Rauzy induction of polygon partitions and toral $\mathbb{Z}^2$-rotations

Sébastien Labbé, CNRS, LaBRI, Université de Bordeaux

Conference Algebraic and Combinatorial Invariants of Subshifts and Tilings, CIRM, Marseille, January 13, 2021

These slides are available in 3 formats:

- html: http://www.slabbe.org/Communications/2021-01-cirm.slides.html
- pdf: http://www.slabbe.org/Communications/2021-01-cirm.pdf
- the source (SageMath Jupyter notebook): http://www.slabbe.org/Communications/2021-01-cirm.ipynb

HELP for navigating in the HTML slides:

- SPACE BAR = next slide,
- SHIFT + SPACE = previous slide,
- ESC = overview

Outline as 5 sections (disposed as columns of slides if viewed in html format):

- 1 - Polyhedrons, Polyhedron partitions and PETs
- 2 - Rauzy induction of PETs and of toral partitions
- 3 - A particular partition $\mathcal{P}_{\mathcal{U}}$ of $\mathbb{T}^2$
- 4 - Inducing the partition $\mathcal{P}_{\mathcal{U}}$ with respect to a toral $\mathbb{Z}^2$-rotation
- 5 - Results

1 - Polyhedrons, Polyhedron partitions, PETs, symbolic representation

Computations (arithmetic, comparisons, etc.) are more efficient when performed in a number field like $\mathbb{Q}(\varphi)$ with $\varphi = (1 + \sqrt{5})/2$.

In [1]:
```
z = polygen(QQ, 'z')
K.<phi> = NumberField(z**2-z-1, 'phi', embedding=RR(1.6)); K
```

Out[1]:
```
Number Field in phi with defining polynomial z^2 - z - 1 with phi = 1.618033988749895?
```

In [2]:
```
phi.n(digits=500)
```

Out[2]:
```
1.61803398874989484820458683436563811772030917980576286213544862270526046281890244970720720418939113748475408807538689175212663
3862223536931793180060766726354433389086595939582905638322661319928290267880675208766892501711696207032221043216269548626296313
6144381497587012203408058879544547492461856953648644492410443207713449470495658467885098743394422125448770664780915884607499887
124007652170575179788341662562494075890697040002812104276217711177780531531714101170466659914669798731761356006708748071
```

In [3]:
```
phi^2 + phi^-10
```

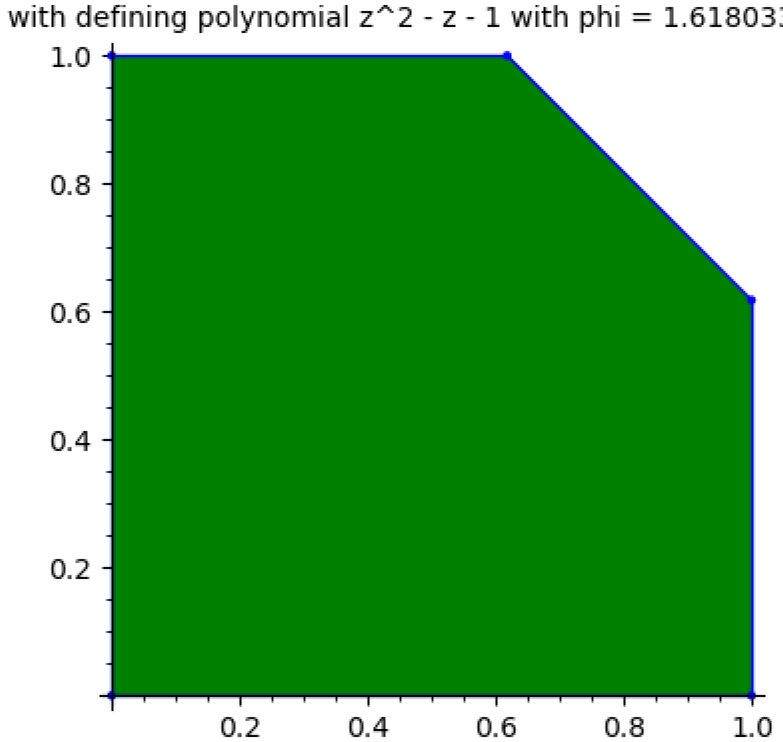
Out[3]:
```
-54*phi + 90
```

Polyhedron in SageMath: from vertices

In [4]:
```
vertices = [(0,0), (1,0), (0,1), (1,1/phi), (1/phi,1)]
bottom = Polyhedron(vertices)
bottom
```

Out[4]:

A 2-dimensional polyhedron in (Number Field in phi with defining polynomial z^2 - z - 1 with phi = 1.618033988749895?)^2 defined as the convex hull of 5 vertices

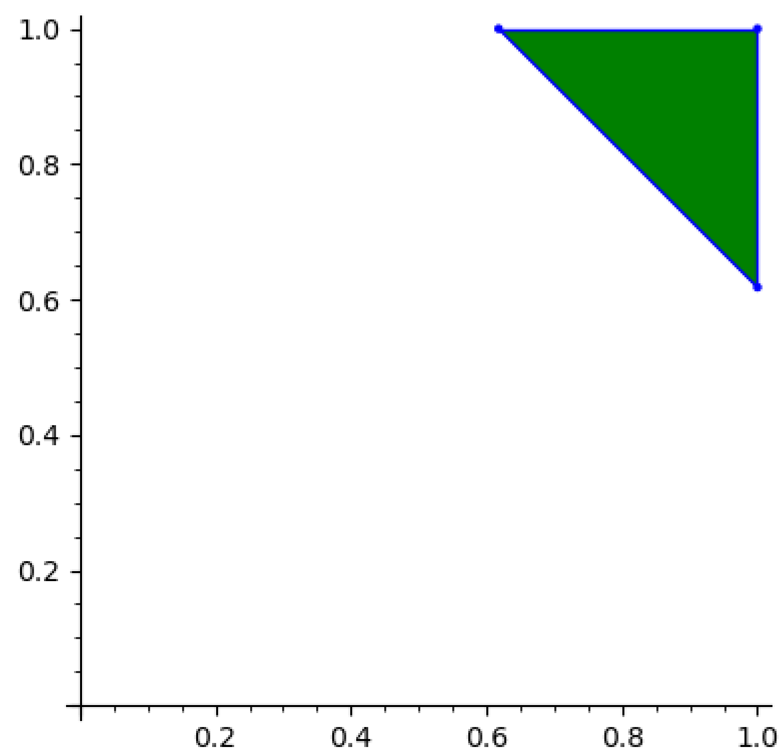


Polyhedron in SageMath: from inequalities

Convention for inequalities: $7 + 2x_1 - 3x_2 \geq 0$ is incoded as `(7, 2, -3)`.

In [5]:
```
top = Polyhedron(ieqs=[(-1/phi-1,1,1), (1,-1,0), (1,0,-1)])
top.plot(xmin=0, ymin=0)
```

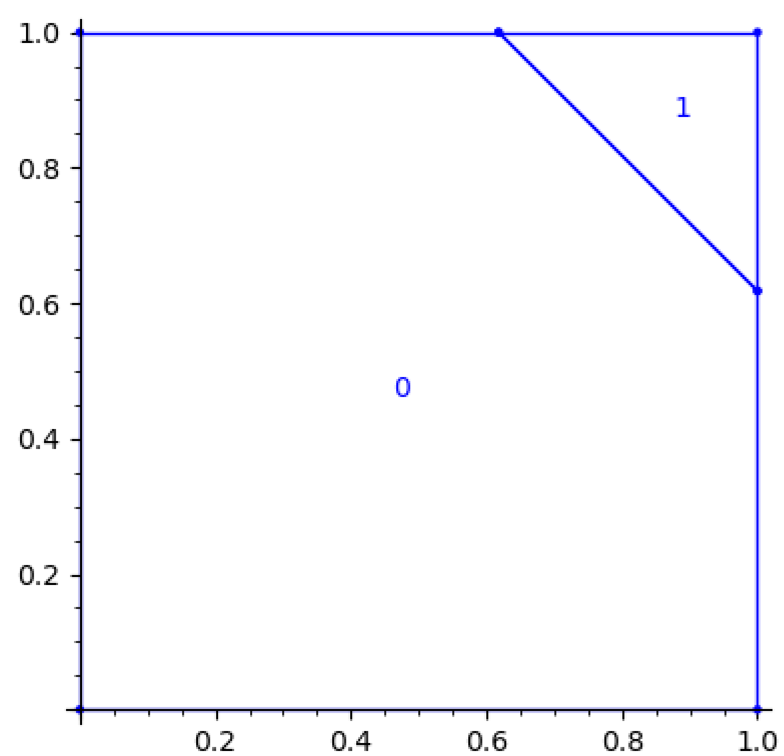
Out[5]:



Polyhedron partition

In [6]:
```
from slabbe import PolyhedronPartition
P = PolyhedronPartition([bottom, top])
P.plot()
```

Out[6]:

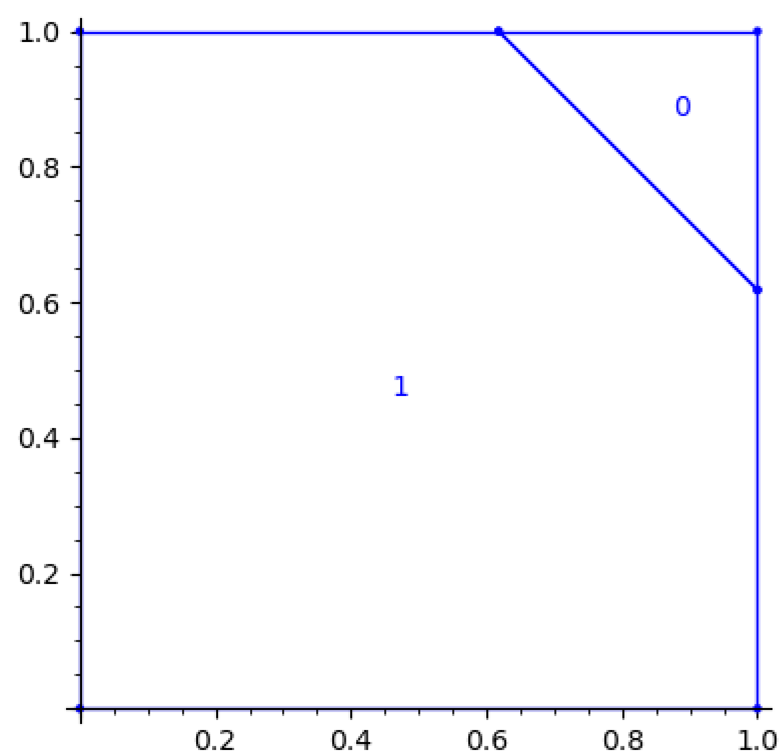


Shortcut: refine a partition by a hyperplane

In [7]:

```
square = Polyhedron([(0,0), (1,0), (0,1), (1,1)])
P = PolyhedronPartition([square])
P = P.refine_by_hyperplane([-1/phi-1,1,1])
P.plot()
```

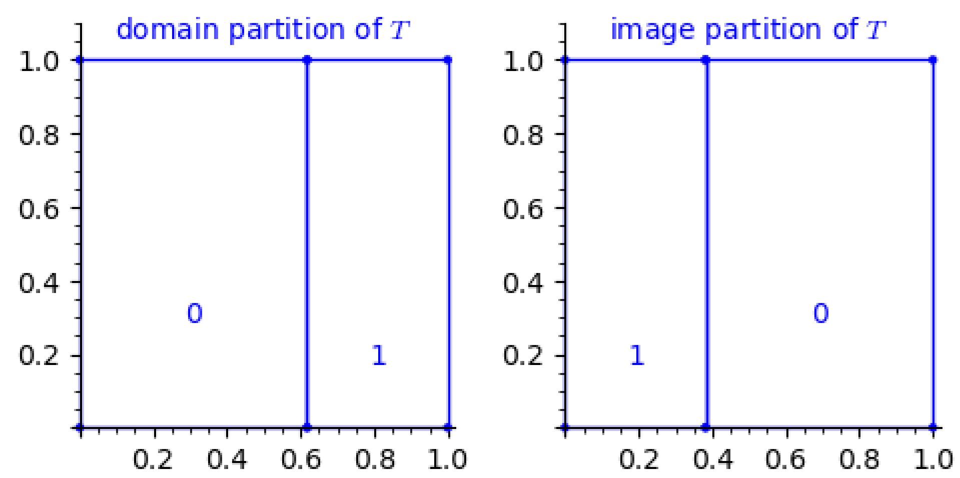
Out[7]:



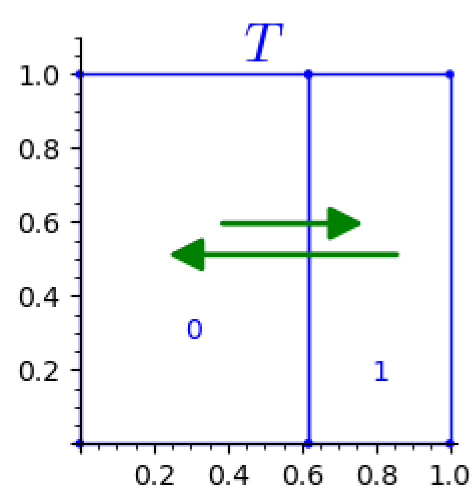
Polyhedron Exchange Transformation on $[0, 1)^2 \simeq \mathbb{T}^2$

In [8]:
```
from slabbe import PolyhedronExchangeTransformation as PET
lattice_base = matrix.column([(1,0), (0,1)])
T = PET.toral_translation(lattice_base, vector((phi^-2,0)))
```

In [9]:
```
def title(content, height=1.08, fontsize=10):
    return text(content, (.5, height), fontsize=fontsize)
t1 = title(r"domain partition of $T$", fontsize=10)
t2 = title(r"image partition of $T$", fontsize=10)
graphics_array([T.partition().plot()+t1, T.image_partition().plot()+t2]).show(figsize=5)
```



In [10]:
```
t = title(r"$T$", fontsize=20)
(T.plot()+t).show(figsize=4)
```



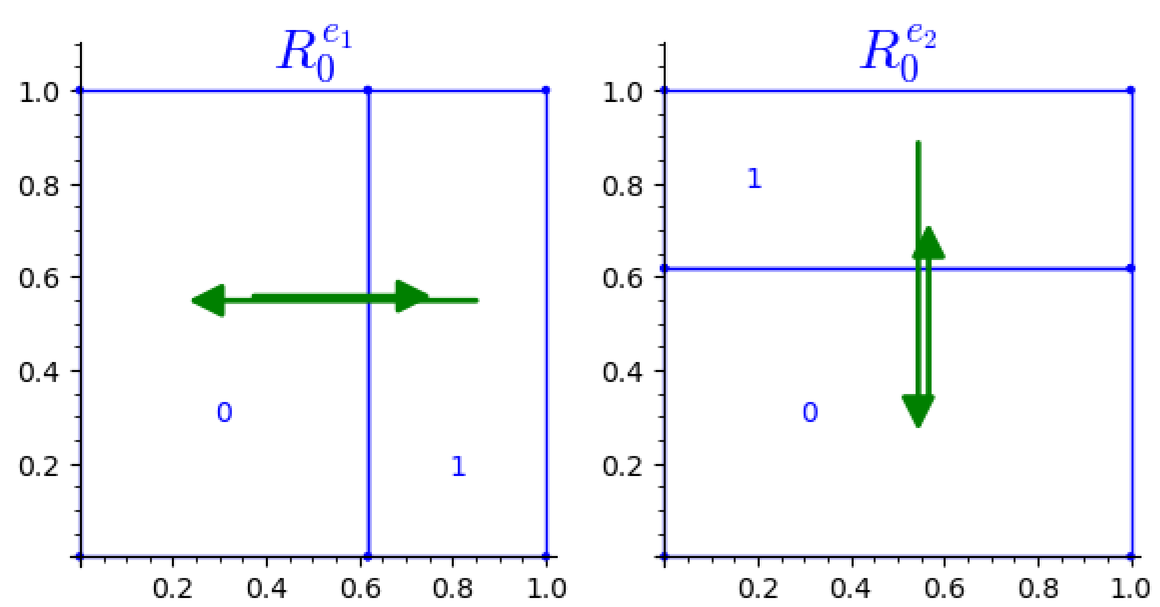
A toral $\mathbb{Z}^2$-rotation

A continuous $\mathbb{Z}^2$-action $\begin{array}{rccc} R_0 : & \mathbb{Z}^2 \times \mathbb{T}^2 & \to & \mathbb{T}^2 \\ & (\boldsymbol{n}, \boldsymbol{x}) & \mapsto & \boldsymbol{x} + \varphi^{-2}\boldsymbol{n} \bmod \mathbb{Z}^2 \end{array}$ can be written as a pair of commuting PETs.

In [11]:
```
R0e1 = PET.toral_translation(lattice_base, vector((phi^-2,0)))
R0e2 = PET.toral_translation(lattice_base, vector((0,phi^-2)))
```

In [12]:
```
t1 = title(r"$R_0^{e_1}$", fontsize=20)
t2 = title(r"$R_0^{e_2}$", fontsize=20)
graphics_array([R0e1.plot()+t1, R0e2.plot()+t2])
```

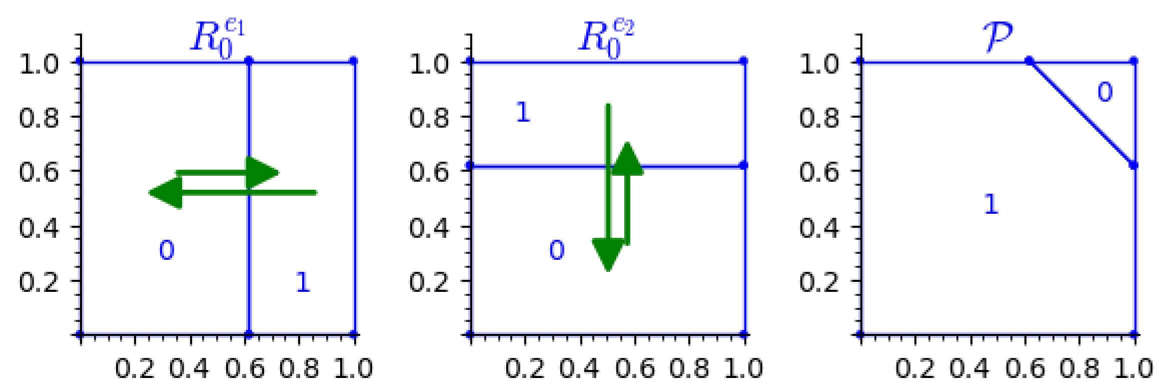
Out[12]:



Symbolic dynamical system

In [13]:
```
t1 = title(r"$R_0^{e_1}$", fontsize=15); t2 = title(r"$R_0^{e_2}$", fontsize=15); t3 = title(r"$\mathcal{P}$", fontsize=15)
graphics_array([R0e1.plot()+t1, R0e2.plot()+t2, P.plot()+t3])
```

Out[13]:



- Let $(\mathbb{T}, \mathbb{Z}^2, R)$ be the dynamical system given by a $\mathbb{Z}^2$-rotation R on $\mathbb{T}$.
- For some finite set $\mathcal{A}$, a **topological partition** of $\mathbb{T}$ is a finite collection $\{P_a\}_{a\in\mathcal{A}}$ of disjoint open sets $P_a \subset \mathbb{T}$ such that $\mathbb{T} = \bigcup_{a\in\mathcal{A}} \overline{P_a}$.
- If $S \subset \mathbb{Z}^2$ is a finite set, we say that a pattern $w \in \mathcal{A}^S$ of support S is **allowed** for $\mathcal{P}, R$ if
$$\bigcap_{\boldsymbol{k}\in S} R^{-\boldsymbol{k}}(P_{w_{\boldsymbol{k}}}) \neq \emptyset.$$
- Let $\mathcal{L}_{\mathcal{P},R}$ be the collection of all allowed patterns for $\mathcal{P}, R$. The set $\mathcal{L}_{\mathcal{P},R}$ is the language of the **symbolic dynamical system** corresponding to $\mathcal{P}, R$, i.e., the subshift $\mathcal{X}_{\mathcal{P},R} \subseteq \mathcal{A}^{\mathbb{Z}^2}$ defined as
$$\mathcal{X}_{\mathcal{P},R} = \{x \in \mathcal{A}^{\mathbb{Z}^2} \mid \pi_S \circ \sigma^{\boldsymbol{n}}(x) \in \mathcal{L}_{\mathcal{P},R} \text{ for every } \boldsymbol{n} \in \mathbb{Z}^2 \text{ and finite subset } S \subset \mathbb{Z}^2\},$$
see Prop. 9.2.4 in the chapter Hochman 2016.

2 - Rauzy induction of a PET and of a partition

Recall that the **first return map** $\widehat{T}|_W$ of a dynamical system (X, T) maps a point $\boldsymbol{x} \in W \subset X$ to the first point in the forward orbit of T lying in W, i.e.

$$\widehat{T}|_W(\boldsymbol{x}) = T^{r(\boldsymbol{x})}(\boldsymbol{x}) \qquad \text{where } r(\boldsymbol{x}) = \min\{k \in \mathbb{Z}_{>0} : T^k(\boldsymbol{x}) \in W\}.$$

Facts:

- From Poincaré's recurrence theorem, if μ is a finite T-invariant measure on X, then the first return map $\widehat{T}|_W$ is well defined for μ-almost all $\boldsymbol{x} \in W$.
- Moreover if T is a PET and W is a polyhedron, then the first return map $\widehat{T}|_W$ is a PET.
- If $\mathcal{P}$ is a partition of X, then there exists a substitution β and an induced partition $\widehat{\mathcal{P}}|_W$ such that $\mathcal{X}_{\mathcal{P},T} = \overline{\beta\left(\mathcal{X}_{\widehat{\mathcal{P}}|_W, \widehat{T}|_W}\right)}^{\sigma}$.
- If W is the intersection of the domain with a half-space, then there is a nice algorithm to compute $\widehat{\mathcal{P}}|_W$, $\widehat{T}|_W$ and β, see arXiv:1906.01104.

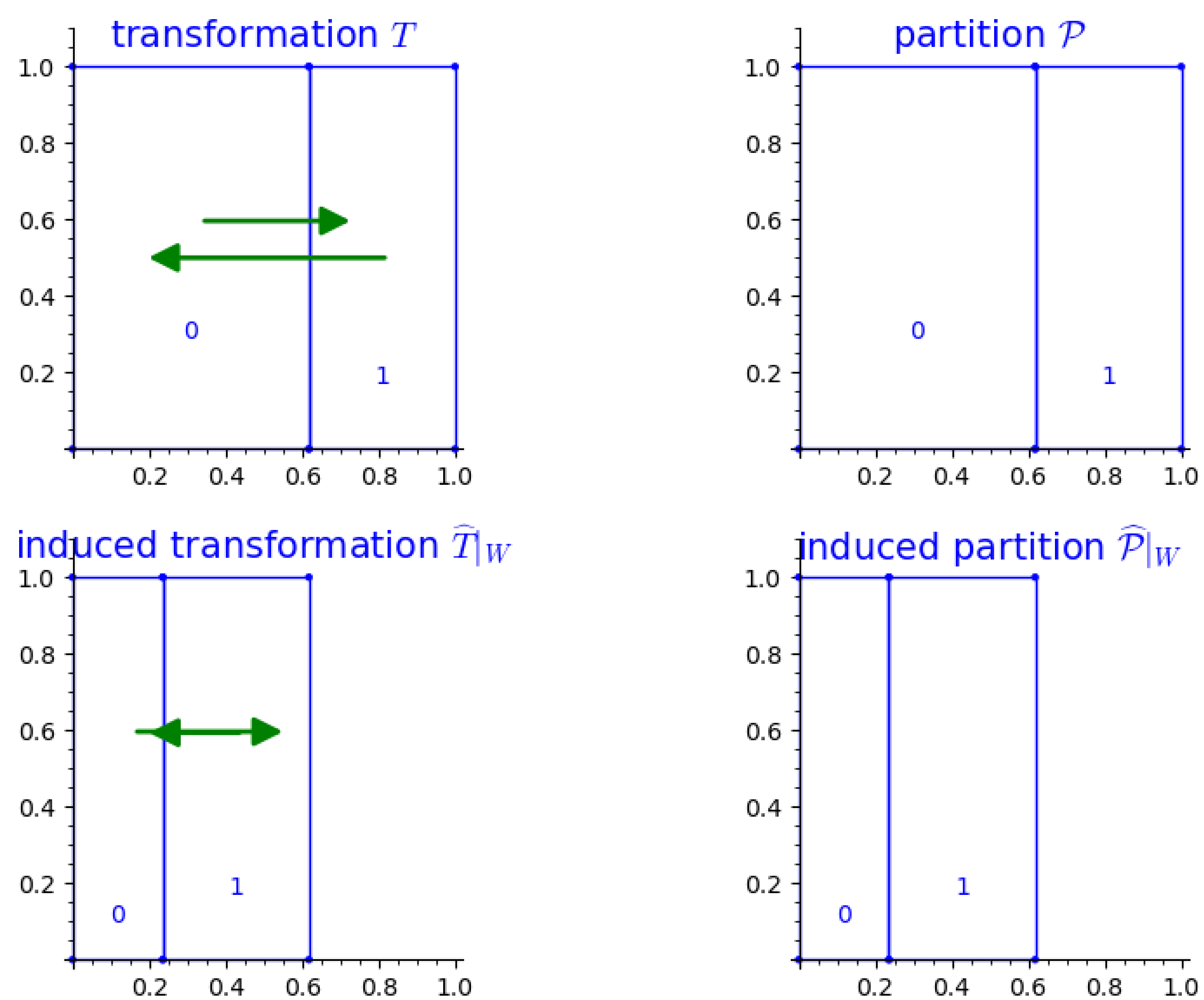
Helper function `please_draw_Rauzy_induction`

This is some code to draw induced transformation on the next slide (you may safely ignore what is below).

In [14]:
```
bb = point([(0,0), (1,1)], color='white') ### hack to make all plots to have the same bounding box
def please_draw_Rauzy_induction(T, P, inducedT, inducedP, beta, figsize=9):
    t1 = title(r'transformation $T$', fontsize=15)
    t2 = title(r'partition $\mathcal{P}$', fontsize=15)
    t3 = title(r'induced transformation $\widehat{T}|_W$', fontsize=15)
    t4 = title(r'induced partition $\widehat{\mathcal{P}}|_W$', fontsize=15)
    graphics_array([T.plot()+bb+t1, P.plot()+bb+t2,
                    inducedT.plot()+bb+t3,
                    inducedP.plot()+bb+t4], ncols=2).show(figsize=figsize)
    show(LatexExpr(r"\text{The induced substitution is }"+
                   r"\beta:{}".format(latex(beta))))
```

Rauzy induction on subdomain W: $\mathcal{X}_{\mathcal{P},T} = \overline{\beta\left(\mathcal{X}_{\widehat{\mathcal{P}}|_W,\widehat{T}|_W}\right)}^{\sigma}$

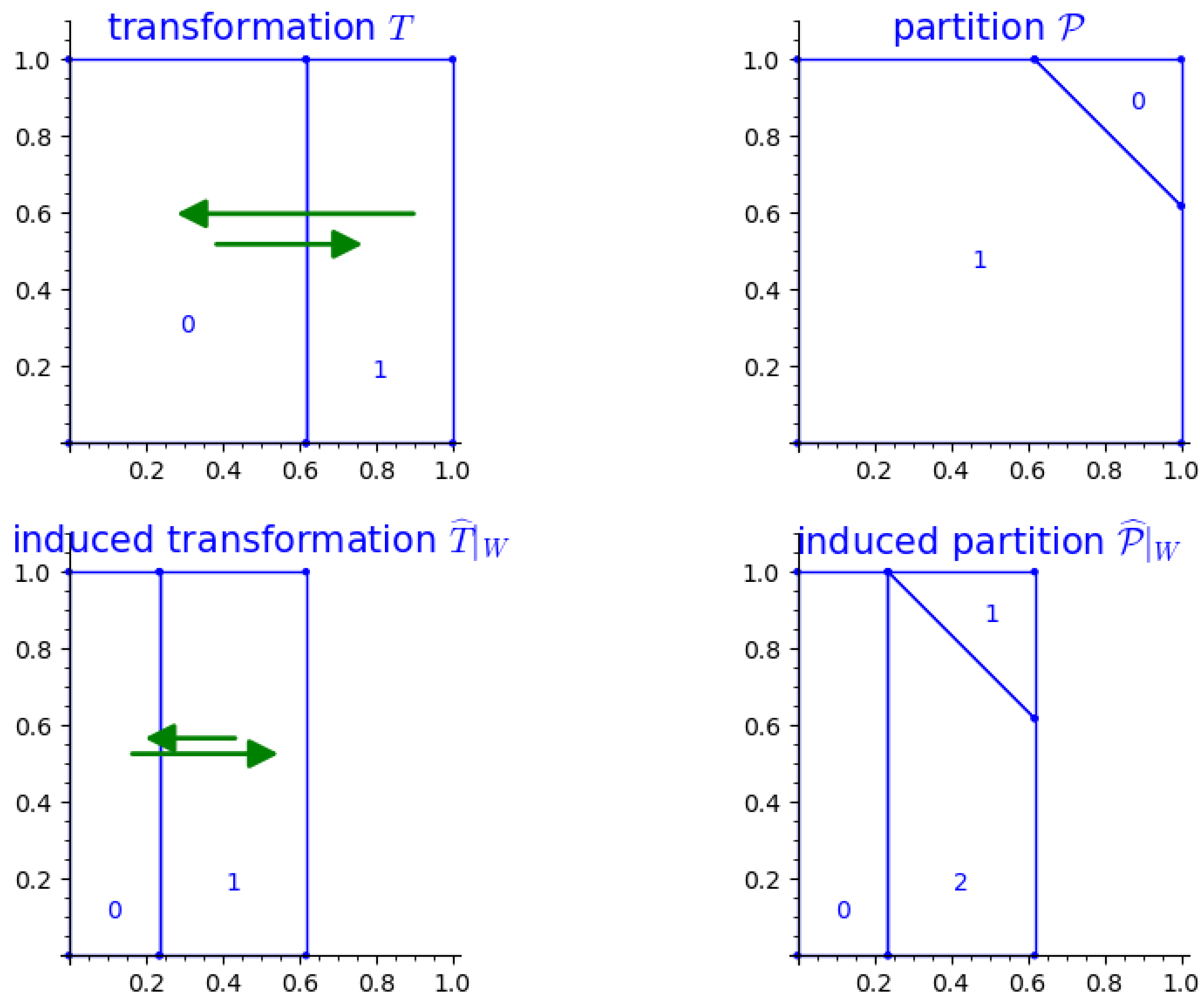
In [15]:
```
x_ineq = [phi^-1, -1, 0] ### x <= phi^-1
inducedT,beta = T.induced_transformation(x_ineq)
please_draw_Rauzy_induction(T, T.partition(), inducedT,  inducedT.partition(), WordMorphism(beta))
```



The induced substitution is $\beta : \begin{array}{l} 0 \mapsto 0 \\ 1 \mapsto 01 \end{array}$

Rauzy induction on subdomain W: $\mathcal{X}_{\mathcal{P},T} = \overline{\beta\left(\mathcal{X}_{\widehat{\mathcal{P}}|_W,\widehat{T}|_W}\right)}^{\sigma}$ for any partition $\mathcal{P}$

In [16]:
```
x_ineq = [phi^-1, -1, 0] ### x <= phi^-1
inducedT,_ = T.induced_transformation(x_ineq)
inducedP,beta = T.induced_partition(x_ineq, P, substitution_type='row')
please_draw_Rauzy_induction(T, P, inducedT, inducedP, beta)
```

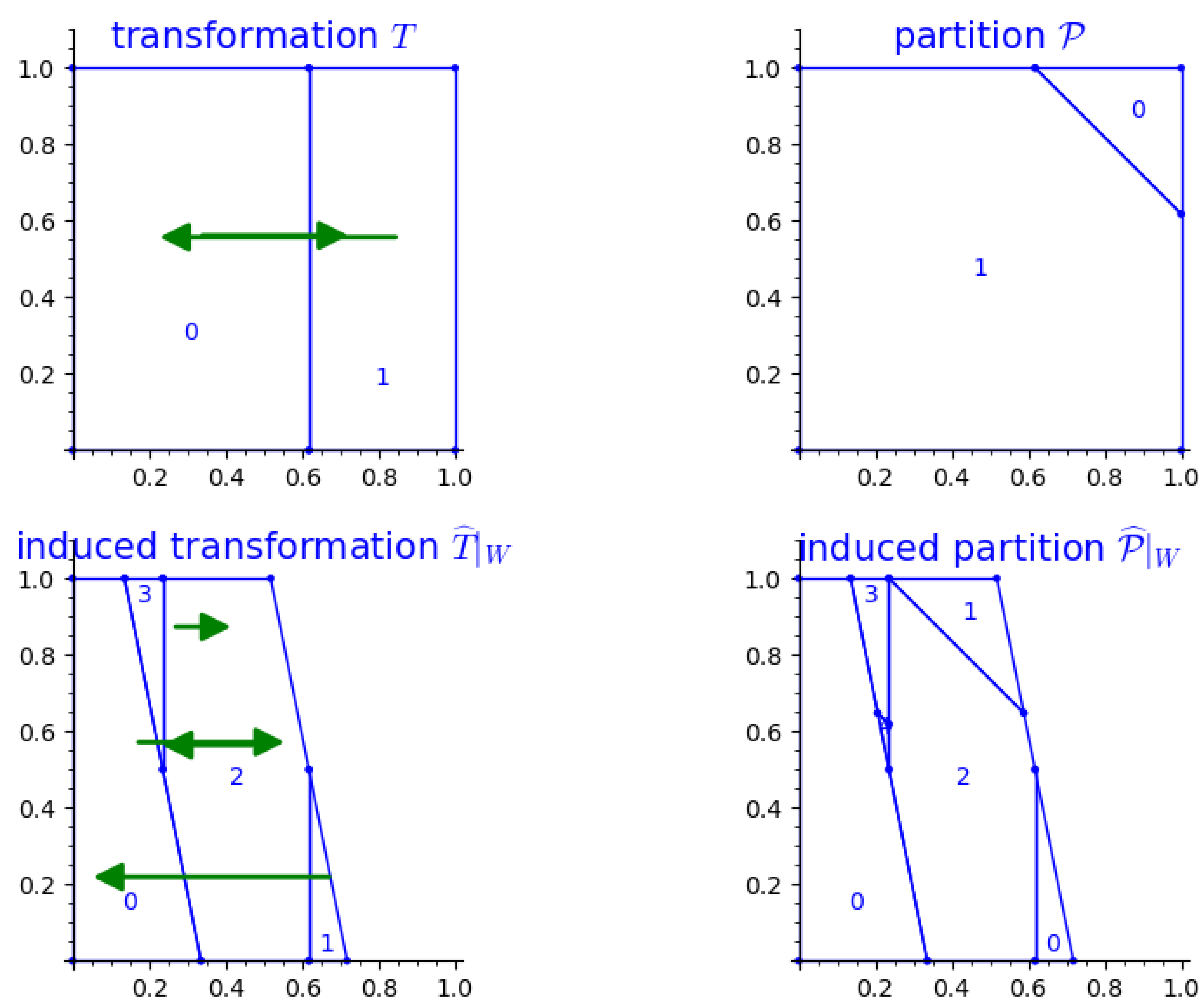


The induced substitution is $\beta: \ 0 \mapsto (1), \quad 1 \mapsto (1, 0), \quad 2 \mapsto (1, 1).$

Rauzy induction on general subdomain W: $\mathcal{X}_{\mathcal{P},T} = \overline{\beta\left(\mathcal{X}_{\widehat{\mathcal{P}}|_W,\widehat{T}|_W}\right)}^{\sigma}$ for any partition $\mathcal{P}$

Of course, for general subdomain W, the induced transformation $\widehat{T}|_W$ of a toral rotation T is not a toral rotation. Today, the induced transformations are toral rotations, so they commute between themselves.

In [17]:
```
x_ineq = [phi^-1+1/10, -1, 0-1/5]
inducedT,_ = T.induced_transformation(x_ineq)
inducedP,beta = T.induced_partition(x_ineq, P, substitution_type='row')
please_draw_Rauzy_induction(T, P, inducedT, inducedP, beta)
```



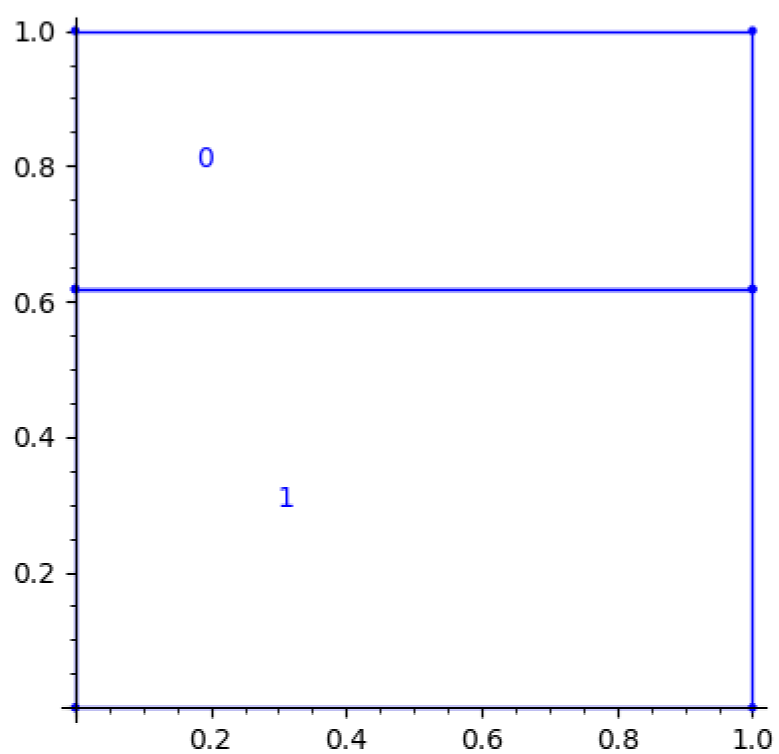
The induced substitution is β

3 - A particular partition $\mathcal{P}_{\mathcal{U}}$ of $\mathbb{T}^2$

The polygon partition P_a:

In [18]:
```
square = Polyhedron([(0,0), (1,0), (0,1), (1,1)])
Pa = PolyhedronPartition([square])
Pa = Pa.refine_by_hyperplane([-1/phi, 0, 1])
Pa.plot()
```

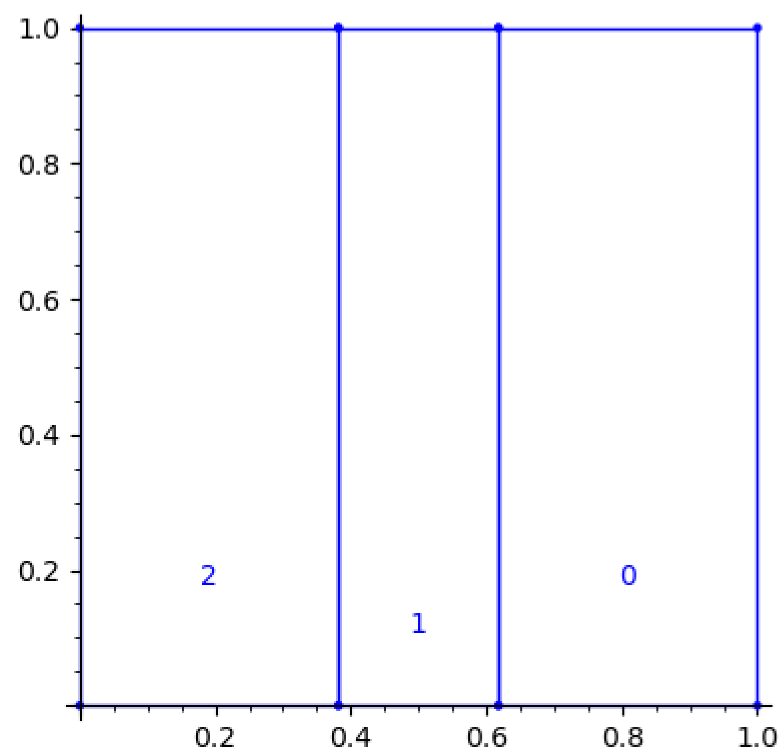
Out[18]:



The polygon partition P_b:

In [19]:
```
Pb = PolyhedronPartition([square])
Pb = Pb.refine_by_hyperplane([-1/phi, 1, 0])
Pb = Pb.refine_by_hyperplane([-1/phi^2, 1, 0])
Pb.plot()
```

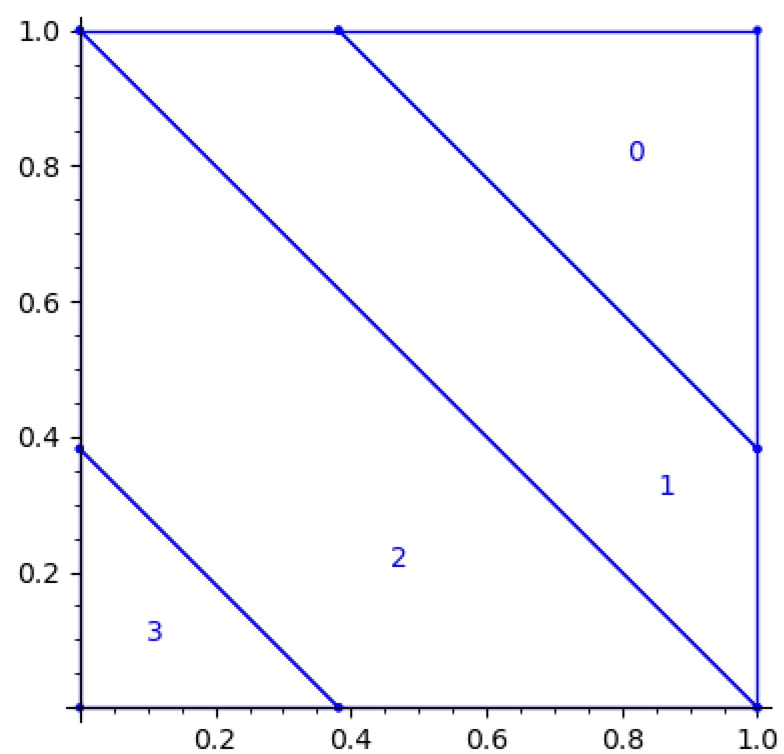
Out[19]:



The polygon partition P_c:

In [20]:
```
Pc = PolyhedronPartition([square])
Pc = Pc.refine_by_hyperplane([-1,1,1])
Pc = Pc.refine_by_hyperplane([-1/phi^2,1,1])
Pc = Pc.refine_by_hyperplane([-1/phi^2-1,1,1])
Pc.plot()
```

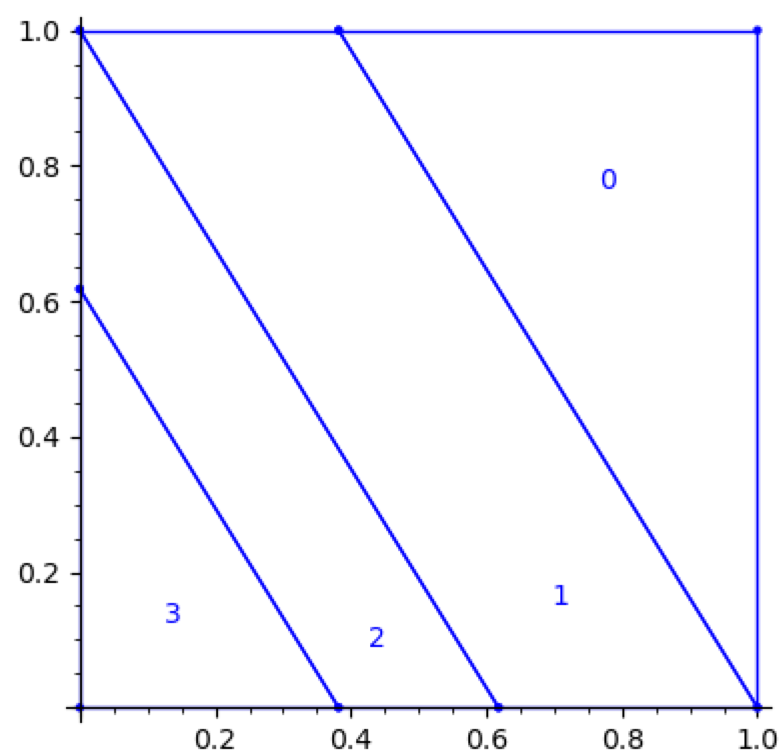
Out[20]:



The polygon partition P_d:

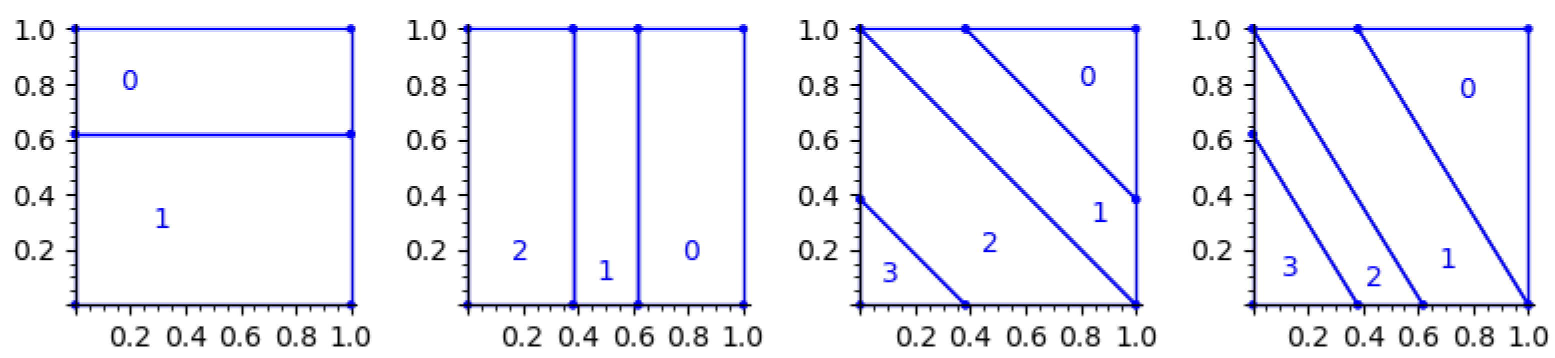
In [21]:
```
Pd = PolyhedronPartition([square])
Pd = Pd.refine_by_hyperplane([-1,phi,1])
Pd = Pd.refine_by_hyperplane([-1/phi,phi,1])
Pd = Pd.refine_by_hyperplane([-1/phi-1,phi,1])
Pd.plot()
```

Out[21]:



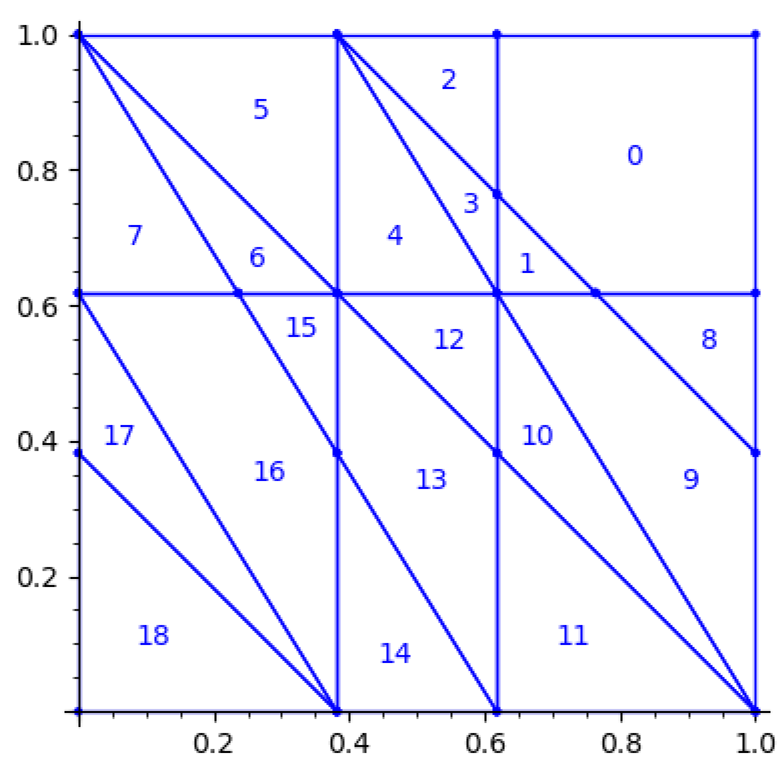
The polygon partitions P_a, P_b, P_c, P_d and their refinement:

In [22]:
```
graphics_array([Pa.plot(), Pb.plot(), Pc.plot(), Pd.plot()]).show(figsize=8)
```



In [23]:
```
Pa.refinement(Pb).refinement(Pc).refinement(Pd).plot()
```

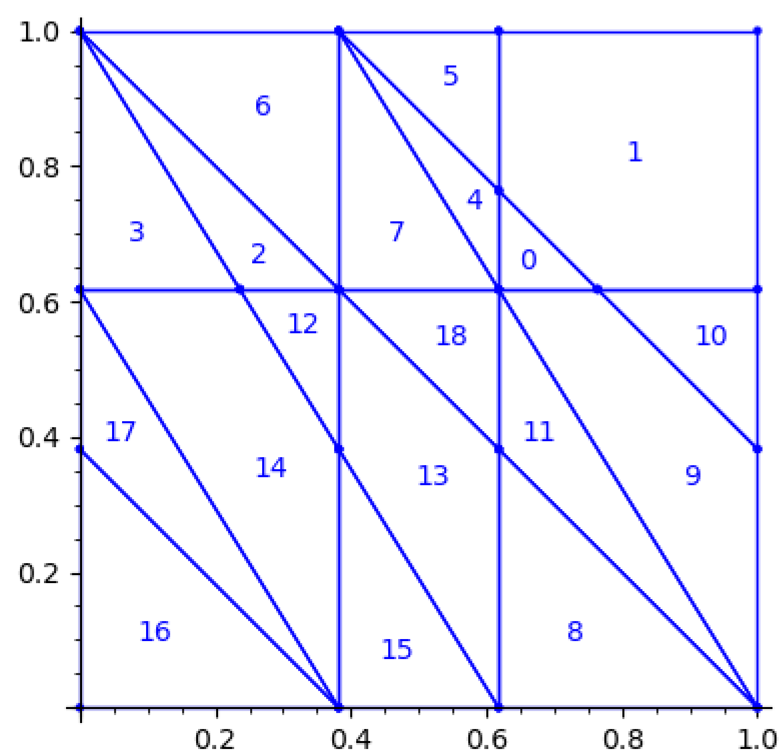
Out[23]:



The partition $\mathcal{P}_{\mathcal{U}}$ using the labelling defined in arXiv:1903.06137

In [24]:
```
from slabbe.arXiv_1903_06137 import self_similar_19_atoms_partition
P0 = PU = self_similar_19_atoms_partition()
P0.plot()
```

Out[24]:



4 - Inducing the partition $\mathcal{P}_\mathcal{U}$ with respect to a toral $\mathbb{Z}^2$-rotation

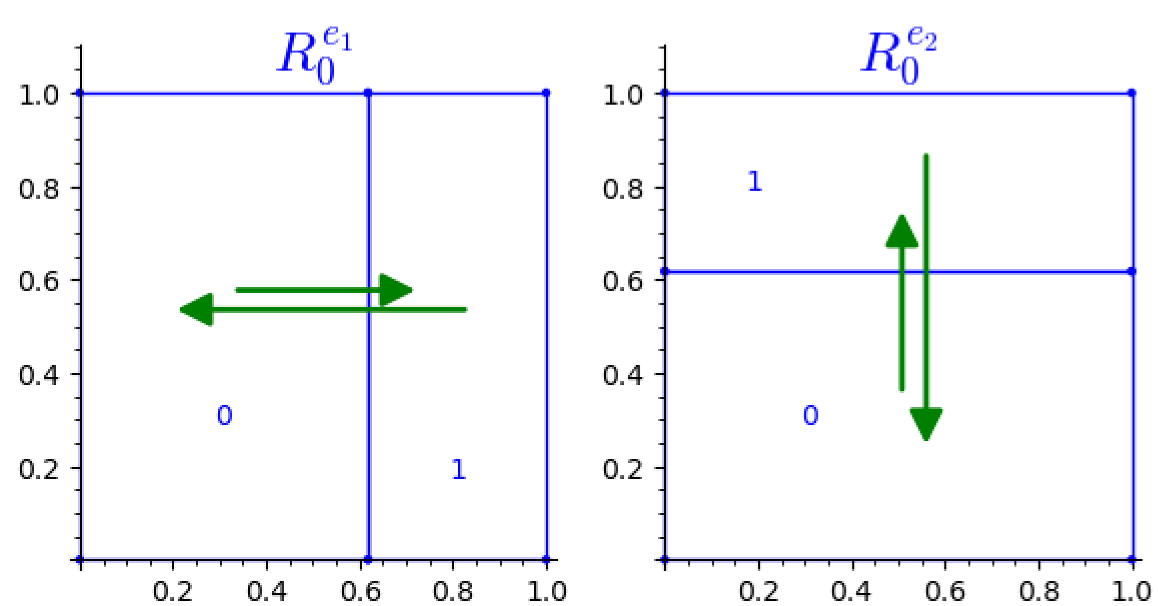
A continuous $\mathbb{Z}^2$-action R_0 on $\mathbb{T}^2$:

$$\begin{array}{rccc} R_0 : & \mathbb{Z}^2 \times \mathbb{T}^2 & \to & \mathbb{T}^2 \\ & (\boldsymbol{n}, \boldsymbol{x}) & \mapsto & \boldsymbol{x} + \varphi^{-2}\boldsymbol{n} \bmod \mathbb{Z}^2 \end{array}$$

In [25]:
```
lattice_base = matrix.column([(1,0), (0,1)])
R0e1 = PET.toral_translation(lattice_base, vector((phi^-2,0)))
R0e2 = PET.toral_translation(lattice_base, vector((0,phi^-2)))
```

In [26]:
```
t1 = title(r"$R_0^{e_1}$", fontsize=20)
t2 = title(r"$R_0^{e_2}$", fontsize=20)
graphics_array([R0e1.plot()+t1, R0e2.plot()+t2])
```

Out[26]:



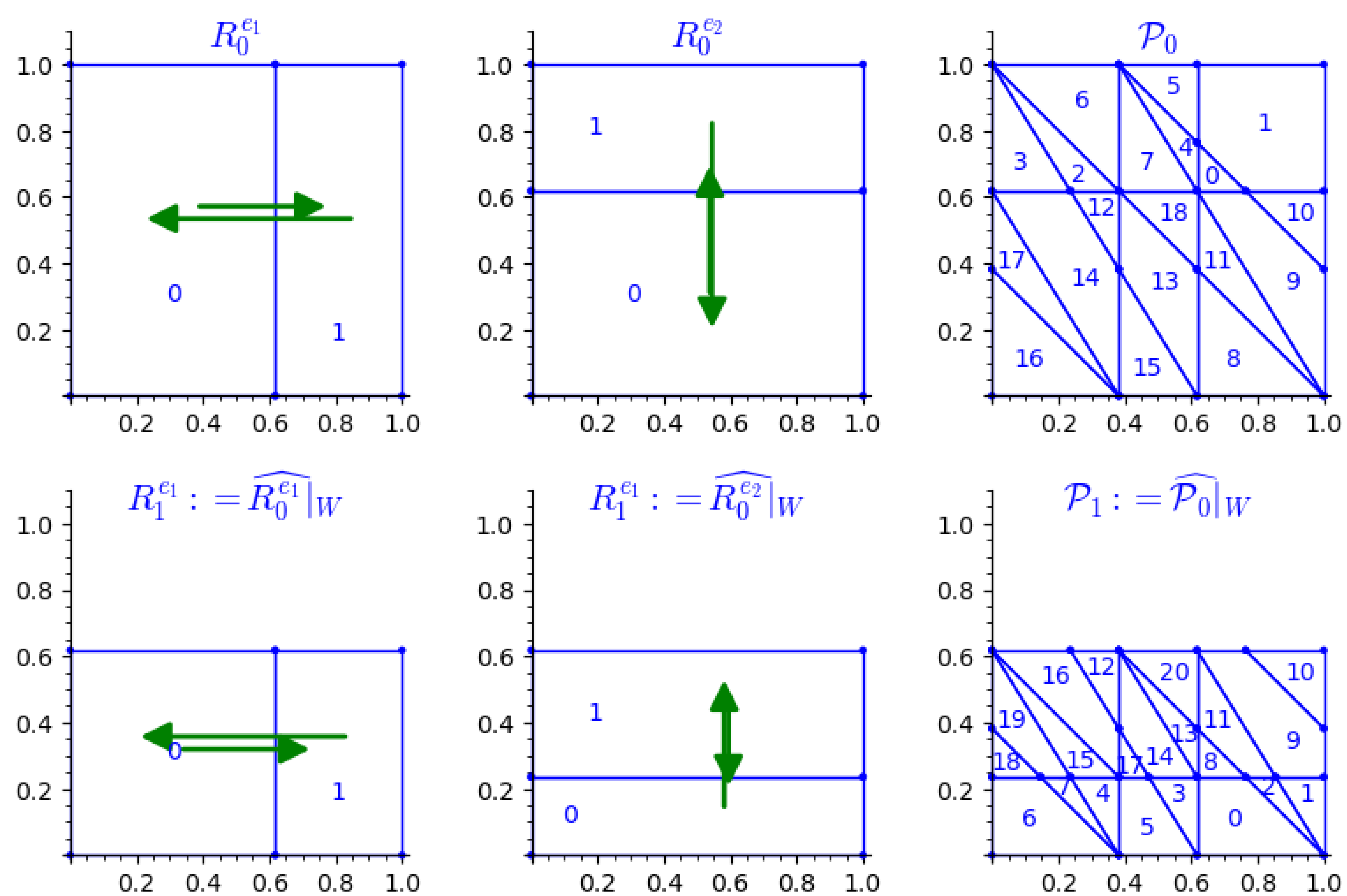
Helper function `please_draw_Rauzy_induction_for_Z2_action`

This is some code to draw induced transformation on the next slide (you may safely ignore what is below).

In [27]:
```
def please_draw_Rauzy_induction_for_Z2_action(T1, T2, P, inducedT1, inducedT2, inducedP, beta,
                                               subscripts=['',''],figsize=9, fontsize=15):
    input_subscript, output_subscript= subscripts
    t1 = title(r'$R^{e_1}%s$'%input_subscript, fontsize=fontsize)
    t2 = title(r'$R^{e_2}%s$'%input_subscript, fontsize=fontsize)
    t3 = title(r'$\mathcal{P}%s$'%input_subscript, fontsize=fontsize)
    t4 = title(r'$R^{e_1}%s:=\widehat{R^{e_1}%s}|_W$'%(output_subscript, input_subscript), fontsize=fontsize)
    t5 = title(r'$R^{e_1}%s:=\widehat{R^{e_2}%s}|_W$'%(output_subscript, input_subscript), fontsize=fontsize)
    t6 = title(r'$\mathcal{P}%s:=\widehat{\mathcal{P}%s}|_W$'%(output_subscript, input_subscript), fontsize=fontsize)
    graphics_array([T1.plot()+bb+t1, T2.plot()+bb+t2, P.plot()+bb+t3,
                    inducedT1.plot()+bb+t4,
                    inducedT2.plot()+bb+t5,
                    inducedP.plot()+bb+t6], ncols=3).show(figsize=figsize)
    show(LatexExpr(r"\text{The substitution is }"+
                   r"\beta{}:{}".format(input_subscript, latex(beta))))
```

Vertical Rauzy induction $\mathcal{X}_{\mathcal{P}_0,R_0} = \overline{\beta_0\left(\mathcal{X}_{\mathcal{P}_1,R_1}\right)}^{\sigma}$

In [28]:
```
y_ineq = [phi^-1, 0, -1] ###   <= phi^-1 (see Polyhedron? for syntax)
R1e1,_ = R0e1.induced_transformation(y_ineq)
R1e2,_ = R0e2.induced_transformation(y_ineq)
P1,beta0 = R0e2.induced_partition(y_ineq, P0, substitution_type='column')
please_draw_Rauzy_induction_for_Z2_action(R0e1, R0e2, PU, R1e1, R1e2, P1, beta0, subscripts=[r'_0','_1'], figsize=8)
```

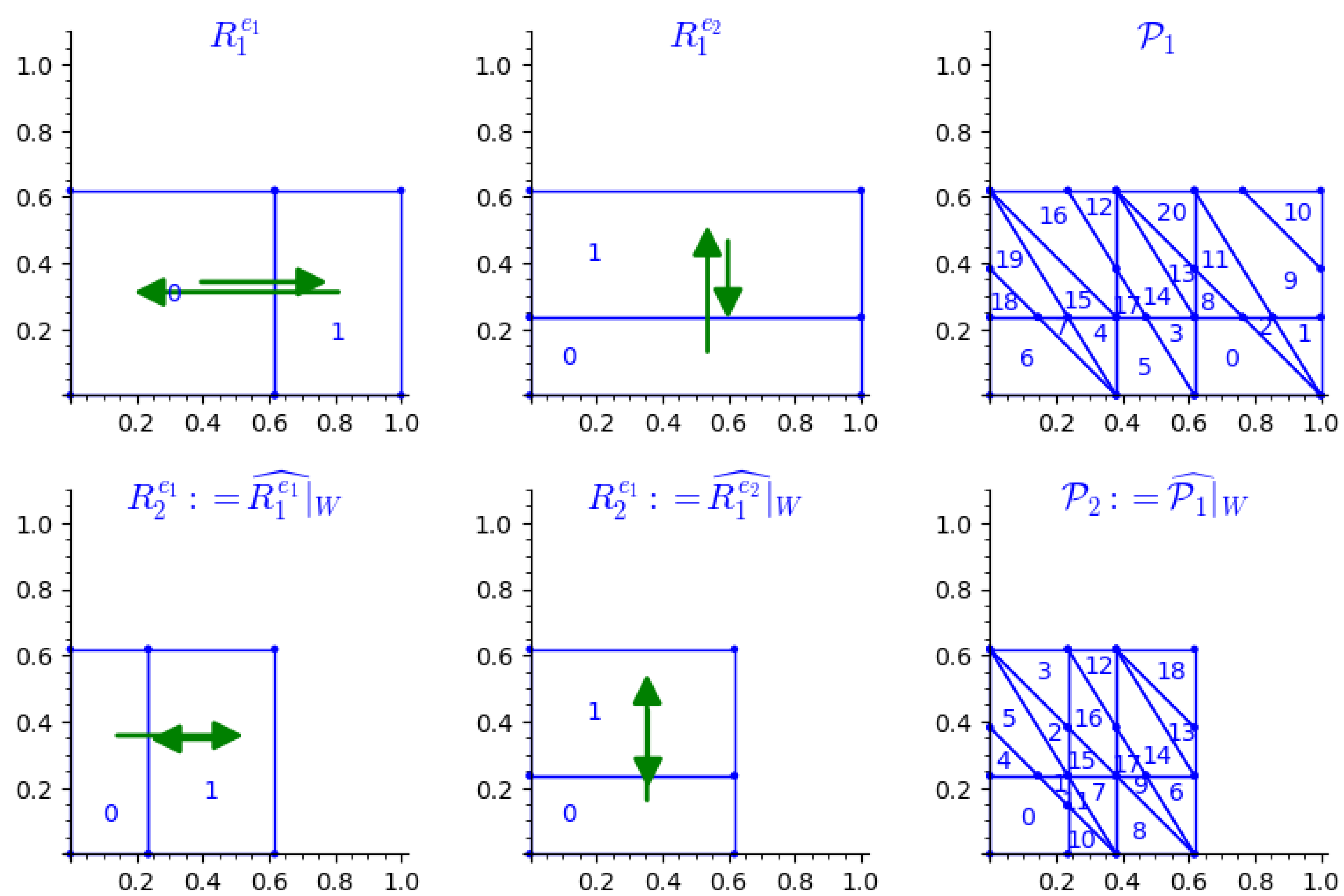


The substitution is β_0

$$: \begin{array}{llllllll}
0 \mapsto (8), & 1 \mapsto (9), & 2 \mapsto (11), & 3 \mapsto (13), & 4 \mapsto (14), & 5 \mapsto (15), & 6 \mapsto (16), & 7 \mapsto (17), \\
8 \mapsto \begin{pmatrix} 0 \\ 8 \end{pmatrix}, & 9 \mapsto \begin{pmatrix} 1 \\ 9 \end{pmatrix}, & 10 \mapsto \begin{pmatrix} 1 \\ 10 \end{pmatrix}, & 11 \mapsto \begin{pmatrix} 1 \\ 11 \end{pmatrix}, & 12 \mapsto \begin{pmatrix} 6 \\ 12 \end{pmatrix}, & 13 \mapsto \begin{pmatrix} 4 \\ 13 \end{pmatrix}, & 14 \mapsto \begin{pmatrix} 7 \\ 13 \end{pmatrix}, & 15 \mapsto \begin{pmatrix} 2 \\ 14 \end{pmatrix}, \\
16 \mapsto \begin{pmatrix} 6 \\ 14 \end{pmatrix}, & 17 \mapsto \begin{pmatrix} 7 \\ 15 \end{pmatrix}, & 18 \mapsto \begin{pmatrix} 3 \\ 16 \end{pmatrix}, & 19 \mapsto \begin{pmatrix} 3 \\ 17 \end{pmatrix}, & 20 \mapsto \begin{pmatrix} 5 \\ 18 \end{pmatrix}. & & &
\end{array}$$

Horizontal Rauzy induction $\mathcal{X}_{\mathcal{P}_1,R_1} = \overline{\beta_1\left(\mathcal{X}_{\mathcal{P}_2,R_2}\right)}^{\sigma}$

In [29]:
```
x_ineq = [phi^-1, -1, 0] ### x <= phi^-1 (see Polyhedron? for syntax)
R2e1,_ = R1e1.induced_transformation(x_ineq)
R2e2,_ = R1e2.induced_transformation(x_ineq)
P2,beta1 = R1e1.induced_partition(x_ineq, P1, substitution_type='row')
please_draw_Rauzy_induction_for_Z2_action(R1e1, R1e2, P1, R2e1, R2e2, P2, beta1, subscripts=[r'_1','_2'], figsize=8)
```



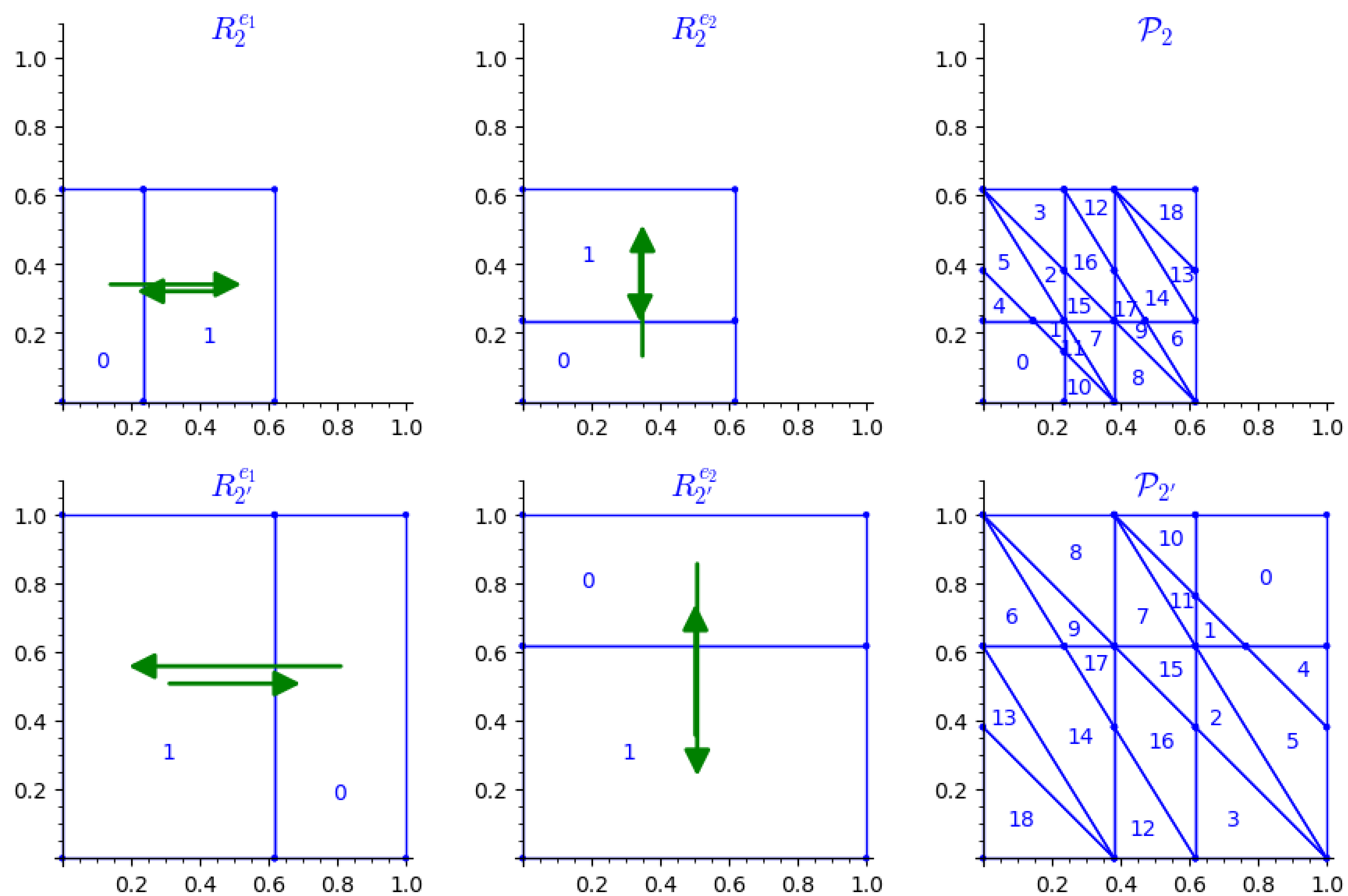
The substitution is β_1

$$: \begin{array}{llllllll} 0 \mapsto (6), & 1 \mapsto (7), & 2 \mapsto (15), & 3 \mapsto (16), & 4 \mapsto (18), & 5 \mapsto (19), & 6 \mapsto (3, 1), & 7 \mapsto (4, 0), \\ 8 \mapsto (5, 0), & 9 \mapsto (5, 2), & 10 \mapsto (6, 0), & 11 \mapsto (7, 0), & 12 \mapsto (12, 9), & 13 \mapsto (13, 9), & 14 \mapsto (14, 9), & 15 \mapsto (15, 8), \\ 16 \mapsto (16, 11), & 17 \mapsto (17, 11), & 18 \mapsto (20, 10). \end{array}$$

Renormalization $\mathcal{X}_{\mathcal{P}_2,R_2} = \mathcal{X}_{\mathcal{P}_{2'},R_{2'}}$

In [30]:
```
R2e1_scaled = (-phi*R2e1).translate_domain((1,1))
R2e2_scaled = (-phi*R2e2).translate_domain((1,1))
P2_scaled = (-phi*P2).translate((1,1))
```

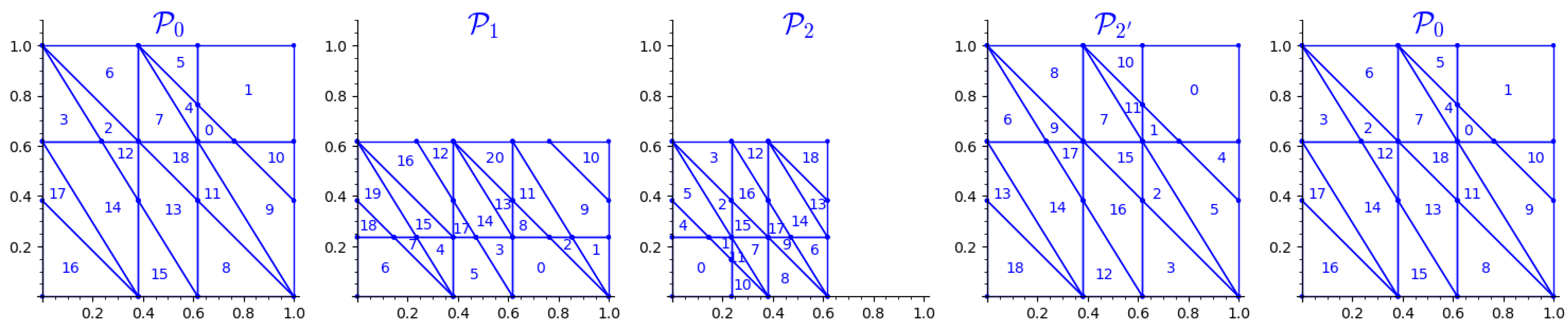
In [31]:
```
t1 = title(r"$R^{e_1}_{2}$", fontsize=15);t2 = title(r"$R^{e_2}_{2}$", fontsize=15);t3 = title(r"$\mathcal{P}_{2}$", fontsize=15
t4 = title(r"$R^{e_1}_{2'}$", fontsize=15);t5 = title(r"$R^{e_2}_{2'}$", fontsize=15);t6 = title(r"$\mathcal{P}_{2'}$", fontsize
graphics_array([R2e1.plot()+bb+t1, R2e2.plot()+bb+t2, P2.plot()+bb+t3,
                R2e1_scaled.plot()+t4, R2e2_scaled.plot()+t5, P2_scaled.plot()+t6], ncols=3).show(figsize=9)
```



Back to the starting partition $\mathcal{P}_0$

We observe that the scaled partition $\mathcal{P}_{2'}$ is the same as $\mathcal{P}_0$ up to a permutation β_2 of the indices of the atoms in such a way that $\mathcal{X}_{\mathcal{P}_{2'},R_{2'}} = \beta_2\left(\mathcal{X}_{\mathcal{P}_0,R_0}\right)$

In [32]:
```
t1 = title(r'$\mathcal{P}_0$', fontsize=20);t2 = title(r'$\mathcal{P}_1$', fontsize=20)
t3 = title(r'$\mathcal{P}_2$', fontsize=20);t4 = title(r"$\mathcal{P}_{2'}$", fontsize=20)
L = [PU.plot()+t1, P1.plot()+bb+t2, P2.plot()+bb+t3, P2_scaled.plot()+t4, PU.plot()+t1]
graphics_array(L).show(figsize=15)
```



In [33]:
```
assert P2_scaled.is_equal_up_to_relabeling(PU)
from slabbe import Substitution2d
beta2 = Substitution2d.from_permutation(PU.keys_permutation(P2_scaled))
show(beta2)
```

$$0 \mapsto (1), \quad 1 \mapsto (0), \quad 2 \mapsto (9), \quad 3 \mapsto (6), \quad 4 \mapsto (11), \quad 5 \mapsto (10), \quad 6 \mapsto (8), \quad 7 \mapsto (7),$$
$$8 \mapsto (3), \quad 9 \mapsto (5), \quad 10 \mapsto (4), \quad 11 \mapsto (2), \quad 12 \mapsto (17), \quad 13 \mapsto (16), \quad 14 \mapsto (14), \quad 15 \mapsto (12),$$
$$16 \mapsto (18), \quad 17 \mapsto (13), \quad 18 \mapsto (15).$$

The self-similarity

In summary, we have

$$\mathcal{X}_{\mathcal{P}_0,R_0} = \overline{\beta_0(\mathcal{X}_{\mathcal{P}_1,R_1})}^{\sigma} = \overline{\beta_0\beta_1(\mathcal{X}_{\mathcal{P}_2,R_2})}^{\sigma} = \overline{\beta_0\beta_1(\mathcal{X}_{\mathcal{P}_{2'},R_{2'}})}^{\sigma} = \overline{\beta_0\beta_1\beta_2\left(\mathcal{X}_{\mathcal{P}_0,R_0}\right)}^{\sigma}$$

with self-similarity $\phi = \beta_0\beta_1\beta_2$:

In [34]:
```
phi_ = beta0 * beta1 * beta2
show(phi_)
```

$$0 \mapsto (17), \quad 1 \mapsto (16), \quad 2 \mapsto (15,\ 11), \quad 3 \mapsto (13,\ 9), \quad 4 \mapsto (17,\ 8), \quad 5 \mapsto (16,\ 8), \quad 6 \mapsto (15,\ 8)$$

$$8 \mapsto \begin{pmatrix} 6 \\ 14 \end{pmatrix}, \quad 9 \mapsto \begin{pmatrix} 3 \\ 17 \end{pmatrix}, \quad 10 \mapsto \begin{pmatrix} 3 \\ 16 \end{pmatrix}, \quad 11 \mapsto \begin{pmatrix} 2 \\ 14 \end{pmatrix}, \quad 12 \mapsto \begin{pmatrix} 7 & 1 \\ 15 & 11 \end{pmatrix}, \quad 13 \mapsto \begin{pmatrix} 6 & 1 \\ 14 & 11 \end{pmatrix}, \quad 14 \mapsto \begin{pmatrix} 7 \\ 13 \end{pmatrix}$$

$$16 \mapsto \begin{pmatrix} 5 & 1 \\ 18 & 10 \end{pmatrix}, \quad 17 \mapsto \begin{pmatrix} 4 & 1 \\ 13 & 9 \end{pmatrix}, \quad 18 \mapsto \begin{pmatrix} 2 & 0 \\ 14 & 8 \end{pmatrix}.$$

Moreover, one can prove (from the study of 2×2 factors) that there is a unique subshift X such that $X = \overline{\phi(X)}^{\sigma}$. Thus

$$\mathcal{X}_{\mathcal{P}_0,R_0} = \mathcal{X}_{\phi}.$$

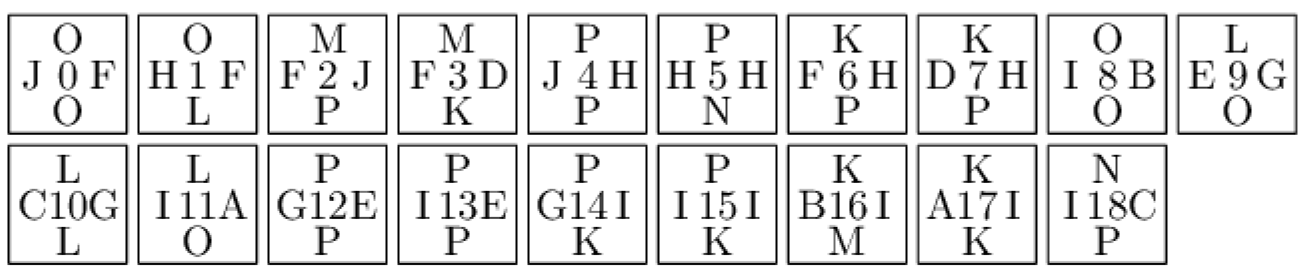
Also ϕ is onto up to a shift and recognizable. Thus $\mathcal{X}_{\phi}$ is aperiodic.

5 - Results

Another characterization of $\mathcal{X}_{\mathcal{P}_0,R_0}$ is the Wang shift $\Omega_{\mathcal{U}} \subseteq [0,18]^{\mathbb{Z}^2}$ defined by a set $\mathcal{U}$ of 19 Wang tiles.

In [35]:
```python
from slabbe import WangTileSet
tiles = ["FOJO", "FOHL", "JMFP", "DMFK", "HPJP", "HPHN", "HKFP", "HKDP",
         "BOIO", "GLEO", "GLCL", "ALIO", "EPGP", "EPIP", "IPGK", "IPIK",
         "IKBM", "IKAK", "CNIP"]
U = WangTileSet([tuple(tile) for tile in tiles])
U.tikz()
```

Out[35]:



which satisfies:

$$\Omega_{\mathcal{U}} = \overline{\alpha_0(\Omega_{\mathcal{V}})}^{\sigma} = \overline{\alpha_0\alpha_1(\Omega_{\mathcal{W}})}^{\sigma} = \overline{\alpha_0\alpha_1\alpha_2(\Omega_{\mathcal{U}})}^{\sigma} = \overline{\phi(\Omega_{\mathcal{U}})}^{\sigma}$$

and

$$\beta_0 = \alpha_0, \quad \beta_1 = \alpha_1, \qquad \beta_2 = \alpha_2.$$

The computation of α_0, α_1 and α_2 is done using subset of marker tiles, see this other 30 minutes talk (online SDA2 meeting, Caen, December 2020) or this chapter arXiv:2012.03892

$\mathcal{P}_{\mathcal{U}}$ is a Markov partition for $\mathbb{Z}^2$-action $R_{\mathcal{U}}$ on $\mathbb{T}^2$

Theorem

- (i) $\mathcal{X}_{\mathcal{P}_{\mathcal{U}},R_{\mathcal{U}}}$ is minimal and aperiodic, and $\mathcal{X}_{\mathcal{P}_{\mathcal{U}},R_{\mathcal{U}}} = \mathcal{X}_{\phi} = \Omega_{\mathcal{U}}$,
- (ii) $\mathcal{P}_{\mathcal{U}}$ is a Markov partition for the dynamical system $(\mathbb{T}^2, \mathbb{Z}^2, R_{\mathcal{U}})$,
- (iii) $(\mathbb{T}^2, \mathbb{Z}^2, R_{\mathcal{U}})$ is the maximal equicontinuous factor of $(\Omega_{\mathcal{U}}, \mathbb{Z}^2, \sigma)$,
- (iv) the set of fiber cardinalities of the factor map $\Omega_{\mathcal{U}} \to \mathbb{T}^2$ is $\{1, 2, 8\}$,
- (v) the dynamical system $(\Omega_{\mathcal{U}}, \mathbb{Z}^2, \sigma)$ is strictly ergodic and the measure-preserving dynamical system $(\Omega_{\mathcal{U}}, \mathbb{Z}^2, \sigma, \nu)$ is isomorphic to $(\mathbb{T}^2, \mathbb{Z}^2, R_{\mathcal{U}}, \lambda)$ where ν is the unique shift-invariant probability measure on $\Omega_{\mathcal{U}}$ and λ is the Haar measure on $\mathbb{T}^2$.

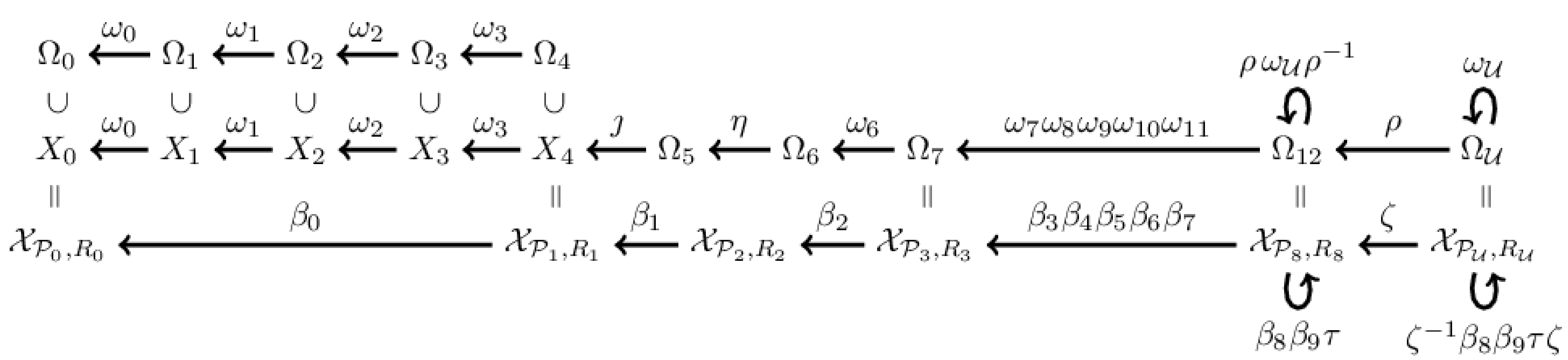
Theorem There exists a 4-to-2 cut and project scheme such that for every configuration $w \in \Omega_{\mathcal{U}}$, the set $Q \subseteq \mathbb{Z}^2$ of occurrences of a pattern in w is a regular model set. If w is a generic (resp. singular) configuration, then Q is a generic (resp. singular) model set.

Both $\mathcal{X}_{\mathcal{P}_\mathcal{U},R_\mathcal{U}}$ and $\Omega_\mathcal{U}$ come from the description of the Jeandel-Rao Wang shift

In [36]:

```
from slabbe import TikzPicture
with open('figure4.tex','r') as f:
    s = f.read()
TikzPicture(s)
```

Out[36]:



- A self-similar aperiodic set of 19 Wang tiles, *Geometriae Dedicata* 201 (2019) 81-109, doi, arXiv:1802.03265
- Substitutive structure of Jeandel-Rao aperiodic tilings. *Discrete Comput. Geom.*, 2019, doi, arXiv:1808.07768
- Markov partitions for toral $\mathbb{Z}^2$-rotations featuring Jeandel-Rao Wang shift and model sets. April 2020. to appear in *Annales Henri Lebesgue*. arXiv:1903.06137v3
- Rauzy induction of polygon partitions and toral $\mathbb{Z}^2$-rotations, last update January 2021, arXiv:1906.01104v3
- Chapter: Three characterizations of a self-similar aperiodic 2-dimensional subshift, Dec 2020, arXiv:2012.03892

Code

- PyPI: https://pypi.org/project/slabbe/ (version 0.6.2, Dec 2020, running with SageMath 9.2)
- documentation: http://www.slabbe.org/docs/
- gitlab: http://gitlab.com/seblabbe/slabbe

Installation:

```
sage -pip install slabbe
```

In case of trouble: email me.

```
In [ ]:
```