Nonlinear Phenomena in Complex Systems, vol. **17**, no. 3 (2014), pp. 318 - 326

Illustrations of Irreducibility and Tops of Umbrellas in the PostScript Methodology

Ivan A. Kovalew[*] and Dmitry W. Serow[†]
St.Petersburg State Polytechnic University, 29 Politechnicheskaya Str, 195251 Saint-Petersburg, RUSSIA
(Received 29 November, 2013)

Sierpiński carpets and similar objects, irreducible and indecomposable continua are visually realized based on the PostScript language. Algorithmic problems associated with rendering effects and the presence of tops of umbrellas and irreducible points have been solved within the PostScript language methodology.

AMS Subject Classification: 68T35, 28A80
Keywords: PostScript, operator, operand, stack, dictionary, Sierpiński sieve, Sierpiński carpet, rectangular snowflake, top of umbrella, irreducible point, irreducible continuum, indecomposable continuum

The problem of effectiveness of computations is a most important for nonlinear dynamic and geometric models realizations.

The PostScript interpreter controls actions of the output device according to instructions provided in the PostScript program generated by an application [1]. The proposed in what follows approach was not intended by the authors of PostScript.

1. Introduction: aesthetics and effectiveness of computations problem

PostScript aesthetics is formed by short code without loss of functionality of high level programming languages. At the same time the direct recourse to the stack are appeared due to PostScript is an interesting hybrid of high and low level programming languages. This leads to a reduction of algorithmically complex pieces of code.

The main elements of Postscript are stack, *back polish notation* (BPN)

$$operand_1 \ldots operand_n \; operator$$

and dictionary structure. Thus it is possible *to create operators on one's own initiative.*

Although the number of built-in operators is large, the names that represent operators are not reserved by the language. A PostScript program may change the meanings of the operator names.

A form geometric complexity of the dynamic system action (on the plane, for instance) produces the algorithmic process complexity. It is clear that a computational complexity grows exponentially for fractal sets. Ad hoc the geometric (qualitative) illustrations of the dynamic system action realization lead to algorithmic complexity that grows exponentially too.

PostScript allows to produce the plane construction on the delivery plane immediately from algorithm avoiding the increase of algorithmic complexity. IN this way complex geometric constructions can be algorithmically implemented especially easy.

Bill Casselman [2] established the foundations of the methodology for basic features of the PostScript language and showed how to apply it for producing mathematical graphics in a good style of mathematical illustration. Further we put the following problem for "bad" topogical situations.

Effectiveness of computations problem: *Every step of the algorithm depends on the state of the stack defined by the previous steps. The PostScript algorithm is effective in the sense of*

[*]E-mail: letau@yandex.ru
[†]E-mail: dimusum@yandex.ru

direct work with the stack.

2. Tops of umbrellas and irreducible points

The *n-dimension umbrella* is said to be an arc and n-dimension ball union, such that their intersection is one point being the center of the ball (e. g. see [4]).

The continuum C is called *irreducible between points a and b* if $a, b \in C$ and two points can be joined only by the set. If continuum C is irreducible between every points pair then it is called *irreducible continuum.*

Example 1. 1-*dimension umbrella.* Let us consider a numeric function with unique irreducible point.

1° Let the sequence of functions

$$u_n^p(x) = A_n \frac{x^{2p}}{(x^2+\varepsilon_n)^{2p}}, \quad \forall p \in \mathbb{N},$$

is such that $\max\limits_{|x|<1} u_n^p(x) = 2^{1-n}$, $\forall\, n \in \mathbb{N}$, $p \geqslant 2$ and m are odd numbers;

2° and sequences defined by the formulae

$$\sigma_{n+1}^p = \sigma_n^p + (-1)^n \alpha_{n+1} u_{n+1}^m,$$

$\sigma_1^p(x) = u_1^m(x)$, $\alpha > 0$, such that

$$\forall \max_{|x|<1} \sigma_n^p = 1 \quad \text{and} \quad \forall \min_{|x|<1} \sigma_n^p = -1.$$

defined by the condition $|x| \leqslant 1$ is the irreducible continuum between points $(-1, \sigma_\infty^p(1))$ and $(1, \sigma_\infty^p(1))$.

"Half" of the graphics consisting of points with the positive (negative) abscissa is irreducible between the point $(1, \sigma_\infty^p(1))$ (the point $(-1, \sigma_\infty^p(1))$) and the point $(0,0)$.

It is clear that the limited set of the $\sigma_\infty^p(x)$ contains only irreducible points between the points $(1, \sigma_\infty^p(1))$ and $(-1, \sigma_\infty^p(1))$:

$$\textit{limited set } \sigma_\infty^p(x) = \operatorname{Clos} \operatorname{gr} \sigma_\infty^p(x).$$

But only the point $(0,0)$ is contained in the plot of $\sigma_\infty^p(x)$ because it is a single internal point!

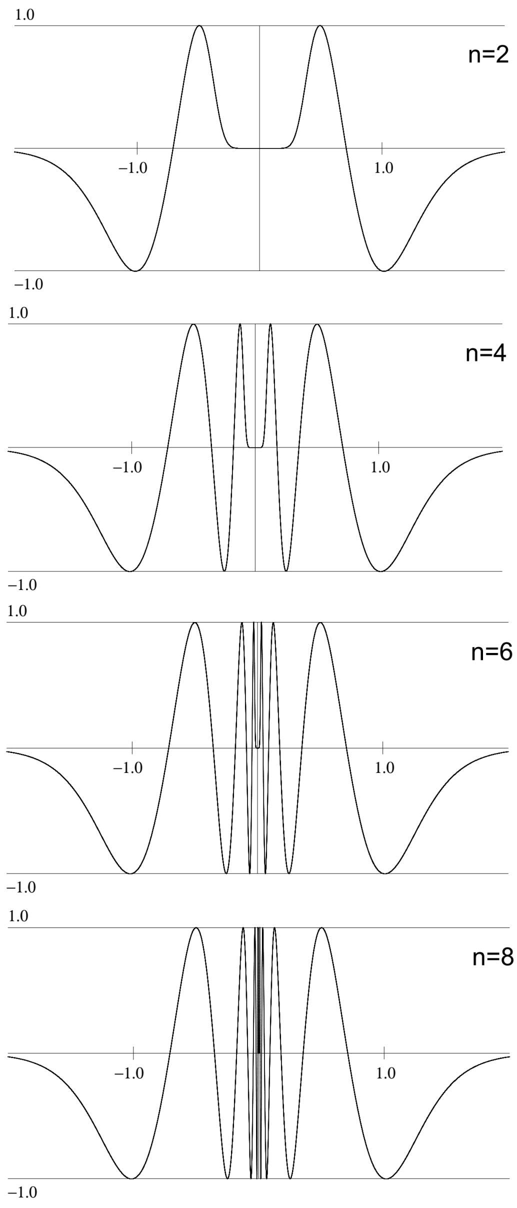


FIG. 1: Functions $\sigma_n^p(x)$, for $p = 8$ and different n.

Proposition 1 *The point $(0,0)$ is the top of 1-dimensional umbrella.*

PROOF Let us consider sequence of unions of arcs on the plot with ends in the points

$$(x_{2k}, -1) \text{ and } (x_{2k+1}, 1), \; k \geqslant 1,$$

such that $\sigma_n^p(x_{2k}) = -1$ and $\sigma_n^p(x_{2k+1}) = 1$ and lengths of tangents in the points

$(o(\sigma_n^p)(k), 0)$ being end point and other end points with identical negative or positive ordinate. $\{o(\sigma_n^p)(k)\}_{k\geqslant 1}$ is a nill function sequence. Topological limit of the sequence is 1-dimension umbrella with top in the point $(0, 0)$. Moreover, point $(0, 0)$ is the top of two umbrellas from right and from left. ■

Remark 1. *Every irreducible point is the top of umbrella but not every top of umbrella is the irreducible point.*

It is clear that there subsist irreducible continua containing only tops of umbrellas.

Remark 2. *It seems that all planar nowhere dense fractals contain only the tops of the 1-dimensional umbrellas (as for instance "Sierpiński Sieve"). However, this is not true (see example "rectangular snowflakes").*

Why is it silly to write fractal programs in PostScript? (see [3], p. 7)

Because the interpreter engine eliminates the need for explicit recursion.

Example 2. *Sierpiński carpet (Sierpinski (universal plane) curve).*

Let us consider the square in the first quadrant, such that $0 \leqslant x, y \leqslant 0$, and coordinates x and y are represented as symbols sequences in triple numerical system (alphabet symbols: {0, 1, 2}). Sierpiński carpet is the set contains only (x, y) points, such that two coordinates are the sequences free of symbol "1".

The algorithm of its construction is based on the points screening from the symbol "1" by the Monte-Carlo method.

It is the nowhere dense fractal example containing only tops of umbrellas.

Executable code for Fig. 2.

```
%!
/N 5 def 100 dup scale 3 3 translate
/B# 2 31 exp 1 sub def 1879 srand
4000000 {
/X rand B# div def /Y rand B# div def
/A X def /C Y def /BlackFlag true def
N {/B A 3 mul truncate def
/D C 3 mul truncate def
B 1 eq D 1 eq and
 {/BlackFlag false def exit}
{/A A 3 mul B sub def
 /C C 3 mul D sub def} ifelse} repeat
BlackFlag
{newpath X Y 1E-4 0 360 arc fill} if} repeat
```

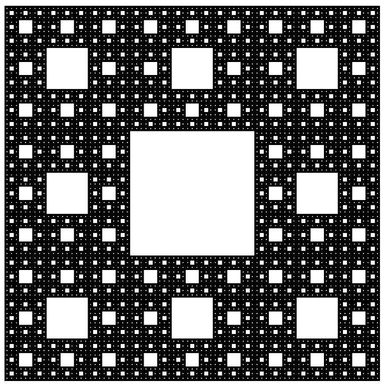


FIG. 2: Sierpiński carpet (fifth iteration).

3. Sierpiński sieve

For instance the Sierpiński Sieve (or Sierpiński triangle curve) is realized by assemblage on the stack of three arrays all elements. The construction algorithm proofs the follow.

Proposition 2 *Sierpiński Sieve contains only tops of umbrellas.*

Executable code for Fig. 3.

```
%!
/n 3 def /Ang 180 n div def

/construct { mark [0 0 2 0 -1 g]
N { counttomark { aload pop /Z exch g def
/Y1 exch def /X1 exch def /Y0 exch def
/X0 exch def /XC X0 X1 add 2 div def
/YC Y0 Y1 add 2 div def
Y0 YC sub  X0 XC sub 2 copy atan
/Phi exch def dup mul exch dup mul add
sqrt /r exch def X0 Y0
1 1 n {/I exch def /Phi1 Phi Ang Z mul add
def newpath XC YC r Phi Phi1 arc
currentpoint 2 copy Z -1 I exp mul 1 mul
3 1 roll 7 2 roll 5 array astore
count 1 roll /Phi Phi1 def} for
        pop pop } repeat
           count 1 roll} repeat
           } def

/draw {translate construct counttomark
{aload pop pop newpath moveto
     lineto stroke } repeat
   pop} def

0 130 translate 50 dup scale

.002 setlinewidth

/g {} def
/N 1 def 4.7 3.5 draw
/N 8 def -1 3.85 draw
/N 2 def -.4 -2 draw
/N 3 def 3 0 draw

/g {neg} def
/N 2 def -1.6 -3.5 draw
/N 3 def 0 -1.8 draw
/N 8 def 0 -2 draw
```

Some program transformations deliver *random Sierpiński sieves.*

Executable code for Fig. 4.

```
%!
/n 3 def /Ang 180 n div def

/vse {counttomark {aload pop /Z exch g def
/Y1 exch def /X1 exch def /Y0 exch def
/X0 exch def /XC X0 X1 add 2 div def
/YC Y0 Y1 add 2 div def
Y0 YC sub  X0 XC sub 2 copy atan
/Phi exch def dup mul exch dup mul add sqrt
/r exch def X0 Y0
1 1 n{/I exch def/Phi1 Phi Ang Z mul add def
newpath XC YC r Phi Phi1 arc currentpoint
2 copy Z -1 I exp mul 1 mul 3 1 roll 7 2 roll
5 array astore count 1 roll
/Phi Phi1 def} for
pop pop}repeat count 1 roll} def /g {} def

/construct { % init_srand
srand mark [0 0 2 0 -1 g] vse
```

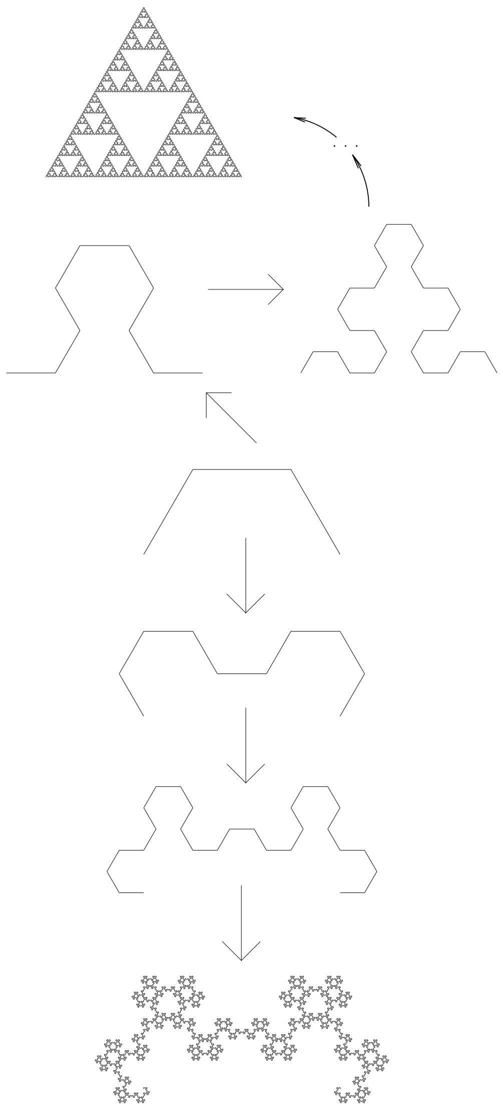


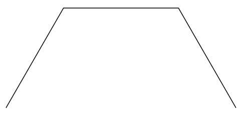
FIG. 3: Regular Sierpiński sieves.

```
N {/V rand 2 31 exp 1 sub div .5 gt {1}{0}
ifelse def /g {V {neg} repeat} def  vse
  } repeat
         } def

/draw { % initsrand X,Y_translate
translate construct counttomark {aload pop
pop newpath moveto lineto stroke} repeat
pop} def

/nshow { % numb -->
1. add exch 2.5 add exch
newpath moveto (n=) show 4 string cvs show} def
```

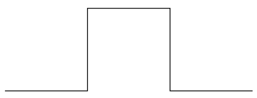
```
50 dup scale .001 setlinewidth

/N 7 def
/Symbol findfont 0.26 scalefont setfont

163 3 2 3 copy nshow draw
167 0 2 3 copy nshow draw
168 0 2 3 copy nshow draw
169 0 2 3 copy nshow draw
170 0 2 3 copy nshow draw
180 0 2 3 copy nshow draw
```

4. Rectangular snowflakes

Let us consider a similar fractal construction. Barely, the trapetional element of the Sierpiński sieves plotting



is exchanged on the square "element" of the *Rectangular Snowflakes*



On the analogy of Sierpiński sieves, the construction algorithm proofs the following

Proposition 3 *Rectangular Snowflakes contain only tops of umbrellas.*

Executable code for Fig. 5.

```
%!
/alop {aload pop} def

/sum { % point0 point1 --> pointsum
alop 3 -1 roll alop 3 -1 roll add
       3 1 roll add exch 2 array astore} def

/vec { % point0 point1 --> vector
exch alop 3 -1 roll alop 3 -1 roll sub
```

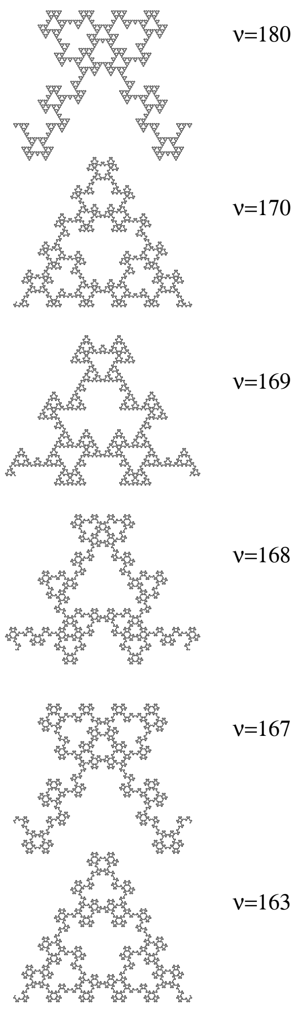


FIG. 4. Random Sierpiński sieves. ν is an "initial srand" parameter.

```
  3 1 roll exch sub exch 2 array astore} def

/kvec { % k vector --> [kx ky]
alop 2 index mul 3 1 roll mul exch
                        2 array astore} def

4 dup scale .01 setlinewidth

/Gr {
N {
  counttomark {alop /Z exch def /P1 exch def
                /P0 exch def /V P0 P1 vec def
               /L V alop abs exch abs add def
                   /VAng V alop exch atan def
```

```
        /l L 3 div def /v 1 3 div V kvec def
                        /CurP P0 v sum def
              [P0 CurP Z ngizn] /P0 CurP def
  P0 alop l VAng dup 90 Z mul add arc
  currentpoint 2 array astore /CurP exch def

  /Tvec P0 CurP vec def

   [P0 P0 Tvec sum Z ng] /P0 P0 Tvec sum def
      [P0 P0 v sum Z ngizn] /P0 P0 v sum def
   [P0 P0 -1 Tvec kvec sum Z ng]
   /P0 P0 -1 Tvec kvec sum def
   [P0 P1 Z ngizn]
   count 5 roll} repeat count 1 roll} repeat

alop pop newpath alop moveto alop lineto
counttomark {alop pop pop alop lineto}repeat
stroke pop} def % Gr

/source {mark [[1 1] [50 1] 1]} def

/ng {neg} def /ngizn {} def
source
/N 1 def 36 1 translate Gr
-36 0 translate
2 1 5 {/N exch def
source 0 32 translate Gr} for

/ng {} def /ngizn {neg} def
70 -128 translate
2 1 5 {/N exch def
source 0 32 translate Gr} for

-11.6 -126 translate
gsave 18 dup scale
/Helvetica findfont 0.2 scalefont setfont
0 1 4 {/N exch def
newpath 0 1.8 N mul moveto
                (n=) show N 3 string cvs show
      } for
grestore
```

In this way one can build many similar examples of sets containing only the tops of umbrellas.

5. Irreducible continua

On the other part, the algorithmic realizations problem of irreducible continua being Cantor set and interval $[0, 1]$ direct product is solved as the following two "children" records is replaced by one "mother" record.

Executable code for Fig. 6.

```
%!
/Eps .01 def
/daught0 {aload /Mom exch def 1 sub abs 3 1
roll 3 div dup 4 1 roll 2 mul sub 3 1 roll
                        3 array astore} def
/daught1 {aload /Mom exch def 1 sub abs 3 1
roll 3 div dup 4 1 roll 4 mul add 3 1 roll
                        3 array astore} def
/building {aload /Mom exch def /Y exch def
/BS exch def /BC exch def
newpath BC 0 moveto BC 1 lineto stroke
newpath BC BS add 0 moveto BC BS add 1
lineto stroke newpath BC Y moveto BC BS add
                       Y lineto stroke} def
/Mom 3 array def
/P {/Mom exch def Mom building Mom  daught0
dup 1 get Eps lt {pop} if Mom daught1 dup 1
                   get Eps lt {pop} if} def
150 dup scale 1 1 translate
.003 setlinewidth
[1 3 div dup 0]
{count 0 ne {P}{exit} ifelse} loop
```

If α and β are closed sets without common points and the continuum C contains one or more points of both sets, then C is irreducible between α and β if no proper sub-continuum of C contains points of both sets [5].

Executable code for Fig. 7.

```
%!
/Pick 1 def

150 dup scale 2 2 translate
.001 setlinewidth

/takecoords {aload pop /B1 exch def
/B0 exch def /T1 exch def /T0 exch def} def

/draw { newpath T0 Pick moveto B0 0 lineto
stroke newpath T1 Pick moveto B1 0 lineto
stroke}def

/build {[T0  T1 T0 sub 4 div T0 add B0 B1
B0 sub 4 div B0 add]
```

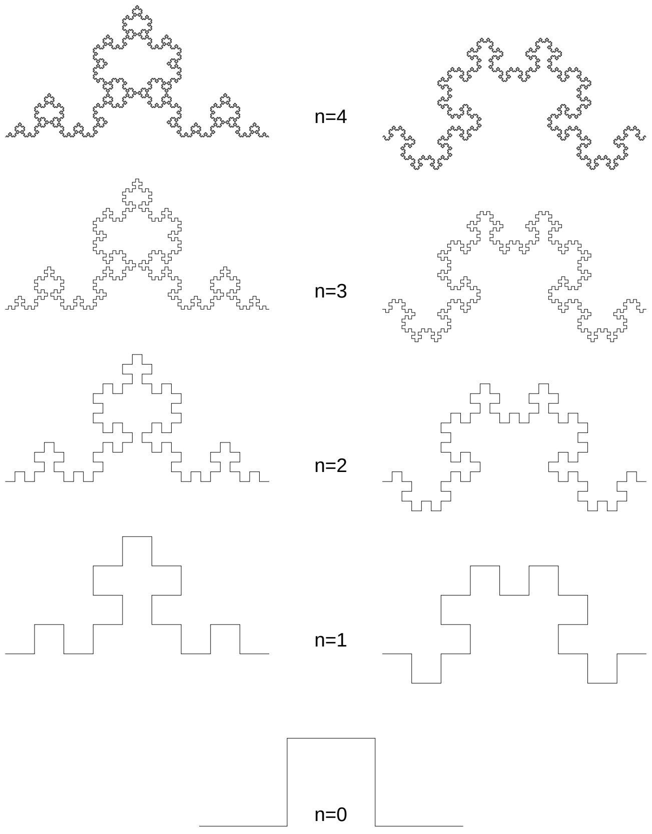


FIG. 5: Rectangular Snowflakes.

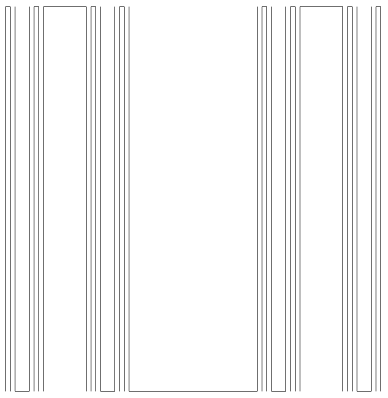


FIG. 6. The irreducible continuum example (from [5], Vol. II, fig. 3 left).

```
[ T1 T0 sub 4 div T0 add  T1 T0 sub 2 div
T0 add  B1 B0 sub 2 div B0 add  B1 B1 B0
sub 4 div sub]
[ T1 T1 T0 sub 4 div sub  T1  B1 B1 B0 sub
                         4 div sub  B1 ]} def

[0 1 0 1] mark 4{{count -1 roll dup mark ne
{takecoords draw build}{exit} ifelse} loop}
repeat clear
```

Executable code for Fig. 8.

```
%!
/Eps .003 def
/daught0{aload/Mom exch def 1 sub abs 3 1
roll 3 div dup 4 1 roll 2 mul sub 3 1 roll
                        3 array astore} def
/daught1{aload/Mom exch def 1 sub abs 3 1
roll 3 div dup 4 1 roll 4 mul add 3 1 roll
                        3 array astore} def
/building {aload /Mom exch def /Y exch def
/BS exch def /BC exch def newpath BC 0
moveto BC 1 lineto stroke newpath BC BS
add 0 moveto BC BS add 1 lineto stroke
newpath BC Y moveto BC BS add Y lineto
                              stroke} def
```

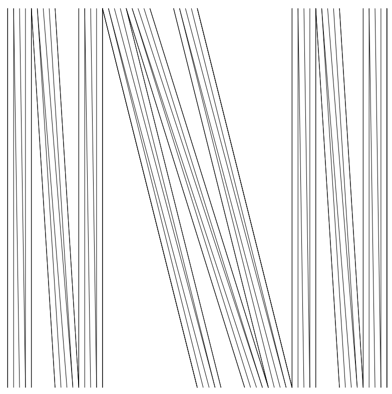


FIG. 7. The irreducible continuum example (from [5], Vol. II, fig. 3 right).

```
/psi{ % coord shift spin
aload /Mom exch def /SP exch def
/SH exch def /CRD exch def SP 0 eq
{/pasta{}def}{/pasta {neg 90 add}def}
ifelse newpath CRD SP moveto
CRD .00001 add .00001 CRD SH add
{dup CRD SH 2 div add sub 2 SH div mul dup
5 exp 1 add exch 5 exp 1 exch sub atan
pasta exec 90 div lineto} for stroke} def
/Mom 3 array def /P {/Mom exch def Mom
%building
psi Mom daught0 dup 1 get Eps lt {pop} if
Mom daught1 dup 1 get Eps lt {pop} if} def
150 dup scale 1 1 translate
.001 setlinewidth
[1 3 div dup 0] {count 0 ne
{P}{exit} ifelse} loop
```

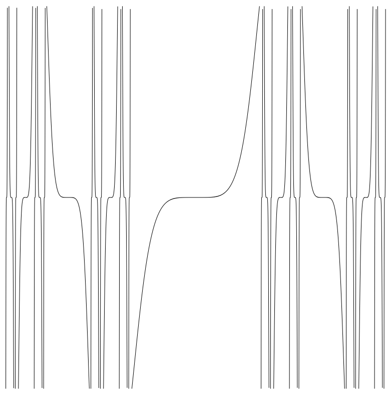


FIG. 8: The irreducible tan-continuum example.

6. Knaster indecomposable continua

Knaster proposed the indecomposable continua prime examples (see i. g. [5], vol. II).

Executable code for Fig. 9.

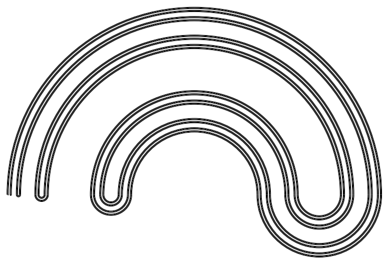


FIG. 9: Knaster bucket handle.

There exists another faster topological algorithm for Knaster bucket handle:

```
%!
.1 dup scale 540 1250 translate
.01 setlinewidth
/W{/B A def/C A def}def

/semiarcs {0 1 C {/R exch def
/S R B 2 div add def /A B def
/A B 3#10 idiv def/O R A idiv def O 1 ne
{{/R R O A mul sub def/A A 3#10 idiv def
/O R A idiv def A 1 eq O 1 eq or{exit}if
 } loop} if

O 1 ne{C D .5 add mul O S F newpath
arc stroke} if} for} def

/A 3#10000000 def W /F {0 180} def
/D 1 def semiarcs
/D 7 def /F {180 360} def
1 1 5 {/I exch def /A 3#10000000 def
/A A 3 I exp cvi idiv def W semiarcs}for
```

which leads to more simple result.

As a final point, we mention that topological algorithms also allow the construction of one another *Knaster bucket poker* example [5].

Executable code for Fig. 10.

```
%!
100 100 translate .018 dup scale
.000001 setlinewidth

/semiarcs {0 1 C {/R exch def
/S R .5 C mul add def /A B def
/A B 5 idiv def /O R A idiv def
O 1 ne O 3 ne and {{/R R O A mul sub def
/A A 5 idiv def /O R A idiv def
A 1 eq O 1 eq or O 3 eq or {exit} if}
loop} if

O 1 ne O 3 ne and
{C D mul 0 S F newpath arc stroke} if
      } for} def

0 1 3 {/I exch def /A 3125 def
/A A 5 I exp cvi idiv def
/B A def /C A def
/D 3.5 def /F {180 360} def semiarcs
/D 5 I 1 add exp 3.5 sub def
/F {0 180} def semiarcs} for
```

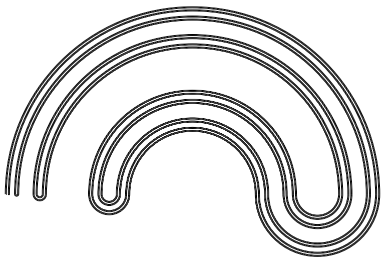


FIG. 10: Knaster bucket poker.

7. Conclusion

In the paper the PostScript visualization algorithms for some concrete "bad" topologic situations due to the tops of umbrellas presence and/or irreducible points have been developed, namely:

1° an algorithm based on the points screening from the symbol "1" by the Monte-Carlo method on the Sierpiński carpet is dynamically realized;

2° regular and random fractal sets upon the Sierpiński carpets and rectangular snowflakes are constructed by the "method is broken";

3° irreducible continua realizations on examples from [5] are constructed;

4° Knaster indecomposable continua examples (Knaster bucket handle and bucket poker) [5] are constructed in two forms, both in metric and topologic (or symbolic) sense.

It has been shown that application of the PostScript language for visualization of some nonlinear dynamics problems could be very convenient.

References

[1] *PostScript Language Reference.* 3rd ed. (Addison-Wesley Publishing Company, 1999).

[2] B. Casselman. *Mathematical Illustrations: A Manual of Geometry and PostScript.* (Cambridge University Press, 2005).

[3] G. C. Reid. *Thinking in PostScript.* (Addison-Wesley Publishing Company, 1990).

[4] K. Borsuk. *Theory of Retracts.* (PWN, Warszawa, 1967).

[5] K. Kuratowski. *Topology,* vol. I, II, (Academic Press, New York & London, 1966).