RauzyFractalRandomSubstitution

October 31, 2024

[58]:
```
# import tools for random substitutions
from random_substitution import *
```

[61]:
```
# Random Tribonnacci
rs = RandomSubstitution({'a':['ab', 'ba'], 'b':['ac'], 'c':['a']})
rs
```

[61]:
```
{'a': [word: ab, word: ba], 'b': [word: ac], 'c': [word: a]}
```

[62]:
```
rs.is_compatible()
```

[62]:
```
True
```

[63]:
```
rs.incidence_matrix()
```

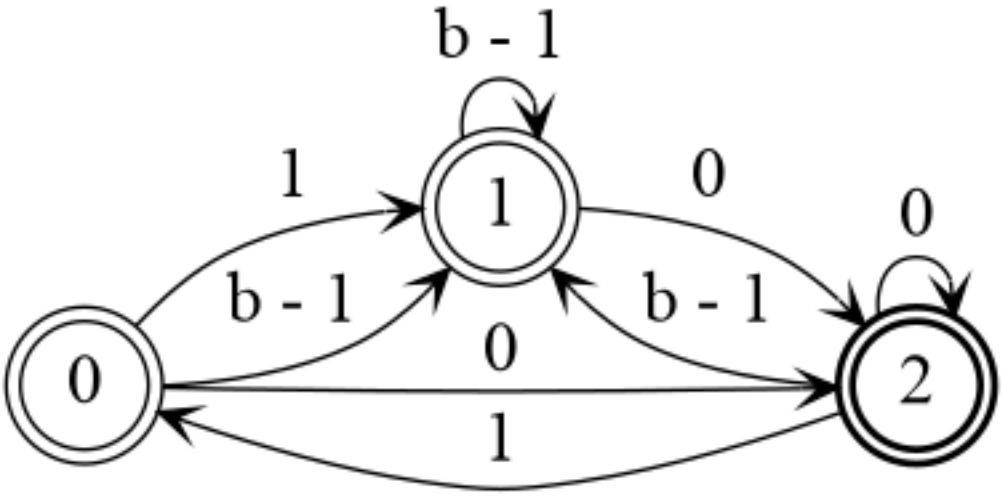
[63]:
```
[1 1 1]
[1 0 0]
[0 1 0]
```

[64]:
```
# BetaAdicSet describing the Rauzy fractal of the random substitution
m = rs.rauzy_fractal()
m
```

[64]:
```
b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 3 states and 3
letters
```

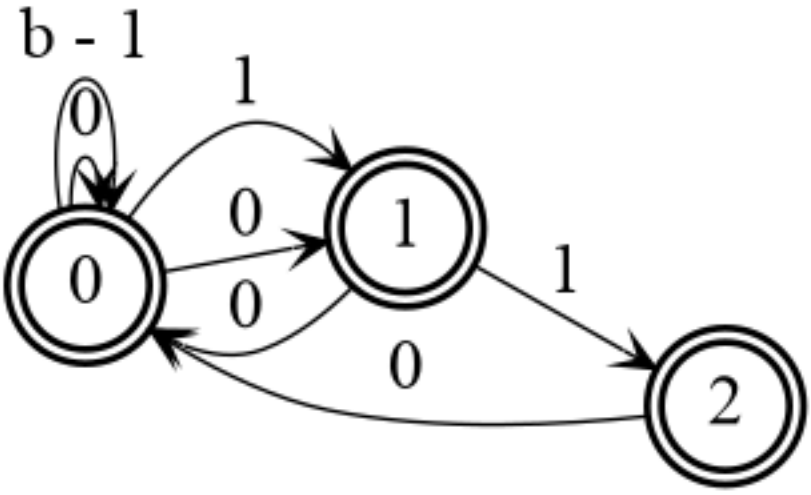
[65]:
```
# automaton describing this BetaAdicSet
m.a.plot()
```

[65]:



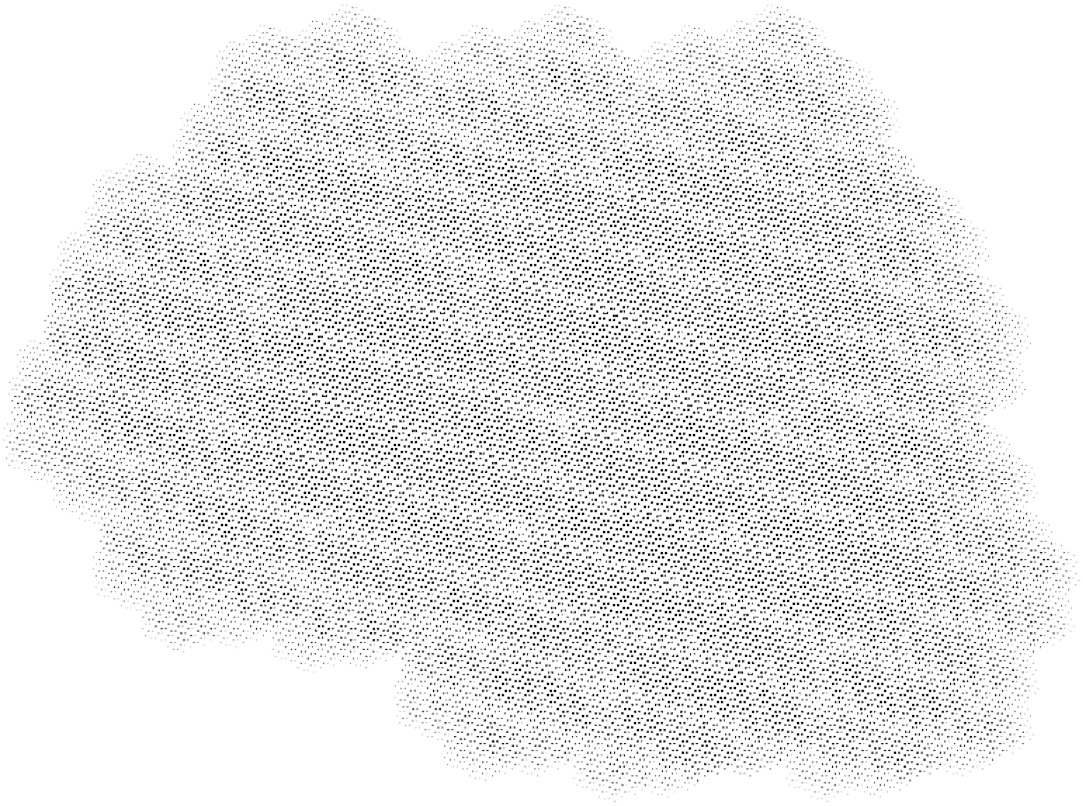
[66]:
```
# plot the (projected) prefix automaton
rs.rauzy_fractal(prefix_aut=True).plot()
```

[66]:



[8]:
```
# the Rauzy fractal doesn't display well because many points have many␣
 ↪differents representations
m.plot()
```

[8]:



[9]:
```
# compute a unique representation for points of the set
mr = m.reduced()
mr
```

[9]:
```
b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 106 states and
3 letters
```

[10]:
```
# it permits to get a nicer plot
mr.plot()
```

[10]:

[12]:
```
# compute a substitution whose Rauzy fractal is this set
s = m.substitution(verb=1)
s
```

```
pl = b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 3 states
and 2 letters
self = b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 51
states and 2 letters
k = 1

*** step 1 ***
Exchange with 7 pieces
12 new pieces: computation to continue

*** step 2 ***
Exchange with 19 pieces
12 new pieces: computation to continue

*** step 3 ***
Exchange with 31 pieces
6 new pieces: computation to continue
```

```
*** step 4 ***
Exchange with 37 pieces
2 new pieces: computation to continue

*** step 5 ***
Exchange with 39 pieces
finished !
```

[12]:
```
WordMorphism: A->lK, B->nI, C->zck, D->Hj, E->Jk, F->M, G->L, H->ofqdt, I->rzcp,
J->oAcp, K->qJp, L->ofqdter, M->phqJp, a->r, b->q, c->p, d->o, e->ck, f->lb,
g->na, h->x, i->w, j->er, k->hq, l->uh, m->vi, n->ue, o->G, p->F, q->E, r->D,
s->C, t->A, u->z, v->y, w->ter, x->phq, y->mgsd, z->mBc
```

[15]:
```
s.is_primitive()
```

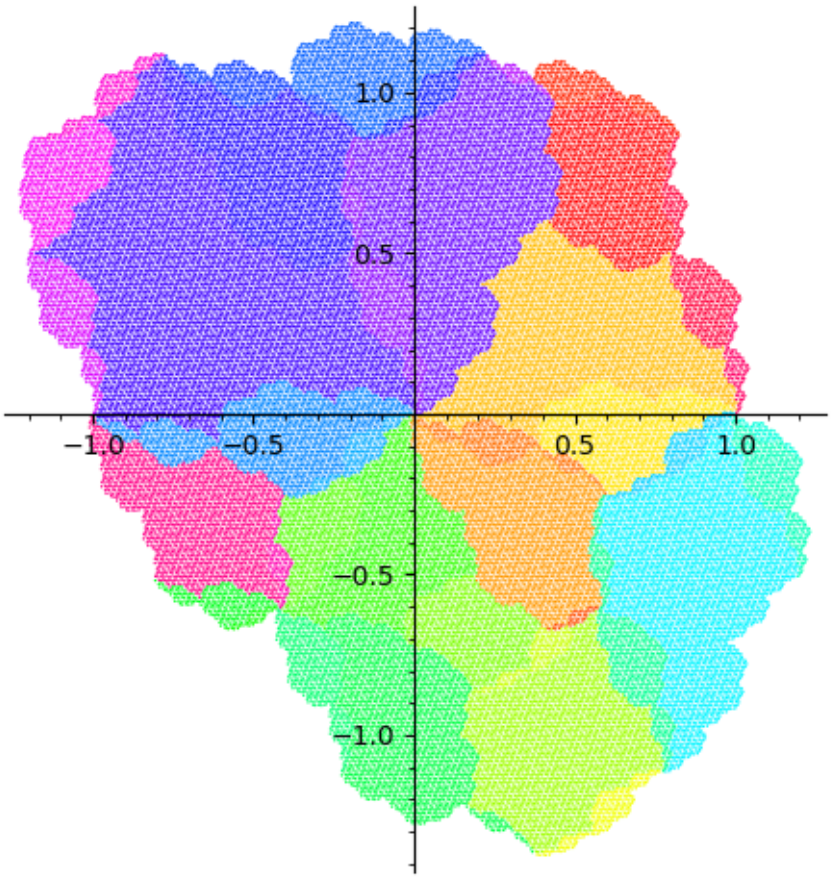
[15]:
```
True
```

[16]:
```
# number of letters of the substitution
len(s.domain().alphabet())
```

[16]:
```
39
```

[17]:
```
# plot the Rauzy fractal with colors
s.rauzy_fractal_plot()
```

[17]:



[18]:
```
# characteristic polynomial
m = s.incidence_matrix()
pi = m.charpoly()
pi.factor()
```

[18]:
```
(x - 1)^3 * x^21 * (x^2 + x + 1)^4 * (x^3 - x^2 - x - 1) * (x^4 - 2*x - 1)
```

[21]:
```
# roots of each factor
for p,_ in pi.factor():
    print(p)
    print([r[0] for r in p.roots(ring=QQbar)])
```

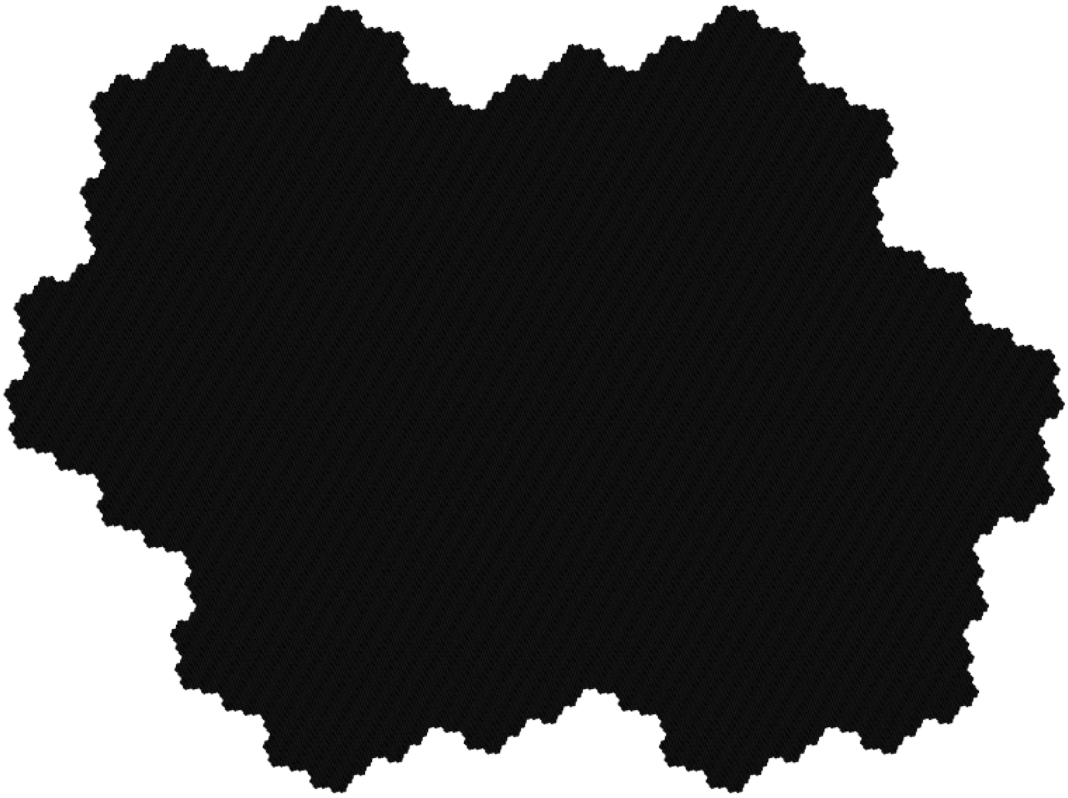
```
x - 1
[1]
x
[0]
x^2 + x + 1
[-0.50000000000000000? - 0.866025403784439?*I, -0.50000000000000000? +
0.866025403784439?*I]
x^3 - x^2 - x - 1
```

```
[1.839286755214161?, -0.4196433776070806? - 0.6062907292071993?*I,
-0.4196433776070806? + 0.6062907292071993?*I]
x^4 - 2*x - 1
[-0.4746266175626056?, 1.395336994467073?, -0.4603551884522338? -
1.139317680301923?*I, -0.4603551884522338? + 1.139317680301923?*I]
```

[22]:
```
# usual Tribonnacci
s = WordMorphism('a->ab,b->ac,c->a')
```

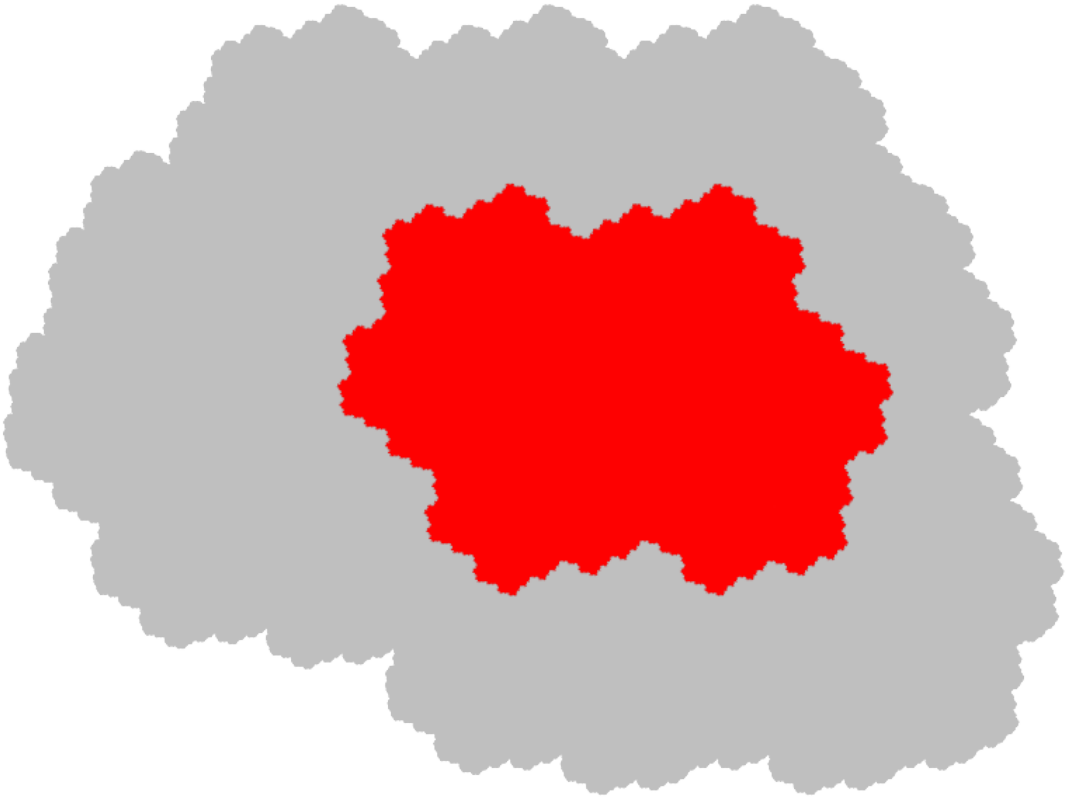
[24]:
```
# BetaAdicSet describing the Rauzy fractal
ms = DumontThomas(s).mirror()
ms.plot()
```

[24]:



[25]:
```
# plot it inside the Rauzy fractal of random Tribonnacci
mr.plot_list([ms.proj(mr)])
```

[25]:



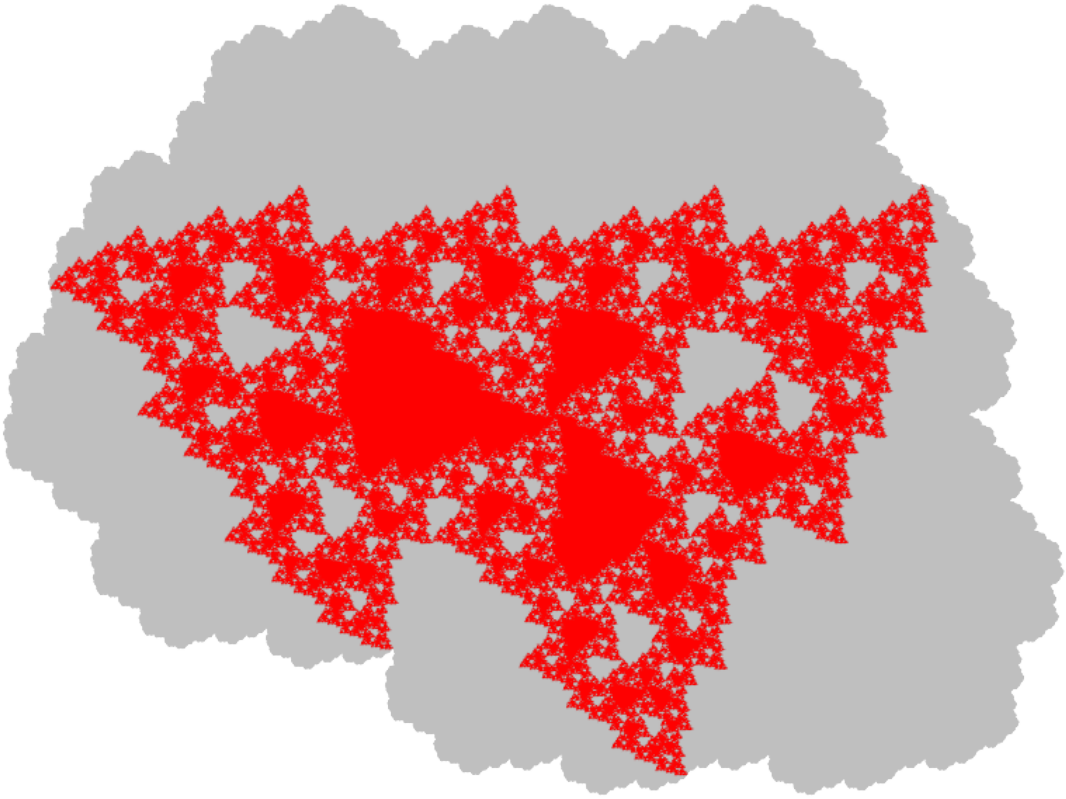
[28]:
```
# flipped Tribonnacci
t = WordMorphism('a->ba,b->ac,c->a')
```

[29]:
```
# BetaAdicSet describing the Rauzy fractal
mt = DumontThomas(t).mirror()
mt.plot()
```

[29]:

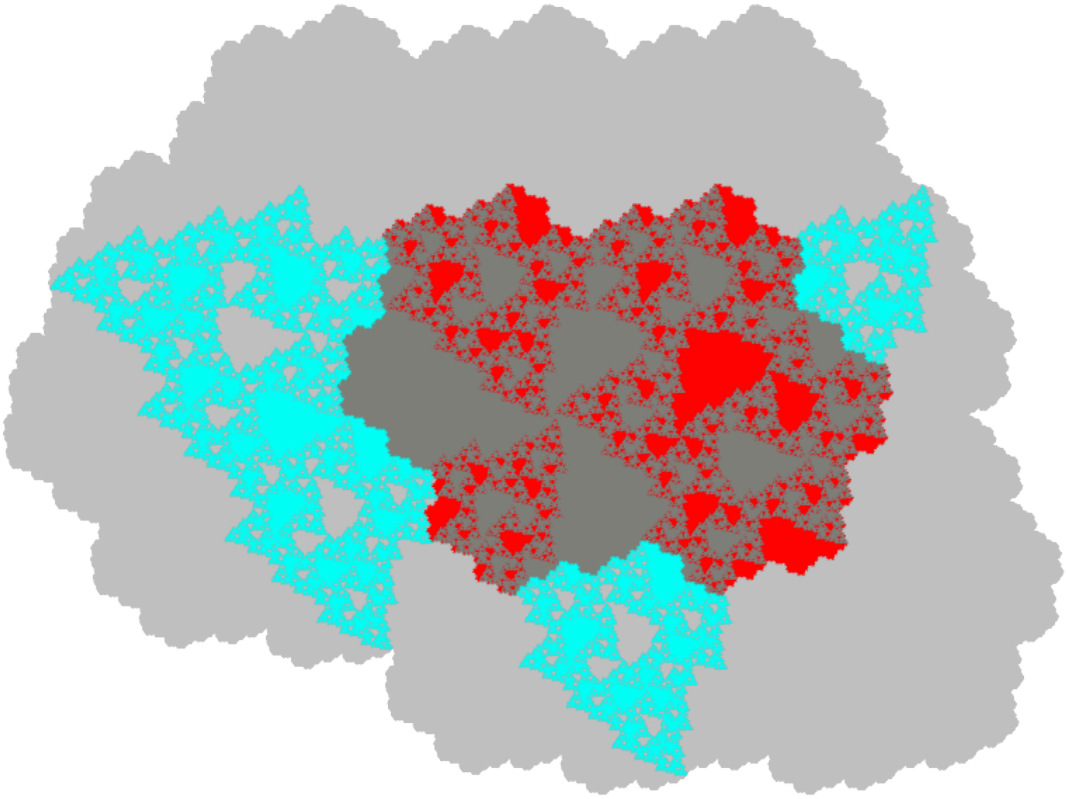
[30]:
```
# plot it inside the Rauzy fractal of random Tribonnacci
mr.plot_list([mt.proj(mr)])
```

[30]:



[31]:
```
# plot both
mr.plot_list([ms.proj(mr), mt.proj(mr)])
```

[31]:



[ ]:

[35]:
```
# variant of Random Tribonnacci
rs = RandomSubstitution({'a':['ab'], 'b':['ac', 'ca'], 'c':['a']})
rs
```

[35]:
```
{'a': [word: ab], 'b': [word: ac, word: ca], 'c': [word: a]}
```

[36]:
```
m = rs.rauzy_fractal()
m
```

[36]:
```
b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 4 states and 3
letters
```

[37]:
```
mr = m.reduced()
mr.plot()
```

[37]:

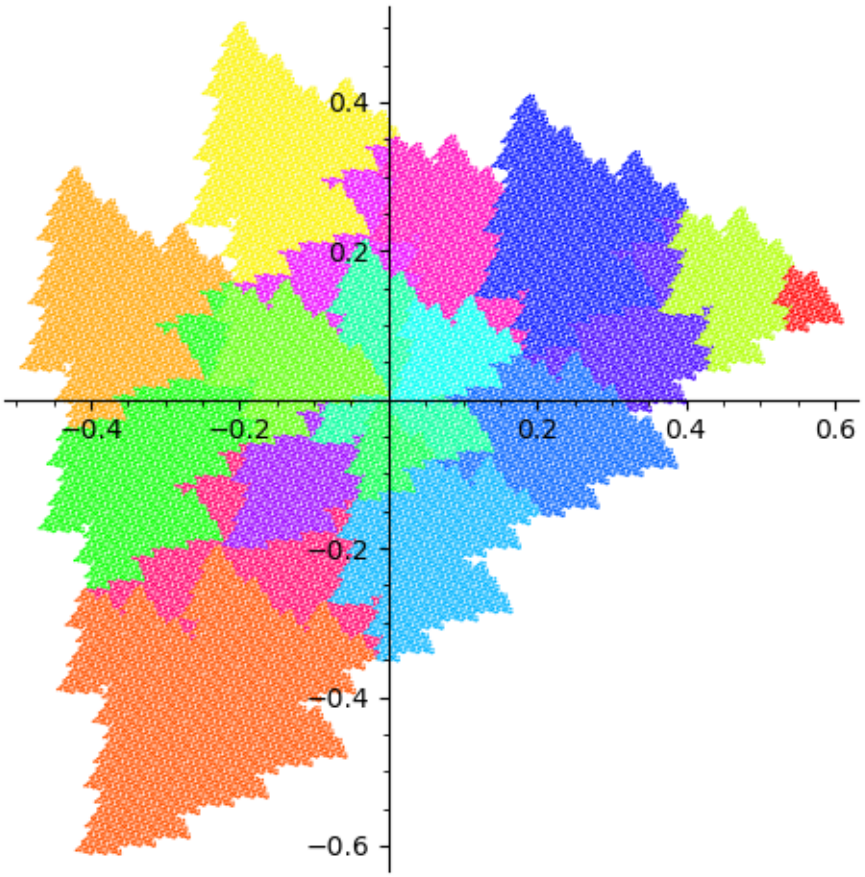
[38]:
```
s = m.substitution(verb=1)
s
```

```
pl = b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 3 states
and 2 letters
self = b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 54
states and 2 letters
k = 1

*** step 1 ***
Exchange with 8 pieces
7 new pieces: computation to continue

*** step 2 ***
Exchange with 15 pieces
3 new pieces: computation to continue

*** step 3 ***
Exchange with 18 pieces
finished !
```

[38]:
```
WordMorphism: a->b, b->d, c->m, d->l, e->mb, f->cj, g->q, h->bcaf, i->bci,
j->bch, k->bef, l->bo, m->gk, n->r, o->mbci, p->ngl, q->ngk, r->rp
```

[39]:
```
s.rauzy_fractal_plot()
```

[39]:



[40]:
```
s.is_primitive()
```

[40]:
```
True
```

[41]:
```
len(s.domain().alphabet())
```

[41]:
```
18
```

[42]:
```
m = s.incidence_matrix()
pi = m.charpoly()
pi.factor()
```

[42]:
```
(x - 1) * x^9 * (x^2 + x + 1) * (x^3 - x^2 - x - 1) * (x^3 - x^2 - 1)
```

[43]:
```
for p,_ in pi.factor():
    print(p)
    print([r[0] for r in p.roots(ring=QQbar)])
```

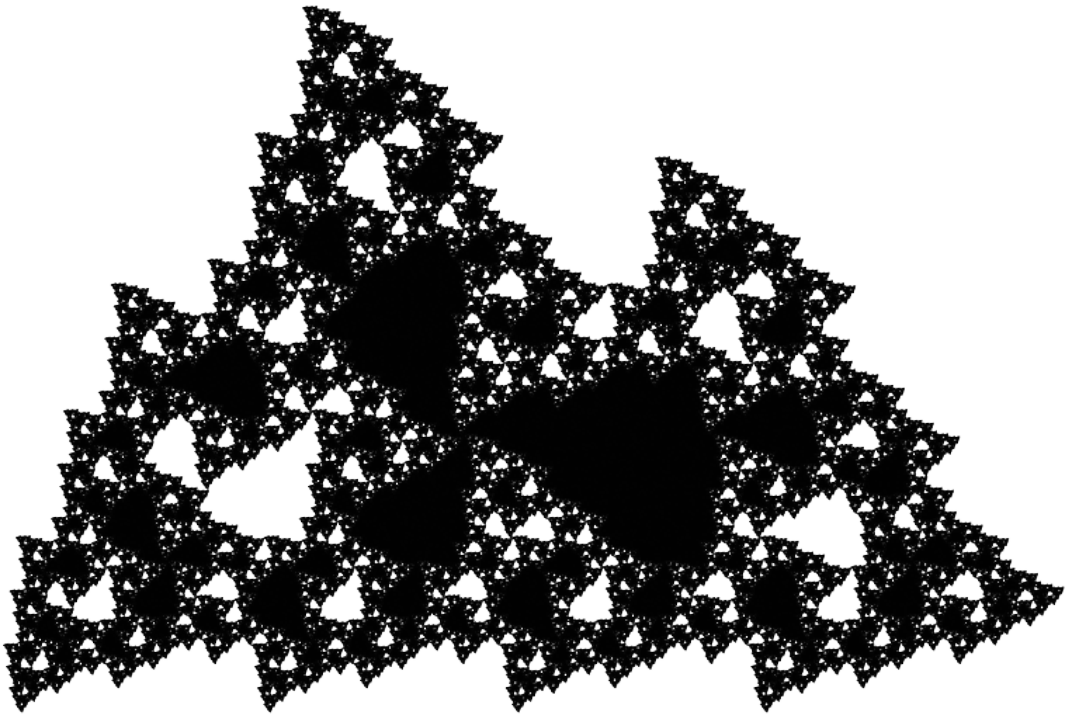
```
x - 1
[1]
x
[0]
x^2 + x + 1
[-0.50000000000000000? - 0.866025403784439?*I, -0.50000000000000000? +
0.866025403784439?*I]
x^3 - x^2 - x - 1
[1.839286755214161?, -0.4196433776070806? - 0.6062907292071993?*I,
-0.4196433776070806? + 0.6062907292071993?*I]
x^3 - x^2 - 1
[1.465571231876768?, -0.2327856159383841? - 0.7925519925154479?*I,
-0.2327856159383841? + 0.7925519925154479?*I]
```

[46]:
```
t = WordMorphism('a->ab,b->ca,c->a')
t
```

[46]:
```
WordMorphism: a->ab, b->ca, c->a
```

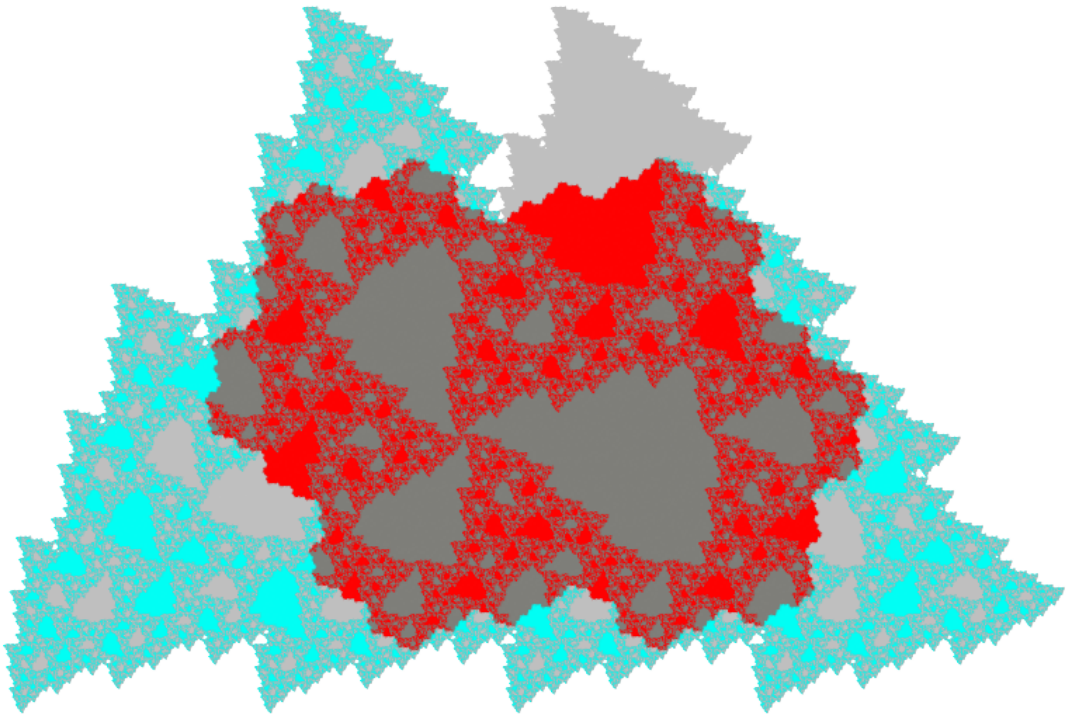
[47]:
```
mt = DumontThomas(t).mirror()
mt.plot()
```

[47]:



[48]:
```
# plot Rauzy fractal of s and t inside the Rauzy fractal of the random␣
↪substitution
mr.plot_list([ms.proj(mr), mt.proj(mr)])
```

[48]:



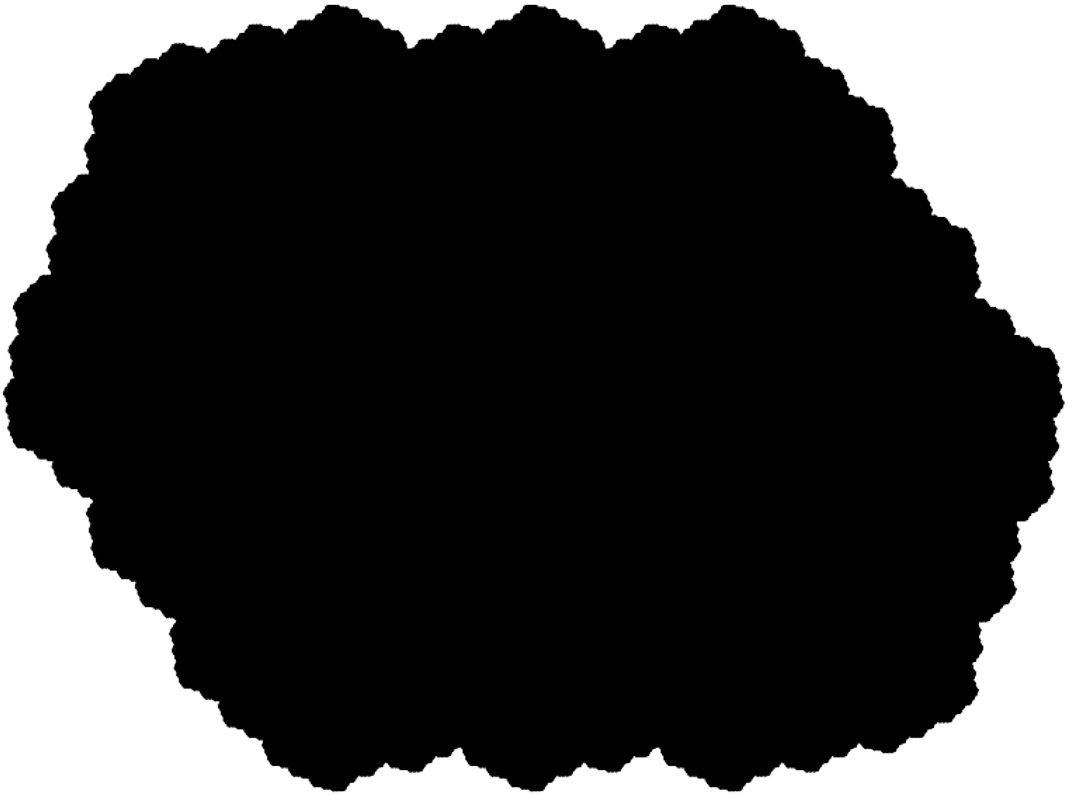
[ ]:

[50]:
```
# another variant of Random Tribonnacci
rs = RandomSubstitution({'a':['ab', 'ba'], 'b':['ac', 'ca'], 'c':['a']})
m = rs.rauzy_fractal()
m
```

[50]:
```
b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 4 states and 4
letters
```

[51]:
```
mr = m.reduced()
mr.plot()
```

[51]:



[52]:
```
s = m.substitution(verb=1)
s
```

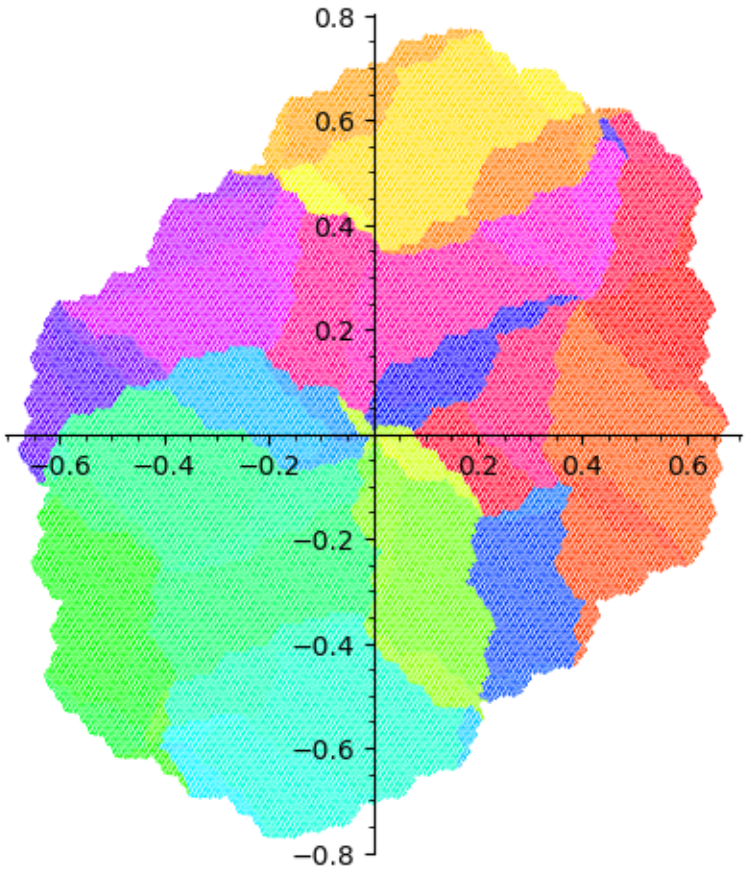
```
pl = b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 3 states
and 2 letters
self = b-adic set with b root of x^3 - x^2 - x - 1, and an automaton of 96
states and 2 letters
k = 1

*** step 1 ***
Exchange with 7 pieces
17 new pieces: computation to continue

*** step 2 ***
Exchange with 24 pieces
30 new pieces: computation to continue

*** step 3 ***
Exchange with 54 pieces
18 new pieces: computation to continue
```

```
*** step 4 ***
Exchange with 72 pieces
2 new pieces: computation to continue

*** step 5 ***
Exchange with 74 pieces
finished !
```

[52]:
```
WordMorphism: 0->35, 1->34, 10->33,4, 11->47, 12->46, 13->45, 14->7,44,
15->6,42, 16->5,17, 17->7,17, 18->8,41, 19->12,43, 2->32, 20->11,41, 21->12,16,
22->23,13, 23->24,10, 24->24,9, 25->36,11, 26->38,13, 27->37,8, 28->39,9,
29->36,6, 3->31, 30->68, 31->67, 32->66, 33->64, 34->62, 35->61, 36->60, 37->59,
38->58, 39->57, 4->30, 40->56, 41->51, 42->50, 43->49, 44->48, 45->35,6,32,
46->34,8,31, 47->35,11,31, 48->1,18,7,17, 49->4,40,7,21, 5->4,40, 50->1,52,14,
51->4,63,19, 52->8,41,1,18, 53->29,2,36,6, 54->25,3,67, 55->29,2,64,
56->25,3,51, 57->24,9,29,2, 58->22,9,33,4, 59->28,53,2, 6->0,20, 60->26,55,4,
61->25,70, 62->27,69, 63->56,1,18, 64->60,0,20, 65->61,0,20, 66->71,15,
67->72,20, 68->73, 69->31,4,35,6,32, 7->1,18, 70->31,4,65,30, 71->34,8,31,4,35,
72->30,54,4,35, 73->34,8,31,4,35,6,32, 8->25,3, 9->29,2
```

[53]:
```
s.rauzy_fractal_plot()
```

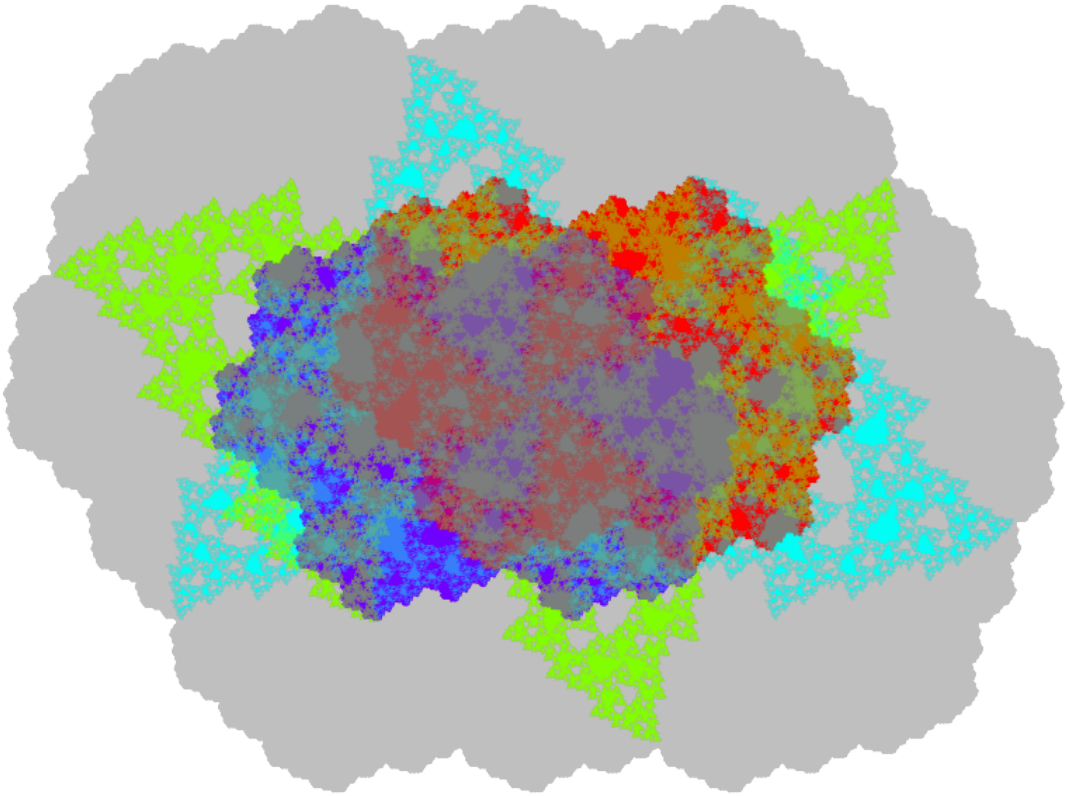
[53]:



[56]:
```
s = WordMorphism('a->ab,b->ac,c->a')
t = WordMorphism('a->ba,b->ac,c->a')
u = WordMorphism('a->ab,b->ca,c->a')
v = WordMorphism('a->ba,b->ca,c->a')
ms = DumontThomas(s).mirror()
mt = DumontThomas(t).mirror()
mu = DumontThomas(u).mirror()
mv = DumontThomas(v).mirror()
```

[57]:
```
# plot Rauzy fractal of s, t, u and v inside the Rauzy fractal of the random␣
↪substitution
mr.plot_list([ms.proj(mr), mt.proj(mr), mu.proj(mr), mv.proj(mr)])
```

[57]:



[ ]: