking which results from removing tokens from the inputs of t_j and adding tokens to the outputs of t_j. The function δ given by equation 1, incorporate a notion of distributed state and a rule for state change of a net via a sequence of markings $(M_0, M_1, M_2 \cdots)$ and a sequence of transitions $(t_{j(0)}, t_{j(1)}, t_{j(2)}, \cdots)$ which were fired. The relationship between these two sequences form the main idea of our proposed methods because of its similarity with

the recursive procedure for generating images of both escape-time algorithm and chaos game. Thus, we formulate equation (1) in terms of such two sequences in a recursive manner as follows:

$$\delta(M_k, t_{j(k)}) = M_{k+1} \qquad for \quad k = 0, 1, 2, \cdots \quad (2)$$

Equation (2), forms an iterative process of marking which exhibit a certain analogy of the iterative procedures for the fractal image generation which we shall discuss later. An example of a petri net execution which illustrate the marking reachability of equation (2) with initial marking $M_0 = (1, 0, 1, 0, 2)$ is shown in Figure 1.

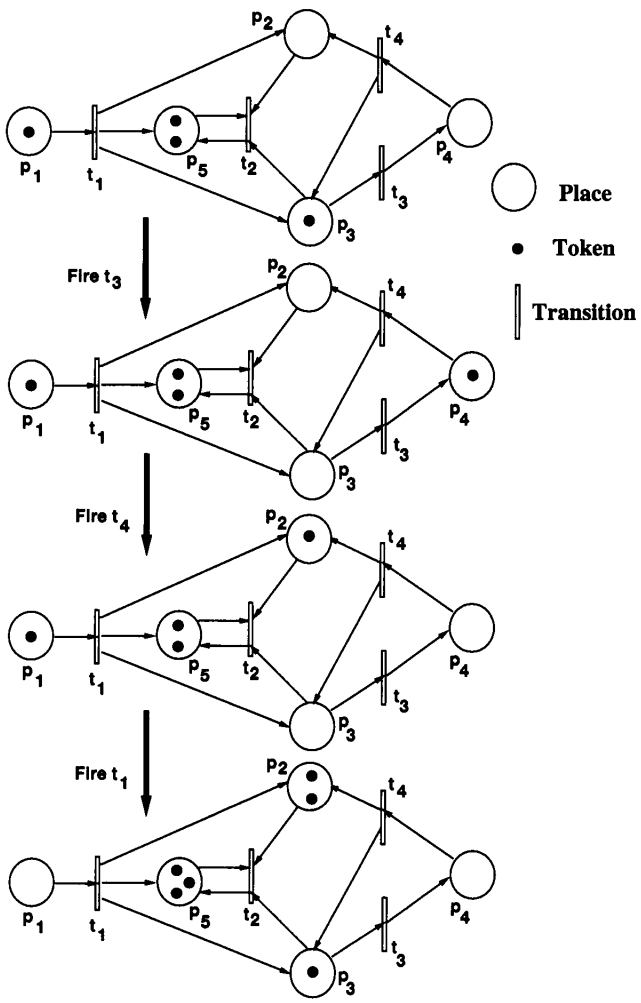


Figure 1. An illustration of marking reachability

In Fig. 1., two transitions t_1 and t_3 are enabled from the marking $M_0 = (1, 0, 1, 0, 2)$. Choosing one arbitrarily, we can fire t_3 producing the marking $\delta(M_0, t_3) = (1, 0, 0, 1, 2) = M_1$. In this marking transitions t_1 and t_4 are enabled. Firing t_4 changes the marking to $\delta(M_1, t_4) = (1, 1, 0, 0, 2) = M_2$. In M_2 t_1 is enabled and firing it leads to $\delta(M_2, t_1) = (0, 2, 1, 0, 3) = M_3$.

3 PN Tools for Fractal Image Generation

Petri nets presents several degree of freedom. For instance, the marking of the net, the sequence of firing or the number of firing could be selected to be displayed. In this section, we'll solve marking reachability problem in order to generate images via two proposed methods. The solution is based on such degree of freedom, equation (2) and the properties of PN. Such properties are initial marking, final marking, transition firing sequence, deadlock, duality, place invariants and the so-called zero-testing through the introduction of inhibitor arcs [6,22]. An inhibitor arc connects a place to a transition and is represented by a dashed line terminating with a small circle instead of an arrowhead at the transition. The introduction of inhibitor arcs adds the ability to test "zero" (i.e., absence of tokens in a place). In principle, the procedure for generating fractal images are given as an iterative mapping function from pixel coordinate on the screen to pixel color. Thus the relationship between the PN and the generated image is to search for a given marked petri net for two functions g_1 and g_2 such that g_1 mapping from the spatial coordinates (x,y) on some region D on the screen to the initial marking M_0 then after some firing sequence of a petri net, the function g_2 maps the obtained final marking M_1 into a color $C(x, y)$. For each of the two proposed methods, we'll discuss some suggested rules for such mapping functions.

3.1 Suggested Rules

To map the space of pixel coordinates (x, y) into the space of markings we need a mapping functions $g1 : N^2 \rightarrow N^m$ and $g_2 : N^m \rightarrow N^3$ where, m is the number of places in the net. We choose two different cases for such mapping:

- $g1 = A \circ B$, as a composition of two functions say A and B where, $A : N^2 \rightarrow N$ and $B : N \rightarrow N^m$. For instance, B may be selected in such a way to put the same number of marks into each place that is, $B_j(y) = y, j = 1, \cdots, m$. The function A strongly characterizes the image pattern. The other case is to choose the function g_1 as a projection function that is, $g_1(x, y) = (x, y, 0, 0, \cdots)$. This means that the values of the point coordinates are assigned to two some places from the m places.

- The function g_2 is given as a map $g_2 : N^m \rightarrow N^3$

which assign some color for a given point (x, y) as an RGB components chosen from the marking of three selected places.

3.2 PN-Escape-time Algorithm

Escape-time algorithm, were originally developed as a method for visualizing the dynamics of complex quadratic fractals. It consists of testing how fast points z outside the attractor diverged to infinity while iterating in complex plane the function:

$$f_c(z) = z^2 + c \tag{3}$$

where, c is some constant. The dynamics of these functions become evident by examining the resulting orbit of their iteration on an initial point. Given an initial point z_0 and a function f_c, the points in the orbit $(z_1, z_2, \cdots)$ are defined recurrently as:

$$z_i = f_c(z_{i-1}) \tag{4}$$

The PN-escape-time method assign pixel colors on the basis of the number of iterations of the net before a deadlock on the marking occur. It is derived from the study of both equations (2) and (4). The difference is that the circle of infinity in case of escape-time algorithm correspond to deadlock case obtained from the duality of the net from the final marking to the initial marking. The resulting image produced by these method performs the same procedure of the escape-time algorithm because color counters indicate the time required for the marking of a net to be in a deadlock case. The algorithm is given as follows: Assume the monitor has a graphical resolution $a \times b$ points and K colors then steps for generating image patterns for this method is given as follows:

Step1 (Initialization)
- Select some suitable attractor domain D for drawing.
- Fix the maximum number of iterations $Maxiter$.

Step2 (Loop)
- For all pixels (i,j) in the domain D do
- set $n = 0$.
- Calculate the initial marking as $M_0 = g_1(i, j)$.

Step3 (iteration)
- Let the enabled transition fire.
- $n = n + 1$.

Step4 (Condition Evaluation)
- If (*no deadlock*) and $(n < Maxiter)$ then determine the PN duality and go back to step 3.
- If (*deadlock*) and $(n < Maxiter)$ then calculate color c as $c = g_2(n)$ and goto step 5.
- If (*no deadlock*)and $(n > Maxiter)$ then choose the red color as a color c and goto step 5.

Step5 (Color Assignment)
- Assign color c to the pixel (i, j) and goto the next pixel starting again with step 2.

3.3 PN-Chaos Algorithm

Fractals in general have an equally valid existence as a limit of random processes. The chaos algorithm generates a sequence of points which fall onto (or near) the attractor. Mathematically the chaos process is described by an IFS and the rate of convergence of points towards the attractor is determined by the contractivity of the IFS mapping [7]. IFS is defined as a pair $\{X; T_n, n = 1, 2, \cdots, N\}$ where, X is a complete metric space and each T_n are affine contractions that is,

$$T_i(x) = C_i x + B_i \tag{5}$$

where, C_i is a square matrix with n rows and B_i is a vector with n elements. By a theorem of Hutchinson [18], there exist for each IFS a single compact non empty set $\hat{A}$, called its attractor which is the union of images of itself under the IFS maps.

In implementing the IFS method, one important question is the prediction a priori of the region of space containing the fractal attractor. Without such a prediction, one could only approximately estimate the spatial extent based on calculating several points of the attractor with no guarantee that these points are near the bounds.

The chaos representation is approximates the attractor of an IFS $\{T_i\}_{i=1}^N$ with a point clouds. The attractor of a random algorithm can be viewed as a limit of random process. It starts with a point x_0 in a metric space X and generates a sequence of points as

$$x_{t+1} = (T_j)(x_t) \tag{6}$$

where j is an integer from one to N randomly chosen for each new point in the sequence. This sequence is dense in the attractor [18].

On the other hand, the PN-Chaos algorithm is derived from the study of both equations (2) and (6), where the new points are calculated in terms of net marking and enabled transitions firing instead of the contraction mappings $\{T_j, j = 1, 2, \cdots, N\}$ of the IFS. The coordinate of the points of the resulting image are calculated as a function of the firing transition and its

color as a function of marking. The algorithm of such method for a given marked PN is given as follows:

Step1 (Initialization)
- $x_0 = 0, y_0 = 0$
- Assign to each transition t a point u_t in the plane which forms a vertices of some polygon.

Step2 (Loop)
- For a fixed number of times do

Step3 (Calculation)
- Randomly fire the enabled transition t_k for some k.
- Calculate : $\dot{x} = (u_k + x_0)/2$, $\dot{y} = (u_k + y_0)/2$.

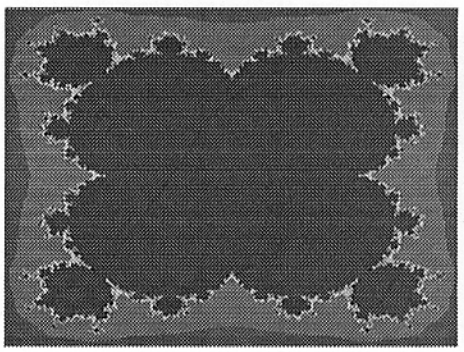
Step4 (Color Assignment)
- Calculate the pixel color from the actual marking.
- Set (x_0, y_0) to the new points: $x_0 = \dot{x}$, $y_0 = \dot{y}$.

According to the above procedure, the firing sequence is assigned to the spatial information and the actual marking to the color information.

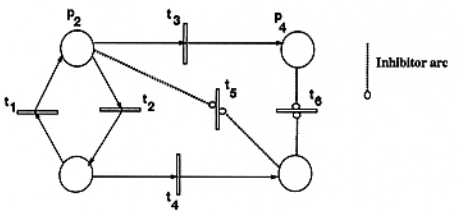
4 Experimental Results and Discussion

The result of the image patterns obtained from the two proposed methods depend on some factors. These factors are given in terms of the class of mapping functions chosen, the upper bound of the number of firing sequences set, the characteristic of the petri nets (live, bounded, fair and reversible), the domain D for drawing, and the properties of the transitions (dead, fair, live and conflict) [22,25].

For instance, the resulting images of the PN-escape time algorithm are mainly affected by the feature of the PN chosen, the number of iteration, the domain D setting and the deadlock case for the transitions. We investigated two classes of petri nets. One with inhibitor arcs is shown in figure 2(b) with its corresponding image given in figure 2(a). The other one without inhibitor arcs is shown in figure 3(b) with its corresponding image in figure 3(a). The area with red color corresponds to a deadlock case for transitions. The resulting images of the PN-chaos algorithm are mainly affected by the characteristic of the petri nets and the properties of the transition with the point coordinates assigned to each transitions. Different point coordinates assigned to each transition will lead to different images. We investigated two petri nets with different coordinate points assigned to each transition. The first one which has the characteristic of bounded, live and reversible is given in figure 4(b), with its corresponding output image shown in figure 4(a). The other PN is shown in figure 6 which has characteristic of non bounded, live and non reversible. The corresponding output images are shown in figure 5. Such images have spatial separation of colors because firing a transition always produces the same marking. In principle, investigating other petri nets with different features and properties of the transitions will lead to other generated images.

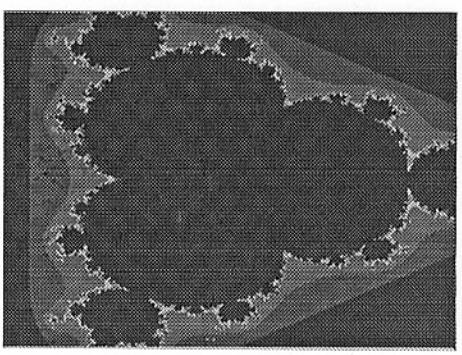


(a) Generated image

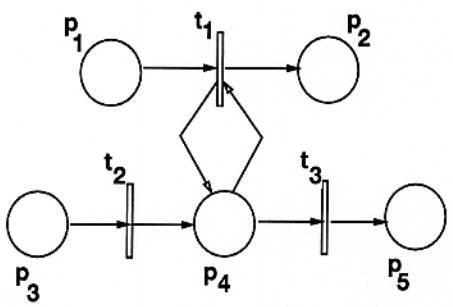


(b) Corresponding PN with inhibitor arc

Figure 2. Image generated using PN-Escape-time algorithm with inhibitor arc

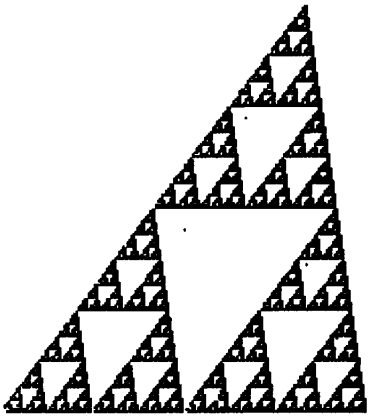


(a) Generated image

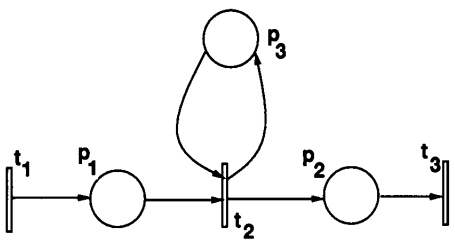


(b) Corresponding PN without inhibitor arc

Figure 3. Image generated using PN-Escape-time algorithm without inhibitor arc

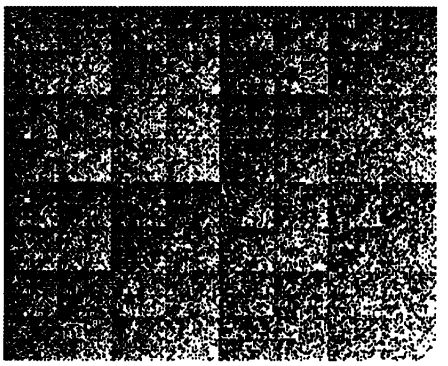


(a) Sierpinski triangle



(b) Corresponding PN

Figure 4. image generated using PN-Chaos algorithm with its corresponding PN



(c)

Figure 5. Some quadrangular images generated using the PN-Chaos algorithm

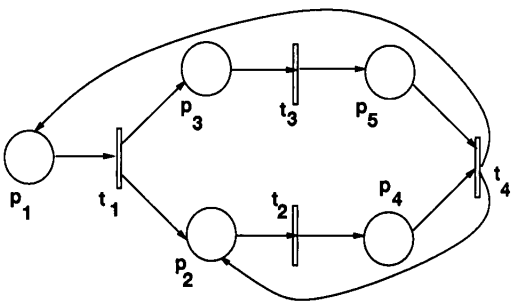
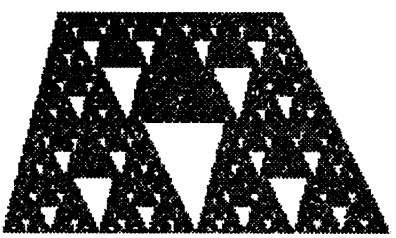
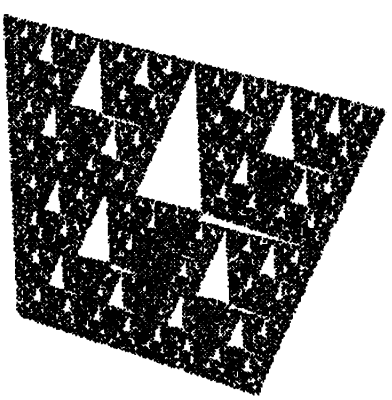


Figure 6. Corresponding Petri net for images of figure 5.



(a)



(b)

5 Conclusion

Many different approaches have been proposed for generating fractal images. In this paper we have presented a new idea for computer image generation via a new kind of analysis method for the dynamics of a petri nets and its reachability problem. This problem is manipulated via two proposed methods which exhibit a certain analogy with fractals. The visual aspects of the resulting images of the two proposed methods depend on the following petri nets features: the final marking of the net after a sequence of firing, the nature and the number of transitions fired during the simulation and the firing sequence. Further research effort is needed to derive more classes of images and to carry out a deeper analysis of the relationships between images and petri nets. Moreover, future computer systems designers and users will require new conceptual mechanisms and theories to deal with their systems. Petri nets incorporate the fundamental concepts which can be used as a basis for these models and theories.

References

[1] Alastair N.Horn, " IFS and Interactive Image Synthesis", Computer Graphics Forum, Vol. 9, pp. 127-137, 1990.

[2] Agerwala, T., " Putting Petri Nets to Work", Computer, Vol. 12, No. 12, pp. 85-94, 1979.

[3] Barnsely, M.F., and Demko S.G., "Iterated function systems and the global construction of Fractals", Proceedings of the Royal Society of London Series A 399, pp. 243-275,1985.

[4] Barnsely, M.F., "Recurrent iterated function systems", Const. Approx. Vol. 1, No. 1, 1989.

[5] Barnsely, M.F., Malassenet F. and Jacquin A., "Harnessing Chaos for Image Synthesis", Proceedings of SIGGRAPH '88.

[6] Bourjij, J., Boutayeb, D., Koenig, T., Cecchin T., " On Generating a Basis of invariants in Petri Nets", IEEE Conf. on Systems, Man, and Cybernetices, Vol. 3, pp. 2228-2233, 1997.

[7] Canright D., "Estimating the spatial extent of attractors of iterated function systems", Computers and Graphics Vol.18, No.2, pp. 231-238,1989.

[8] Casey S. D. and Reingold N.F., "Self similar fractal sets: Theory and Procedure", IEEE Computer Graphics & Applications, Vol. 14, No. 3, 1994.

[9] Daryl H. Heptin, P. Prusinkiewicz and D. Saupe, "Rendering methods for iterated function systems", Fractal in the fundamental and applied sciences, New York, 1991.

[10] Daryl H. Hepting and J.C.Hart,"The Escape Buffer: Efficient Computation of escape-time for linear fractals", Graphics Interface'95 Proc., pp.206-214,1995.

[11] Dario Maio, and A. lumini "image generation by PN ", GKPO'98, Vol. 7, No. 2, pp. 221-232, 1998.

[12] Devaney R., Chaos, Fractals, and Dynamics, Computer Expermints in Mathematics. Addison Wesley, 1990.

[13] Huang, J., and Tadao, M., " Classifications of Petri net transitions and their Application to firing sequence and reachability problems ", IEEE Conf. on Systems, Man, and Cybernetices, Vol. 1, pp. 263-268, 1997.

[14] Hussein, K.H., Aboul Ella, H., M. Nakajima, " Fractal Implementation of Higher order Mapping Based on Escape-time Algorithm ", ITE Annual Conference, D-12-112, 1998.

[15] Hussein, K.H., Aboul Ella, H., M. Nakajima, "Efficient Linear Fractal Algorithm for Computing The Continuous Escape-Time Classifications", Technical Report of IEICE, Vol. 97, No. 558, pp.125-130, 1998.

[16] Hussein, K.H., Aboul Ella, H., M. Nakajima, "Petri net Based Escape-time Fractal Image Generation Algorithm", IEICE Annual Conference, D-11-83, 1998.

[17] Hussein, K.H., Aboul Ella, H., M. Nakajima, "Petri net Modeling Methods For Generating Self-similar Fractal Images", ITE Tech. Rep., Vol. 22, No. 45, pp. 13-18, 1998.

[18] Hutchinson, J.E., "Fractals and self-similarity", Indiana University Journal of Math., 30(5), pp.713-747, 1981.

[19] Jeffrey, H. J., " Chaos Game Visualization of Sequences ", Comp. & Graph, vol. 16, pp. 25-33, 1992.

[20] Johnsobaugh, R., Tadao, M., " Petri Nets and Marked Graphs-Mathematical Models of Concurrent Computation", The American Math. Monthly, Vol. 89, No. 8, pp. 552-566, 1982.

[21] Mandelbrot B.B., "The fractal geometry of nature", W. H. Freeman, New York, 1982.

[22] Peterson, J. L., "Petri nets", ACM Computing Surveys, Vol. 9, No. 3, pp. 223-252, 1977.

[23] Petri, C. A., "Kommunikation mit Automaten", Bonn, W. Germany, 1965.

[24] Peitgen H-O, and Saupe D., "Fractals for the classroom", Springer-Verlag, New York, 1992.

[25] Tadao M., " Petri Nets: properties, analysis and applications ", Proc. of the IEEE, 77(4), 1990.

[26] Prusinkiewicz, P. and Hammel M., "Escape-time visualization for language-restricted iterated function systems", Graphics interface '92 proceedings, pp. 224-231,1992.

[27] Sprott J.C. "Automatic generation of general quadratic map Basins", Computer and Graphics, Vol.19, No.2, pp 309-313,1995.