

In [1]:

```
##### This is some SAGEmath code used during the preparation of the paper
##### "A brute force computer aided proof of an existence result about ex
tremal hyperbolic surfaces"
##### by Ernesto Girondo and Cristian Reyes

##### THE CASE N=14
```

In [2]:

```
##### CELL 1: SOME FUNCTIONS
```

```
##### FUNCTION poli2pi3. Edges, vertices and midpoints of a regular polyg
on of n edges
def poli2pi3(n):
    u'''This is a function of an integer n. The output is a triplet consisting
of the n edges, the n vertices
    and the n edge midpoints of a regular hyperbolic polygon of angle 2Pi/3 ce
ntered at the origin. Points and vertices
    are listed in counterclockwise order. The first midpoint, which lies in th
e negative imaginary axis,
    corresponds to the last edge.
    '''
    H=N(arccosh(1/(tan((pi)/n)*tan((pi)/3))))
    h=N(tanh(H/2))
    R=N(arccosh((cos((pi)/3))/(sin((pi/n)))))
    r=N(tanh(R/2))
    puntosmedios=[N(r*(cos(3*(pi)/2+2*k*pi/n))+I*r*(sin(3*(pi)/2+2*k*pi/n))) f
or k in [0..n-1]
    vertices=[N(h*(cos(3*(pi)/2+(1+2*k)*pi/n))+h*(sin(3*(pi)/2+(1+2*k)*pi/n))*
I) for k in [0..n]]
    lados = [PD.get_geodesic(vertices[k], vertices[k+1]) for k in [0..n-1]]
    return lados, vertices, puntosmedios
####
```

```
##### FUNCTION moverpol returns a plot of the image of the list of edges "la
dos", which is a global variable
def moverpol(x, col):
    u'''This is a function of a transformation x and a color col. The output i
s a plot in color col of the image of the
    set "lados", which is a global variable.
    '''
    global lados
    bpol=Graphics()
    for j in [0..M-1]:
        movido=x*lados[j]
        bpol+=movido.show(color=col)
    return bpol
####
```

In [3]:

```
##### CELL2: SOME CODE DEFINING THE GENERATORS OF A TRIANGLE GROUP (2,3,M)
) and related things

PD=HyperbolicPlane().PD() #The Disc model of the hyperbolic plane

M=14 #The number of edges of the polygons. This is called N in the paper

lados, vertices, puntosmedios = poli2pi3(M) #

#Now computing the points A,B,C and the three generators a,b,c of the extended
triangle group (2,3,M)
H=N(arccosh(1/(tan((pi)/M)*tan((pi)/3))))
h=N(tanh(H/2))
R=N(arccosh((cos((pi)/3))/(sin((pi)/M))))
r=N(tanh(R/2))
LM=N(arccosh((cos((pi)/M))/(sin((pi)/3))))
L=2*LM
l=N(tanh(L/2))

B=PD.get_point(0+0*I)
C=PD.get_point(N(r*cos((pi)*3/2)+r*sin((pi)*3/2)*I))
A=PD.get_point(N(h*cos((pi)/M+3*(pi)/2)+h*sin((pi)/M+3*(pi)/2)*I))
lado_c, lado_b, lado_a =PD.get_geodesic(A,B),PD.get_geodesic(A,C), PD.get_geod
esic(B,C)
a,b,c=lado_a.reflection_involution(), lado_b.reflection_involution(), lado_c.r
eflection_involution()
```

In [4]:

```
##### CELL3: OTHER USEFUL FUNCTIONS

def Rota(t):
    return (c*a)^t*a*b

def rotasionpi(p,q): #THE ORDER 2 ELLIPTIC ELEMENT PERMUTING TWO GIVEN POINTS
p AND q
    ek1=PD.get_point(p)
    ek2=PD.get_point(q)
    geod=PD.get_geodesic(ek1,ek2)
    ek3=geod.midpoint().coordinates()
    ma1=matrix([[I,(ek3)*(-I)],[((ek3.conjugate())*(I),-I)]])
    Rot=PD.get_isometry(ma1)
    rotpii=matrix([[I,0],[0,-I]])
    rotpi=PD.get_isometry(rotpii)
    fuu=Rot*rotpi*Rot^-1
    return fuu

def G0(i):
    return a*b*(c*a)^i*a*b
def G1(i):
    return (c*a)*a*b*(c*a)^i*a*b*(c*a)^-1
def G2(i):
    return (c*a)^2*a*b*(c*a)^i*a*b*(c*a)^-2
def G3(i):
    return (c*a)^3*a*b*(c*a)^i*a*b*(c*a)^-3
def G4(i):
    return (c*a)^4*a*b*(c*a)^i*a*b*(c*a)^-4
def G5(i):
    return (c*a)^5*a*b*(c*a)^i*a*b*(c*a)^-5
```

In [5]:

```
##### CELL4: plotdom IS THE FUNDAMENTAL DOMAIN F DESCRIBED IN THE TEXT AND sidepairings ARE THE SET
##### OF SIDE PAIRING TRANSFORMATIONS GENERATING THE GROUP K

plotdom=moverpol(a*a^-1,'black')
for k in [0..1]:
    plotdom+=moverpol((c*a)^k*a*b*(c*a)^-k,'black')

show(plotdom)

sidepairings=['comodin']
sidepairingscode=['cero']

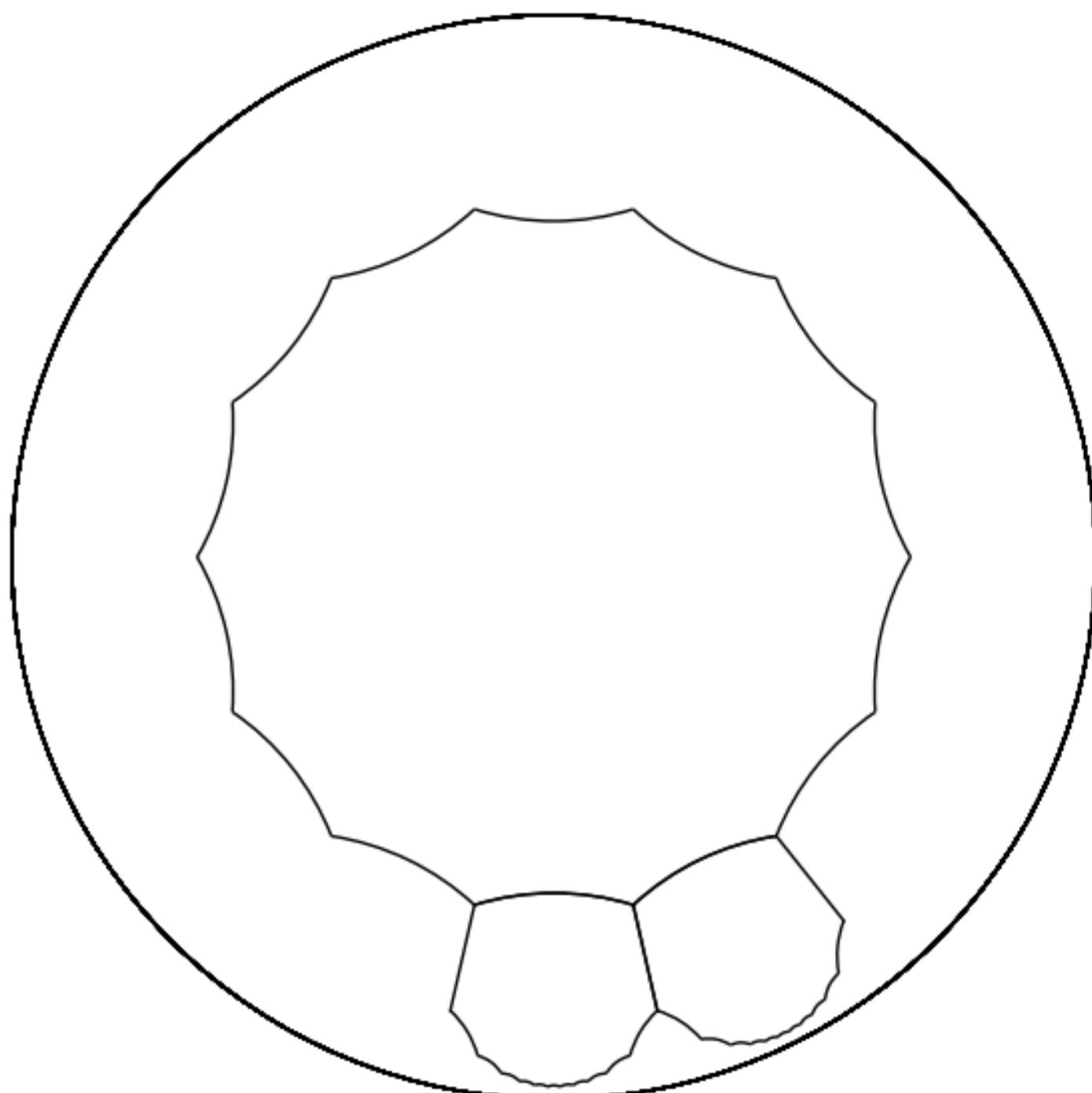
sidepairings.append((c*a)^-4*b*(c*a)) #1 Side pairing (-, 1_{13}, 1_{10})
sidepairingscode.append('(-, 1_{13}, 1_{10})')
sidepairings.append(G0(7)*(c*a)^-2) #2 Side pairing (+, 1_2, 2_7)
sidepairingscode.append('(+, 1_2, 2_7)')
sidepairings.append((c*a)^-3*a*G0(-1)) #3 Side pairing (-, 2_1, 1_{11})
```

```

sidepairingscode.append('(-, 2_1, 1_{11}))')
sidepairings.append(G0(9)*G1(2)) #4 Side pairing (+
, 3_{13}, 2_8)
sidepairingscode.append('(+, 3_{13}, 2_8)')
sidepairings.append((c*a)^7*a*b*(c*a)^2) #5 Side pairing (+
, 1_{12}, 1_7)
sidepairingscode.append('(+, 1_{12}, 1_7)')
sidepairings.append(G0(2)*a*(c*a)^6) #6 Side pairing (-
, 1_8, 2_2)
sidepairingscode.append('(-, 1_8, 2_2)')
sidepairings.append(G0(3)*(c*a)^9) #7 Side pairing (+
, 1_5, 2_3)
sidepairingscode.append('(+, 1_5, 2_3)')
sidepairings.append((c*a)^6*b*(c*a)^5) #8 Side pairing (-
, 1_9, 1_6)
sidepairingscode.append('(-, 1_9, 1_6)')
sidepairings.append(G1(5)*(c*a)^{-3}) #9 Side pairing (+
, 1_4, 3_5)
sidepairingscode.append('(+, 1_4, 3_5)')
sidepairings.append(G0(-4)*(c*a)^{-3}) #10 Side pairing (
+, 1_3, 2_{10})
sidepairingscode.append('(+, 1_3, 2_{10})')
sidepairings.append(G1(5)*G0(4)) #11 Side pairing (
+, 2_9, 3_6)
sidepairingscode.append('(+, 2_9, 3_6)')
sidepairings.append(G1(7)*(c*a)*a*b*(c*a)^{-1}*G1(2)) #12 Side pairing (
+, 3_{12}, 3_7)
sidepairingscode.append('(+, 3_{12}, 3_7)')
sidepairings.append(G0(-3)*a*b*G0(-6)) #13 Side pairing (
+, 2_6, 2_{11})
sidepairingscode.append('(+, 2_6, 2_{11})')
sidepairings.append(G1(3)*G0(-5)) #14 Side pairing (
+, 2_4, 3_4)
sidepairingscode.append('(+, 2_4, 3_4)')
sidepairings.append(G1(-3)*(c*a)*a*b*(c*a)^{-1}*G1(-3)) #15 Side pairing (
+, 3_3, 3_{11})
sidepairingscode.append('(+, 3_3, 3_{11})')
sidepairings.append(G1(-5)*G0(-6)) #16 Side pairing (
+, 2_5, 3_{10})
sidepairingscode.append('(+, 2_5, 3_{10})')
sidepairings.append(G1(-6)*G0(1)) #17 Side pairing (
+, 2_{12}, 3_9)
sidepairingscode.append('(+, 2_{12}, 3_9)')
sidepairings.append(G1(-6)*(c*a)*a*b*(c*a)^{-1}*G1(-2)) #18 Side pairing (
+, 3_2, 3_8)
sidepairingscode.append('(+, 3_2, 3_8)')

```

/Applications/SageMath-8.2.app/Contents/Resources/sage/local/lib/python2.7/site-packages/sage/repl/ipython_kernel/__main__.py:31: DeprecationWarning: show is deprecated. Please use plot instead.
See <http://trac.sagemath.org/20530> for details.

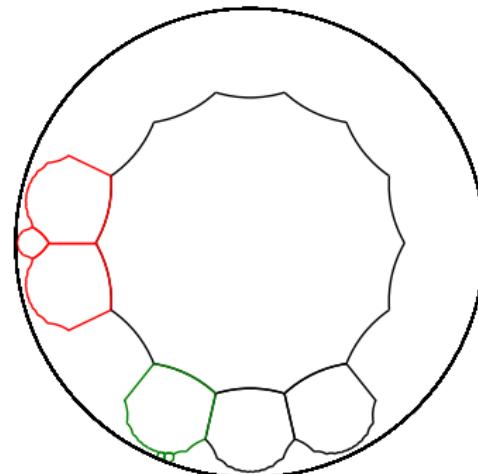


In [6]:

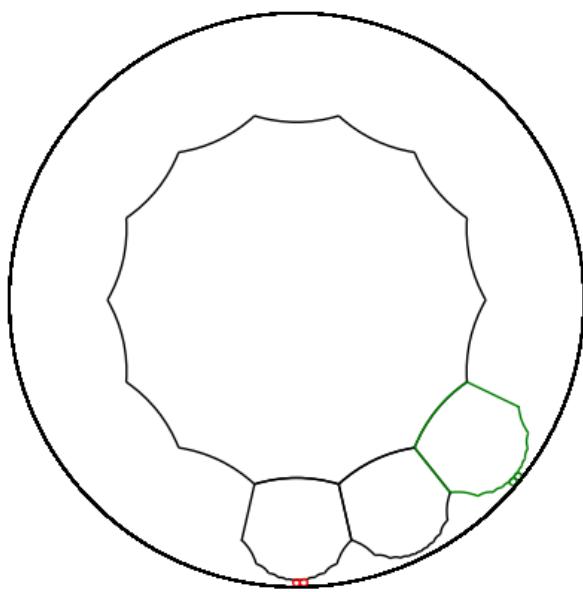
```
##### CELL 5: USE IT IF YOU WANT TO CHECK GRAPHICALLY HOW OUR SIDE PAIRINGS ACT.  
##### THE IMAGE OF plotdom (THE FUNDAMENTAL DOMAIN F) BY THE SIDE PAIRING IS DEPICTED IN RED.  
##### THE IMAGE BY THE INVERSE IS DEPICTED IN GREEN.  
##### THE SIDE PAIRING MAPS THE GREEN EDGE(S) OF plotdom TO THE RED ONE(S).  
##### THE CONFORMALITY/ANTICONFORMALITY OF THE SIDE PAIRING IS DETERMINED BY THE PARITY OF ITS LENGTH AS A WORD IN a,b,c
```

```
total=plotdom;  
for j in [1..len(sidepairings)-1]:  
  
    pairing=sidepairings[j]  
    resultado=plotdom+moverpol(pairing,'red')  
    total=total+moverpol(pairing,'black')  
    for k in [0..1]:  
        resultado=resultado+moverpol(pairing*(c*a)^k*a*b*(c*a)^-k,'red')  
        total=total+moverpol(pairing*(c*a)^k*a*b*(c*a)^-k,'black')  
  
    pairing_inv=sidepairings[j]^(-1)  
    resultado=resultado+moverpol(pairing_inv,'green')  
    total=total+moverpol(pairing_inv,'black')  
    for k in [0..1]:  
        resultado=resultado+moverpol(pairing_inv*(c*a)^k*a*b*(c*a)^-k,'green')  
        total=total+moverpol(pairing_inv*(c*a)^k*a*b*(c*a)^-k,'black')  
    show(resultado, title='Side pairing '+sidepairingscode[j]+' maps the black domain to the red one, its inverse maps the black domain to the green one', title_pos=(0.35,-0.1))  
  
show(total, title='The tesselation induced by the group generated by these side pairings', title_pos=(0.4,-0.05))
```

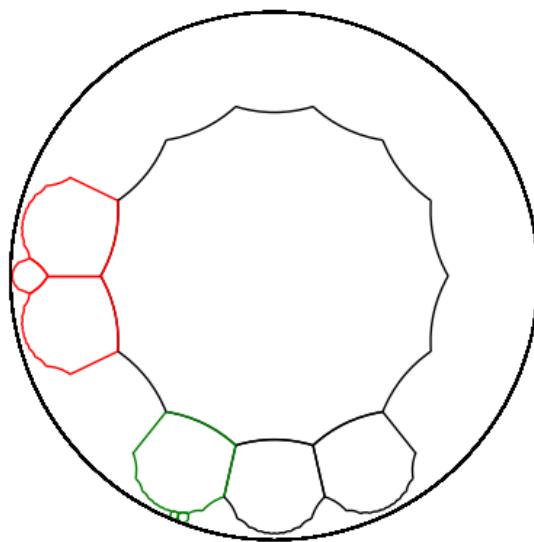
```
/Applications/SageMath-8.2.app/Contents/Resources/sage/local/lib/python2.7/site-packages/sage/repl/ipython_kernel/__main__.py:31: DeprecationWarning: show is deprecated. Please use plot instead.  
See http://trac.sagemath.org/20530 for details.
```



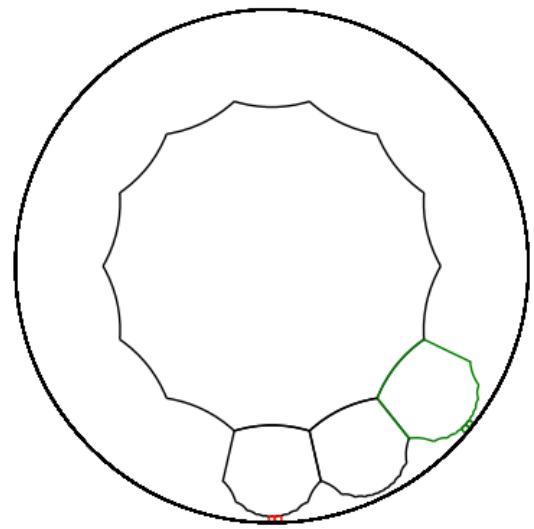
Side pairing $(-, 1_{\{13\}}, 1_{\{10\}})$ maps the black domain to the red one, its inverse maps the black domain to the green one



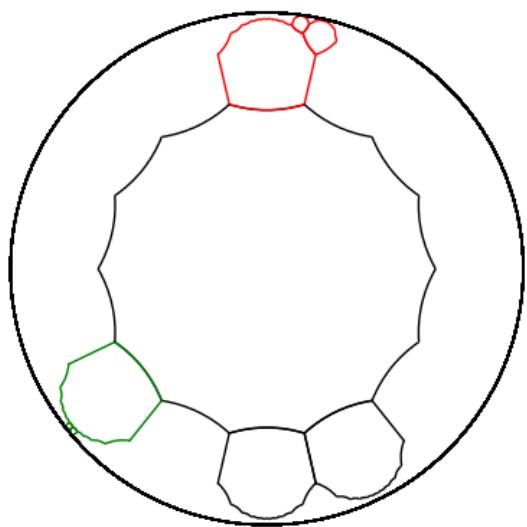
Side pairing $(+, 1_2, 2_7)$ maps the black domain to the red one, its inverse maps the black domain to the green one



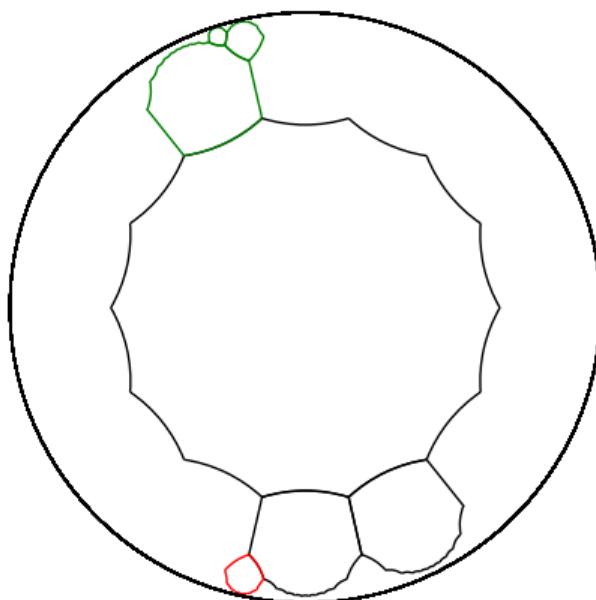
Side pairing $(-, 2_1, 1_{\{11\}})$ maps the black domain to the red one, its inverse maps the black domain to the green one



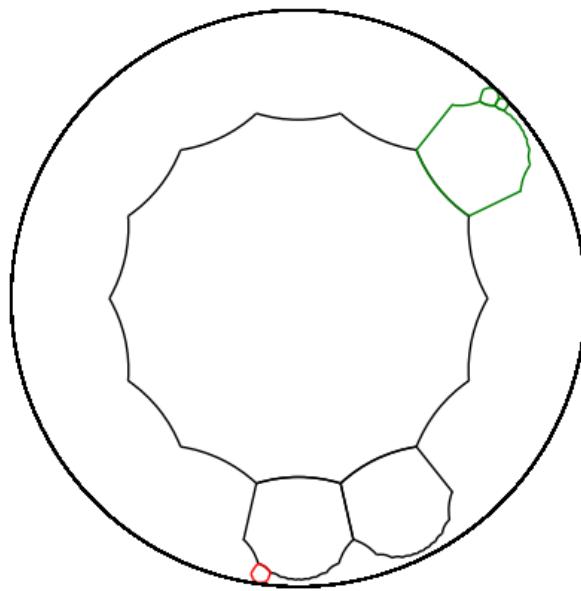
Side pairing $(+, 3_{\{13\}}, 2_8)$ maps the black domain to the red one, its inverse maps the black domain to the green one



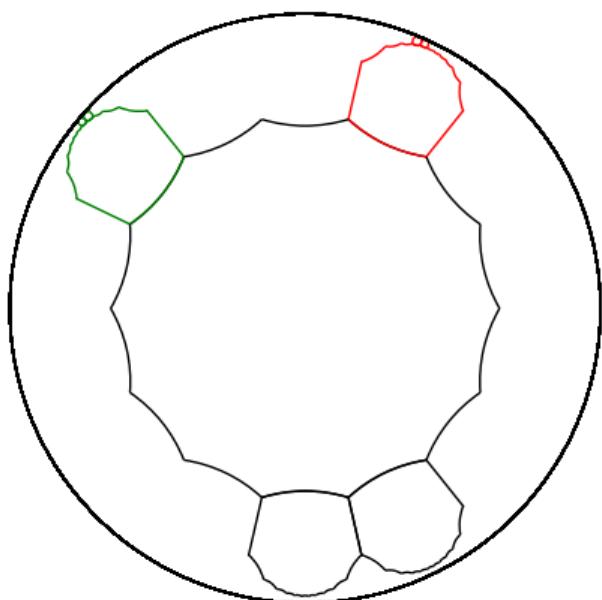
Side pairing $(+, 1_{\{12\}}, 1_7)$ maps the black domain to the red one, its inverse maps the black domain to the green one



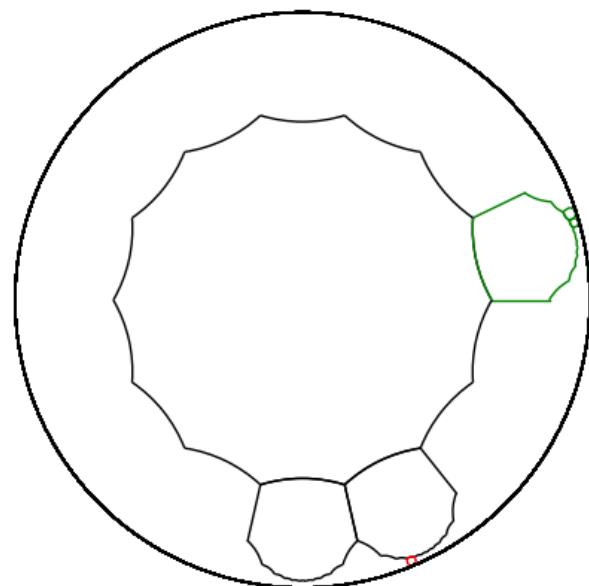
Side pairing $(-, 1_8, 2_2)$ maps the black domain to the red one, its inverse maps the black domain to the green one



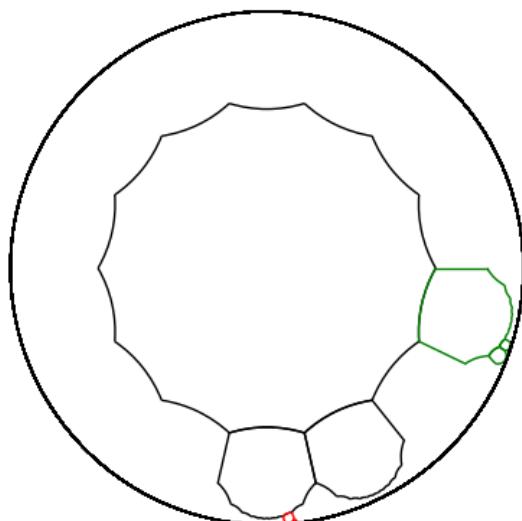
Side pairing $(+, 1_5, 2_3)$ maps the black domain to the red one, its inverse maps the black domain to the green one



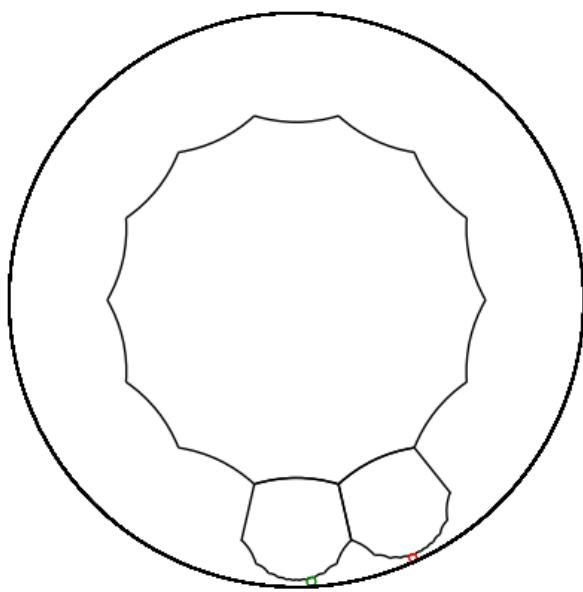
Side pairing $(-, 1_9, 1_6)$ maps the black domain to the red one, its inverse maps the black domain to the green one



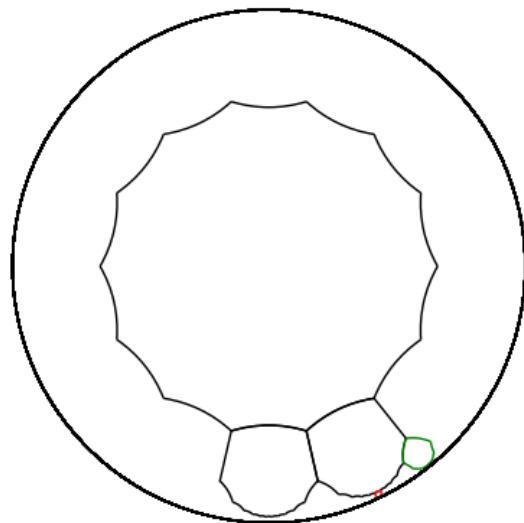
Side pairing $(+, 1_4, 3_5)$ maps the black domain to the red one, its inverse maps the black domain to the green one



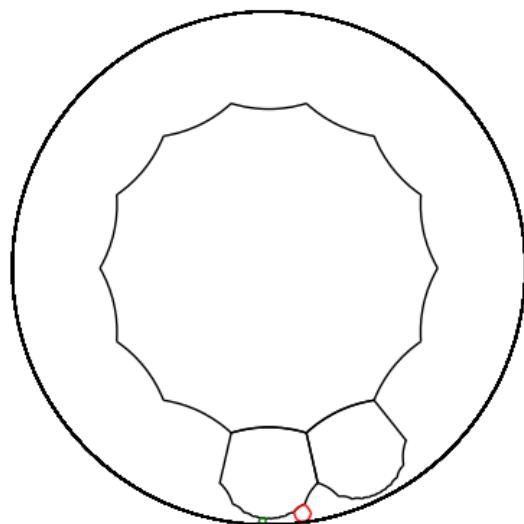
Side pairing $(+, 1_3, 2_{\{10\}})$ maps the black domain to the red one, its inverse maps the black domain to the green one



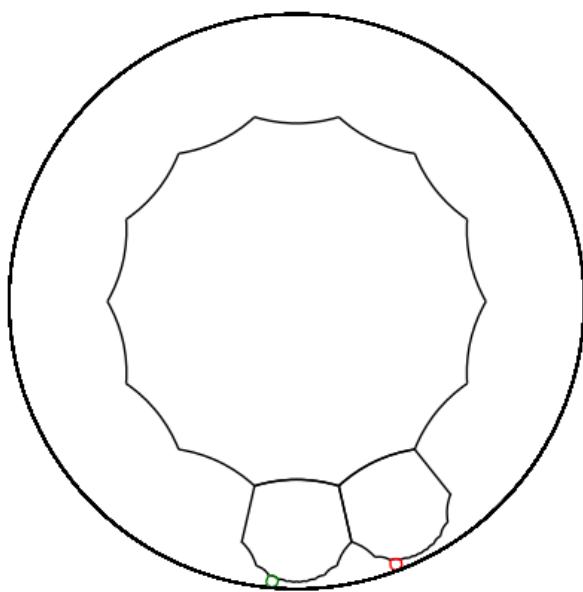
Side pairing $(+, 2_9, 3_6)$ maps the black domain to the red one, its inverse maps the black domain to the green one



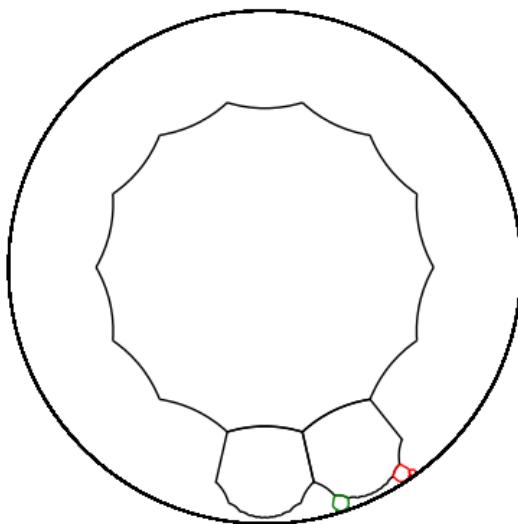
Side pairing $(+, 3_{12}, 3_7)$ maps the black domain to the red one, its inverse maps the black domain to the green one



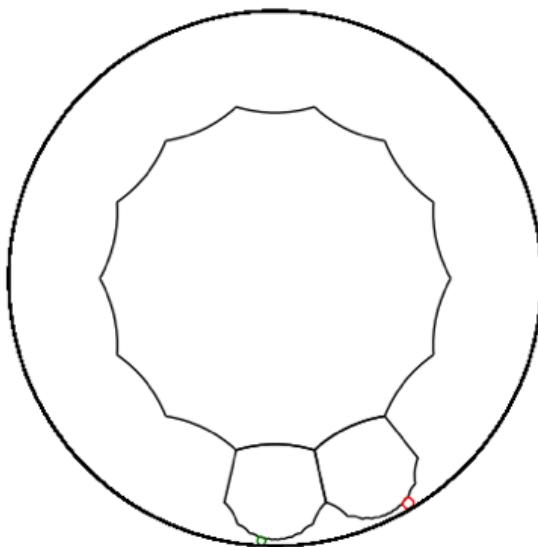
Side pairing $(+, 2_6, 2_{11})$ maps the black domain to the red one, its inverse maps the black domain to the green one



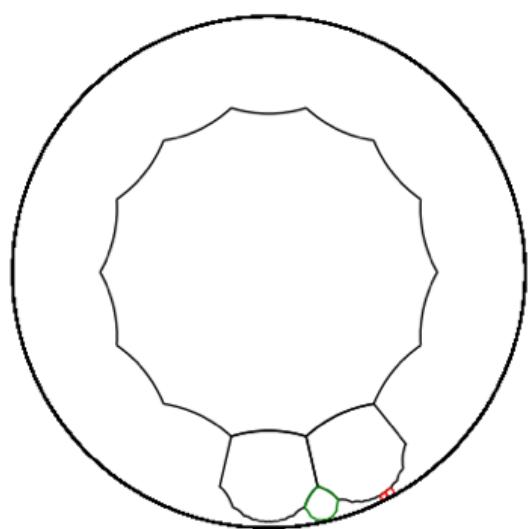
Side pairing $(+, 2_4, 3_4)$ maps the black domain to the red one, its inverse maps the black domain to the green one



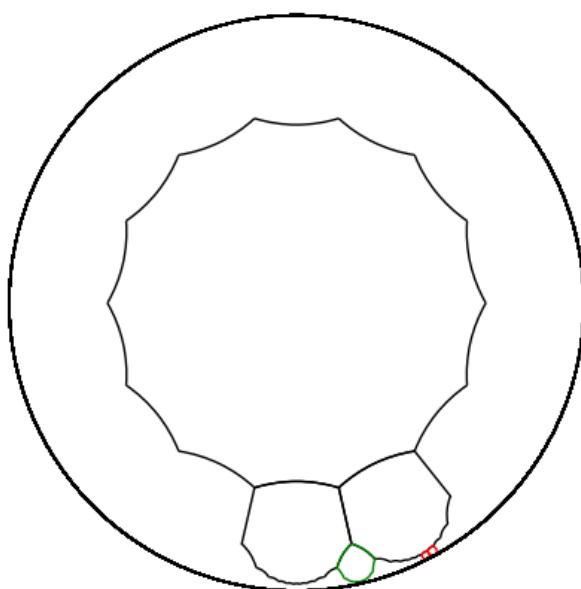
Side pairing $(+, 3_3, 3_{\{11\}})$ maps the black domain to the red one, its inverse maps the black domain to the green one



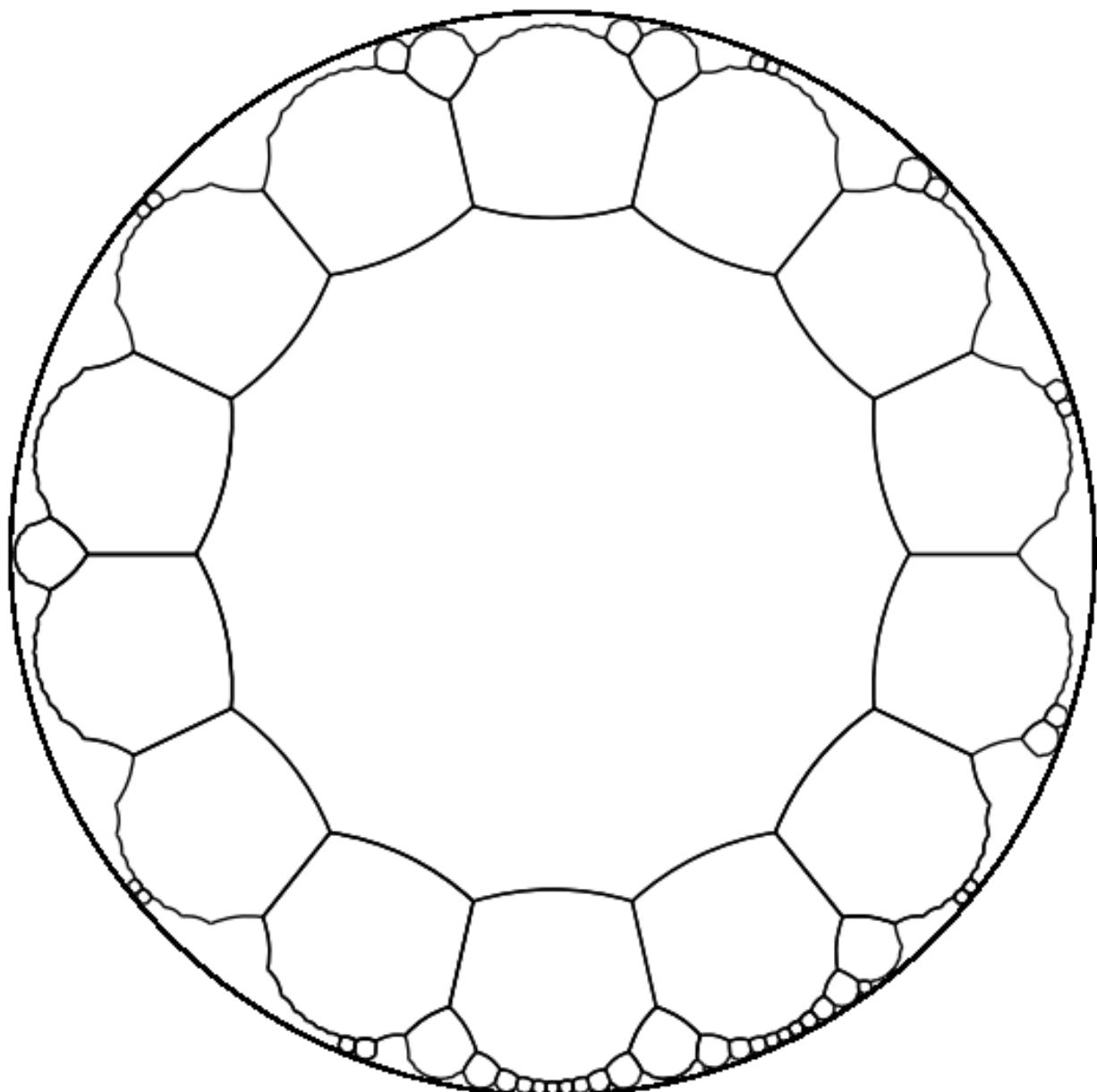
Side pairing $(+, 2_5, 3_{\{10\}})$ maps the black domain to the red one, its inverse maps the black domain to the green one



Side pairing $(+, 2_{\{12\}}, 3_9)$ maps the black domain to the red one, its inverse maps the black domain to the green one



Side pairing $(+, 3_2, 3_8)$ maps the black domain to the red one, its inverse maps the black domain to the green one



The tessellation induced by the group generated by these side pairings

In [7]:

```
##### CELL 6: DEFINING AN ELLIPTIC ELEMENT tau OF ORDER TWO THAT NORMALIZES THE GROUP K
##### candidate IS A POSSIBLE CENTER OF ONE OF THE DISCS OF A HIDDEN EXTR EMAL PACKING, OBTAINED BY THE BRUTE FORCE PROCEDURE
##### conjugates IS A LIST OF CERTAIN ELEMENTS OF THE GROUP K
##### WE SHOW HERE NUMERICALLY THAT conjugates[j]*tau*sidepairing[j]*tau^(-1) IS THE IDENTITY

candidate=0.515984860898361 - 0.248485212697688*I
tau=rotasionpi(((c*a)^2*b*B).coordinates(),candidate)

conjugates=['cero',
sidepairings[2]^(-1)*sidepairings[7]*sidepairings[6]^(-1)*sidepairings[2],
sidepairings[7]^(-1)*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[7]*sidepairings[6]^(-1)*sidepairings[2],
sidepairings[7]^(-1)*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[6]*sidepairings[1]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[3]^(-1)*sidepairings[7]^(-1)*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[7]^(-1)*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[3]^(-1)*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[14]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[11]^(-1)*sidepairings[14]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[14]^(-1)*sidepairings[11]*sidepairings[14]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[16]^(-1)*sidepairings[12]*sidepairings[16]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[7]*sidepairings[2]^(-1)*sidepairings[11]^(-1)*sidepairings[14]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[7]*sidepairings[14]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[14]^(-1)*sidepairings[16]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[7]*sidepairings[16]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[14]^(-1)*sidepairings[11]*sidepairings[4]*sidepairings[16]*sidepairings[2],
sidepairings[2]^(-1)*sidepairings[14]^(-1)*sidepairings[11]*sidepairings[4]*sidepairings[16]*sidepairings[2]]

for j in [1..len(conjugates)-1]:
    print(conjugates[j]*tau*sidepairings[j]*tau^-1)
```

Isometry in PD

```
[ -0.999999999999862 - 1.45218503888600e-10*I 8.16400280712060e
-11 + 1.20077281451358e-10*I]
[8.16400280712060e-11 - 1.20077281451358e-10*I - 0.999999999999
862 + 1.45218503888600e-10*I]
```

Isometry in PD

```
[ -1.00000000000002 + 1.97619698383278e-12*I 8.96172025477426e
-13 - 1.66977542903624e-12*I]
[8.96172025477426e-13 + 1.66977542903624e-12*I - 1.000000000000
002 - 1.97619698383278e-12*I]
```

Isometry in PD

```
[ 0.999999999999861 + 1.45588430200405e-10*I -8.1849194089045
3e-11 - 1.20383703006155e-10*I]
[-8.18491940890453e-11 + 1.20383703006155e-10*I 0.9999999999
99861 - 1.45588430200405e-10*I]
```

Isometry in PD
[-1.00000000000001 + 2.08144612656724e-12*I 8.37996338987068e-13 - 1.75859327100625e-12*I]
[8.37996338987068e-13 + 1.75859327100625e-12*I -1.000000000000001 - 2.08144612656724e-12*I]
Isometry in PD
[-0.999999999999171 + 1.35607081119815e-10*I -7.82485187755810e-11 - 1.10772724326580e-10*I]
[-7.82485187755810e-11 + 1.10772724326580e-10*I -0.999999999999171 - 1.35607081119815e-10*I]
Isometry in PD
[0.999999999999887 + 4.97881291749991e-10*I -3.01067171193381e-10 - 3.96529920010380e-10*I]
[-3.01067171193381e-10 + 3.96529920010380e-10*I 0.999999999999887 - 4.97881291749991e-10*I]
Isometry in PD
[1.00000000000122 - 4.38295177929149e-10*I 2.73256528515731e-10 + 3.42694761457096e-10*I]
[2.73256528515731e-10 - 3.42694761457096e-10*I 1.000000000000122 + 4.38295177929149e-10*I]
Isometry in PD
[1.00000000000034 + 1.45123468797692e-10*I -8.71174243854966e-11 - 1.16053833210117e-10*I]
[-8.71174243854966e-11 + 1.16053833210117e-10*I 1.0000000000000034 - 1.45123468797692e-10*I]
Isometry in PD
[-0.999999999999023 + 1.43418166231868e-10*I -9.38320532384296e-11 - 1.08473230397976e-10*I]
[-9.38320532384296e-11 + 1.08473230397976e-10*I -0.9999999999999023 - 1.43418166231868e-10*I]
Isometry in PD
[-1.00000000000008 + 2.67754707294898e-11*I -2.00812699802100e-11 - 1.76632042325764e-11*I]
[-2.00812699802100e-11 + 1.76632042325764e-11*I -1.0000000000000008 - 2.67754707294898e-11*I]
Isometry in PD
[0.999999999991378 - 2.16585416268344e-9*I 1.17033938096256e-9 + 1.82246395752372e-9*I]
[1.17033938096256e-9 - 1.82246395752372e-9*I 0.999999999991378 + 2.16585416268344e-9*I]
Isometry in PD
[0.99999999999853 - 5.68927127631014e-11*I 2.16719975298929e-11 + 5.26290122593309e-11*I]
[2.16719975298929e-11 - 5.26290122593309e-11*I 0.999999999999853 + 5.68927127631014e-11*I]
Isometry in PD
[0.99999999999820 - 2.35841435269890e-9*I 1.28181332215149e-9 + 1.97968486048694e-9*I]
[1.28181332215149e-9 - 1.97968486048694e-9*I 0.999999999999820 + 2.35841435269890e-9*I]
Isometry in PD
[-0.999999999998419 + 1.77716508176218e-9*I -9.72010028021941e-10 - 1.48780721076491e-9*I]
[-9.72010028021941e-10 + 1.48780721076491e-9*I -0.9999999999998419 - 1.77716508176218e-9*I]
Isometry in PD
[0.999999999999722 - 1.02573061155908e-10*I 5.29860599840504e

In [8]:

```
##### Cell 7: the fixed point of tau
tau.fixed_point_set()
```

Out[8]:

[Point in PD 0.619750015795811 - 0.459297917966208*I]