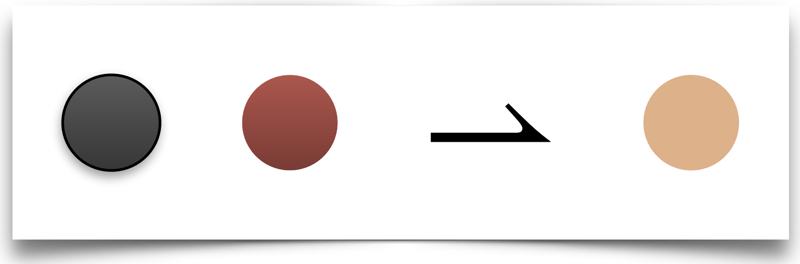
Tracelet Hopf algebras and decomposition spaces

Joint work with ***Joachim Kock (UA Barcelona)***

ACT 2021, University of Cambridge, July 15, 2021

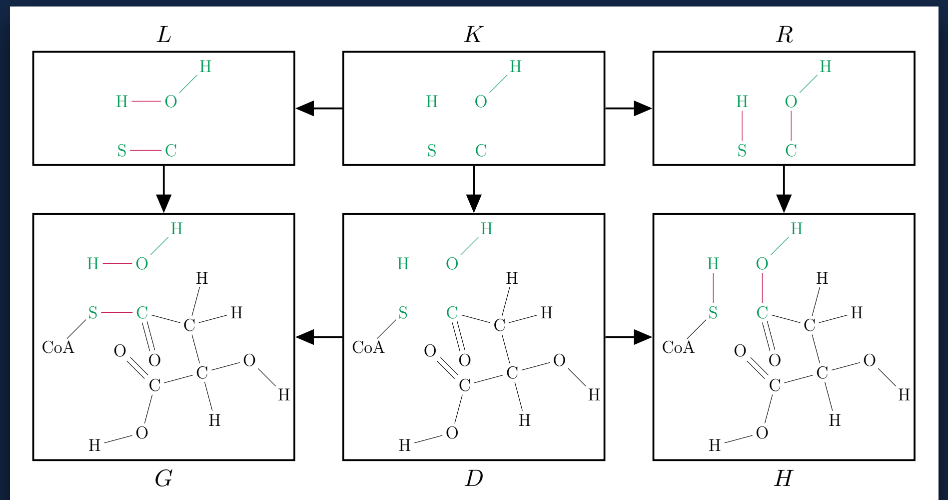
Nicolas Behr

Université de Paris, CNRS, IRIF

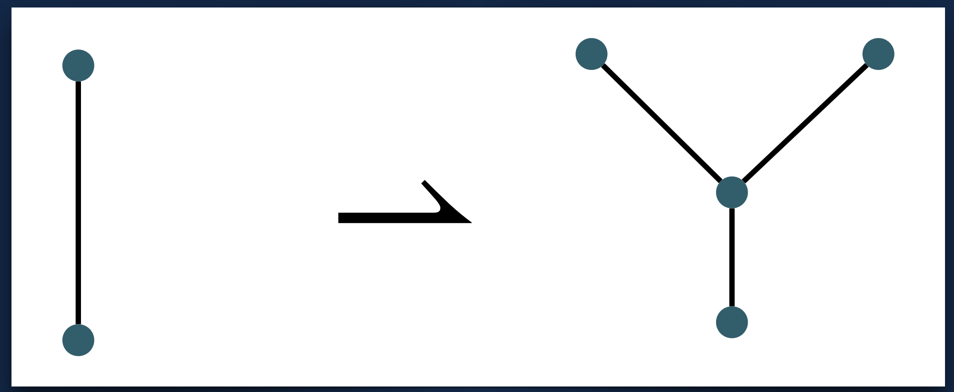


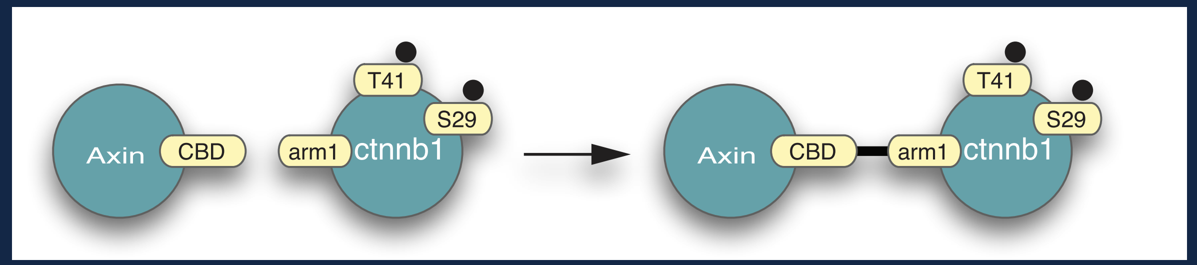
COMPUTER SCIENCE

- **semantics** and **stochastic rewriting theory**
- **concurrency theory**
- **algorithms** for **bio-** and **organo-chemistry**

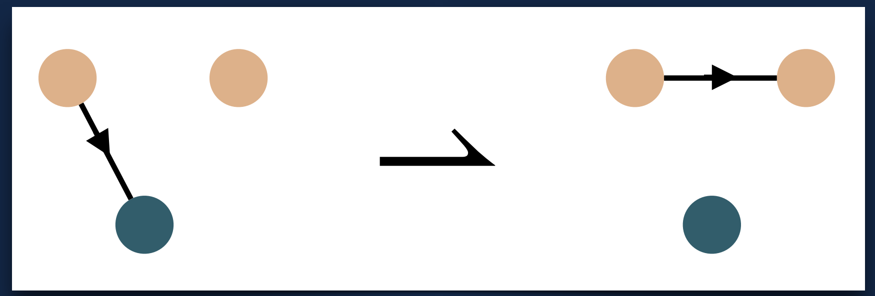


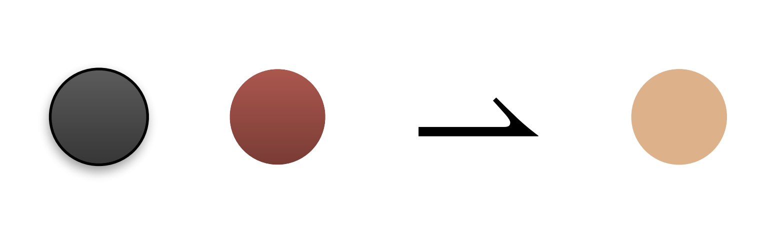
organic chemistry





biochemistry





MATHEMATICAL PHYSICS

- **continuous-time Markov chains (CTMCs)**
- **statistical mechanics**

MATHEMATICS

- enumerative/algebraic **combinatorics**
- theory of **M-adhesive categories**

Plan of the talk

1. Discrete rewriting and diagram Hopf Algebras
2. Categorical rewriting theory
3. From rewriting to tracelets
4. Tracelet decomposition spaces
5. Tracelet Hopf algebras

On the interesting special case of **discrete graph rewriting**

Adv. Theor. Math. Phys. **14** (2010) 1209–1243

Combinatorial algebra for second-quantized Quantum Theory

Pawel Blasiak[1], Gerard H.E. Duchamp[2], Allan I. Solomon[3,4], Andrzej Horzela[1] and Karol A. Penson[3]

The algebras of graph rewriting

Nicolas Behr*[1], Vincent Danos†[2], Ilias Garnier‡[1] and Tobias Heindel§[3]

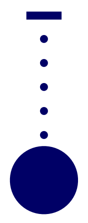
[1]Laboratory for Foundations of Computer Science, School of Informatics, University of Edinburgh, Informatics Forum, 10 Crichton Street, Edinburgh, EH8 9AB, Scotland, UK
[2]LFCS, CNRS & Équipe Antique, Département d'Informatique de l'École Normale Supérieure Paris, 45 rue d'Ulm, 75230 Paris Cedex 05, France
[3]Department of Computer Science, Datalogisk Institut (DIKU), Københavns Universitet, Universitetsparken 5, 2100 København Ø, Denmark

December 20, 2016

On the interesting special case of **discrete graph rewriting**

Idea: represent transformations of **discrete** (= vertex-only) **graphs** as a certain form of **diagrams**

Elementary "one-step" diagrams:

- **Create** a vertex: $v^{\dagger} \;\hat{=}$  output: a vertex
- **Delete** a vertex: $v \;\hat{=}$  input: a vertex

Adv. Theor. Math. Phys. **14** (2010) 1209–1243

Combinatorial algebra for second-quantized Quantum Theory

Pawel Blasiak[1], Gerard H.E. Duchamp[2], Allan I. Solomon[3,4], Andrzej Horzela[1] and Karol A. Penson[3]

The algebras of graph rewriting

Nicolas Behr*[1], Vincent Danos†[2], Ilias Garnier‡[1] and Tobias Heindel§[3]

[1]Laboratory for Foundations of Computer Science, School of Informatics, University of Edinburgh, Informatics Forum, 10 Crichton Street, Edinburgh, EH8 9AB, Scotland, UK
[2]LFCS, CNRS & Équipe Antique, Département d'Informatique de l'École Normale Supérieure Paris, 45 rue d'Ulm, 75230 Paris Cedex 05, France
[3]Department of Computer Science, Datalogisk Institut (DIKU), Københavns Universitet, Universitetsparken 5, 2100 København Ø, Denmark

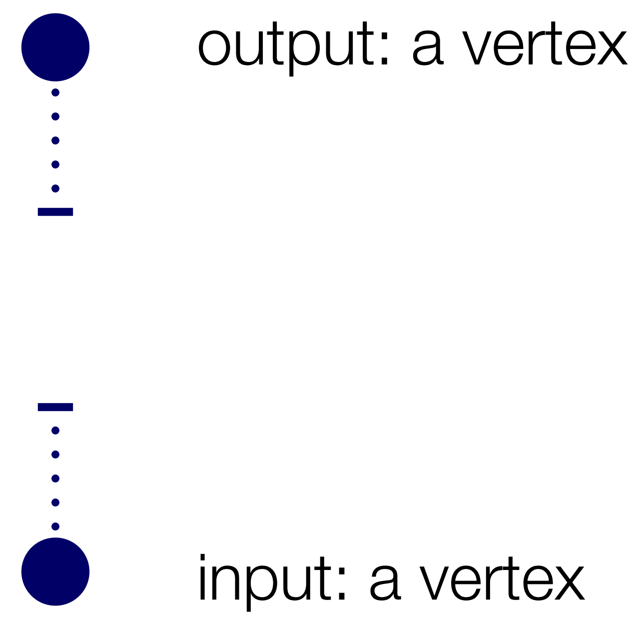
December 20, 2016

On the interesting special case of **discrete graph rewriting**

Idea: represent transformations of **discrete** (= vertex-only) **graphs** as a certain form of **diagrams**

Elementary "one-step" diagrams:

- **Create** a vertex: $v^\dagger \quad \hat{=}$
- **Delete** a vertex: $v \quad \hat{=}$



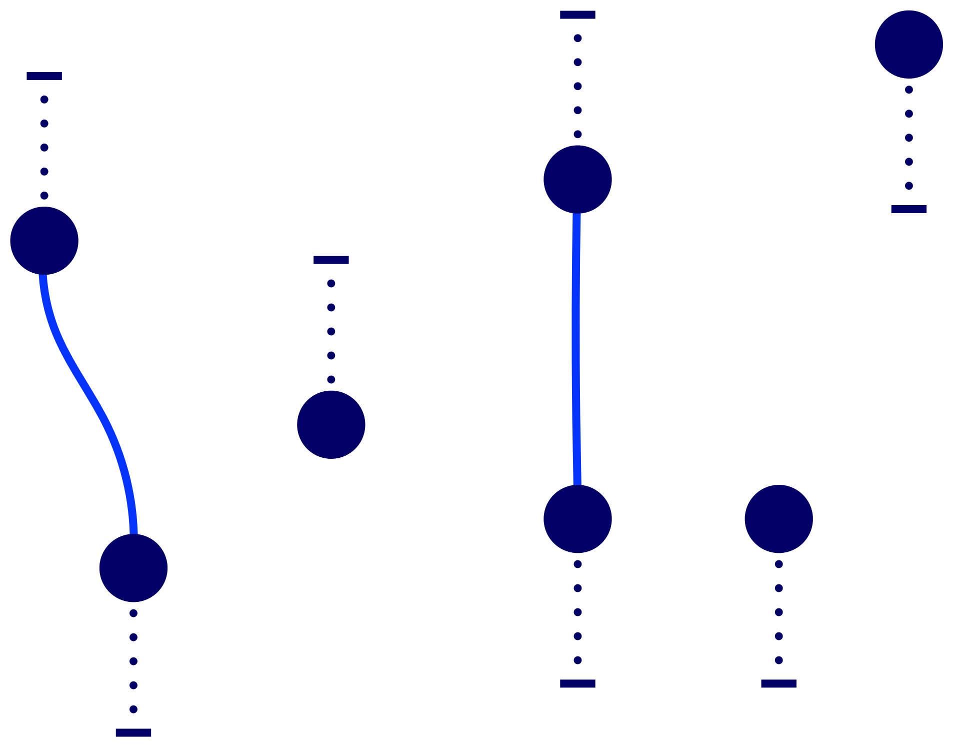
Generic diagrams:

$d = [(O, I, m)]_\sim$

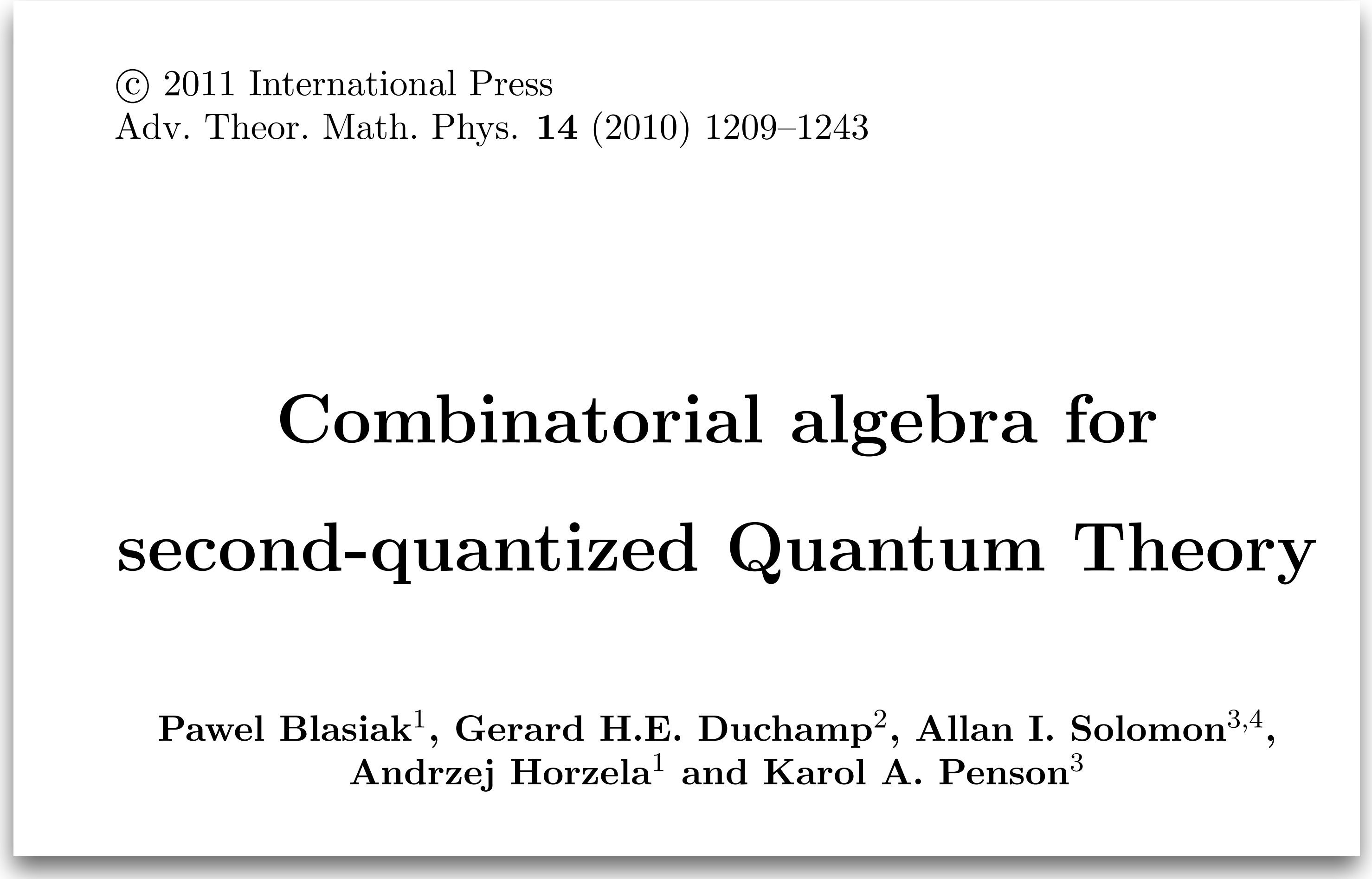
O — set of **output** vertices

I — set of **input** vertices

$m \subseteq O \times I$ — (one-to-one) binary relation

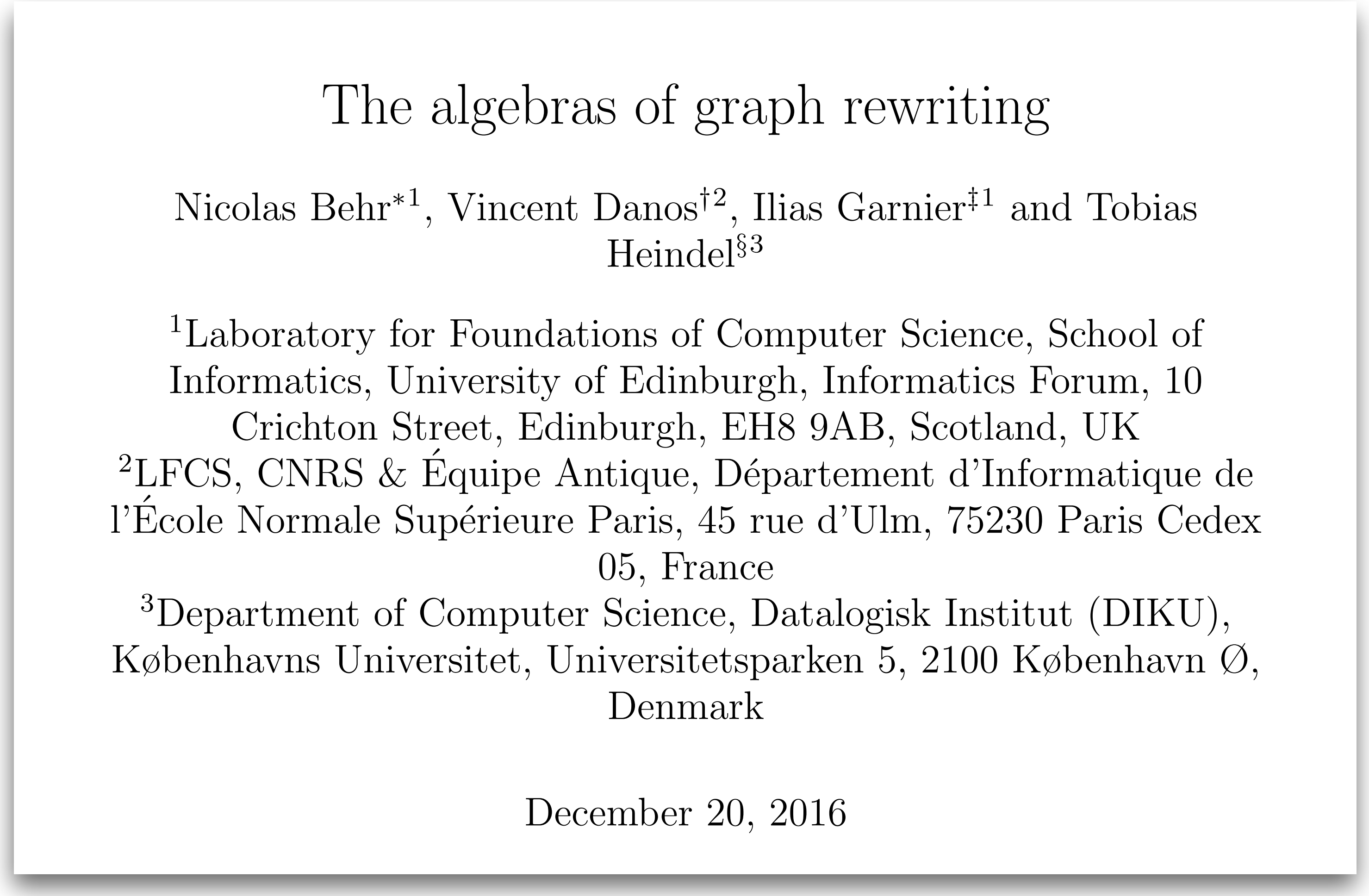


$$(O, I, m) \sim (O', I', m') \quad :\Leftrightarrow \quad \exists (\omega : O \xrightarrow{\cong} O'), (\iota : I \xrightarrow{\cong} I') : \ \big((o, i) \in m \Leftrightarrow (\omega(o), \iota(i)) \in m'\big)$$


© 2011 International Press
Adv. Theor. Math. Phys. **14** (2010) 1209–1243

Combinatorial algebra for second-quantized Quantum Theory

Pawel Blasiak[1], Gerard H.E. Duchamp[2], Allan I. Solomon[3,4], Andrzej Horzela[1] and Karol A. Penson[3]


The algebras of graph rewriting

Nicolas Behr*[1], Vincent Danos†[2], Ilias Garnier‡[1] and Tobias Heindel§[3]

[1]Laboratory for Foundations of Computer Science, School of Informatics, University of Edinburgh, Informatics Forum, 10 Crichton Street, Edinburgh, EH8 9AB, Scotland, UK
[2]LFCS, CNRS & Équipe Antique, Département d'Informatique de l'École Normale Supérieure Paris, 45 rue d'Ulm, 75230 Paris Cedex 05, France
[3]Department of Computer Science, Datalogisk Institut (DIKU), Københavns Universitet, Universitetsparken 5, 2100 København Ø, Denmark

December 20, 2016

On the interesting special case of **discrete graph rewriting**

Notation: let D denote the **set of equivalence classes** $d = [(O, I, m)]_{\sim}$ **of diagrams**

Idea: define a **vector space** $\mathscr{D} \equiv (\mathscr{D}, +, \cdot) := \text{span}_{\mathbb{K}}(D)$ (with $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$), and denote the **basis vector** labelled by $d \in D$ with $\delta(d) \in \mathscr{D}$

On the interesting special case of **discrete graph rewriting**

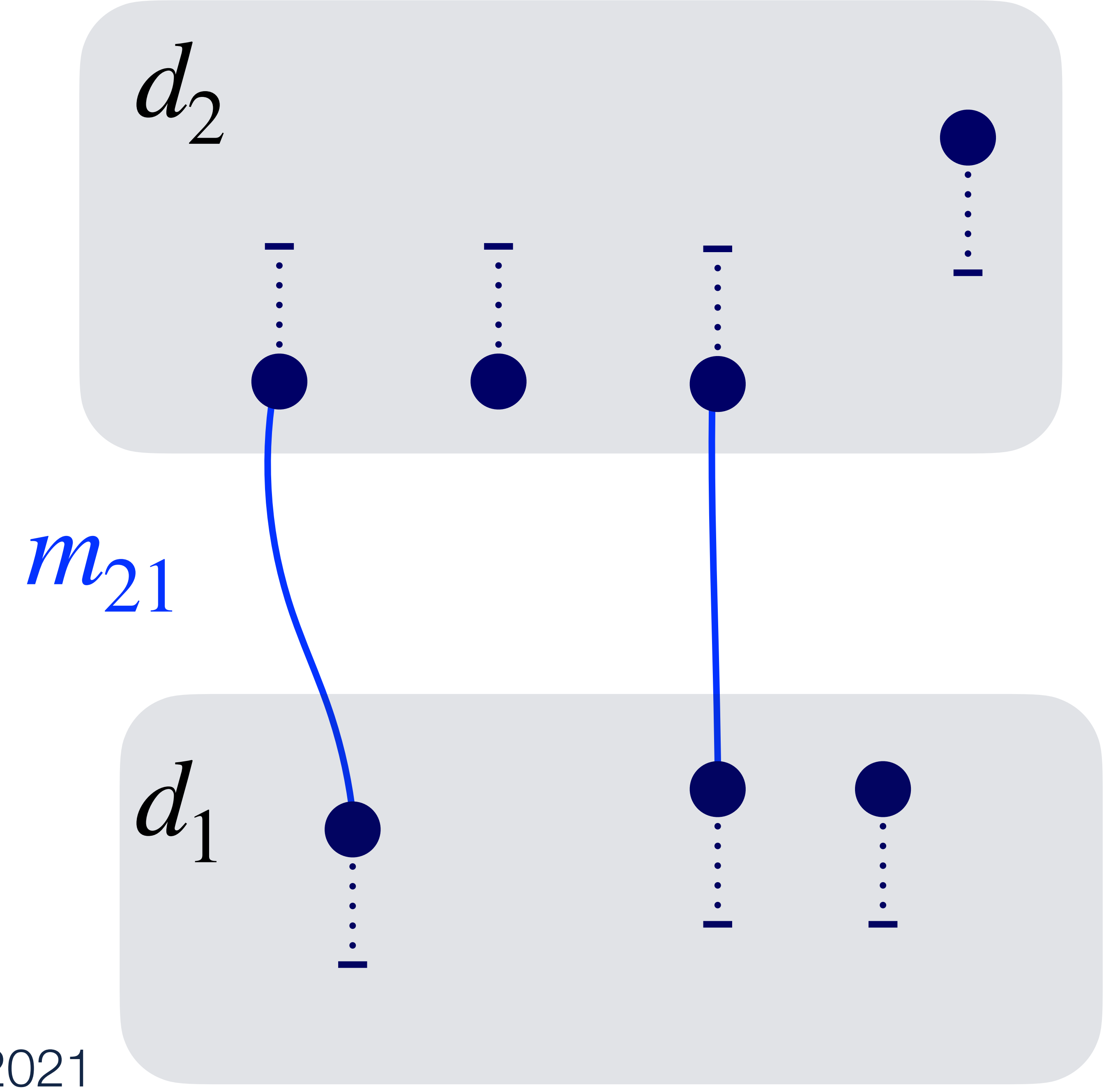
Notation: let D denote the **set of equivalence classes** $d = [(O, I, m)]_{\sim}$ **of diagrams**

Idea: define a **vector space** $\mathscr{D} \equiv (\mathscr{D}, +, \cdot) := \text{span}_{\mathbb{K}}(D)$ (with $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$), and denote the **basis vector** labelled by $d \in D$ with $\delta(d) \in \mathscr{D}$

Diagrammatic composition:

$$\delta(d_2) *_{\mathscr{D}} \delta(d_1) := \sum_{m_{21} \in \mathscr{M}_{d_2}(d_1)} \delta\left(d_2 \triangleleft_{m_{21}} d_1\right), \qquad d_2 \triangleleft_{m_{21}} d_1 := [(O_2 + O_1, I_2 + I_1, m_2 + m_{21} + m_1]_{\sim}$$

matchings (i.e. one-to-one mappings) of **outputs of** d_2 into **inputs of** d_1



On the interesting special case of **discrete graph rewriting**

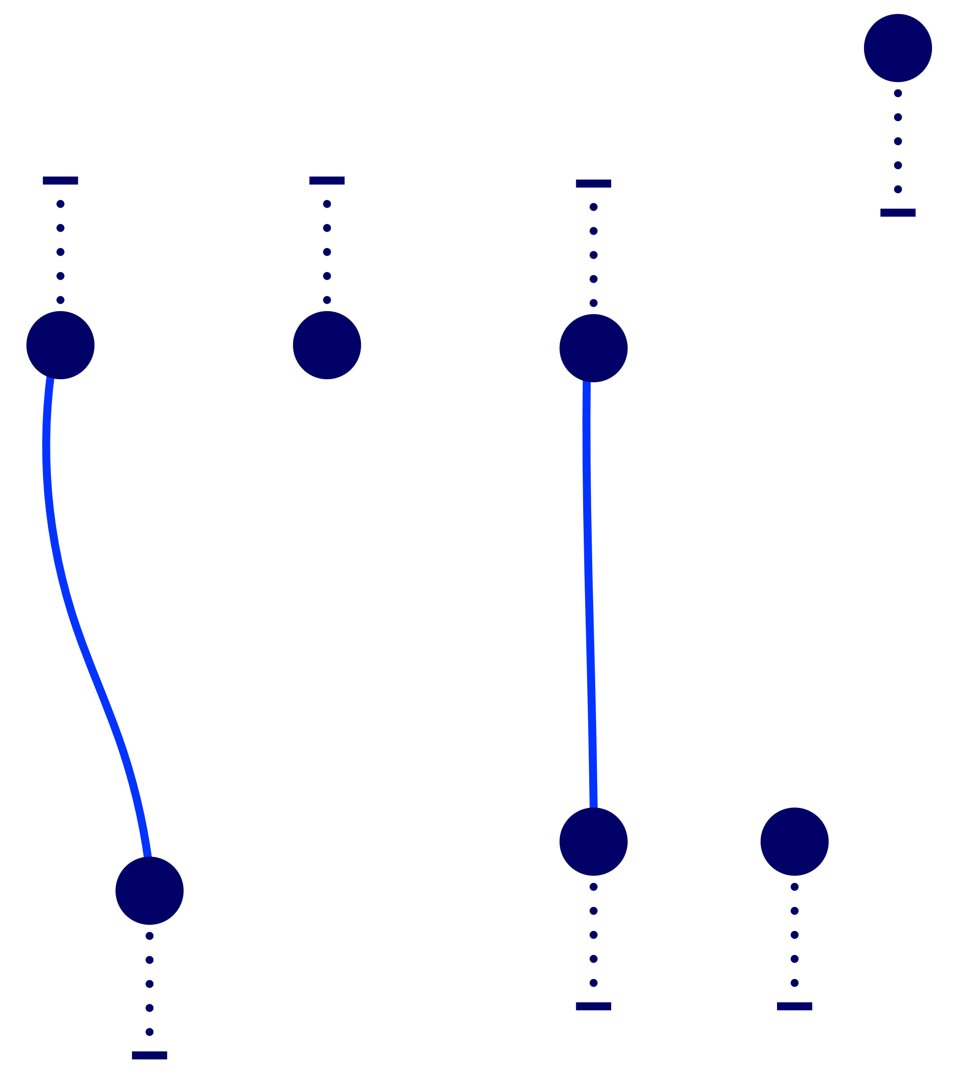
Notation: let D denote the **set of equivalence classes** $d = [(O, I, m)]_{\sim}$ **of diagrams**

Idea: define a **vector space** $\mathscr{D} \equiv (\mathscr{D}, +, \cdot) := \text{span}_{\mathbb{K}}(D)$ (with $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$), and denote the **basis vector** labelled by $d \in D$ with $\delta(d) \in \mathscr{D}$

Diagrammatic composition:

$$\delta(d_2) *_{\mathscr{D}} \delta(d_1) := \sum_{m_{21} \in \mathscr{M}_{d_2}(d_1)} \delta\left(d_2 \triangleleft_{m_{21}} d_1\right), \qquad d_2 \triangleleft_{m_{21}} d_1 := [(O_2 + O_1, I_2 + I_1, m_2 + m_{21} + m_1]_{\sim}$$

matchings (i.e. one-to-one mappings) of **outputs of** d_2 into **inputs of** d_1



On the interesting special case of **discrete graph rewriting**

Notation: let D denote the **set of equivalence classes** $d = [(O, I, m)]_{\sim}$ **of diagrams**

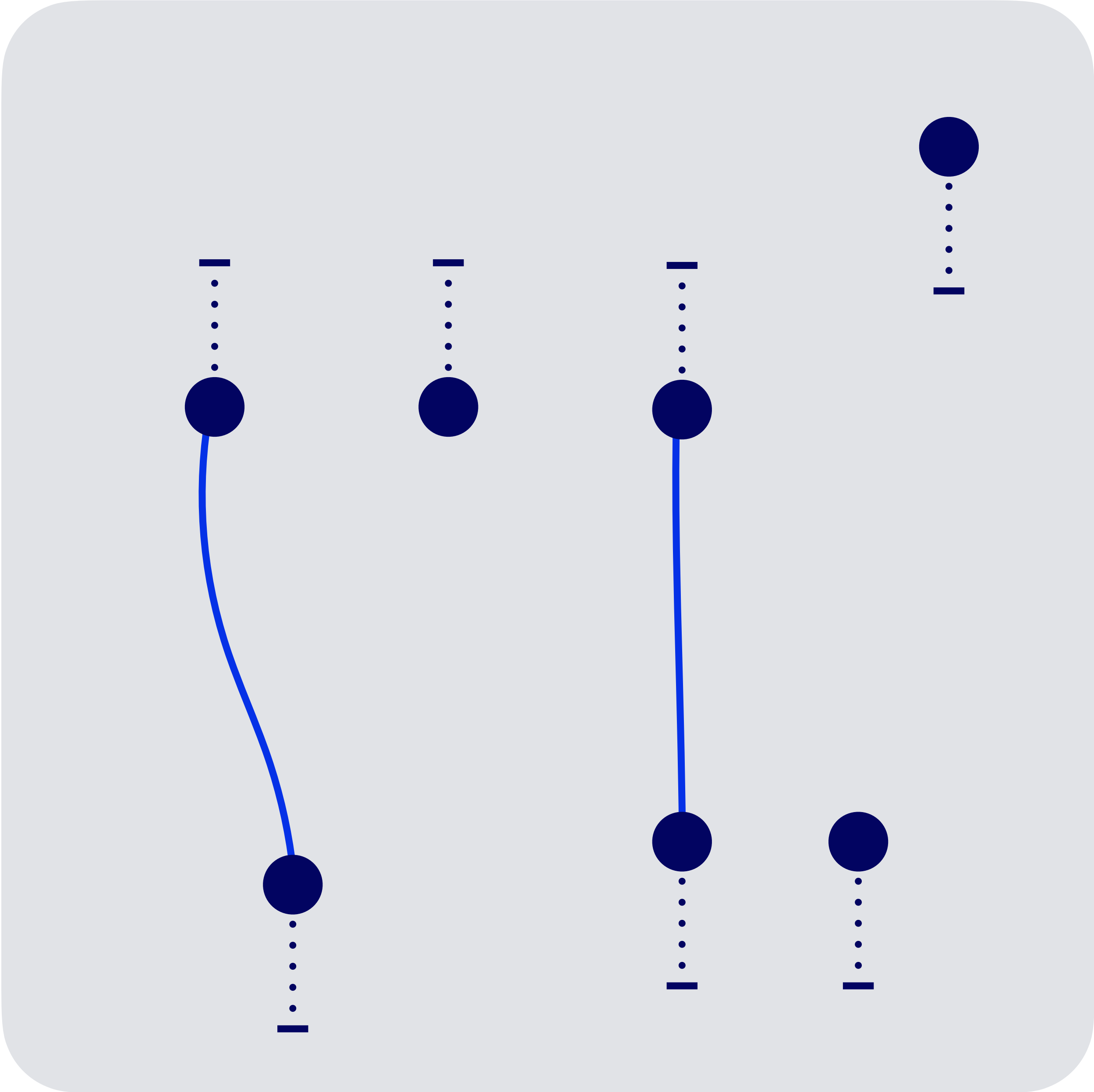
Idea: define a **vector space** $\mathscr{D} \equiv (\mathscr{D}, +, \cdot) := \text{span}_{\mathbb{K}}(D)$ (with $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$), and denote the **basis vector** labelled by $d \in D$ with $\delta(d) \in \mathscr{D}$

Diagrammatic composition:

$$\delta(d_2) *_{\mathscr{D}} \delta(d_1) := \sum_{m_{21} \in \mathscr{M}_{d_2}(d_1)} \delta\left(d_2 \triangleleft_{m_{21}} d_1\right), \qquad d_2 \triangleleft_{m_{21}} d_1 := [(O_2 + O_1, I_2 + I_1, m_2 + m_{21} + m_1]_{\sim}$$

matchings (i.e. one-to-one mappings) of **outputs of** d_2 into **inputs of** d_1

$$d_2 \triangleleft_{m_{21}} d_1 =$$



On the interesting special case of **discrete graph rewriting**

Notation: let D denote the **set of equivalence classes** $d = [(O, I, m)]_{\sim}$ **of diagrams**

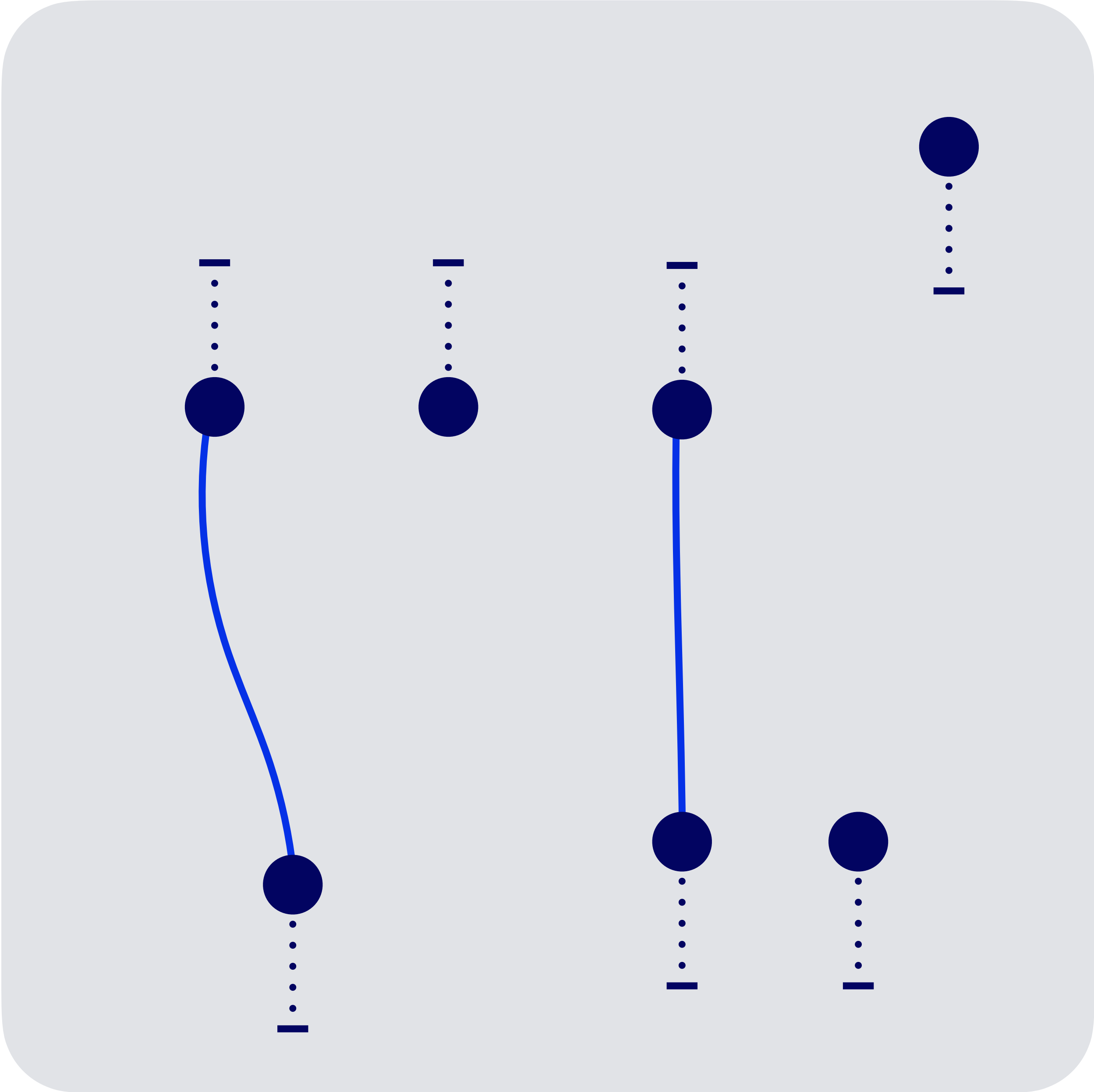
Idea: define a **vector space** $\mathscr{D} \equiv (\mathscr{D}, +, \cdot) := \mathrm{span}_{\mathbb{K}}(D)$ (with $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$), and denote the **basis vector** labelled by $d \in D$ with $\delta(d) \in \mathscr{D}$

Diagrammatic composition:

$$\delta(d_2) *_{\mathscr{D}} \delta(d_1) := \sum_{m_{21} \in \mathscr{M}_{d_2}(d_1)} \delta\left(d_2 \triangleleft_{m_{21}} d_1\right), \qquad d_2 \triangleleft_{m_{21}} d_1 := [(O_2 + O_1, I_2 + I_1, m_2 + m_{21} + m_1]_{\sim}$$

matchings (i.e. one-to-one mappings) of **outputs of** d_2 into **inputs of** d_1

$d_2 \triangleleft_{m_{21}} d_1 =$



Theorem

$(\mathscr{D}, *_{\mathscr{D}})$ is an **associative unital algebra**, with **unit element** $d_{\varnothing} := \delta([(\varnothing, \varnothing, \varnothing)]_{\sim})$

On the interesting special case of **discrete graph rewriting**

Notation: let D denote the **set of equivalence classes** $d = [(O, I, m)]_{\sim}$ **of diagrams**

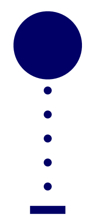
Idea: define a **vector space** $\mathscr{D} \equiv (\mathscr{D}, +, \cdot) := \text{span}_{\mathbb{K}}(D)$ (with $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$), and denote the **basis vector** labelled by $d \in D$ with $\delta(d) \in \mathscr{D}$

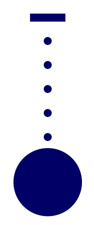
On the interesting special case of **discrete graph rewriting**

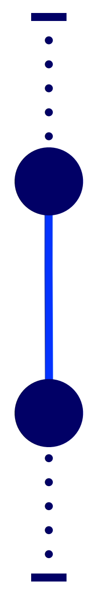
Notation: let D denote the **set of equivalence classes** $d = [(O, I, m)]_\sim$ **of diagrams**

Idea: define a **vector space** $\mathscr{D} \equiv (\mathscr{D}, +, \cdot) := \mathrm{span}_{\mathbb{K}}(D)$ (with $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$), and denote the **basis vector** labelled by $d \in D$ with $\delta(d) \in \mathscr{D}$

Elementary diagrams:

$v^\dagger := [(\{\bullet\}, \varnothing, \varnothing)]_\sim \quad \hat{=}$ 

$v := [(\varnothing, \{\bullet\}, \varnothing)]_\sim \quad \hat{=}$ 

$e := [(\{\bullet\}, \{\bullet\}, \{(\bullet, \bullet)\})]_\sim \quad \hat{=}$ 

On the interesting special case of **discrete graph rewriting**

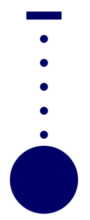
Notation: let D denote the **set of equivalence classes** $d = [(O, I, m)]_\sim$ **of diagrams**

Idea: define a **vector space** $\mathscr{D} \equiv (\mathscr{D}, +, \cdot) := \text{span}_{\mathbb{K}}(D)$ (with $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$), and denote the **basis vector** labelled by $d \in D$ with $\delta(d) \in \mathscr{D}$

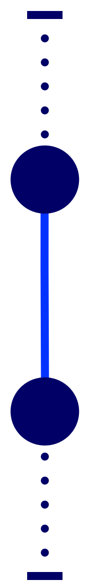
Elementary diagrams:

$$v^\dagger := [(\{\bullet\}, \emptyset, \emptyset)]_\sim \quad \hat{=}$$

$$v := [(\emptyset, \{\bullet\}, \emptyset)]_\sim \quad \hat{=}$$



$$e := [(\{\bullet\}, \{\bullet\}, \{(\bullet, \bullet)\})]_\sim \quad \hat{=}$$



Notation: disjoint union on diagrams $\quad d_2 \uplus d_1 := [(O_2 + O_1, I_2 + I_1, m_2 + m_1)]_\sim \; = \; d_2 \triangleleft_\emptyset d_1$

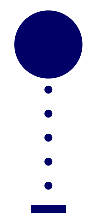
$\Rightarrow$ every equivalence class d may be completely characterized by its **"connected components"**, in the sense that

$$\forall d \in D : \exists k, \ell, m \in \mathbb{Z}_{\geq 0} : d = d_{k,\ell,m}\,, \quad d_{k,\ell,m} := v^{\dagger\, \uplus k} \uplus v^{\uplus \ell} \uplus e^{\uplus m}$$

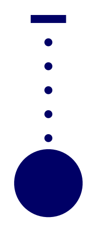
On the interesting special case of **discrete graph rewriting**

Elementary diagrams:

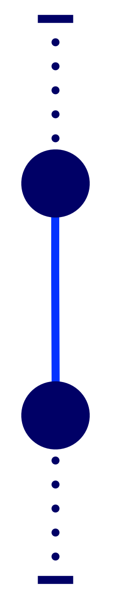
$$v^\dagger := [(\{\,\bullet\,\}, \varnothing, \varnothing)]_\sim \quad \hat{=}$$



$$v := v^\dagger := [(\varnothing, \{\,\bullet\,\}, \varnothing)]_\sim \quad \hat{=}$$



$$e := [(\{\,\bullet\,\}, \{\,\bullet\,\}, \{(\,\bullet\,,\,\bullet\,)\})]_\sim \quad \hat{=}$$



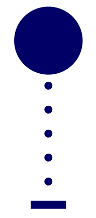
Heisenberg-Lie algebra

$\mathscr{L}_{\mathscr{D}} := (\{\delta(v), \delta(v^\dagger), \delta(e)\}, [\,.\,,.\,])$ (with $[A,B] := A *_{\mathscr{D}} B - B *_{\mathscr{D}} A$), with the only non-zero commutator given by $[\delta(v), \delta(v^\dagger)] = \delta(e)$.

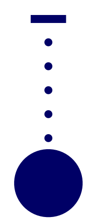
On the interesting special case of **discrete graph rewriting**

Elementary diagrams:

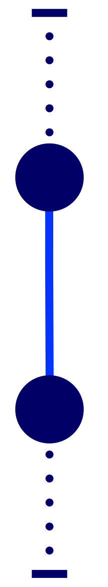
$$v^{\dagger} := [(\{\,\bullet\,\}, \varnothing, \varnothing)]_{\sim} \quad \hat{=}$$



$$v := v^{\dagger} := [(\varnothing, \{\,\bullet\,\}, \varnothing)]_{\sim} \quad \hat{=}$$



$$e := [(\{\,\bullet\,\}, \{\,\bullet\,\}, \{(\,\bullet\,,\,\bullet\,)\})]_{\sim} \quad \hat{=}$$



Heisenberg-Lie algebra

$\mathscr{L}_{\mathscr{D}} := (\{\delta(v), \delta(v^{\dagger}), \delta(e)\}, [\,.\,,\,.\,])$ (with $[A, B] := A *_{\mathscr{D}} B - B *_{\mathscr{D}} A$), with the only non-zero commutator given by $[\delta(v), \delta(v^{\dagger})] = \delta(e)$.

Poincaré-Birkhoff-Witt Theorem

The **universal enveloping algebra of the Heisenberg-Lie algebra**,

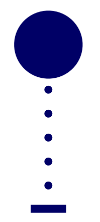
$$\mathscr{U}(\mathscr{L}_{\mathscr{D}}) := \frac{T(\mathscr{L}_{\mathscr{D}})}{\langle \delta(v) \otimes \delta(v^{\dagger}) - \delta(v^{\dagger}) \otimes \delta(v) - \delta(e) \rangle}$$

has a **normal-ordered basis** with elements of the form $U_{k,l,m} := \delta(v^{\dagger})^{\otimes k} \otimes \delta(v)^{\otimes \ell} \otimes \delta(e)^{\otimes m} \quad (k, l, m \in \mathbb{Z}_{\geq 0})$

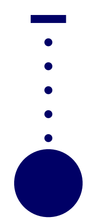
On the interesting special case of **discrete graph rewriting**

Elementary diagrams:

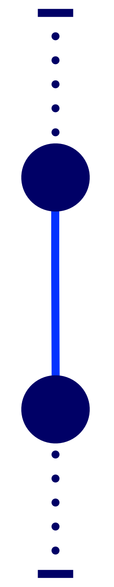
$$v^\dagger := [(\{\bullet\}, \varnothing, \varnothing)]_\sim \quad \hat{=}$$



$$v := v^\dagger := [(\varnothing, \{\bullet\}, \varnothing)]_\sim \quad \hat{=}$$



$$e := [(\{\bullet\}, \{\bullet\}, \{(\bullet, \bullet)\})]_\sim \quad \hat{=}$$



Heisenberg-Lie algebra

$\mathscr{L}_{\mathscr{D}} := (\{\delta(v), \delta(v^\dagger), \delta(e)\}, [\,.\,,.\,])$ (with $[A, B] := A *_{\mathscr{D}} B - B *_{\mathscr{D}} A$), with the only non-zero commutator given by $[\delta(v), \delta(v^\dagger)] = \delta(e)$.

Poincaré-Birkhoff-Witt Theorem

The **universal enveloping algebra of the Heisenberg-Lie algebra**,

$$\mathscr{U}(\mathscr{L}_{\mathscr{D}}) := \frac{T(\mathscr{L}_{\mathscr{D}})}{\langle \delta(v) \otimes \delta(v^\dagger) - \delta(v^\dagger) \otimes \delta(v) - \delta(e) \rangle}$$

has a **normal-ordered basis** with elements of the form $U_{k,l,m} := \delta(v^\dagger)^{\otimes k} \otimes \delta(v)^{\otimes \ell} \otimes \delta(e)^{\otimes m} \quad (k, l, m \in \mathbb{Z}_{\geq 0})$

On the interesting special case of **discrete graph rewriting**

Poincaré-Birkhoff-Witt Theorem

The **universal enveloping algebra of the Heisenberg-Lie algebra**,

$$\mathcal{U}(\mathcal{L}_{\mathcal{D}}) := \frac{T(\mathcal{L}_{\mathcal{D}})}{\langle \delta(v) \otimes \delta(v^\dagger) - \delta(v^\dagger) \otimes \delta(v) - \delta(e) \rangle}$$

has a **normal-ordered basis** with elements of the form $U_{k,l,m} := \delta(v^\dagger)^{\otimes k} \otimes \delta(v)^{\otimes \ell} \otimes \delta(e)^{\otimes m} \quad (k, l, m \in \mathbb{Z}_{\geq 0})$

Notations:

- disjoint union on diagrams $\quad d_2 \uplus d_1 := [(O_2 + O_1, I_2 + I_1, m_2 + m_1)]_{\sim} \; = \; d_2 \triangleleft_{\varnothing} d_1$
- $d_{k,\ell,m} := v^{\dagger \, \uplus k} \uplus v^{\uplus \ell} \uplus e^{\uplus m}$

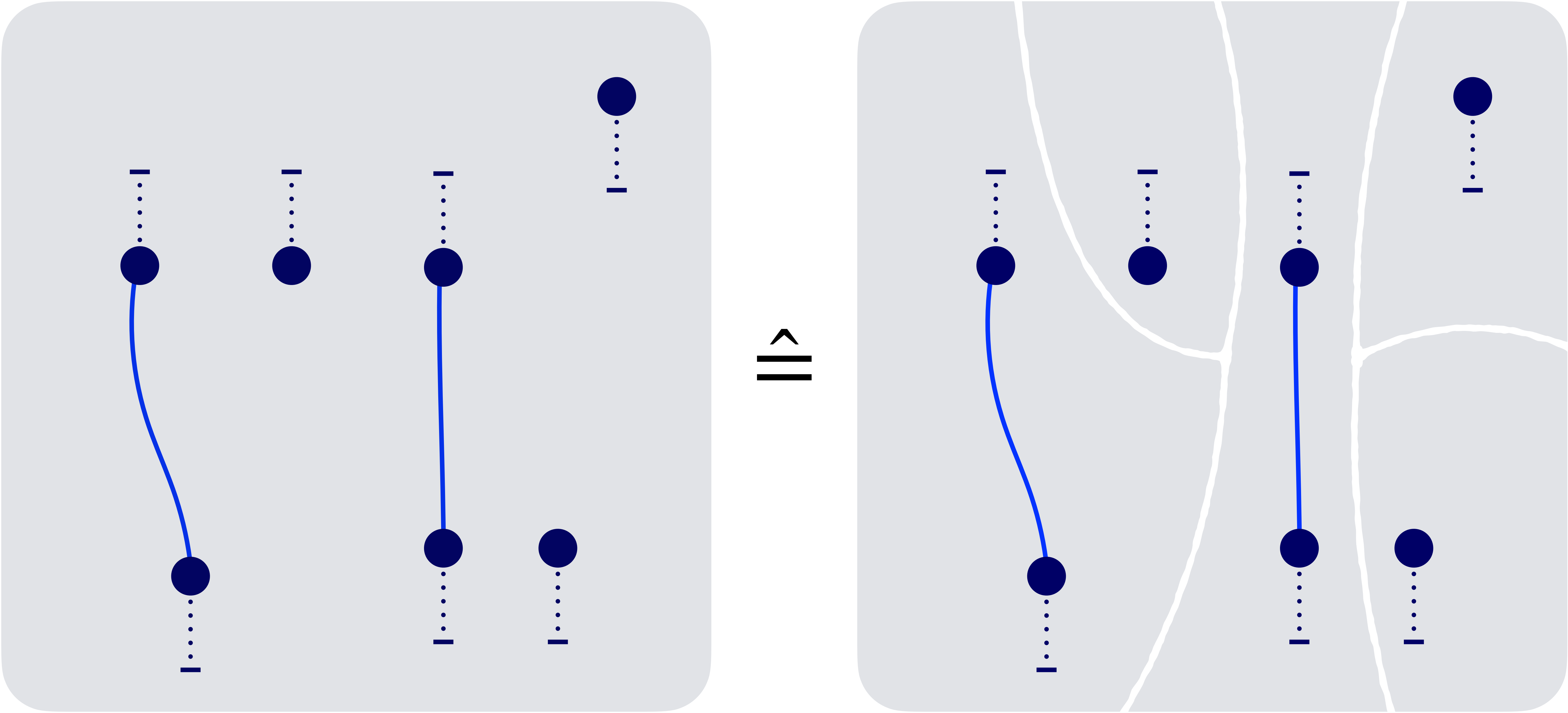
Theorem [Behr et al. 2016]

There exists a **isomorphism of algebras** $(\mathcal{D}, *_{\mathcal{D}}) \xrightarrow[\cong]{\varphi} \mathcal{U}(\mathcal{L}_{\mathcal{D}})$, defined via $\varphi(\delta(d_{k,\ell,m})) = U_{k,\ell,m}$.

On the interesting special case of **discrete graph rewriting**

Interesting fact: the universal enveloping algebra $\mathcal{U}(\mathcal{L}_{\mathcal{D}})$ is a (non-commutative, co-commutative) **Hopf algebra**.

$\Rightarrow$ one may verify that the isomorphism φ extends to a **Hopf-algebra isomorphism** !



Coproduct of the diagram algebra

$$\delta(d) = \delta(\biguplus_{x\in X} d_x) \qquad (d_x \in \{v^\dagger, v, e\})$$

$$\delta(\biguplus_{x\in\emptyset} d_x) := \delta(d_\emptyset)$$

$$\Delta(\delta(d)) := \sum_{Y\subseteq X} \delta(\biguplus_{y\in Y} d_y) \otimes \delta(\biguplus_{z\in X\setminus Y} d_z)$$

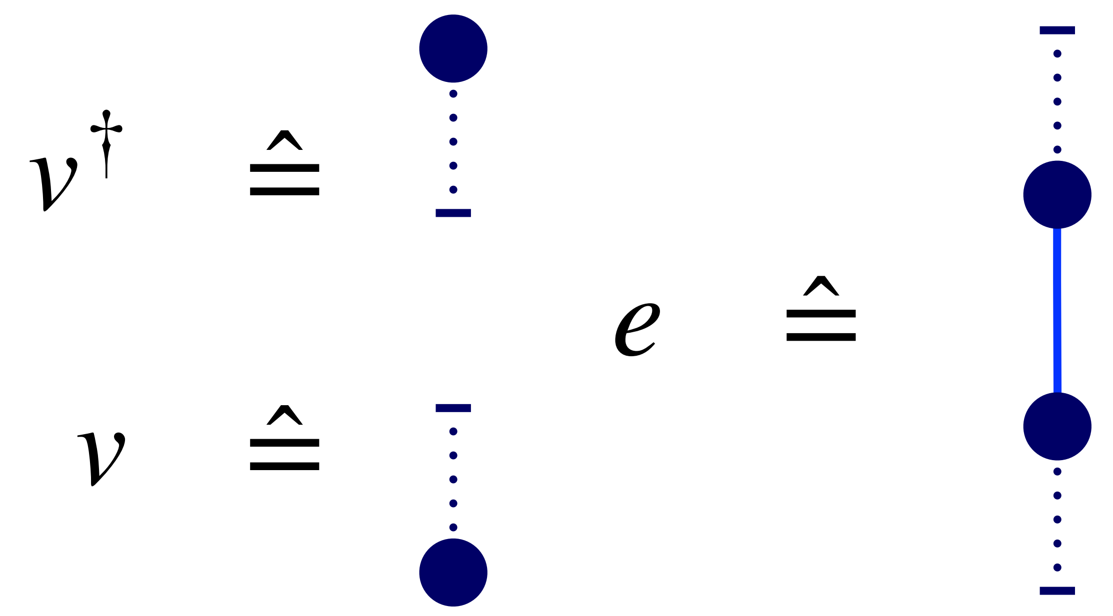
Theorem [Blasiak et al. 2011, Behr et al. 2016]

$(\mathcal{D}, *_{\mathcal{D}}, \Delta)$ is a **Hopf algebra**, with **unit** $\eta : \mathbb{K} \to \mathcal{D} : 1_{\mathbb{K}} \mapsto \delta(d_\emptyset)$ and **counit** $\epsilon : \mathcal{D} \to \mathbb{K} : \delta(d) \mapsto \delta_{d,d_\emptyset}$

On the interesting special case of **discrete graph rewriting**

$$d_{k,\ell,m} := v^{\dagger \uplus k} \uplus v^{\uplus \ell} \uplus e^{\uplus m}$$

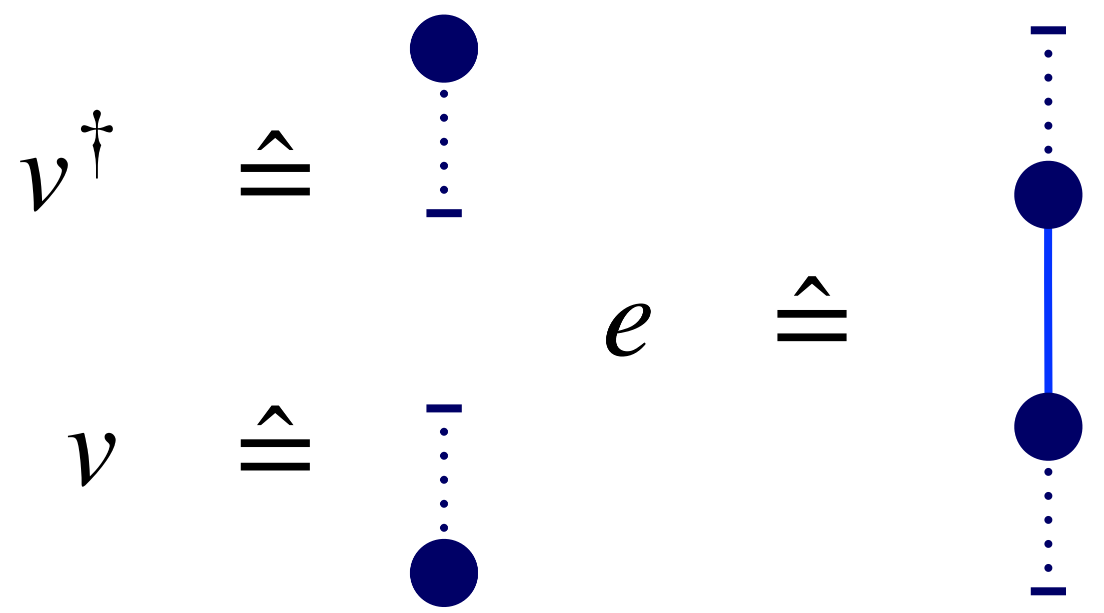
Elementary diagrams:



On the interesting special case of **discrete graph rewriting**

$$d_{k,\ell,m} := v^{\dagger\,\uplus k} \uplus v^{\uplus \ell} \uplus e^{\uplus m}$$

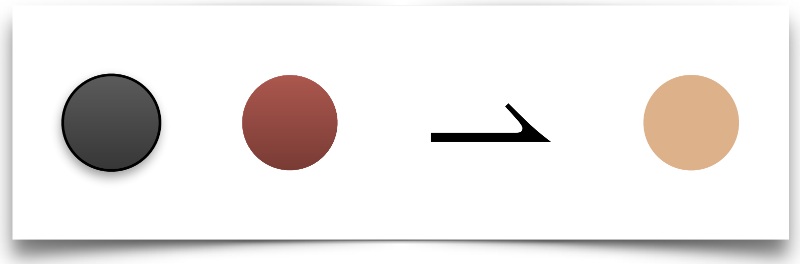
Elementary diagrams:



Example diagrammatic **normal-ordering** formula

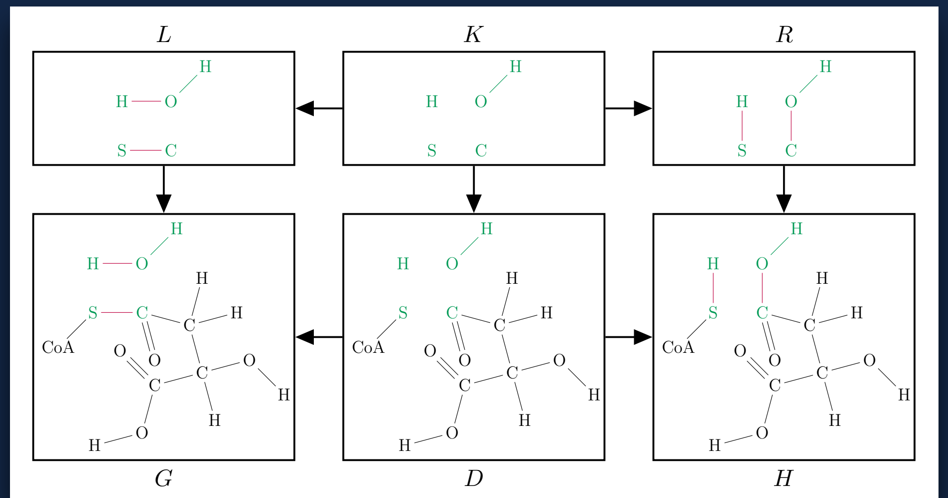
$$\delta(d_{k_2,\ell_2,m_2}) *_{\mathscr{D}} \delta(d_{k_1,\ell_1,m_1}) = \sum_{r\geq 0} \binom{\ell_2}{r} r! \binom{k_1}{r} \delta(d_{k_1+k_2-r,\ell_1+\ell_2-r,m_1+m_2+r})$$

\# of ways to form
r output-to-input "wirings
(disregarding the order)

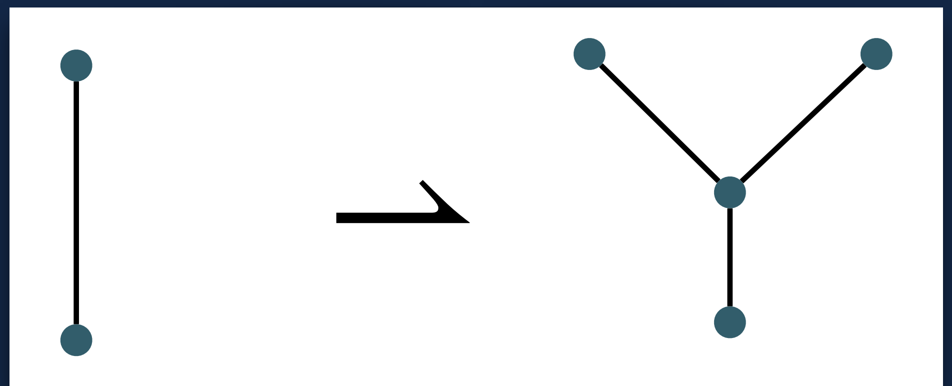


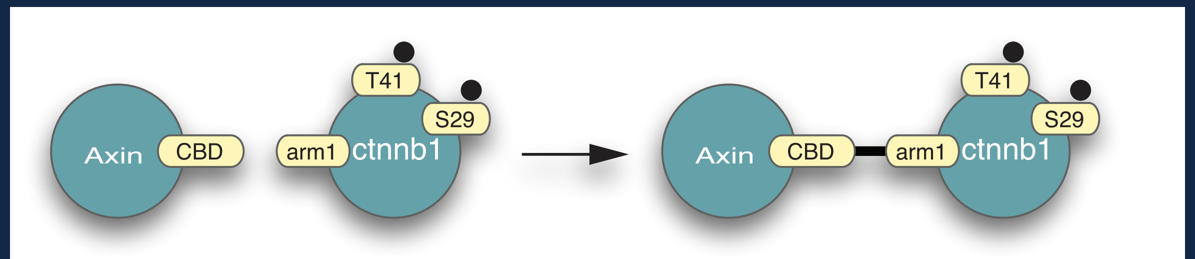
COMPUTER SCIENCE

- **semantics** and **stochastic rewriting theory**
- **concurrency theory**
- **algorithms** for **bio-** and **organo-chemistry**

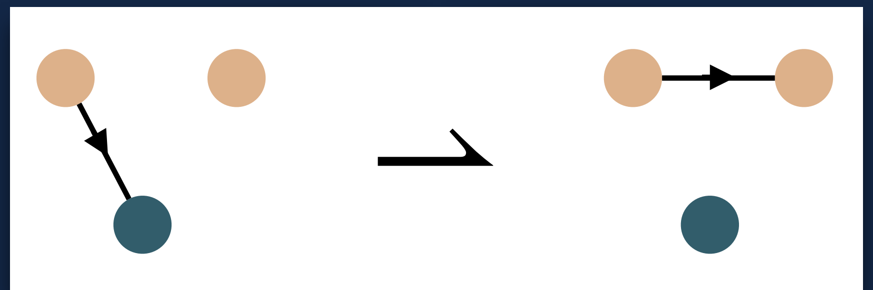


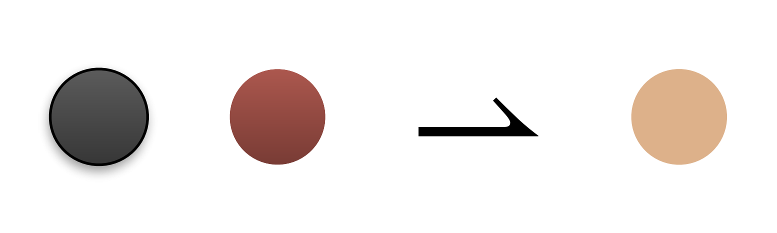
organic chemistry





biochemistry





MATHEMATICAL PHYSICS

- **continuous-time Markov chains (CTMCs)**
- **statistical mechanics**

MATHEMATICS

- enumerative/algebraic **combinatorics**
- theory of **M-adhesive categories**

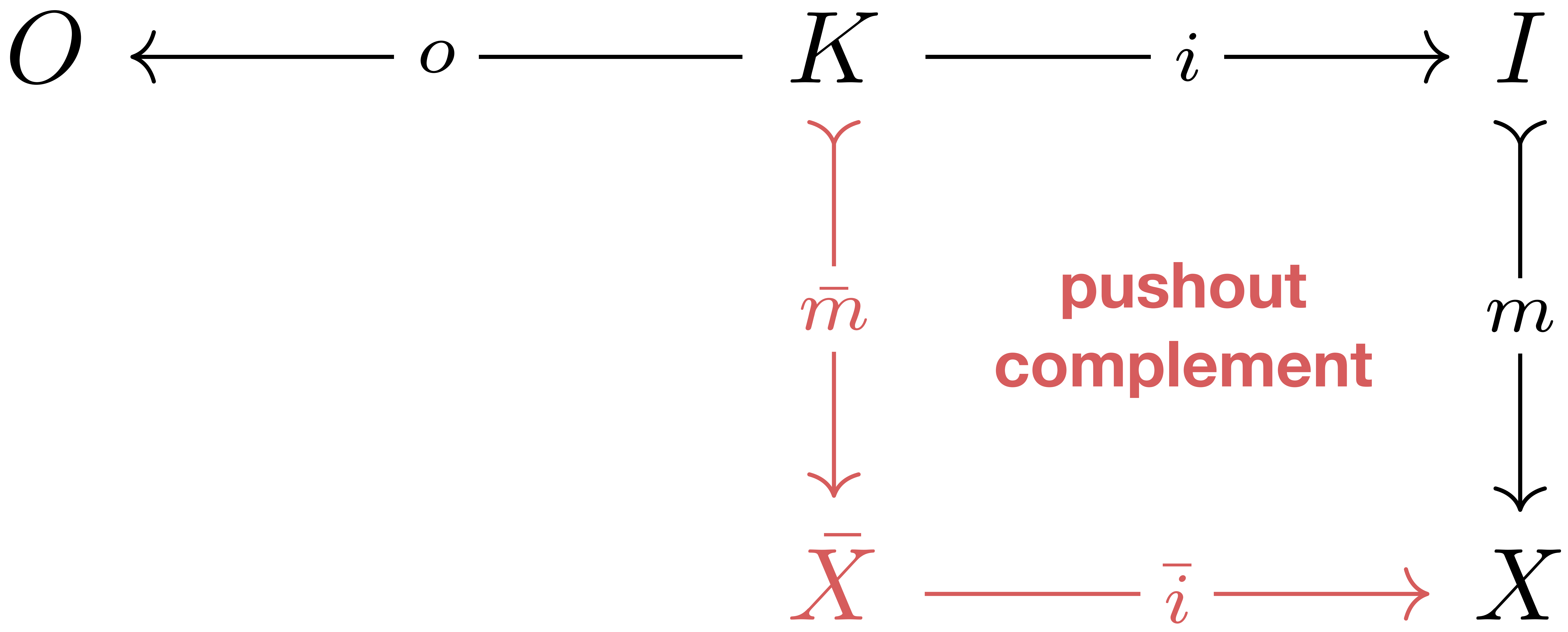
Plan of the talk

1. Discrete rewriting and diagram Hopf Algebras
2. Categorical rewriting theory
3. From rewriting to tracelets
4. Tracelet decomposition spaces
5. Tracelet Hopf algebras

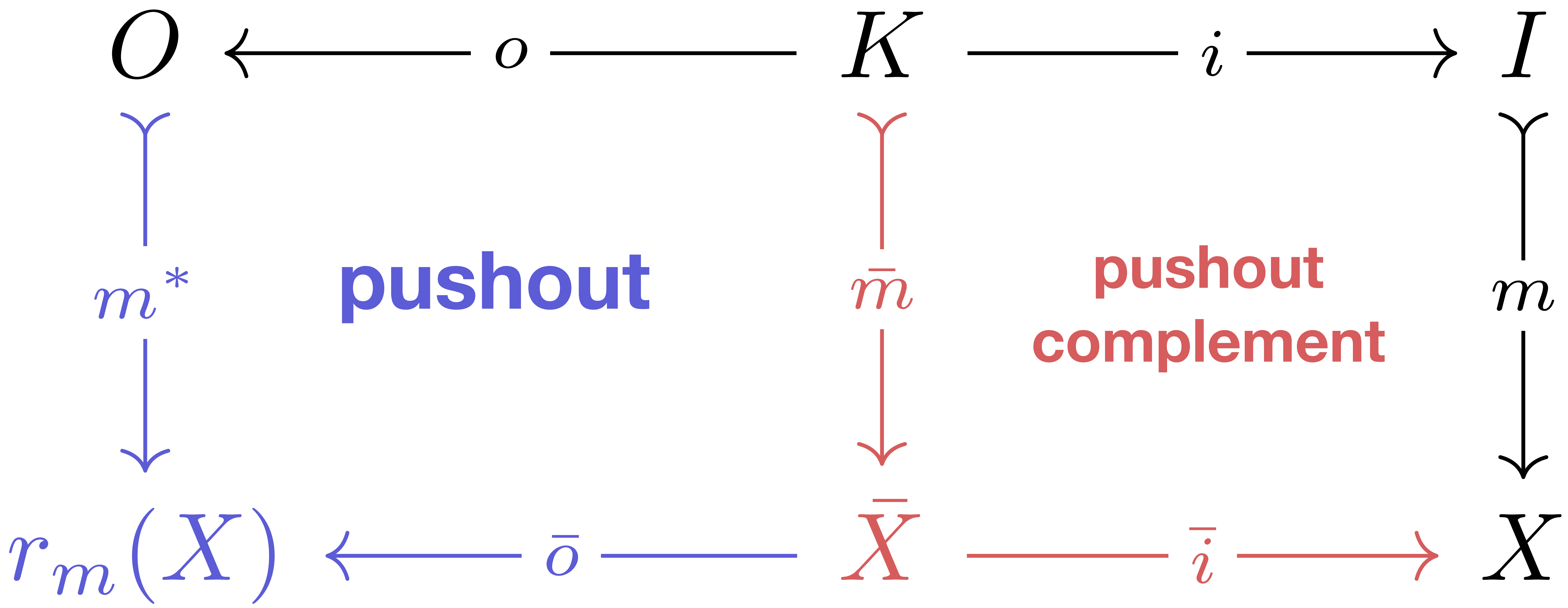
Double Pushout (DPO) rewriting

$$\begin{array}{ccccc} O & \xleftarrow{\;o\;} & K & \xrightarrow{\;i\;} & I \\ & & & & \big\downarrow{\scriptstyle m} \\ & & & & X \end{array}$$

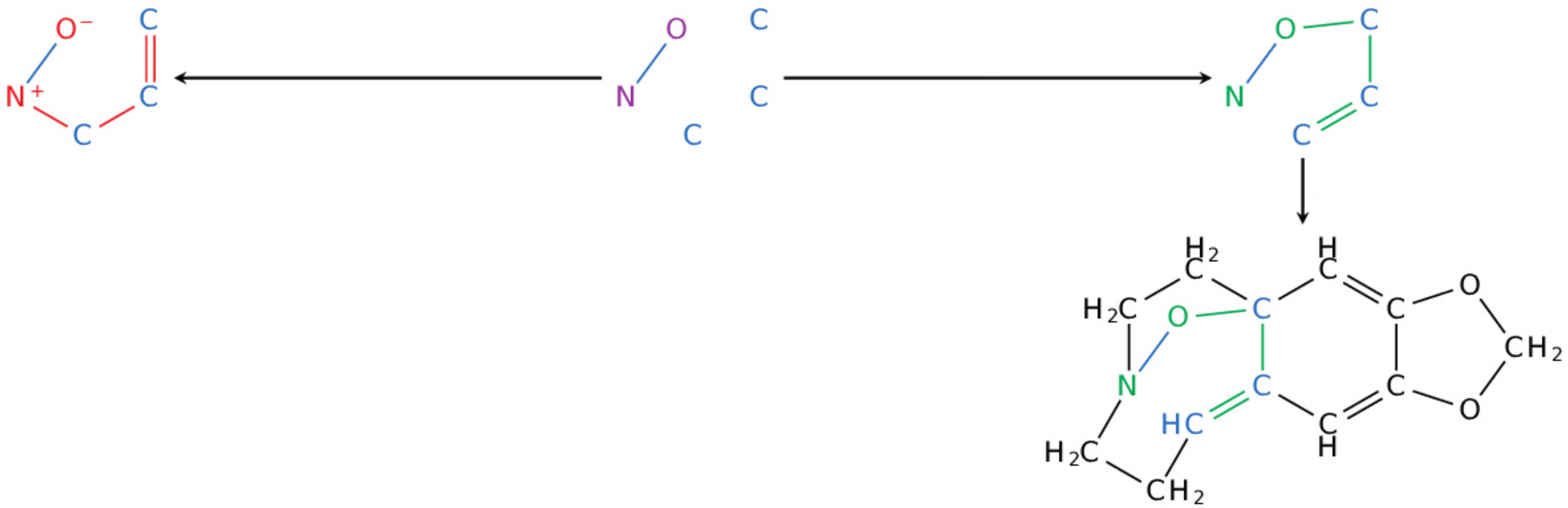
Double Pushout (DPO) rewriting



Double Pushout (DPO) rewriting

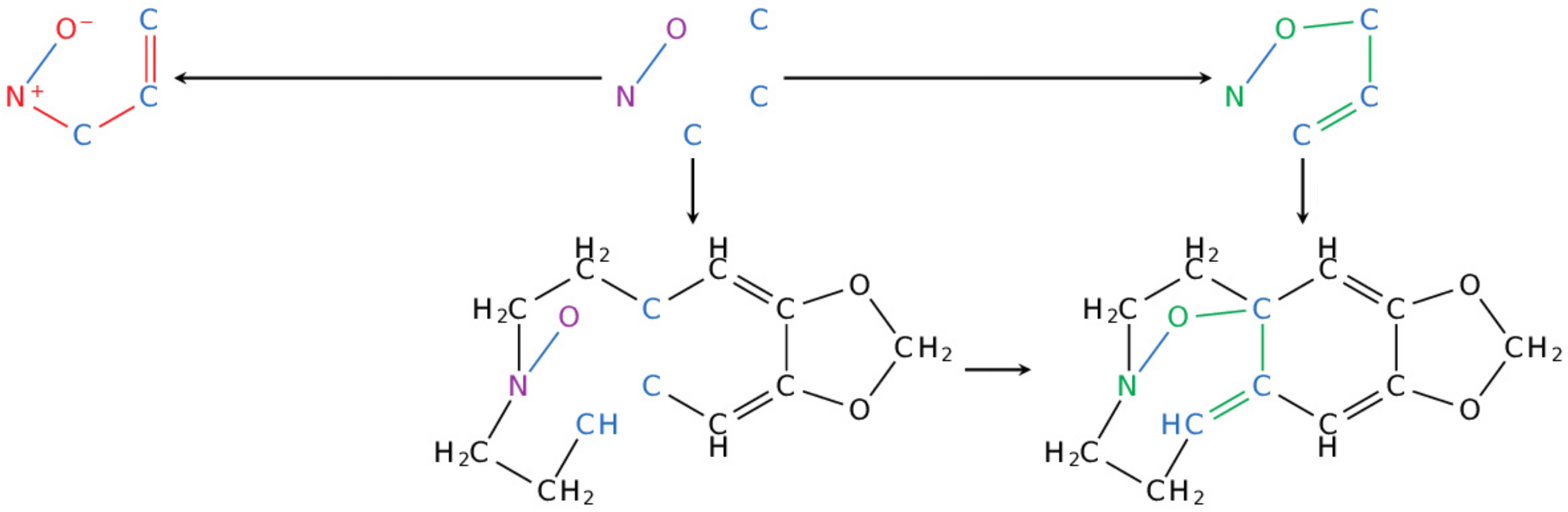


Organic chemistry via DPO-type rewriting (!)



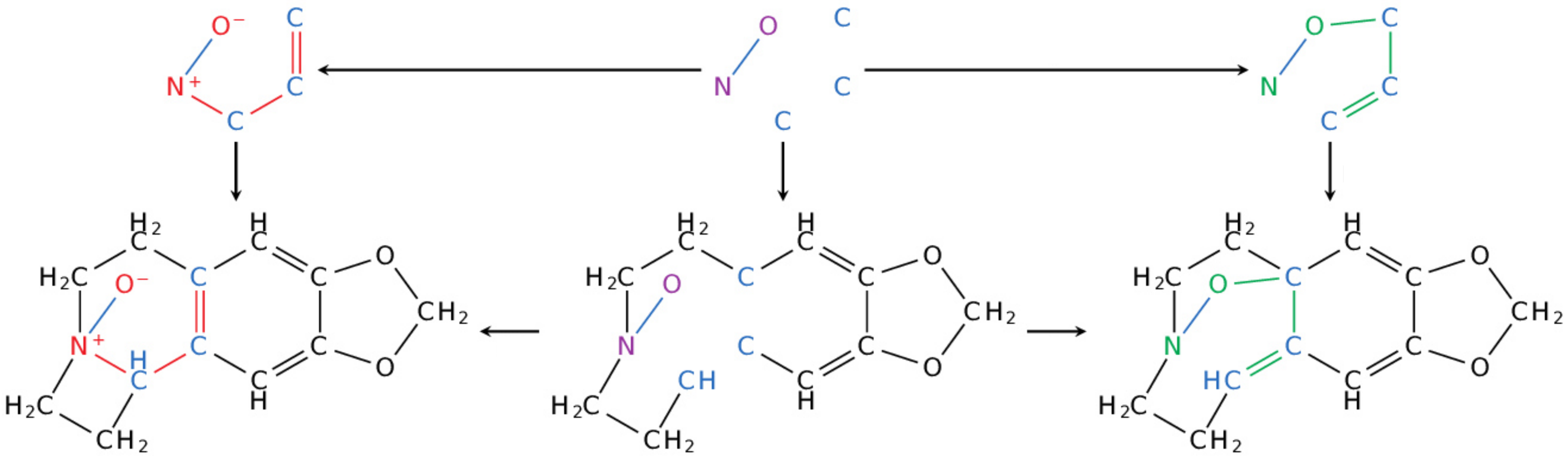
Source: *Algorithmic Cheminformatics Group, SDU Odense*

Organic chemistry via DPO-type rewriting (!)



Source: *Algorithmic Cheminformatics Group, SDU Odense*

Organic chemistry via DPO-type rewriting (!)



Source: *Algorithmic Cheminformatics Group, SDU Odense*

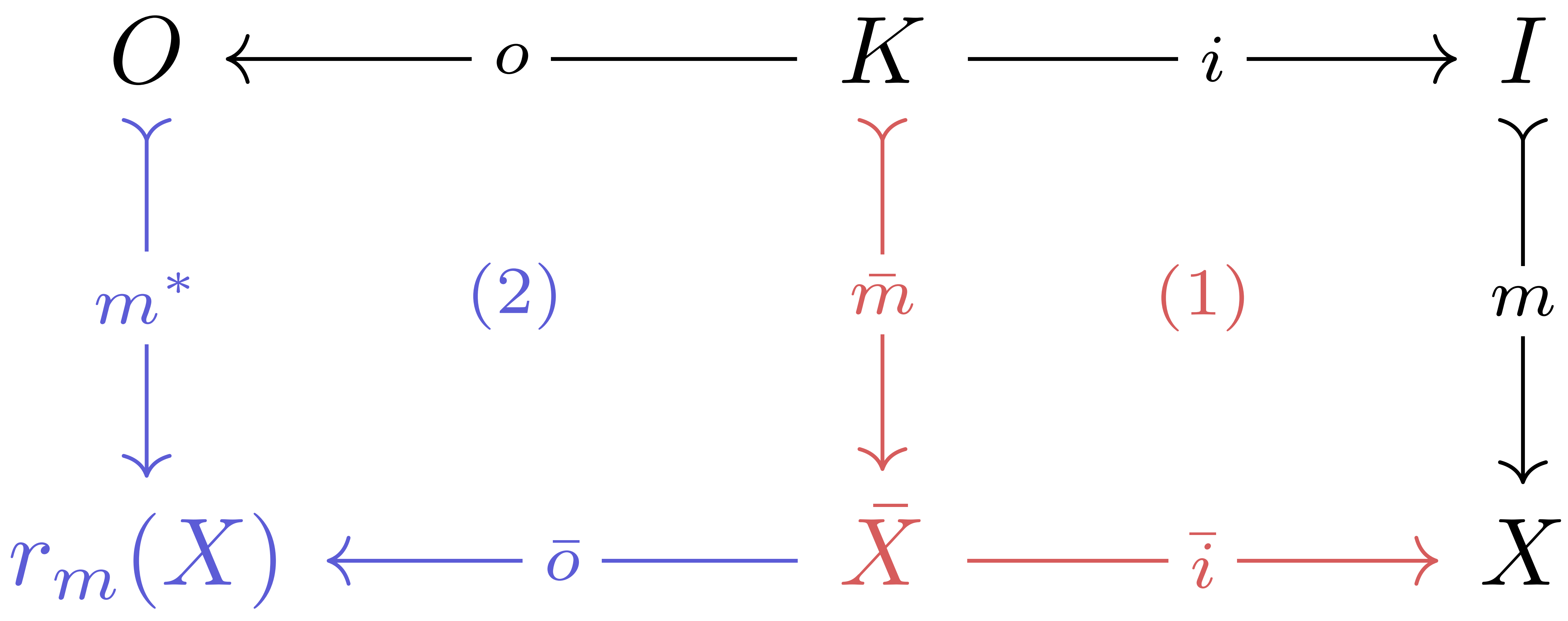
GReTA - Graph Transformation Theory and Applications

New seminar series since **November 2020**,

co-hosted by Nicolas Behr, Jean Krivine and Reiko Heckel

https://www.irif.fr/~greta/

DPO rewriting theory does not really stop at this first definition…



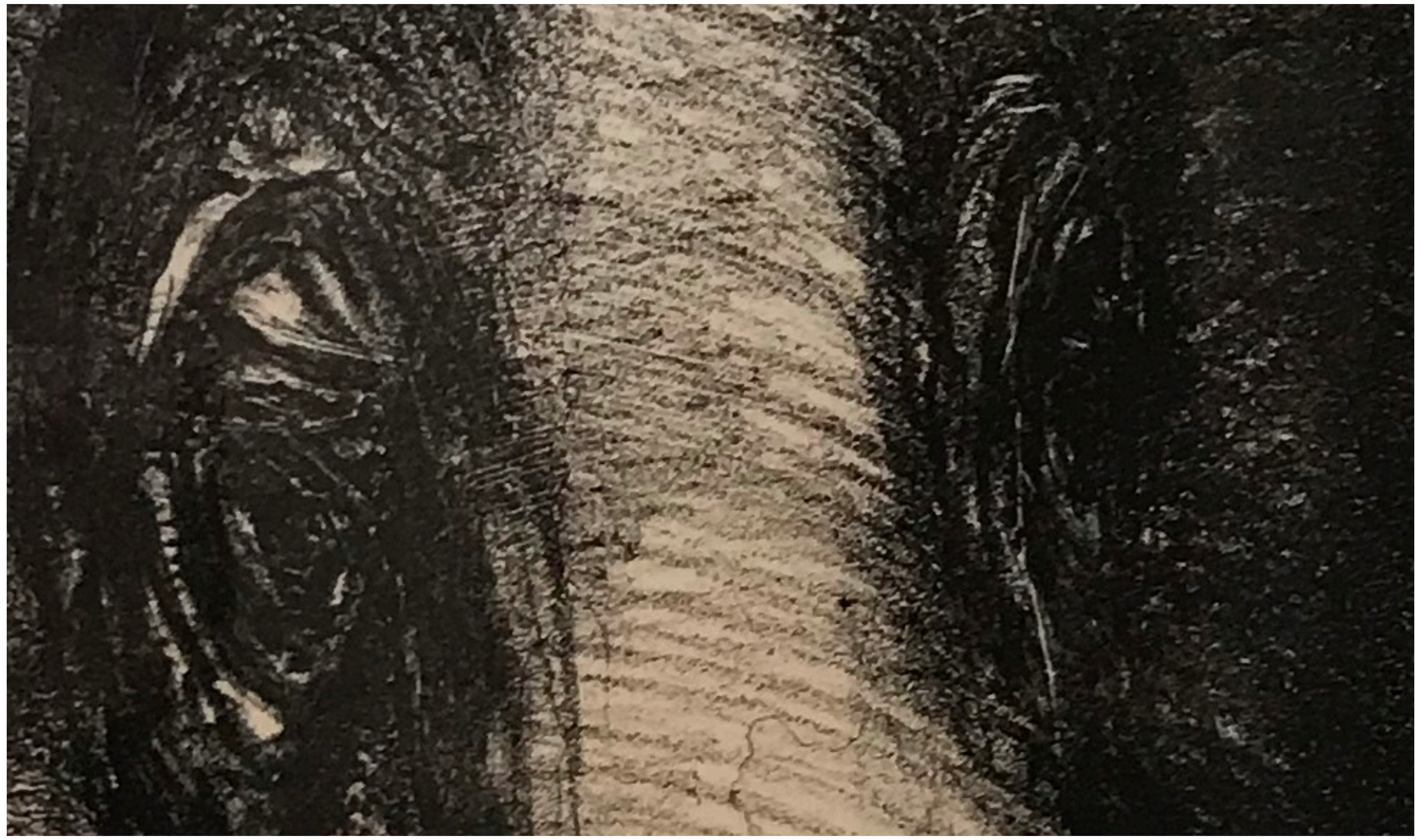
DPO rewriting theory does not really stop at this first definition…

$$
\begin{array}{ccccc}
O & \xleftarrow{\;o\;} & K & \xrightarrow{\;i\;} & I \\
\downarrow{\scriptstyle m^*} & (2) & \downarrow{\scriptstyle \bar{m}} & (1) & \downarrow{\scriptstyle m} \\
r_m(X) & \xleftarrow{\;\bar{o}\;} & \bar{X} & \xrightarrow{\;\bar{i}\;} & X
\end{array}
$$

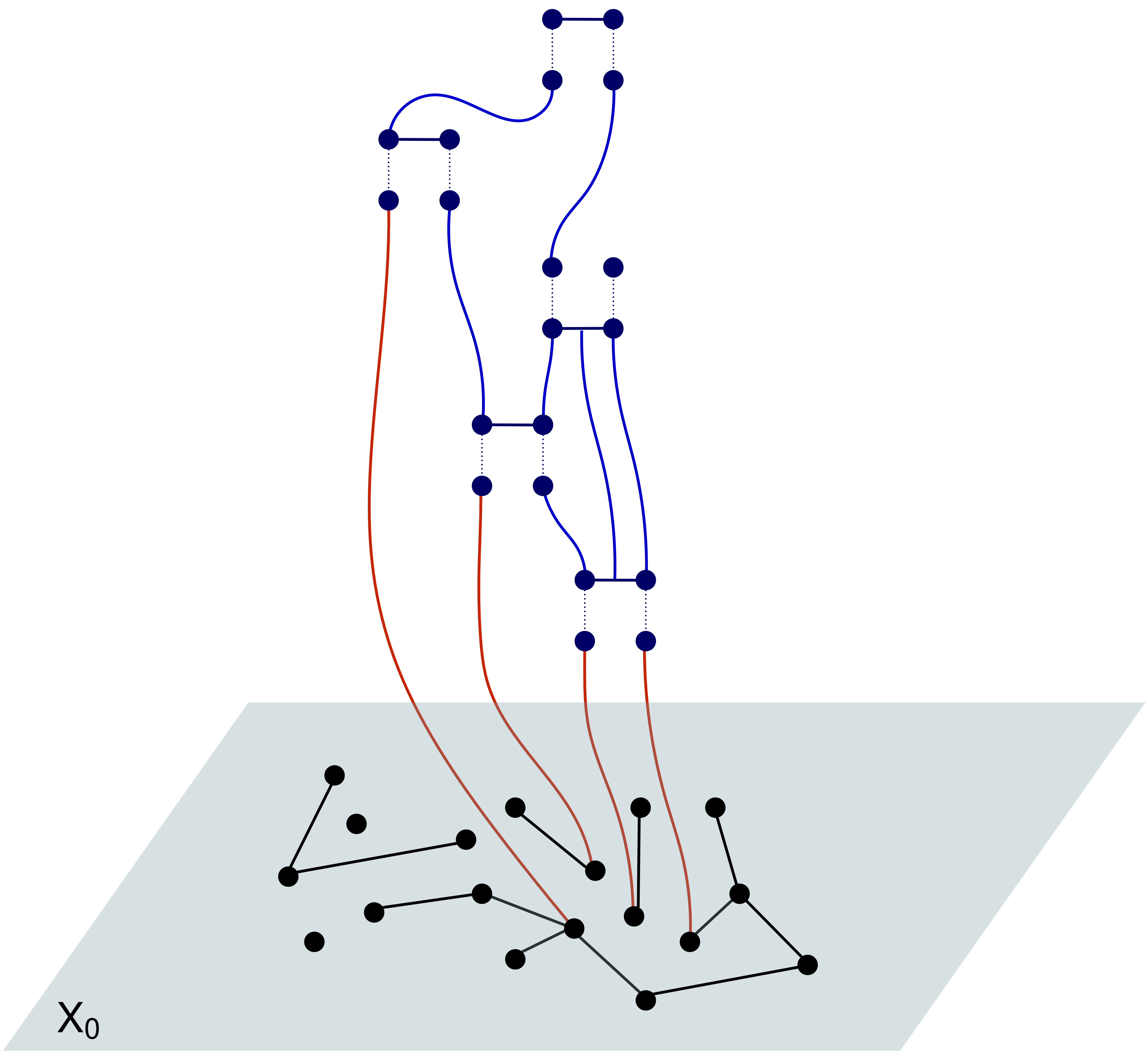
Artwork by Angelika Villagrana

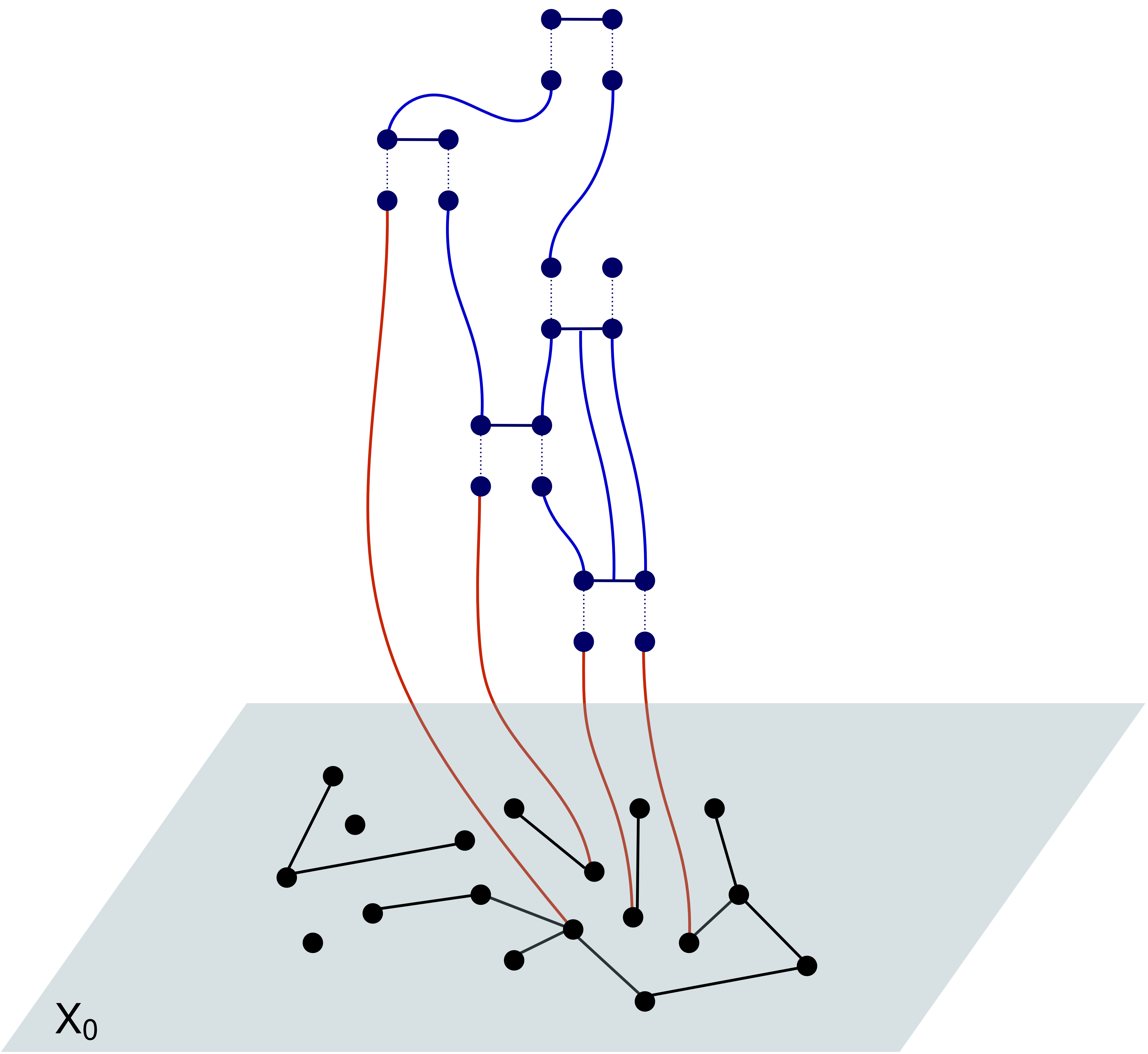
DPO rewriting theory does not really stop at this first definition…

$$\begin{array}{ccccc} O & \xleftarrow{\quad o \quad} & K & \xrightarrow{\quad i \quad} & I \\ \big\downarrow{\scriptstyle m^*} & (2) & \big\downarrow{\scriptstyle \bar{m}} & (1) & \big\downarrow{\scriptstyle m} \\ r_m(X) & \xleftarrow{\quad \bar{o} \quad} & \bar{X} & \xrightarrow{\quad \bar{i} \quad} & X \end{array}$$

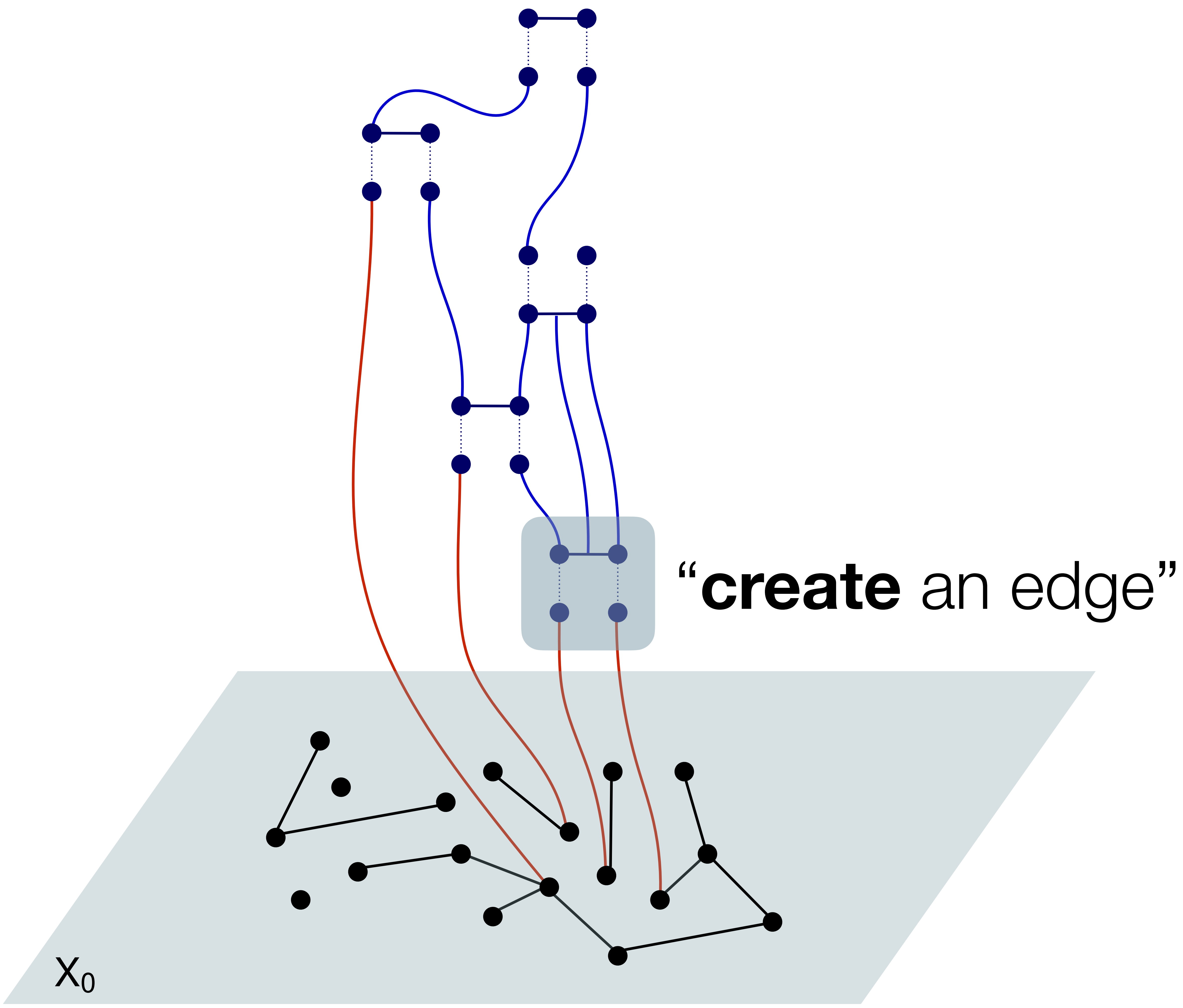


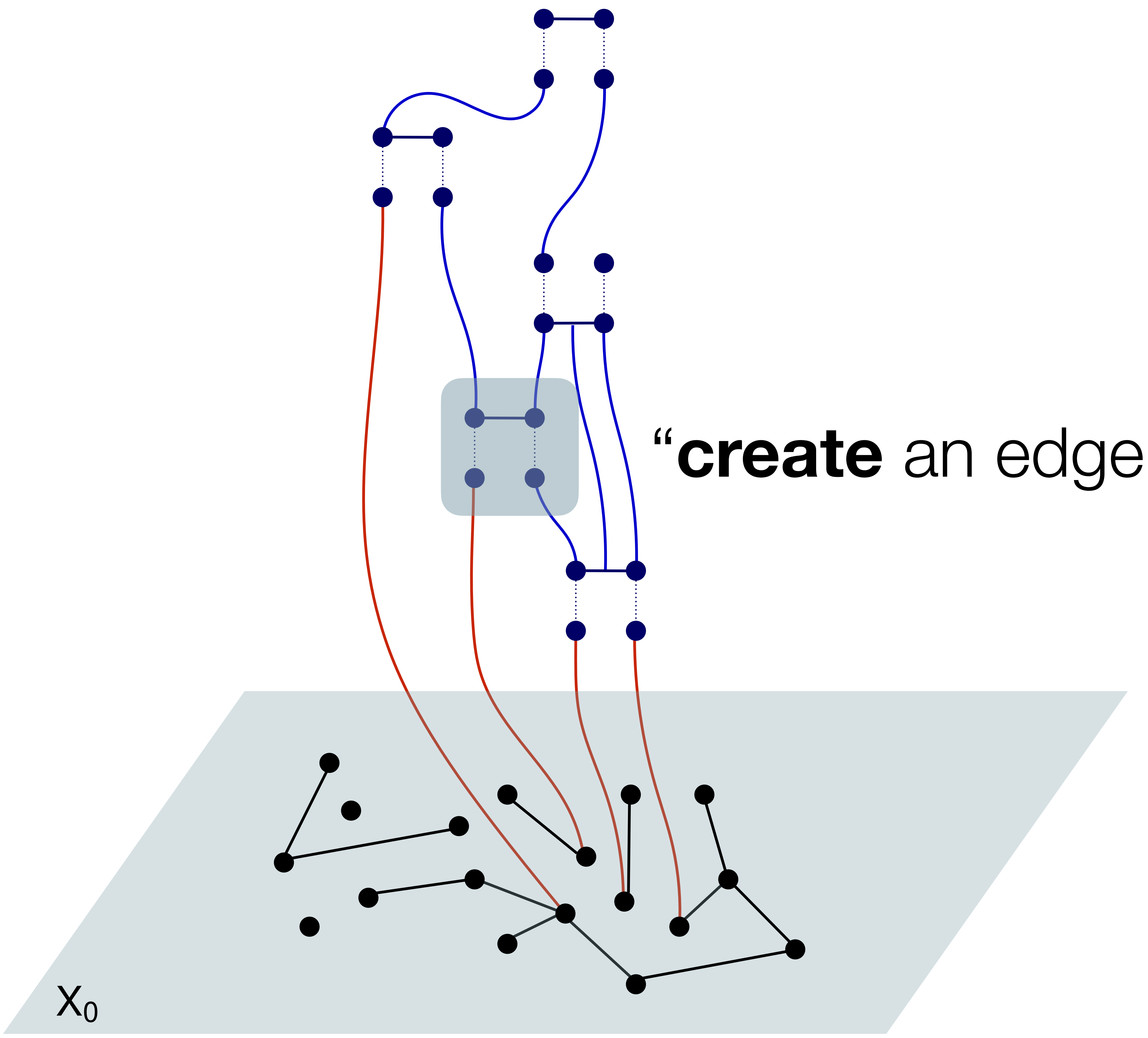
Artwork by Angelika Villagrana


X_0

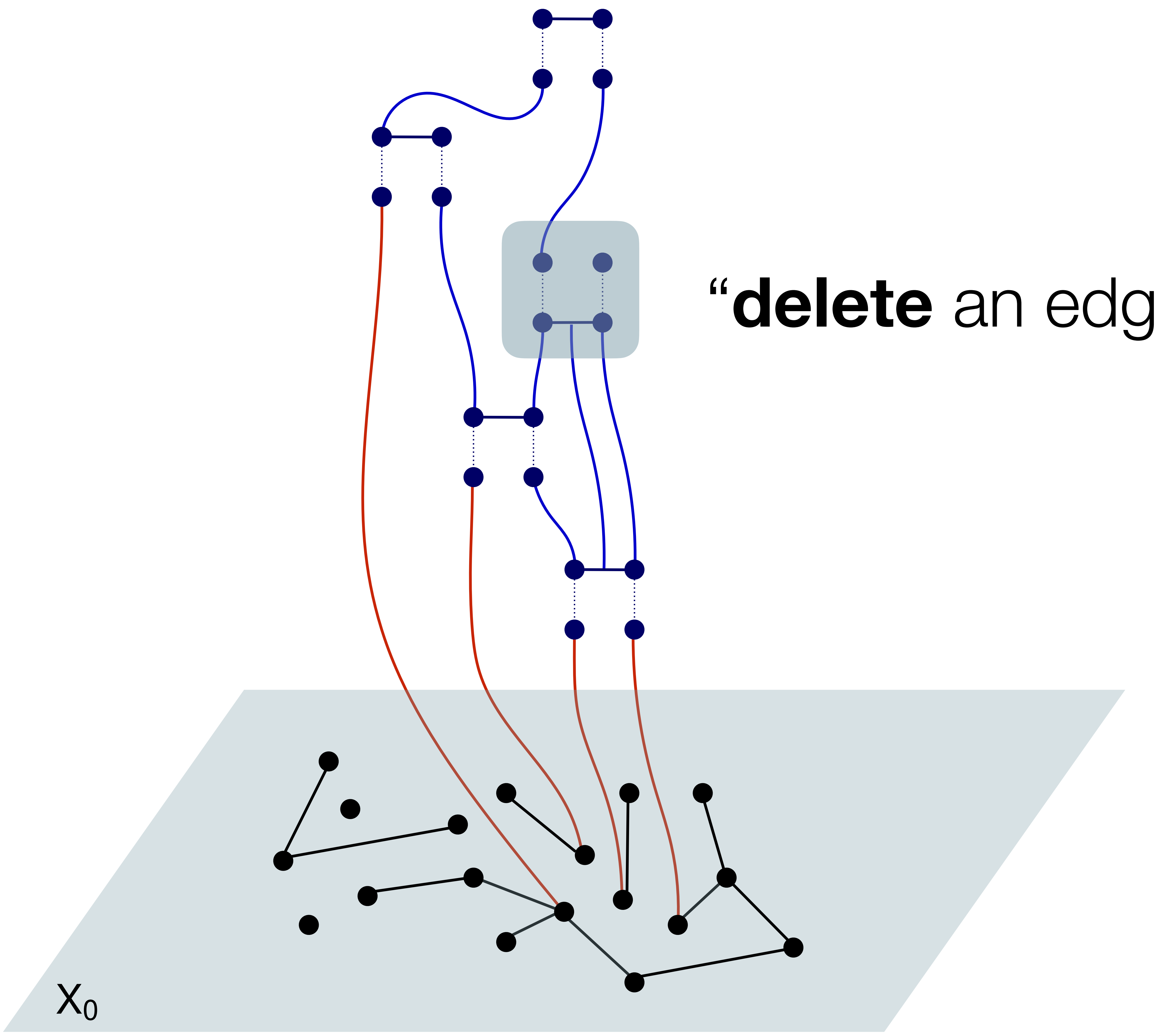


input graph

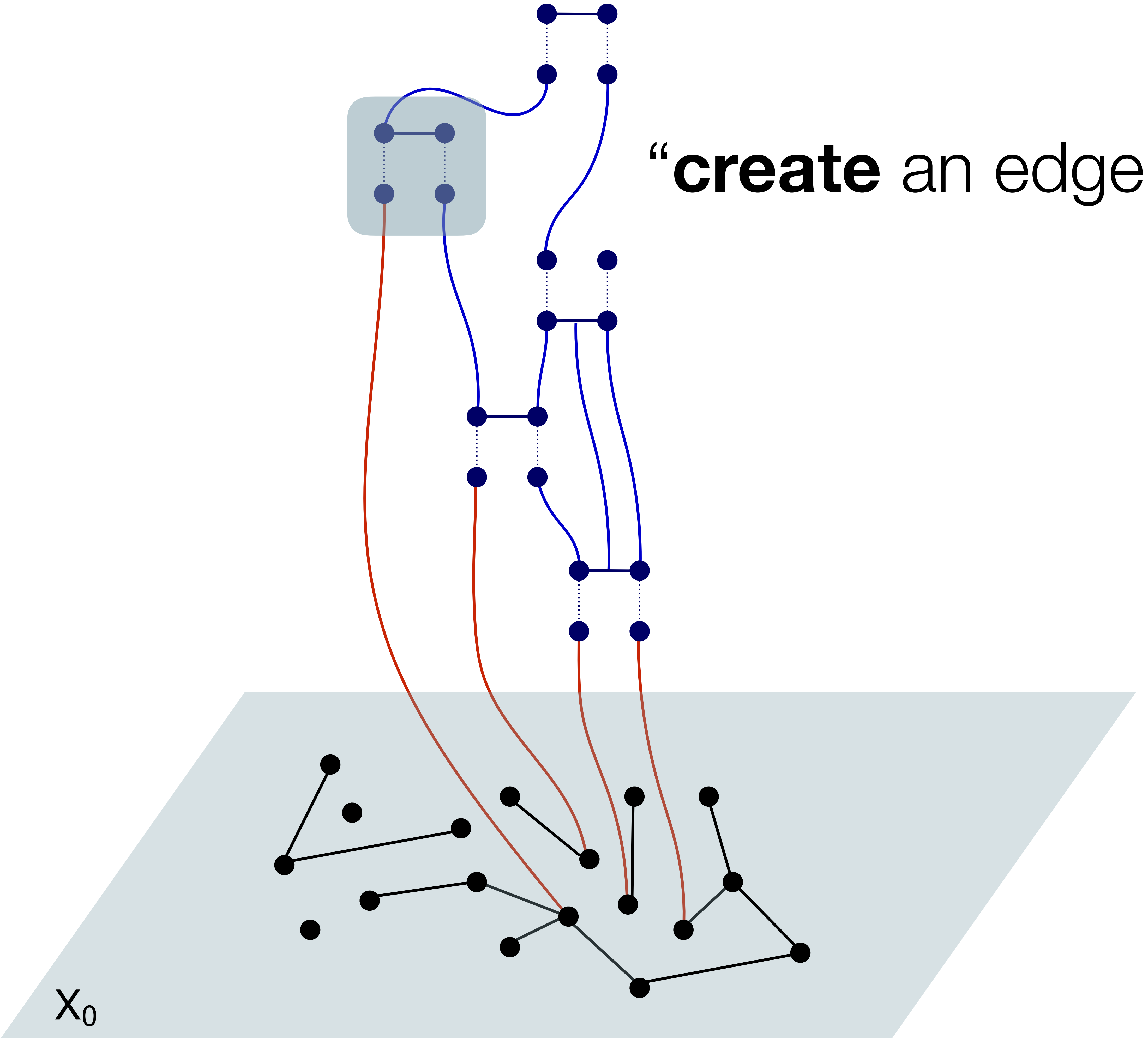

“**create** an edge”
X_0



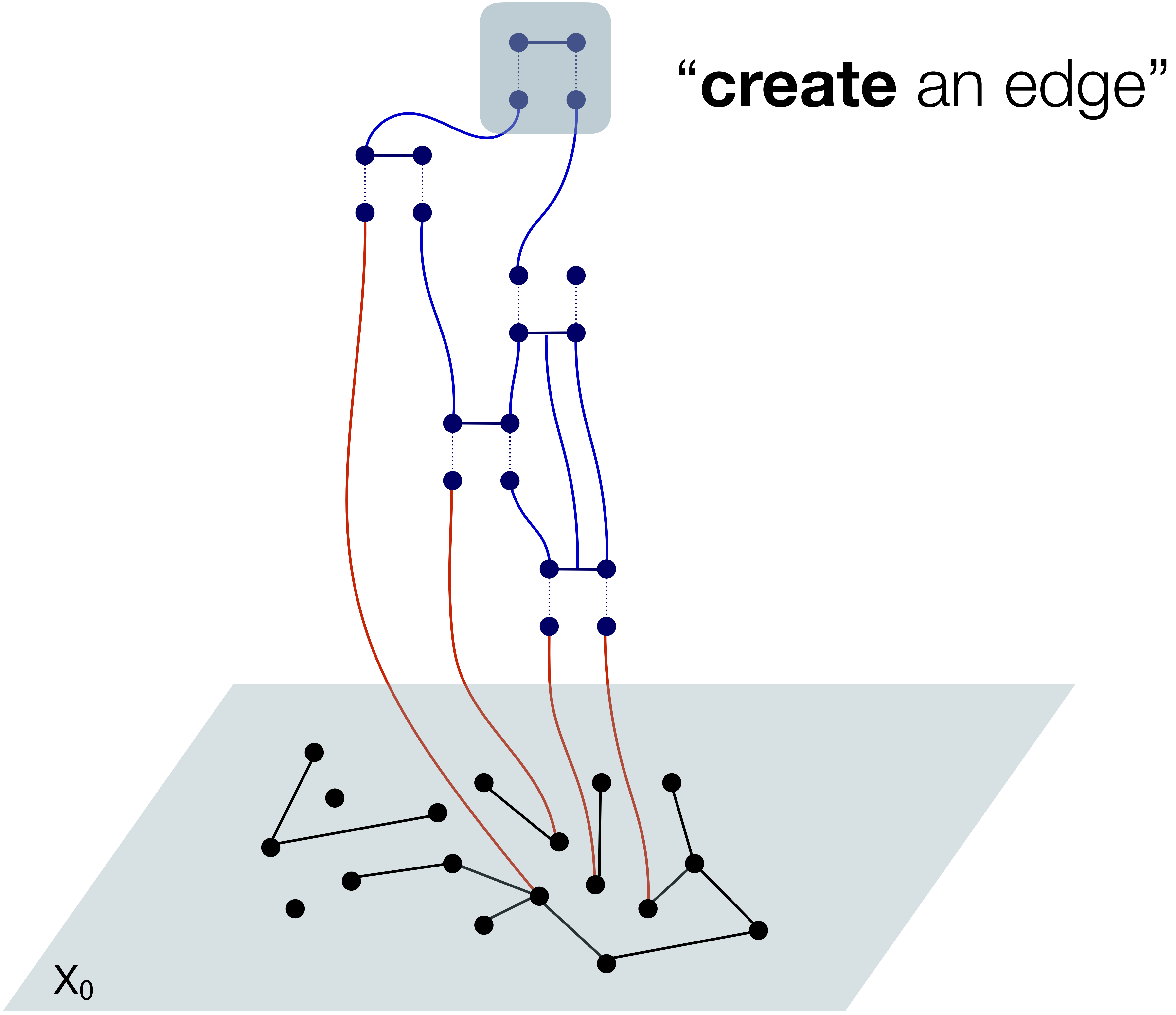
"**create** an edge"

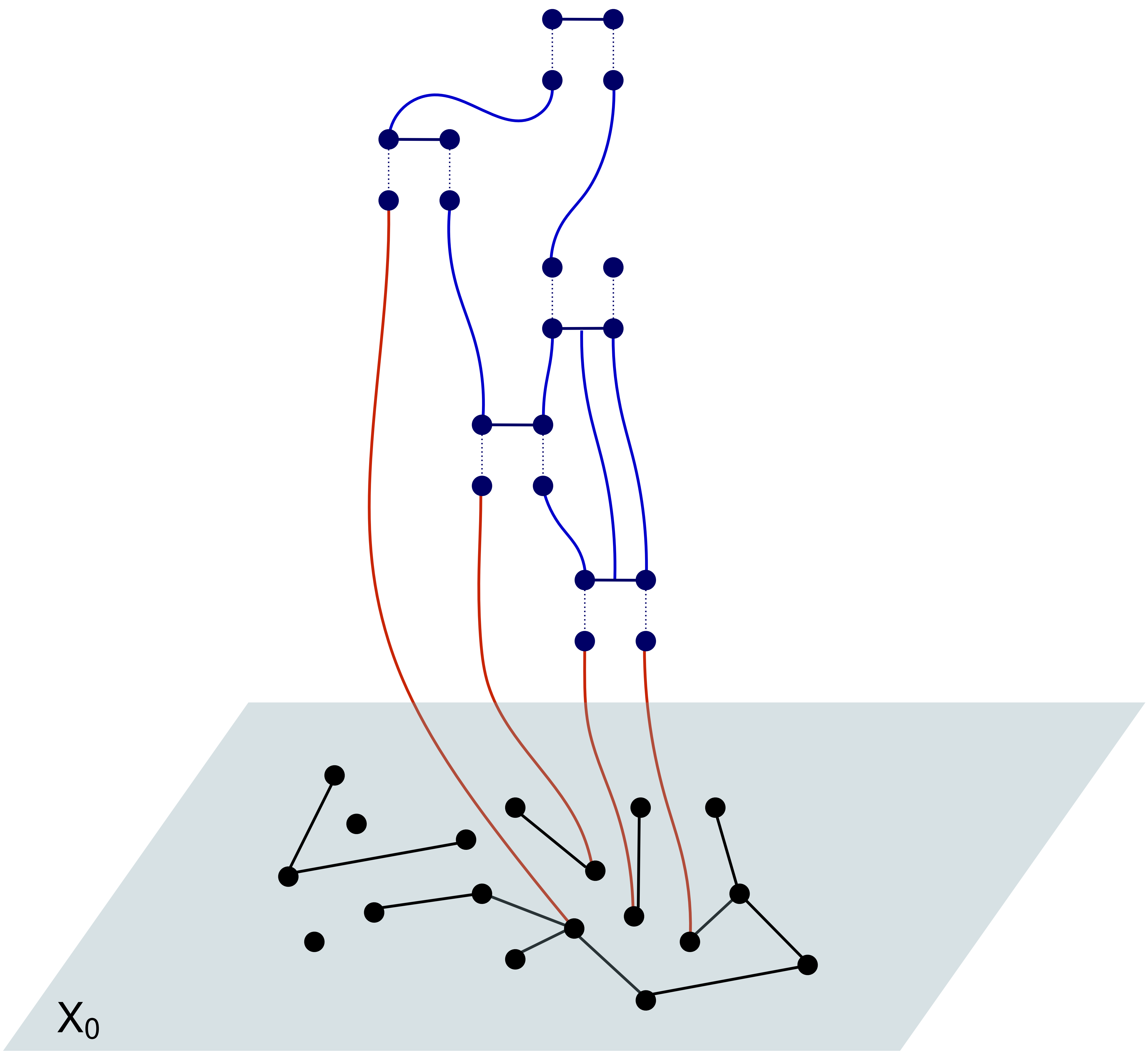


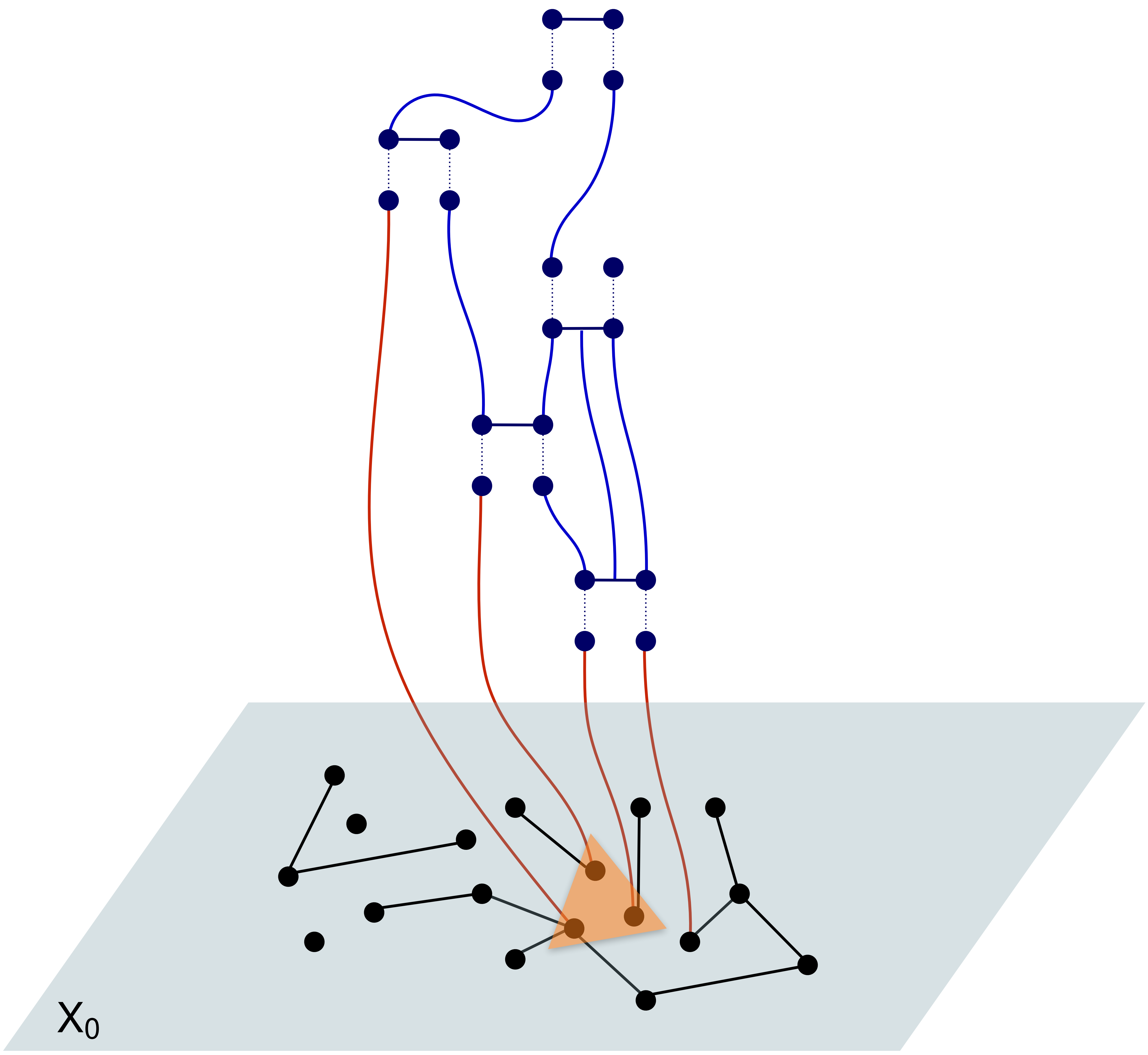
“**delete** an edge”



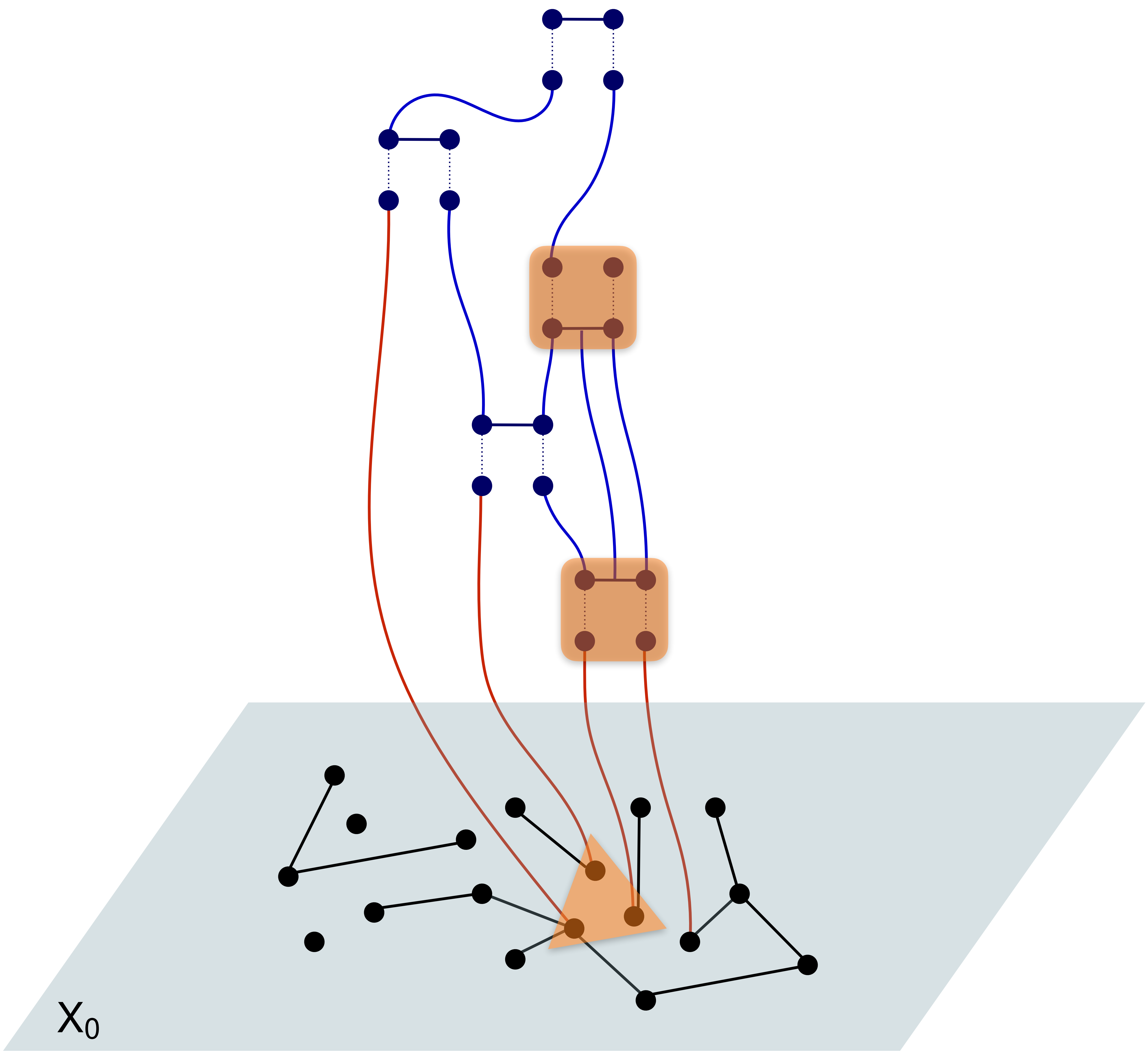
"**create** an edge"

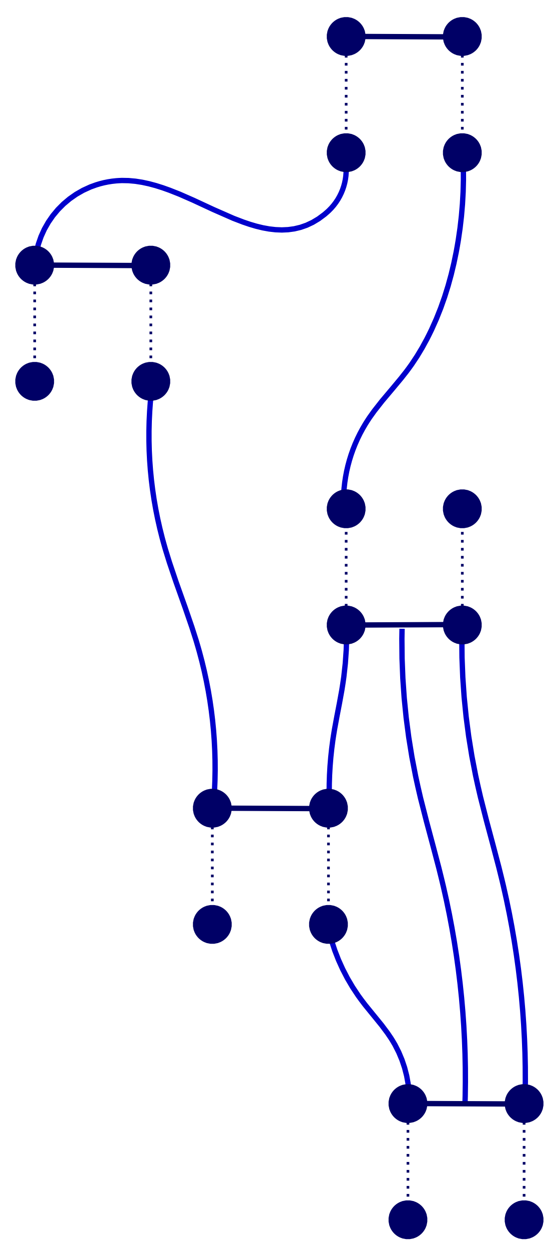

"**create** an edge"
X_0

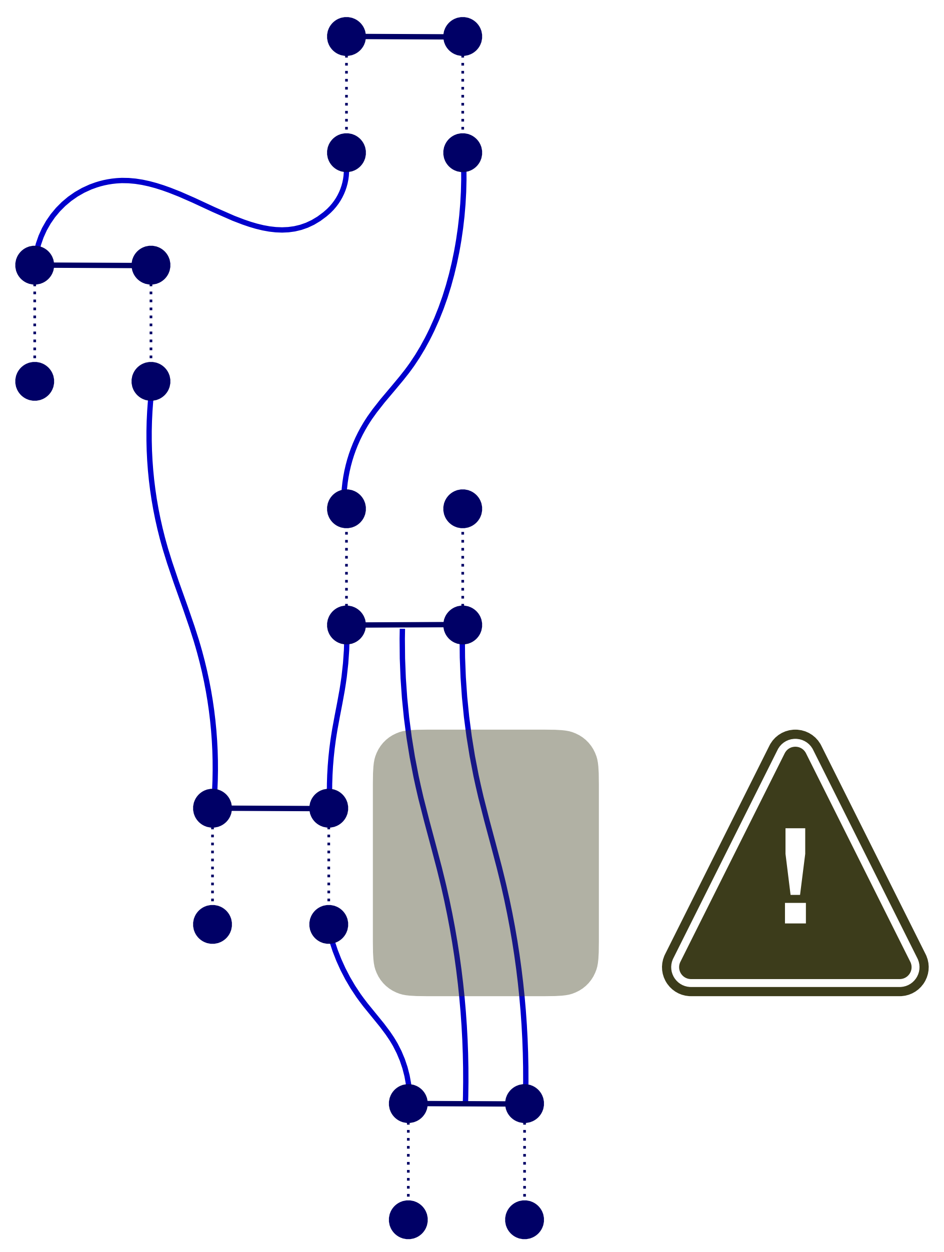

X_0

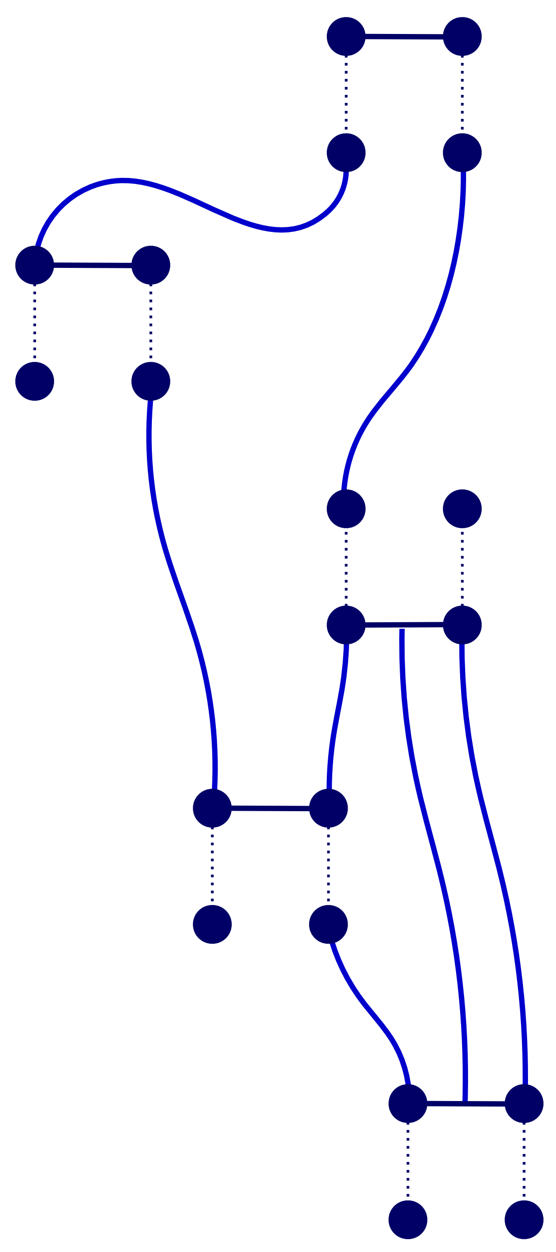


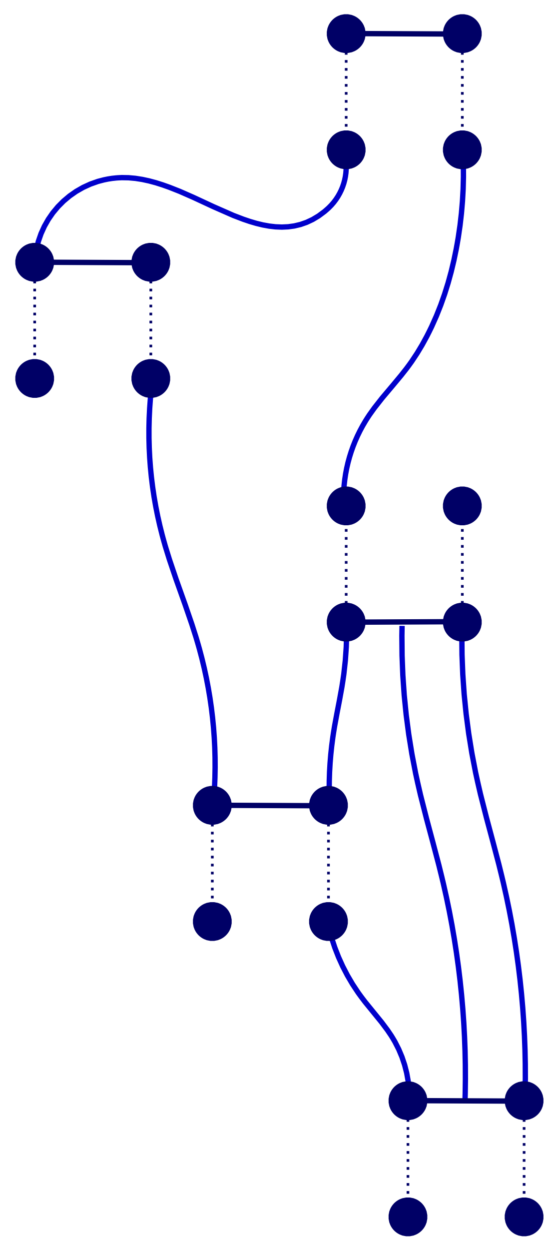
X_0


X0









a **TRACELET**
(of length 5)

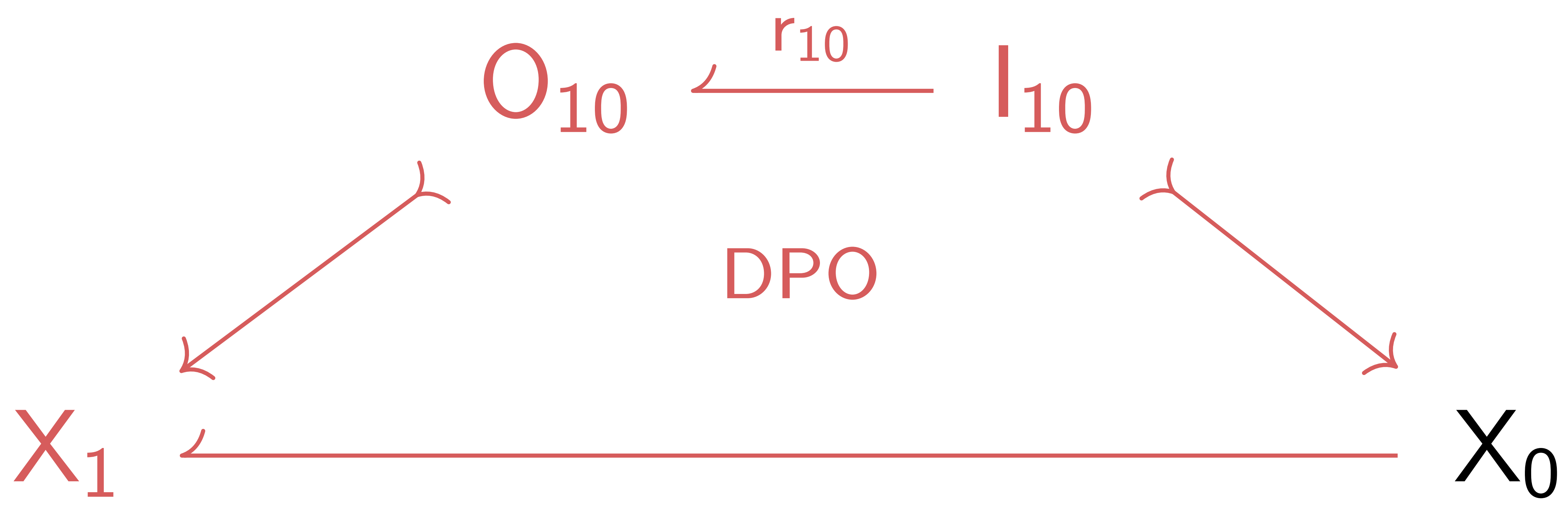
Plan of the talk

1. Discrete rewriting and diagram Hopf Algebras
2. Categorical rewriting theory
3. From rewriting to tracelets
4. Tracelet decomposition spaces
5. Tracelet Hopf algebras

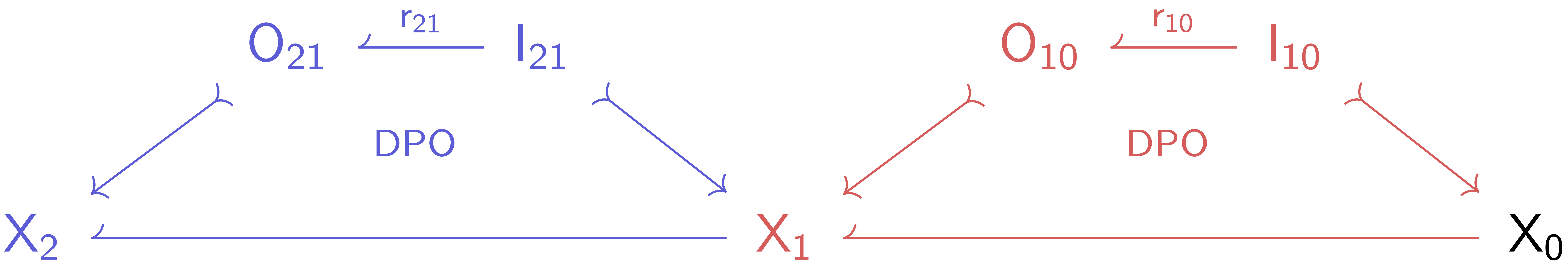
DPO-type **concurrency theorem**

X_0

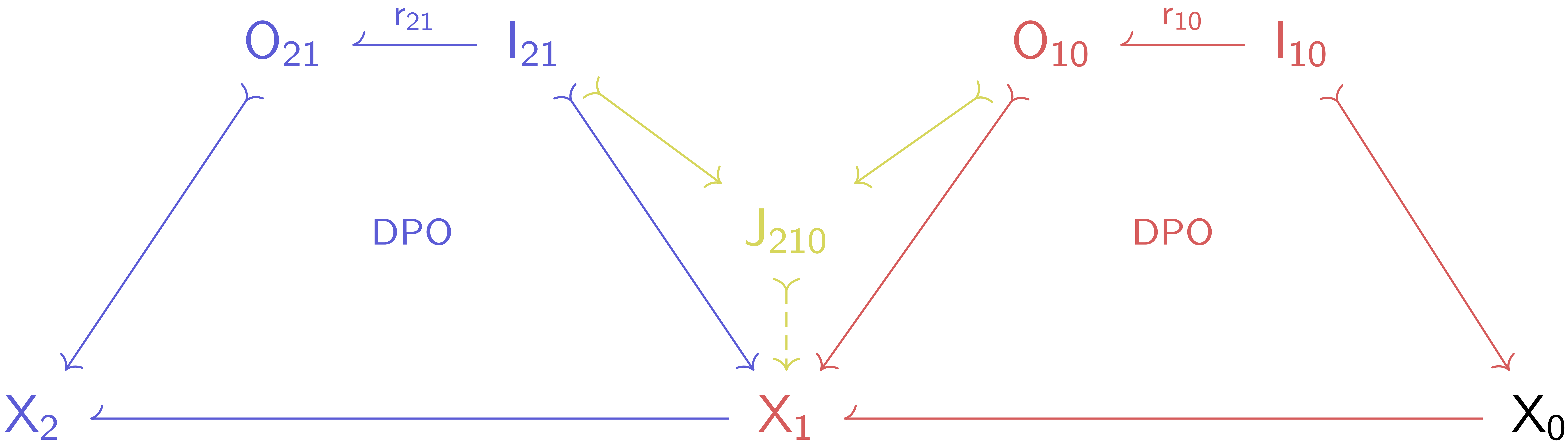
DPO-type **concurrency theorem**



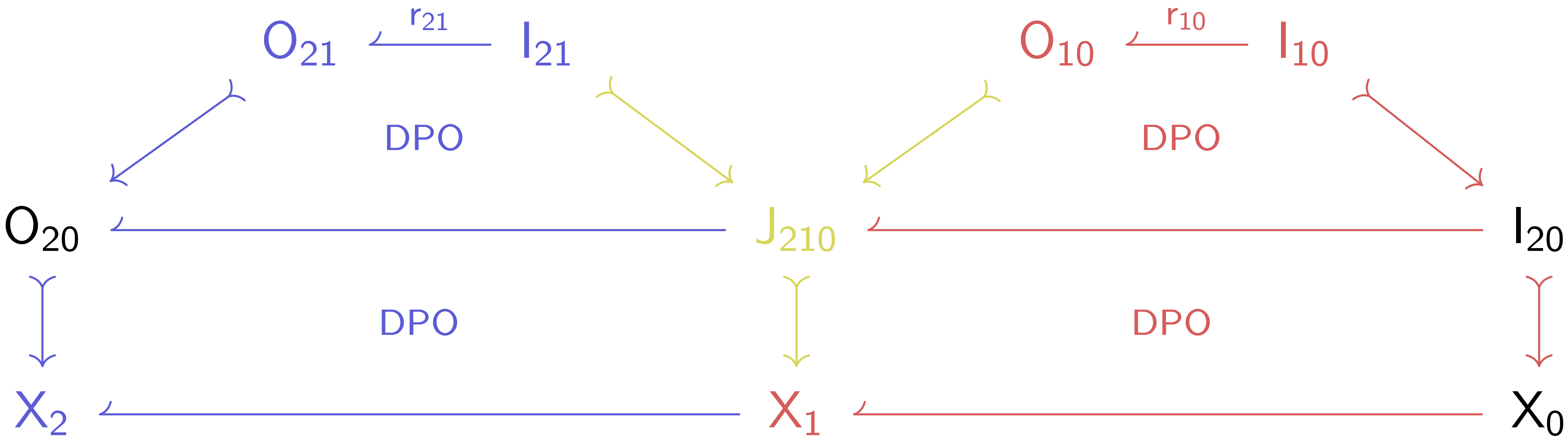
DPO-type **concurrency theorem**



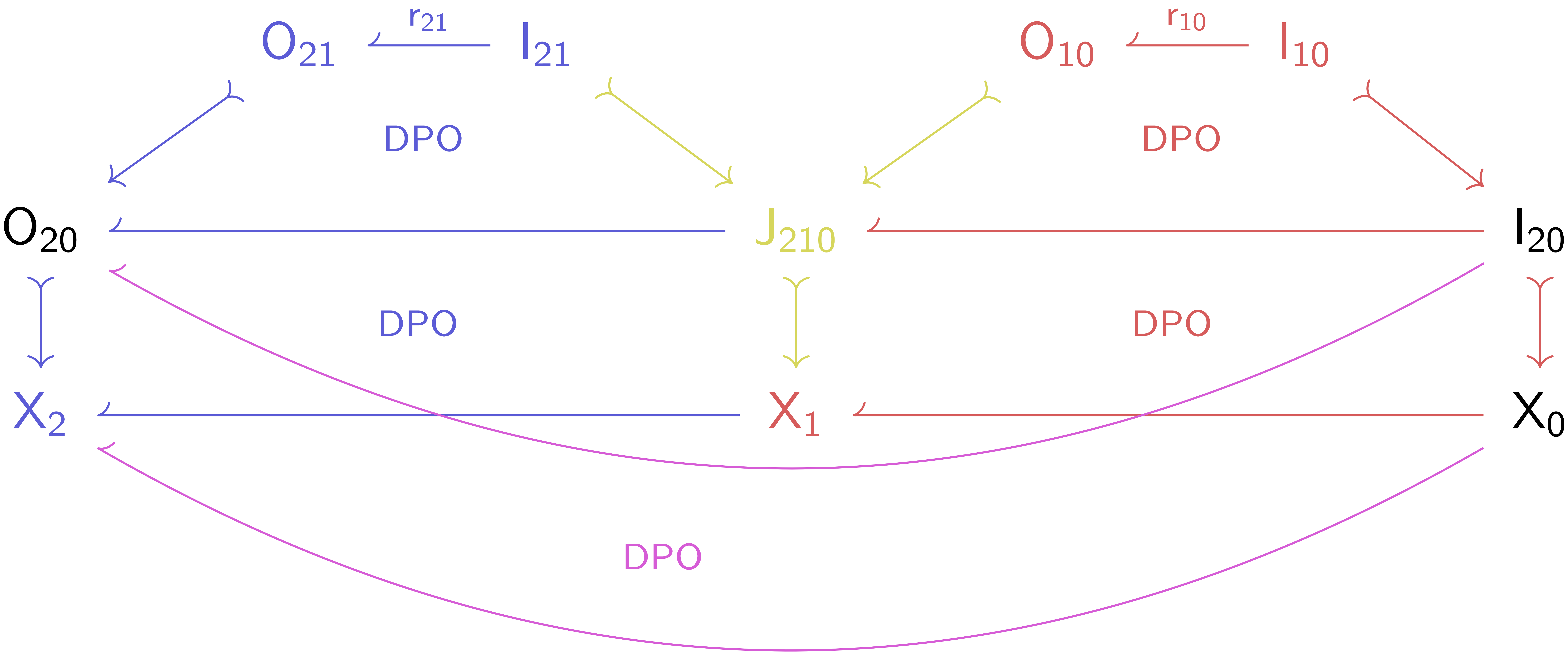
DPO-type **concurrency theorem**



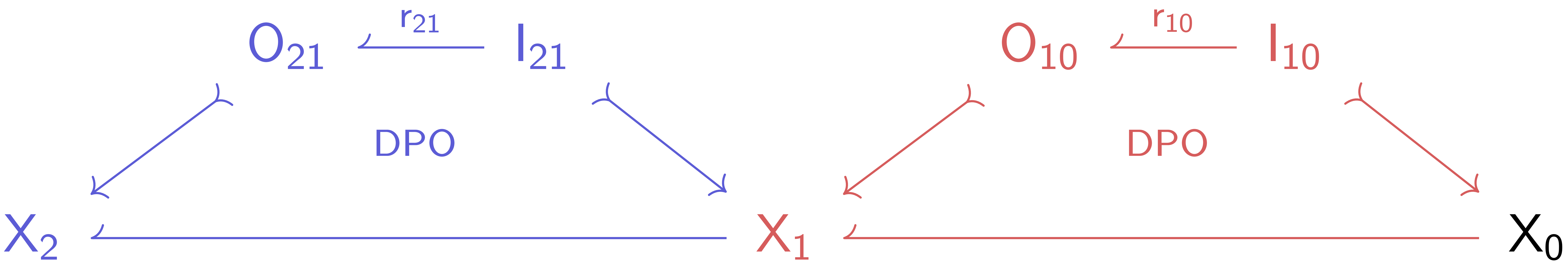
DPO-type **concurrency theorem**



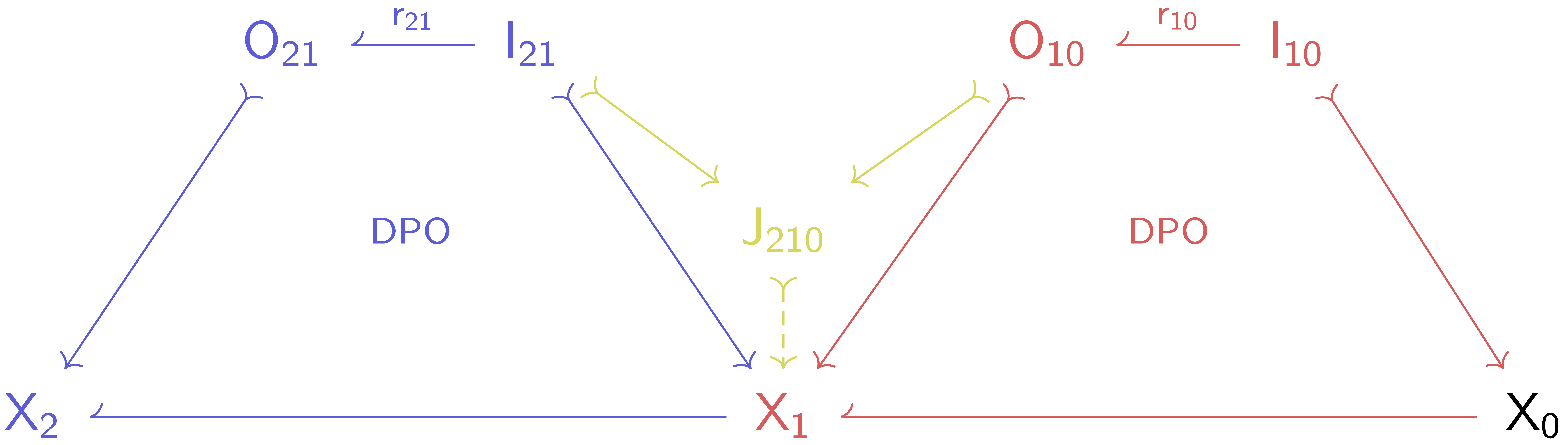
DPO-type **concurrency theorem**



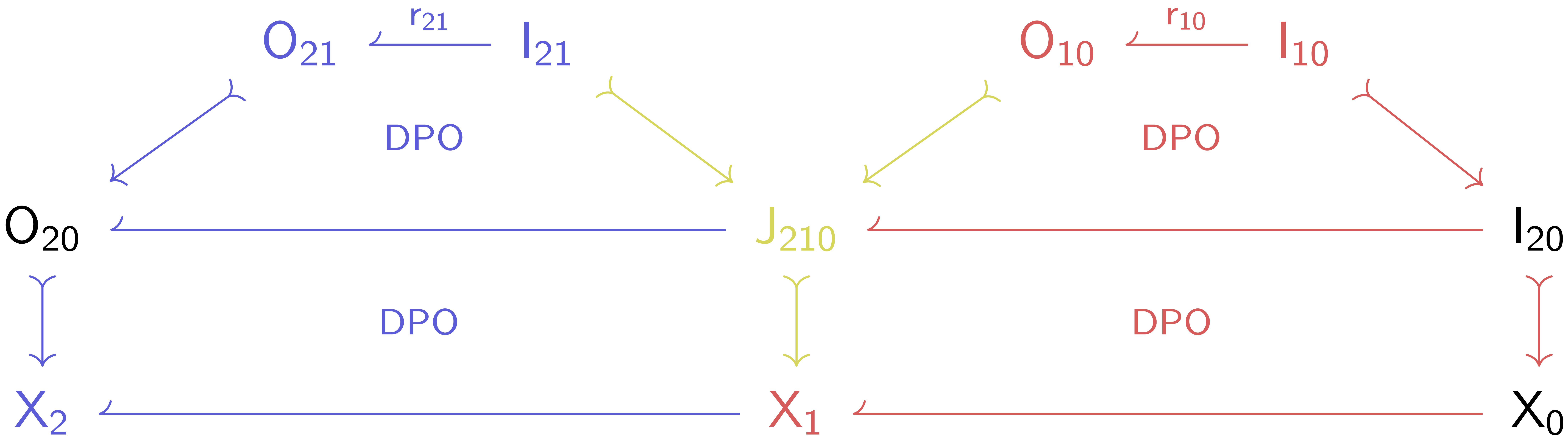
DPO-type **concurrency theorem — “synthesis”**



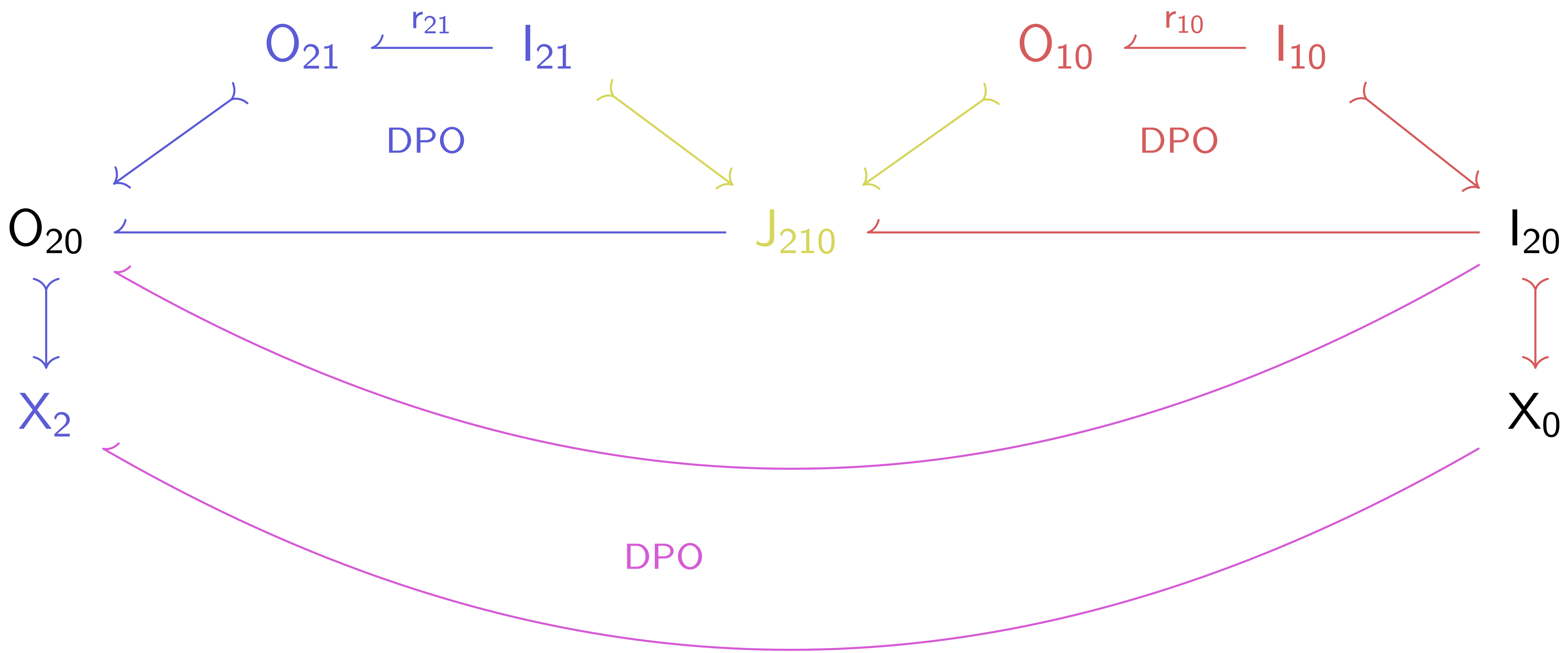
DPO-type **concurrency theorem — “synthesis”**



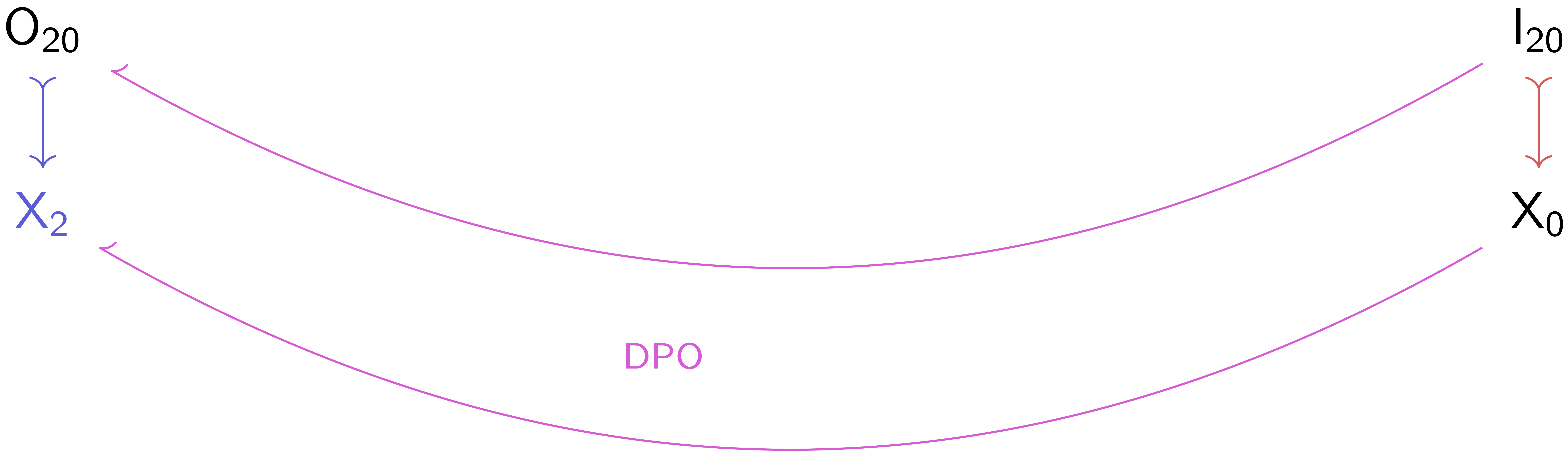
DPO-type **concurrency theorem — "synthesis"**



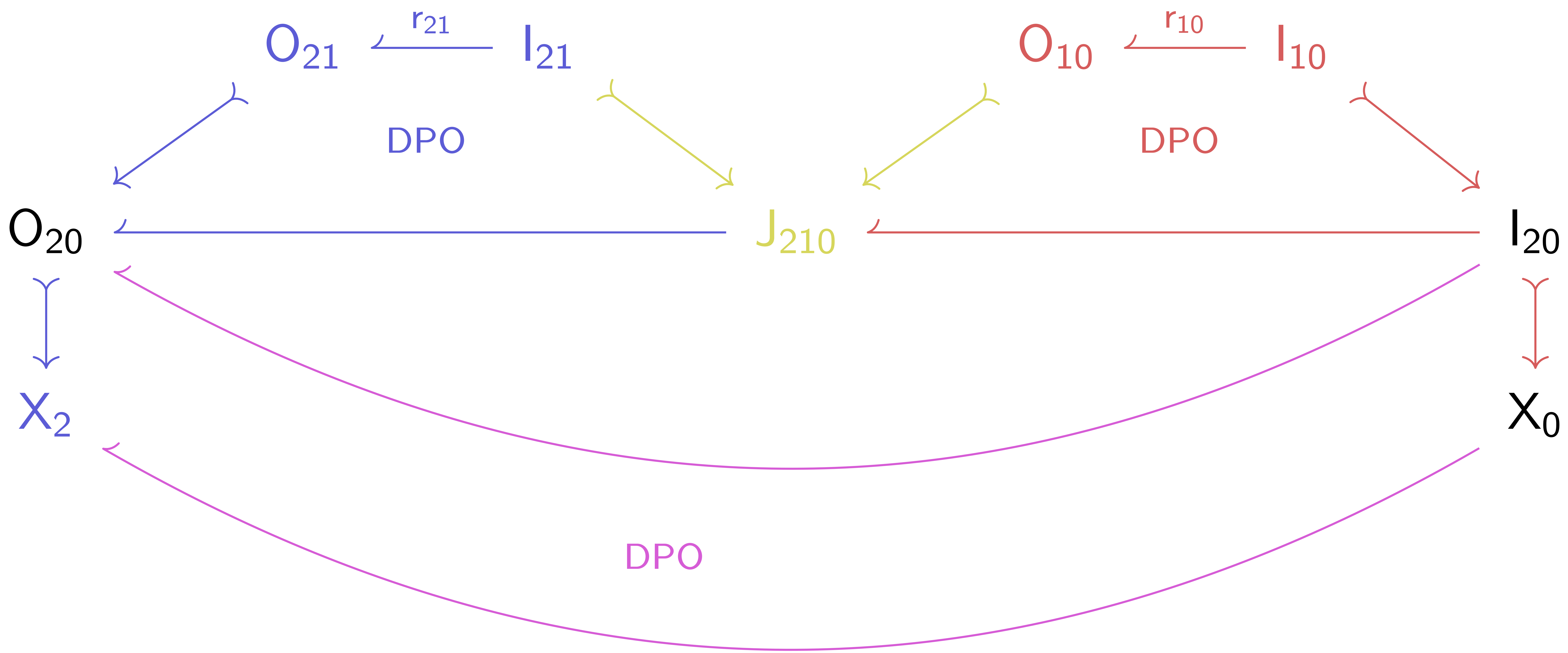
DPO-type **concurrency theorem — "synthesis"**



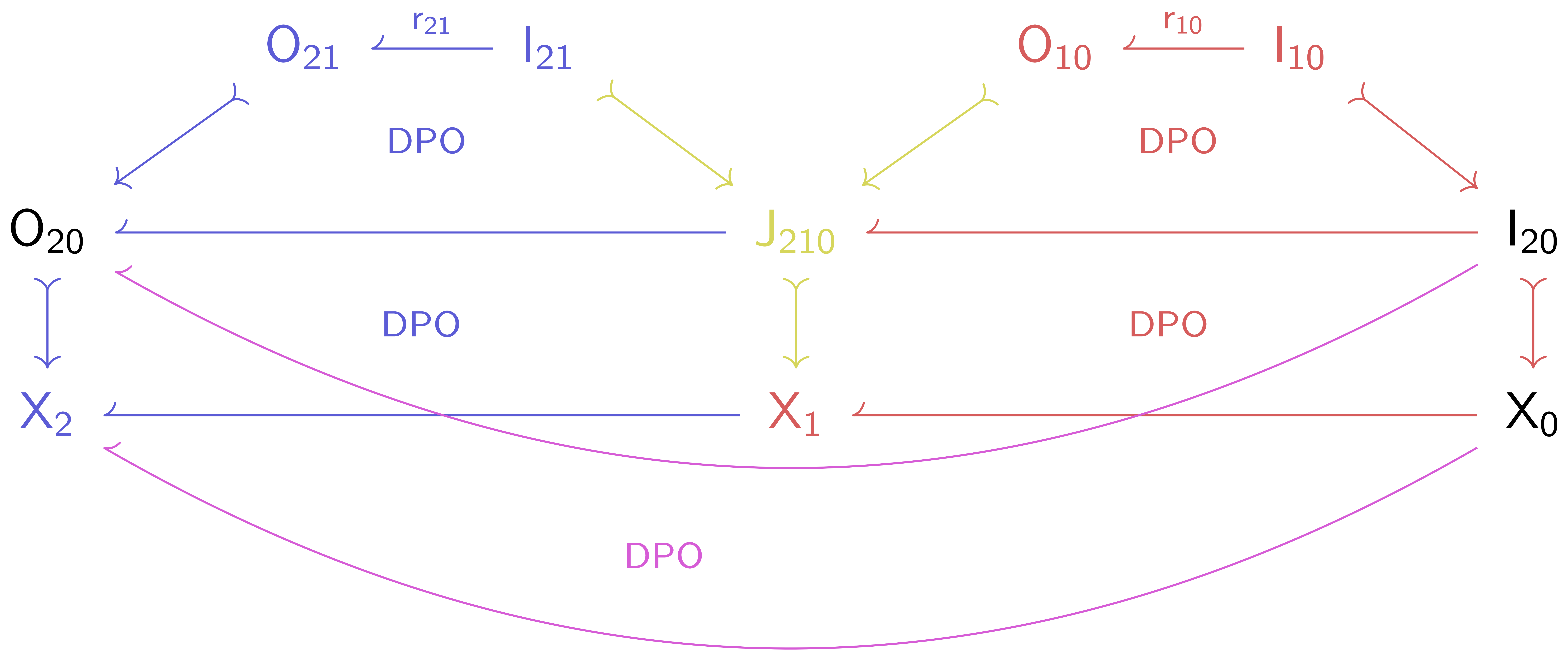
DPO-type **concurrency theorem** — **“synthesis”**



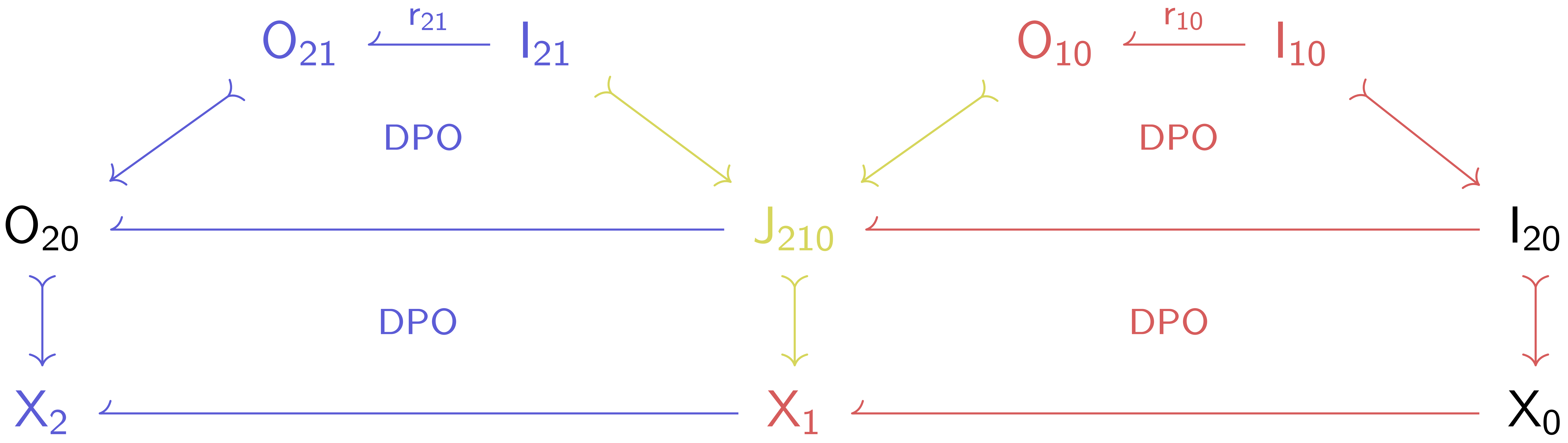
DPO-type **concurrency theorem — “analysis”**



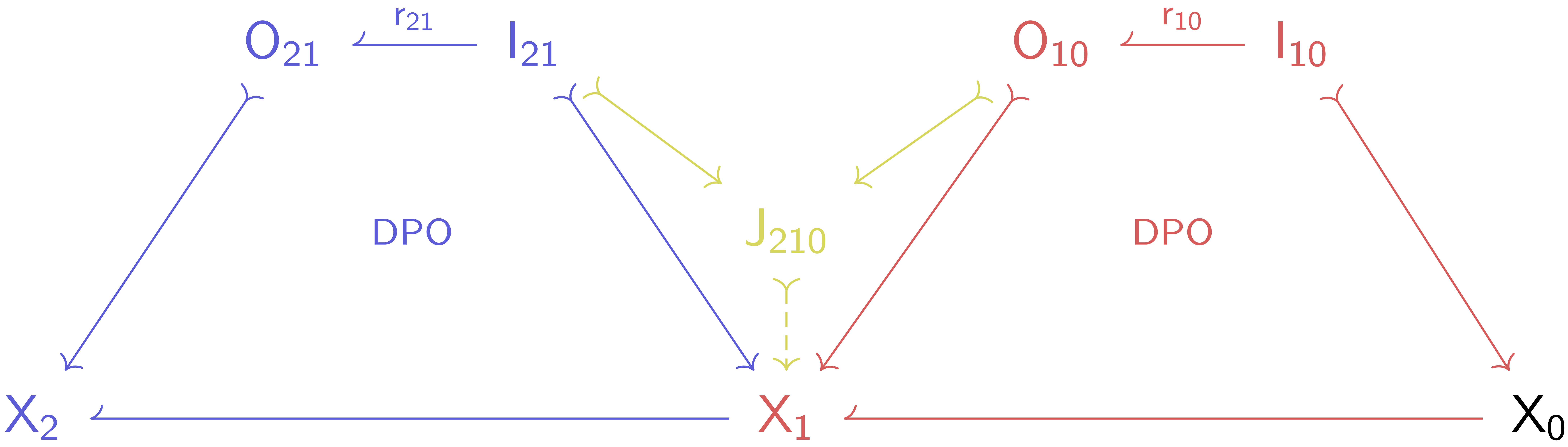
DPO-type **concurrency theorem** — **“analysis”**



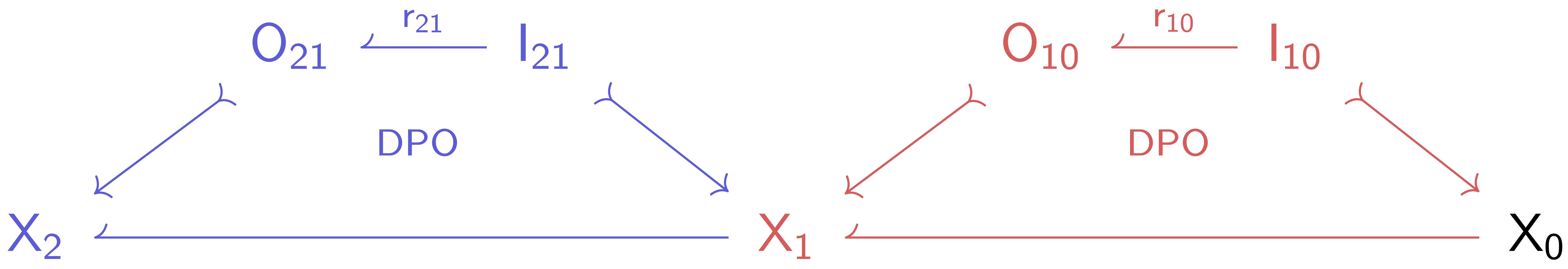
DPO-type **concurrency theorem — “analysis”**



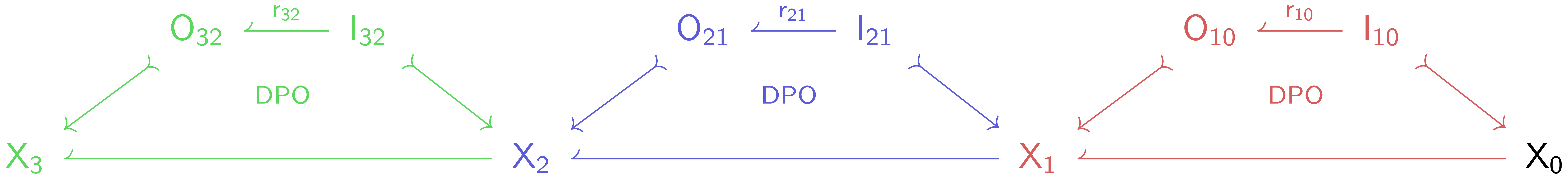
DPO-type **concurrency theorem — “analysis”**



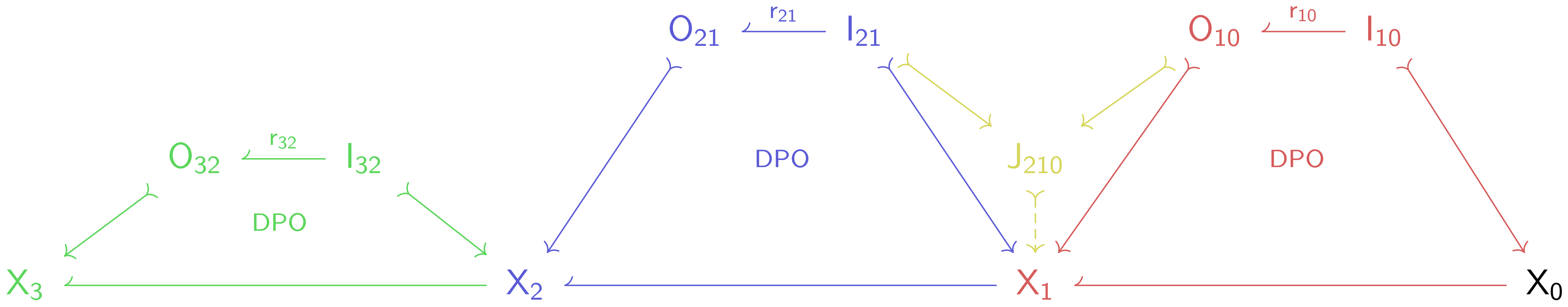
DPO-type **concurrency theorem — "analysis"**



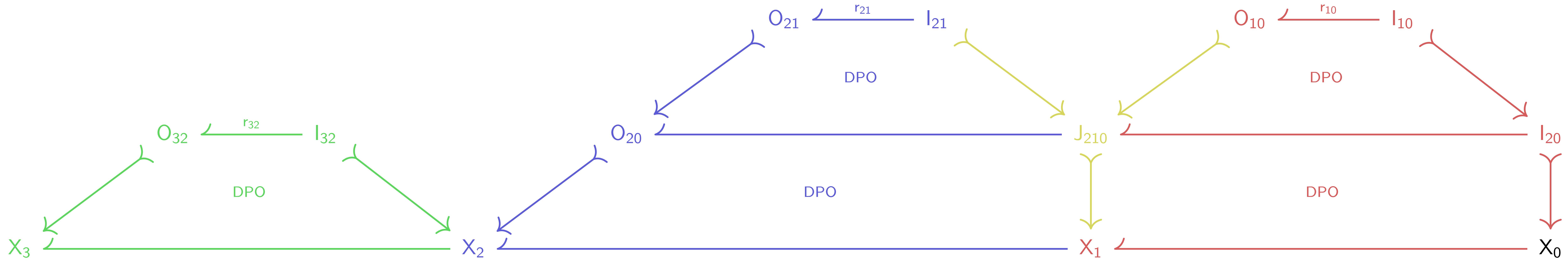
From the **DPO-type concurrency theorem** to **tracelets**



From the **DPO-type concurrency theorem** to **tracelets**

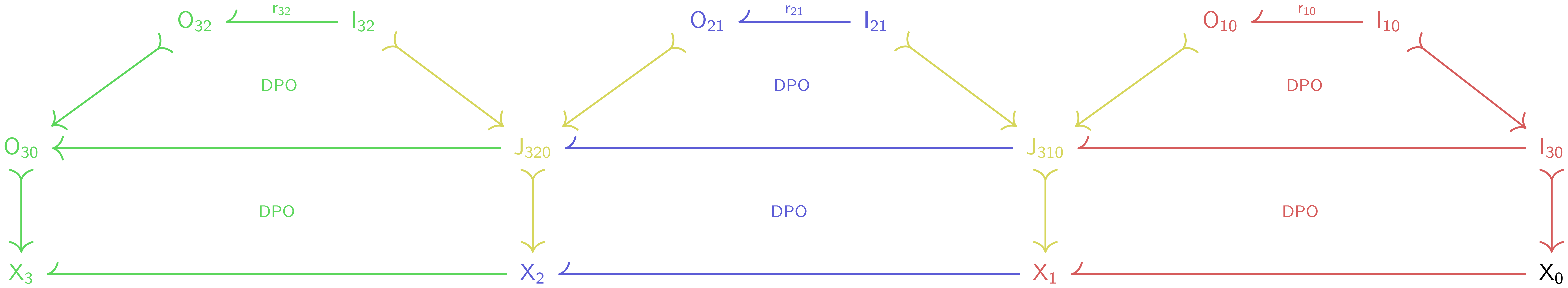


From the **DPO-type concurrency theorem** to **tracelets**



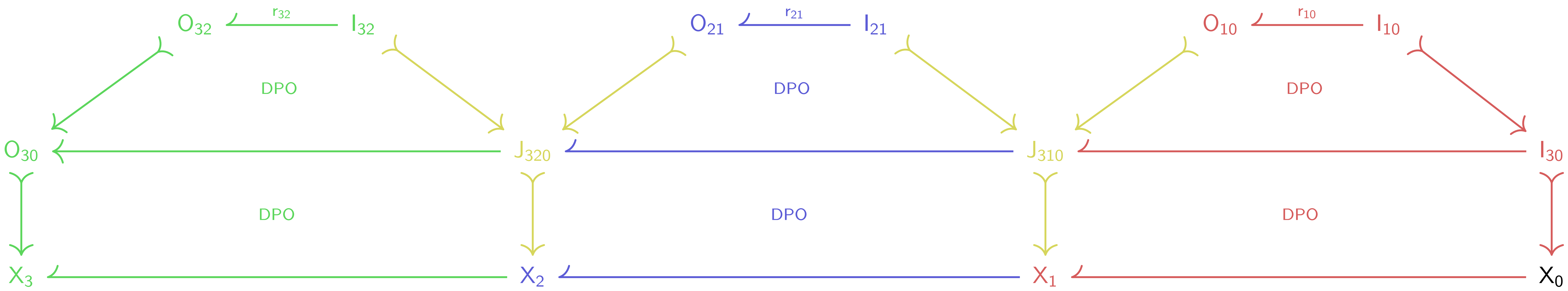
From the **DPO-type concurrency theorem** to **tracelets**

From the **DPO-type concurrency theorem** to **tracelets**



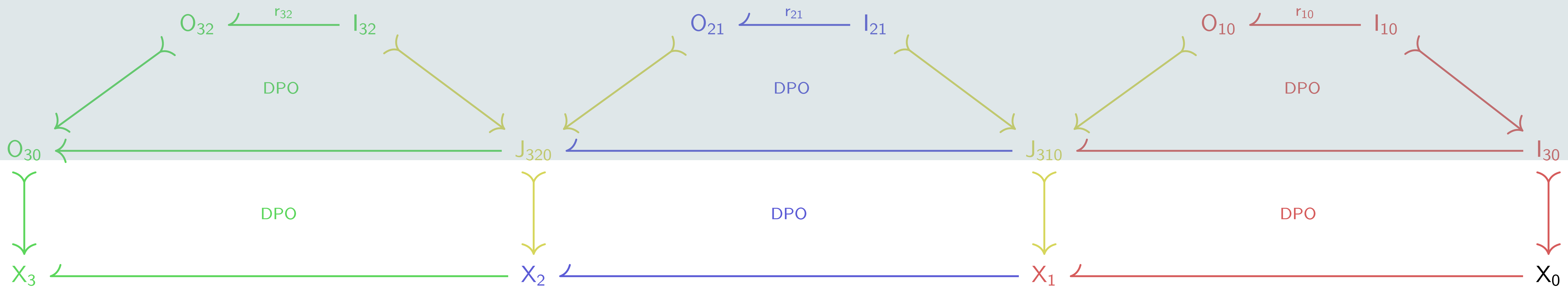
From the **DPO-type concurrency theorem** to **tracelets**

From the **DPO-type concurrency theorem** to **tracelets**



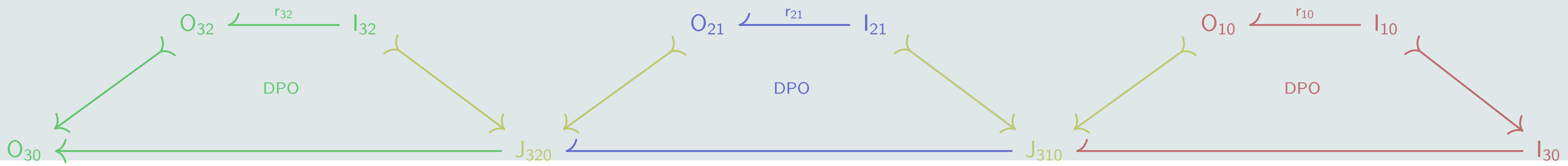
From the **DPO-type concurrency theorem** to **tracelets**

TRACELET (of length 3)



From the **DPO-type concurrency theorem** to **tracelets**

TRACELET (of length 3)



Tracelet generation

Definition: tracelets of **length 1**

$$
\begin{array}{ccc}
O & \hookleftarrow & I \\
\| & \text{DPO} & \| \\
O & \hookleftarrow & I
\end{array}
\quad := \quad
\begin{array}{ccccc}
O & \leftarrowtail & K & \rightarrowtail & I \\
\| & & \| & & \| \\
O & \leftarrowtail & K & \rightarrowtail & I
\end{array}
$$

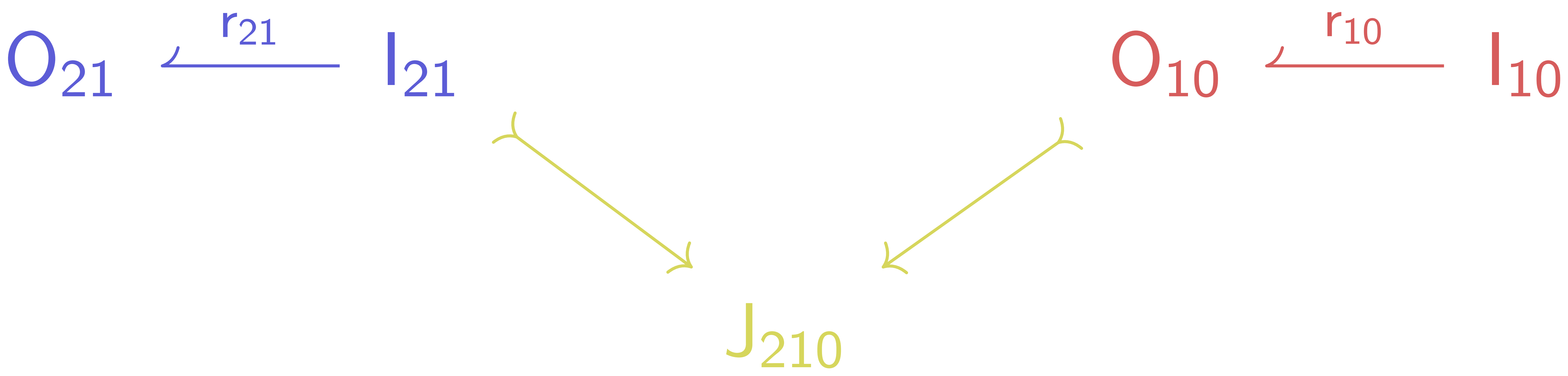
Tracelet generation

Definition: tracelets of **length 2**

$$O_{21} \xleftarrow{r_{21}} I_{21} \qquad\qquad O_{10} \xleftarrow{r_{10}} I_{10}$$

Tracelet generation

Definition: tracelets of **length 2**

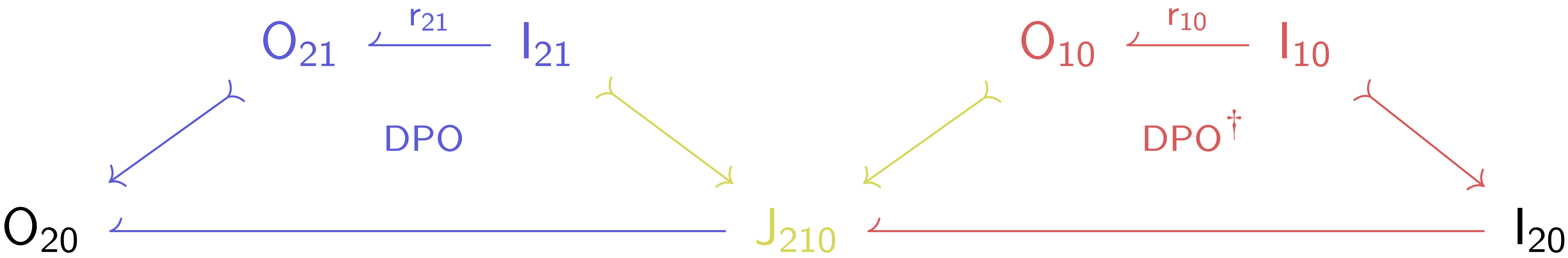


Tracelet generation

Definition: tracelets of **length 2**

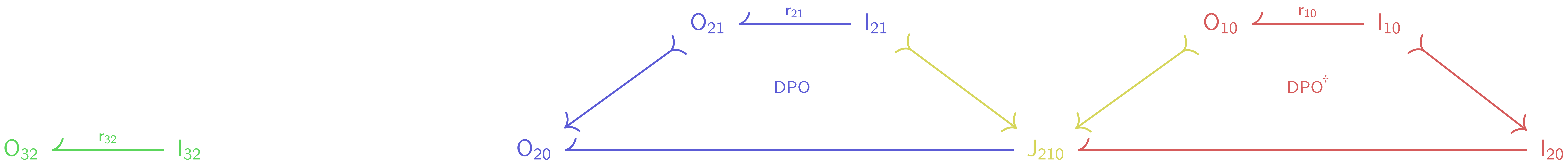
Tracelet generation

Definition: tracelets of **length 2**



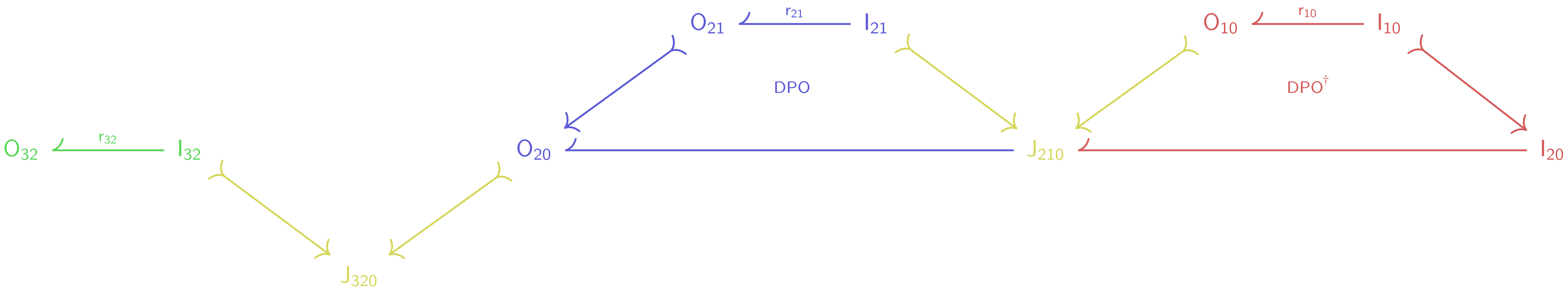
Tracelet generation

Definition: tracelets of **length 3**



Tracelet generation

Definition: tracelets of **length 3**



Tracelet generation

Definition: tracelets of **length 3**

Tracelet generation

Definition: tracelets of **length 3**

Tracelet generation

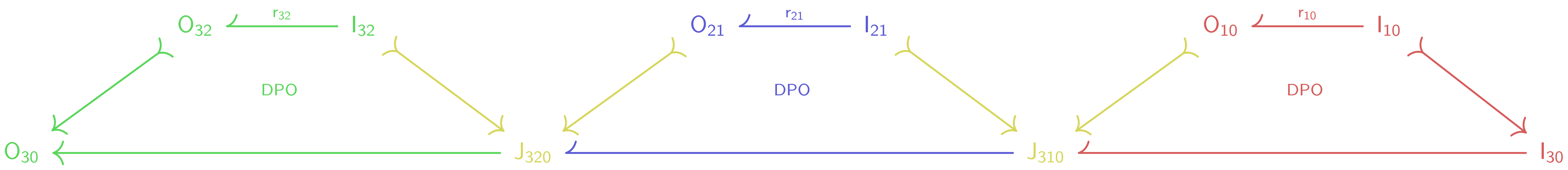
Definition: tracelets of **length 3**

Tracelet generation

Definition: tracelets of **length 3**

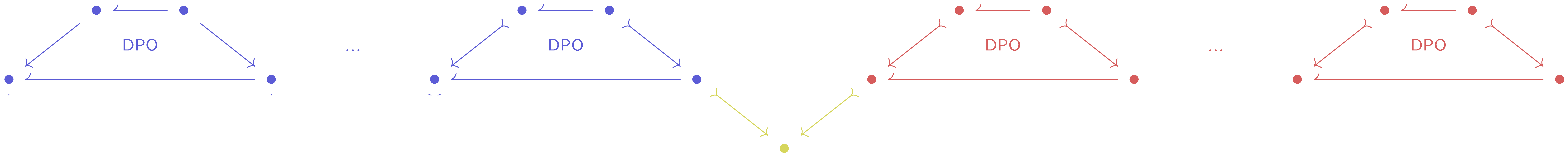
Tracelet generation

Definition: tracelets of **length 3**

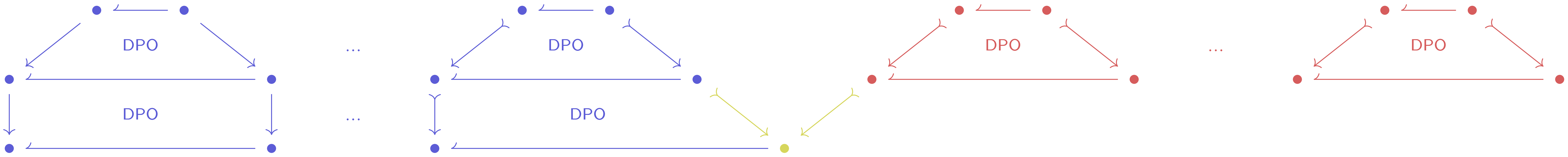


Tracelet composition

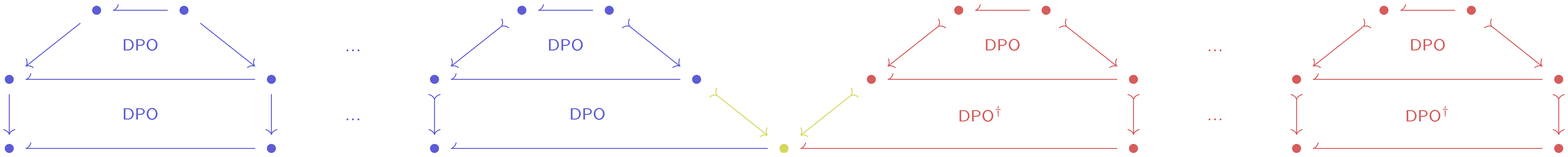
Tracelet composition



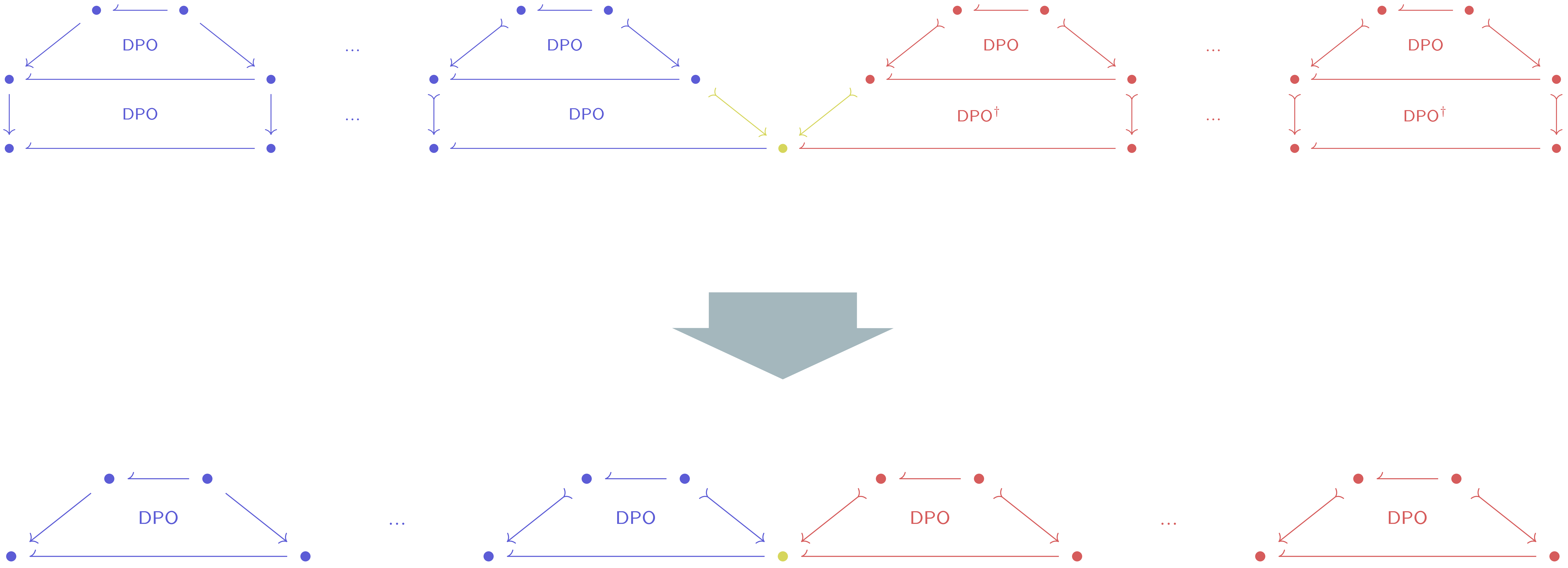
Tracelet composition



Tracelet composition



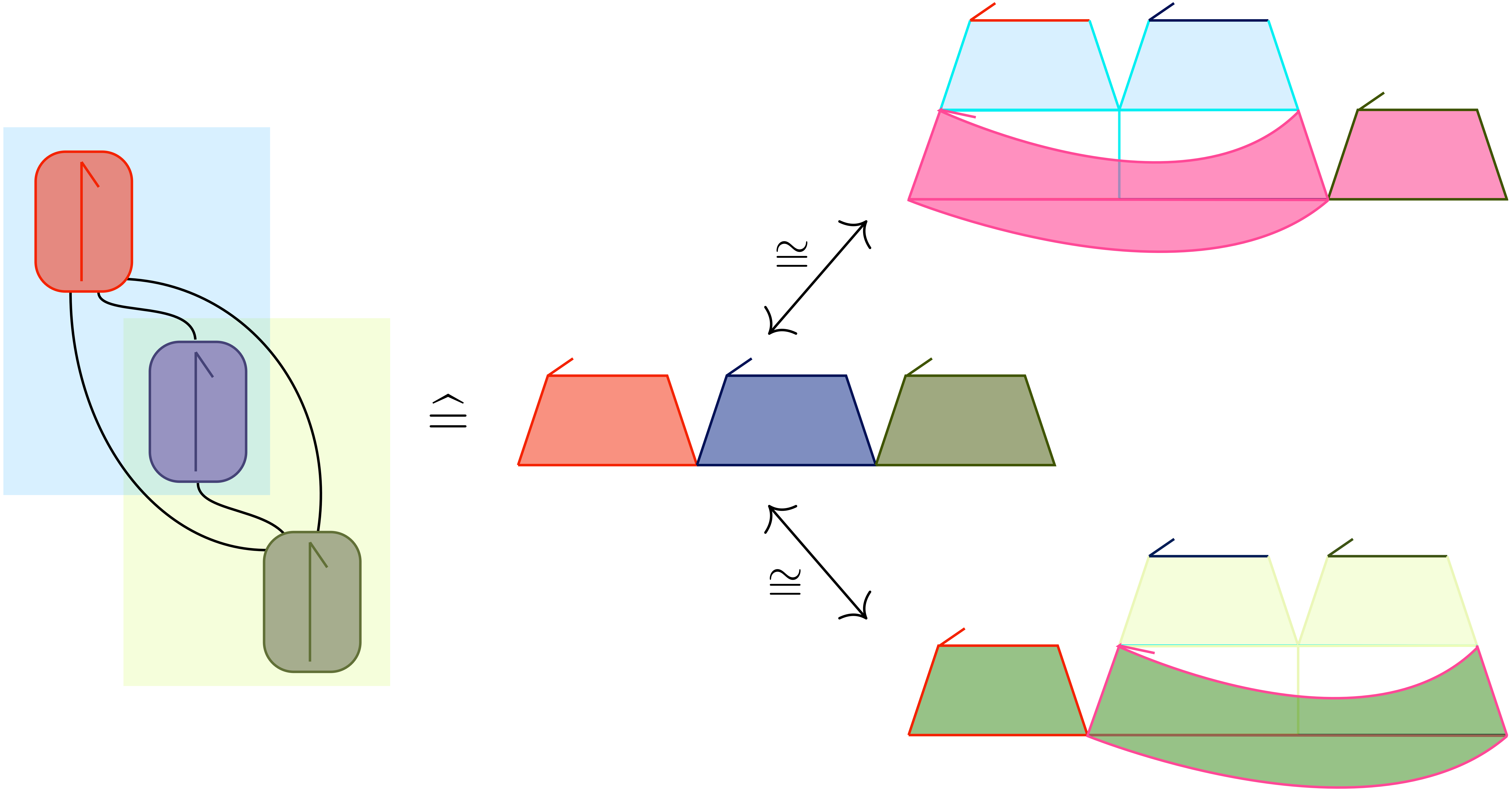
Tracelet composition



Plan of the talk

1. Discrete rewriting and diagram Hopf Algebras
2. Categorical rewriting theory
3. From rewriting to tracelets
4. Tracelet decomposition spaces
5. Tracelet Hopf algebras

Motivation: key property of compositional rewriting theory

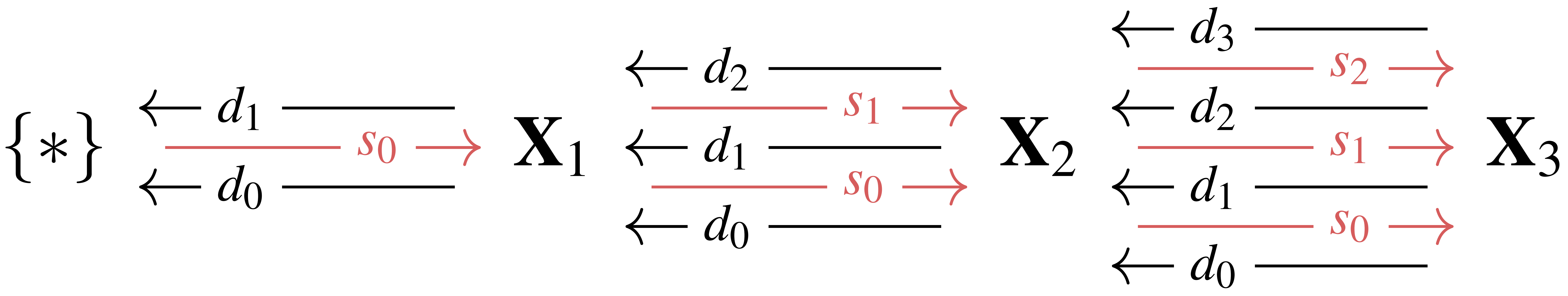


Construction of a suitable **decomposition space**

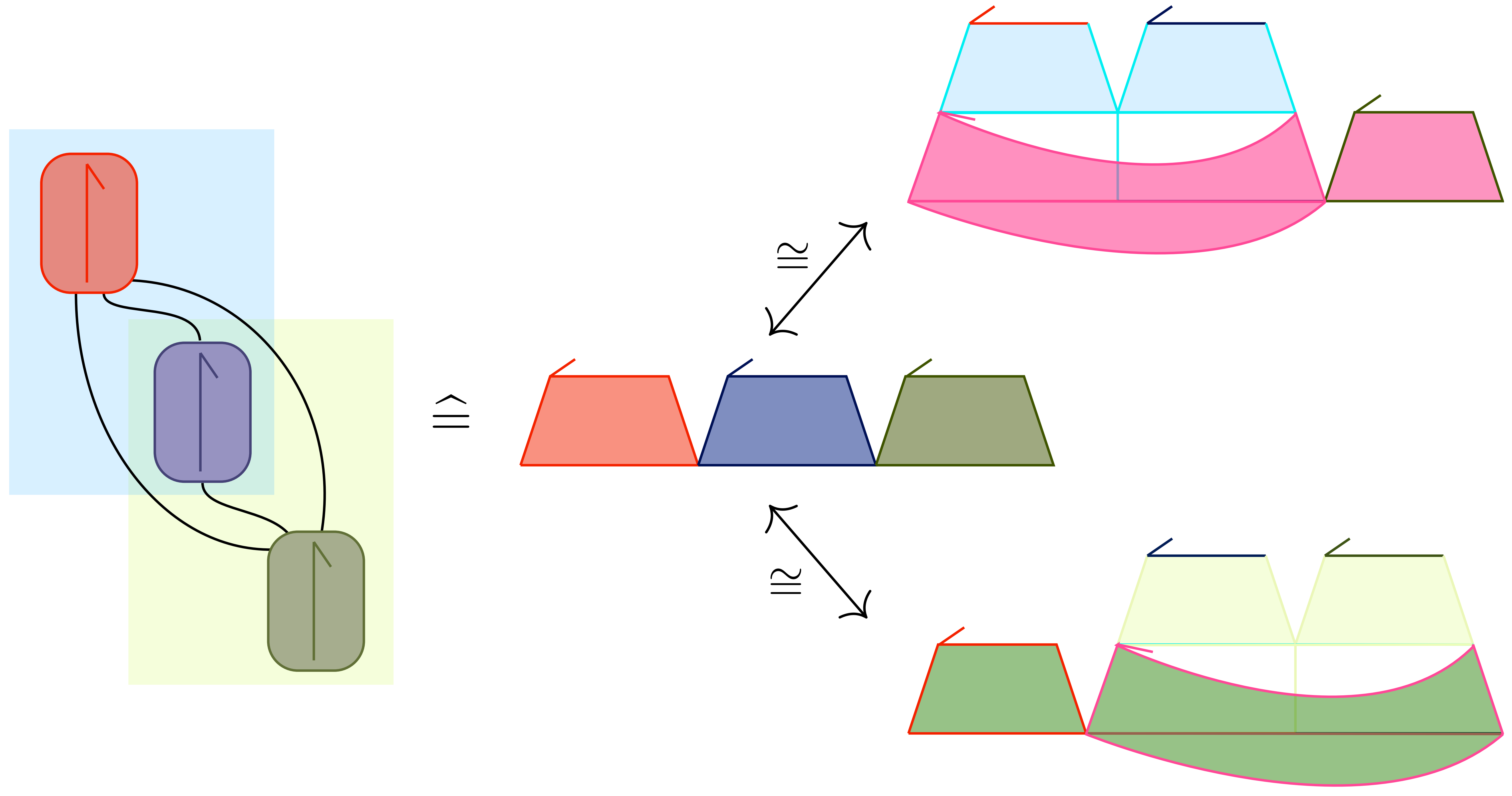
Tracelet Hopf algebras and decomposition spaces

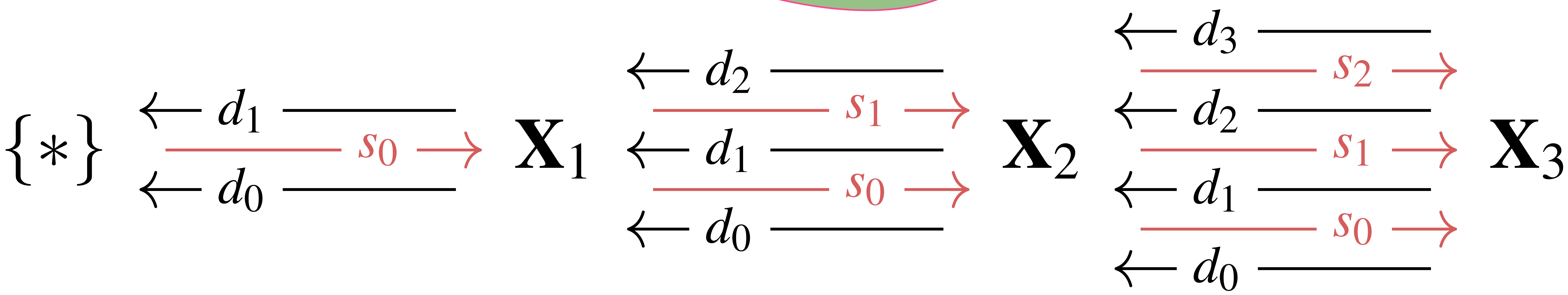
Nicolas Behr
Univ. de Paris, CNRS, IRIF, F-75006, Paris, France
nicolas.behr@irif.fr

Joachim Kock
Universitat Autònoma de Barcelona &
Centre de Recerca Matemàtica
kock@mat.uab.cat

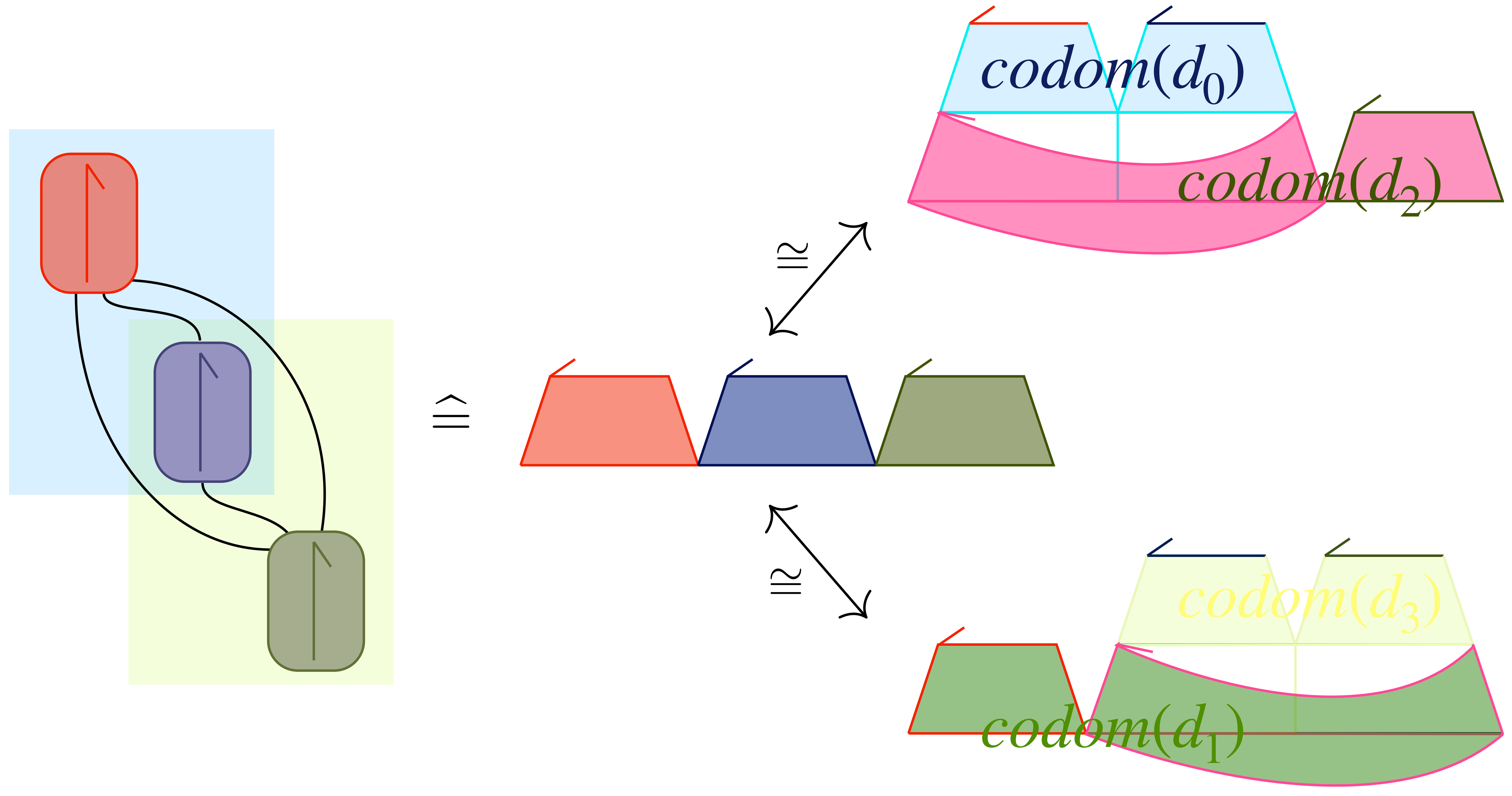


Construction of a suitable **decomposition space**





Construction of a suitable **decomposition space**



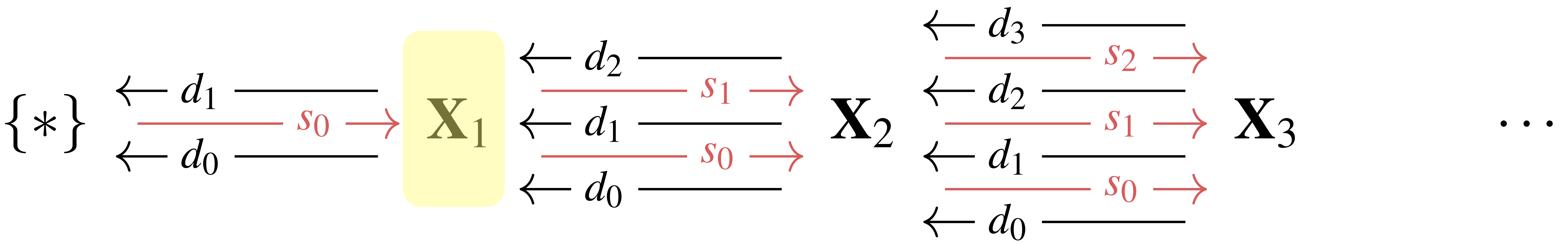
First hint: at **length 3**, the top and bottom diagrams in the equivalence suggest **four "forgetful" mappings**, which are the candidates for the **face maps** $d_0, \ldots, d_3$

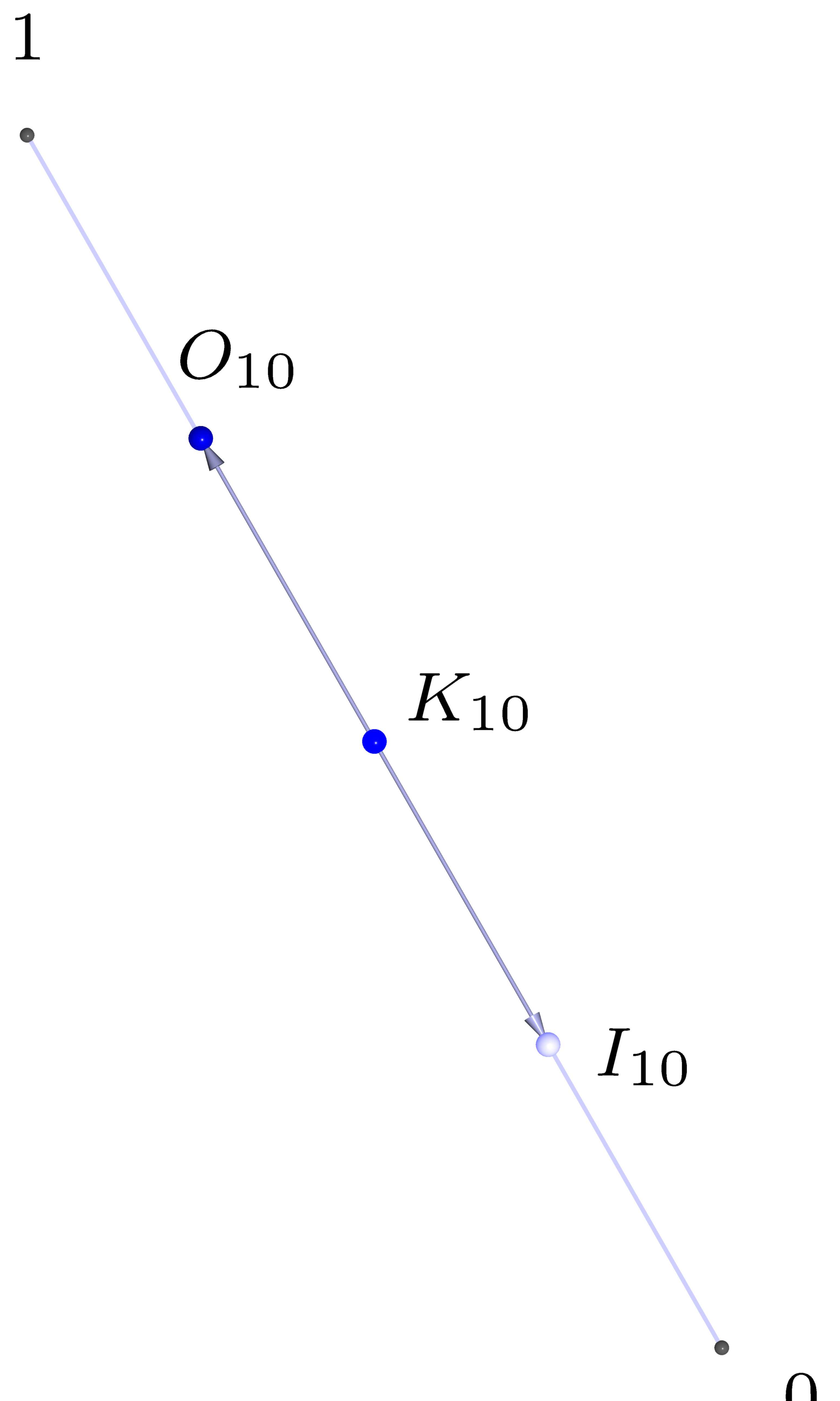
$$\{*\} \underset{\longleftarrow d_0 \longrightarrow}{\overset{\longleftarrow d_1 \longrightarrow}{\xrightarrow{s_0}}} \mathbf{X}_1 \quad \begin{matrix} \xleftarrow{d_2} \\ \xrightarrow{s_1} \\ \xleftarrow{d_1} \\ \xrightarrow{s_0} \\ \xleftarrow{d_0} \end{matrix} \quad \mathbf{X}_2 \quad \begin{matrix} \xleftarrow{d_3} \\ \xrightarrow{s_2} \\ \xleftarrow{d_2} \\ \xrightarrow{s_1} \\ \xleftarrow{d_1} \\ \xrightarrow{s_0} \\ \xleftarrow{d_0} \end{matrix} \quad \mathbf{X}_3$$

Construction of a suitable **decomposition space**

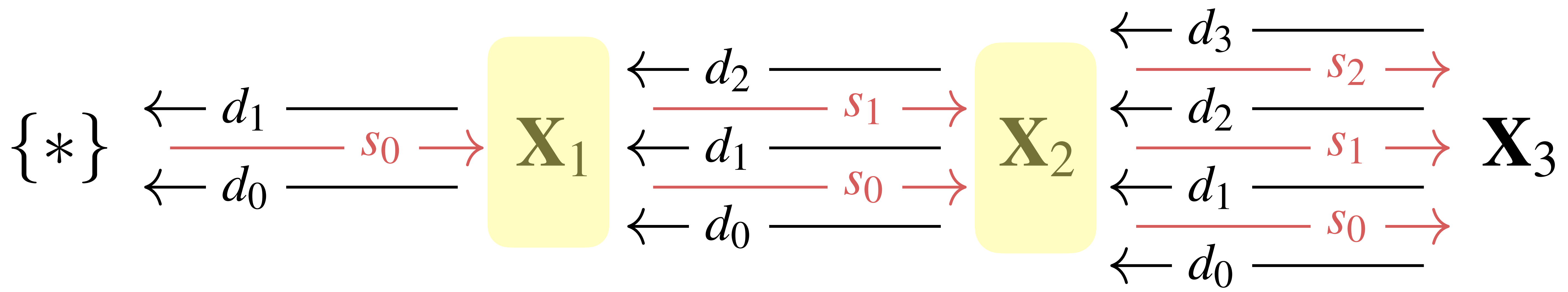
$$\{*\}\ \begin{array}{c}\xleftarrow{d_1}\\ \xrightarrow{s_0}\\ \xleftarrow{d_0}\end{array}\ \mathbf{X}_1\ \begin{array}{c}\xleftarrow{d_2}\\ \xrightarrow{s_1}\\ \xleftarrow{d_1}\\ \xrightarrow{s_0}\\ \xleftarrow{d_0}\end{array}\ \mathbf{X}_2\ \begin{array}{c}\xleftarrow{d_3}\\ \xrightarrow{s_2}\\ \xleftarrow{d_2}\\ \xrightarrow{s_1}\\ \xleftarrow{d_1}\\ \xrightarrow{s_0}\\ \xleftarrow{d_0}\end{array}\ \mathbf{X}_3 \qquad \cdots$$

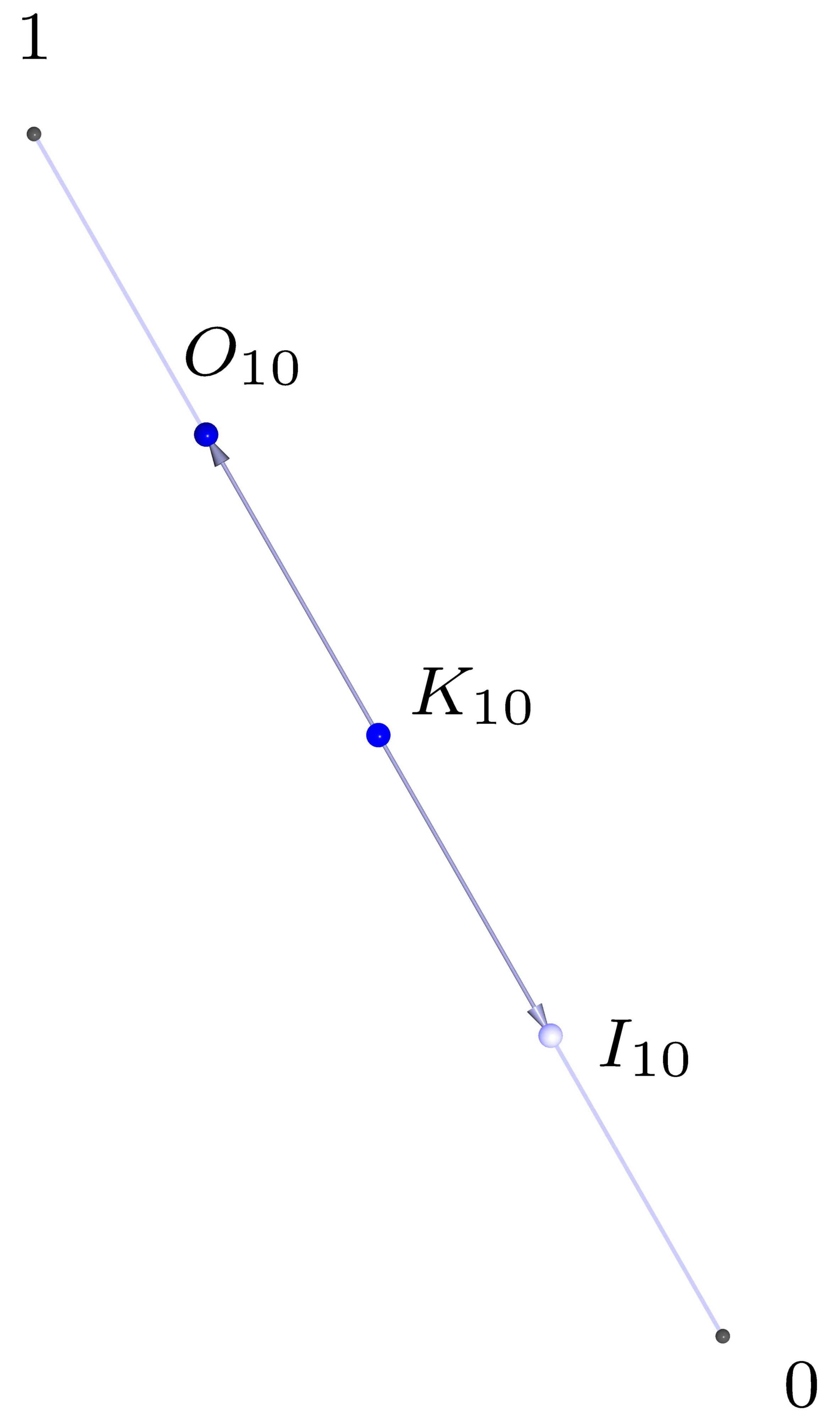
Construction of a suitable **decomposition space**

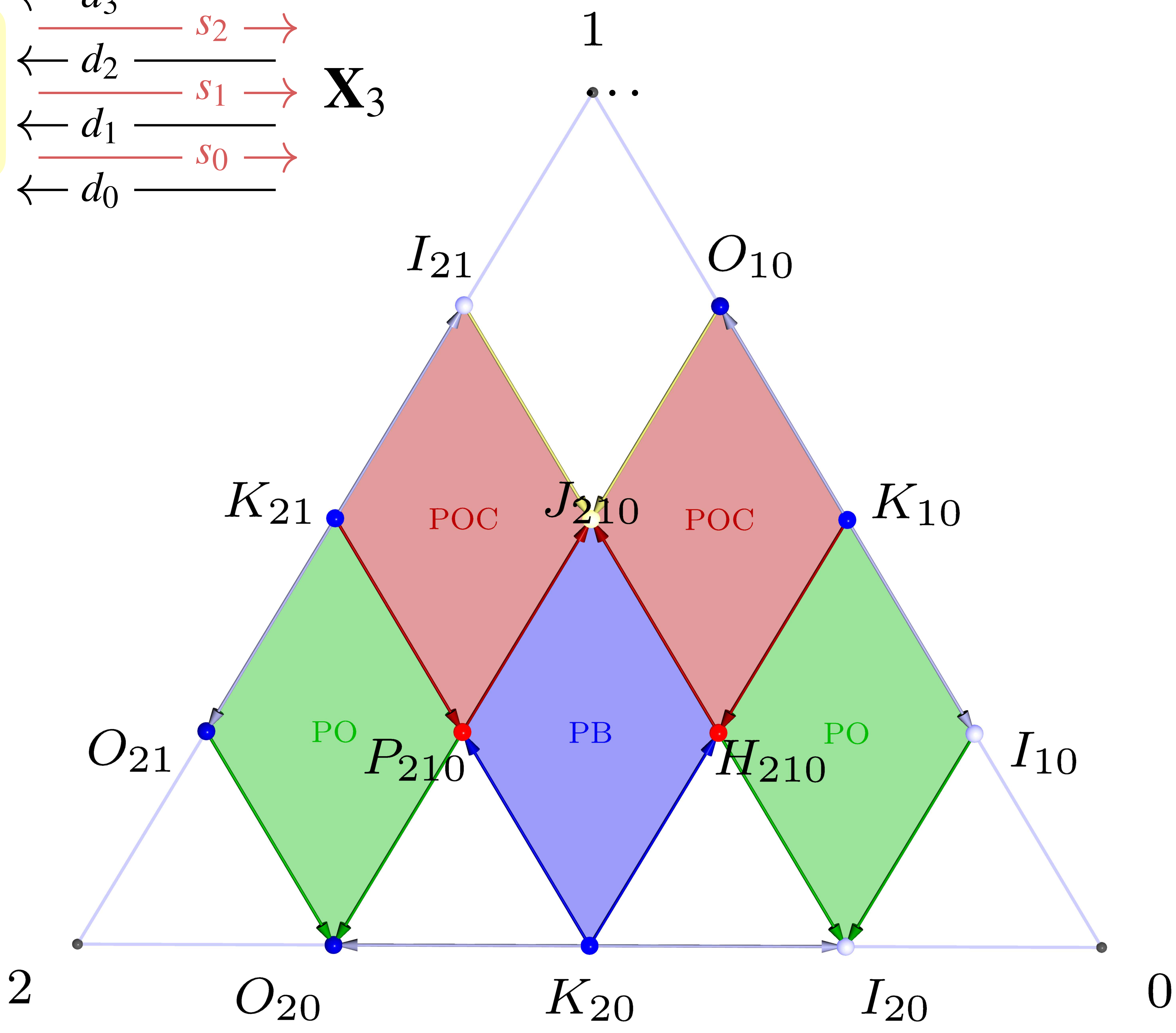




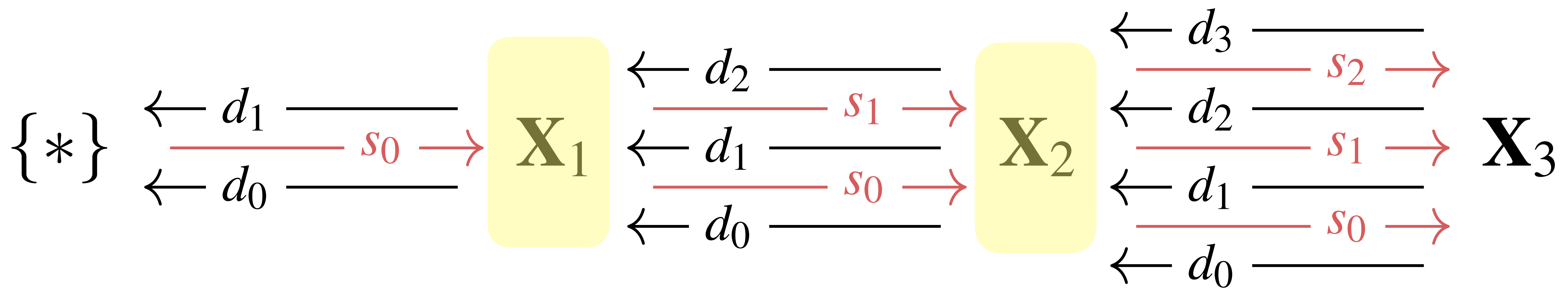
Construction of a suitable **decomposition space**

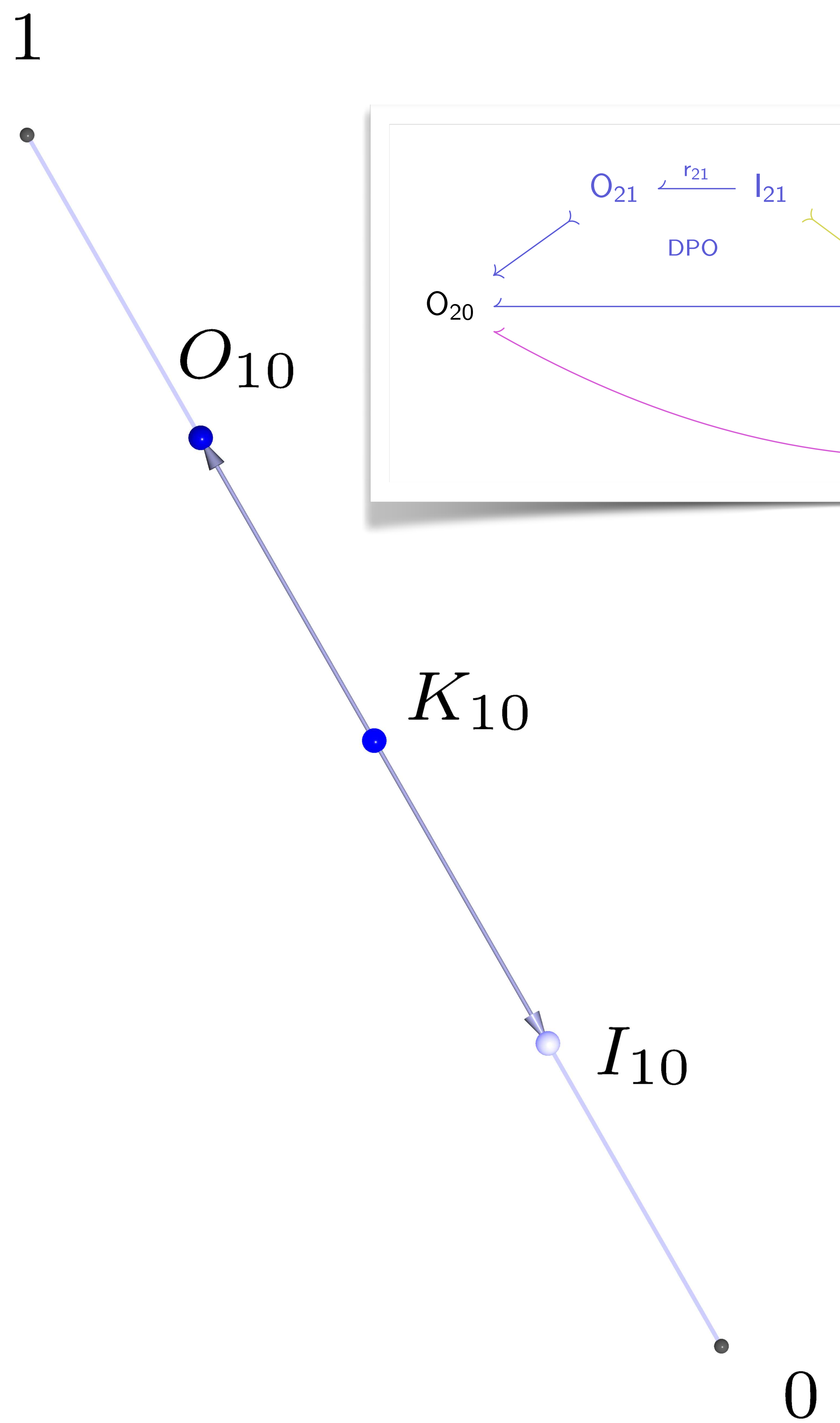


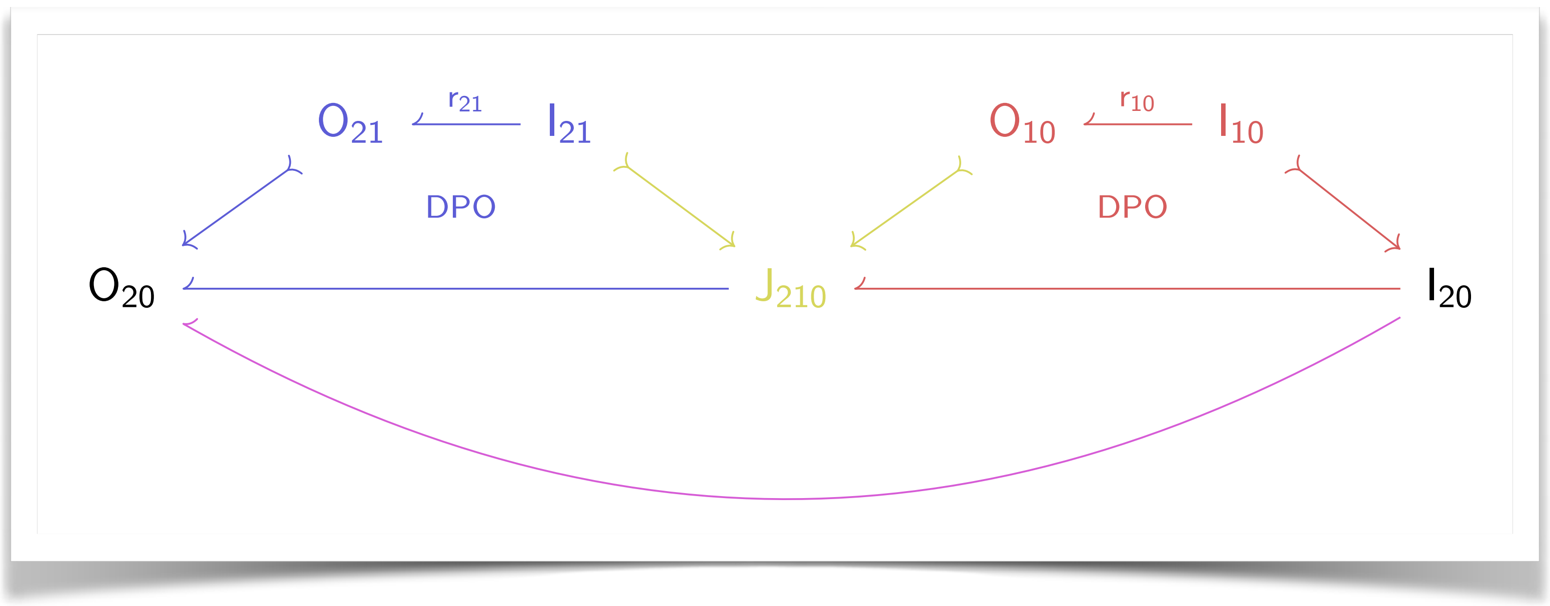


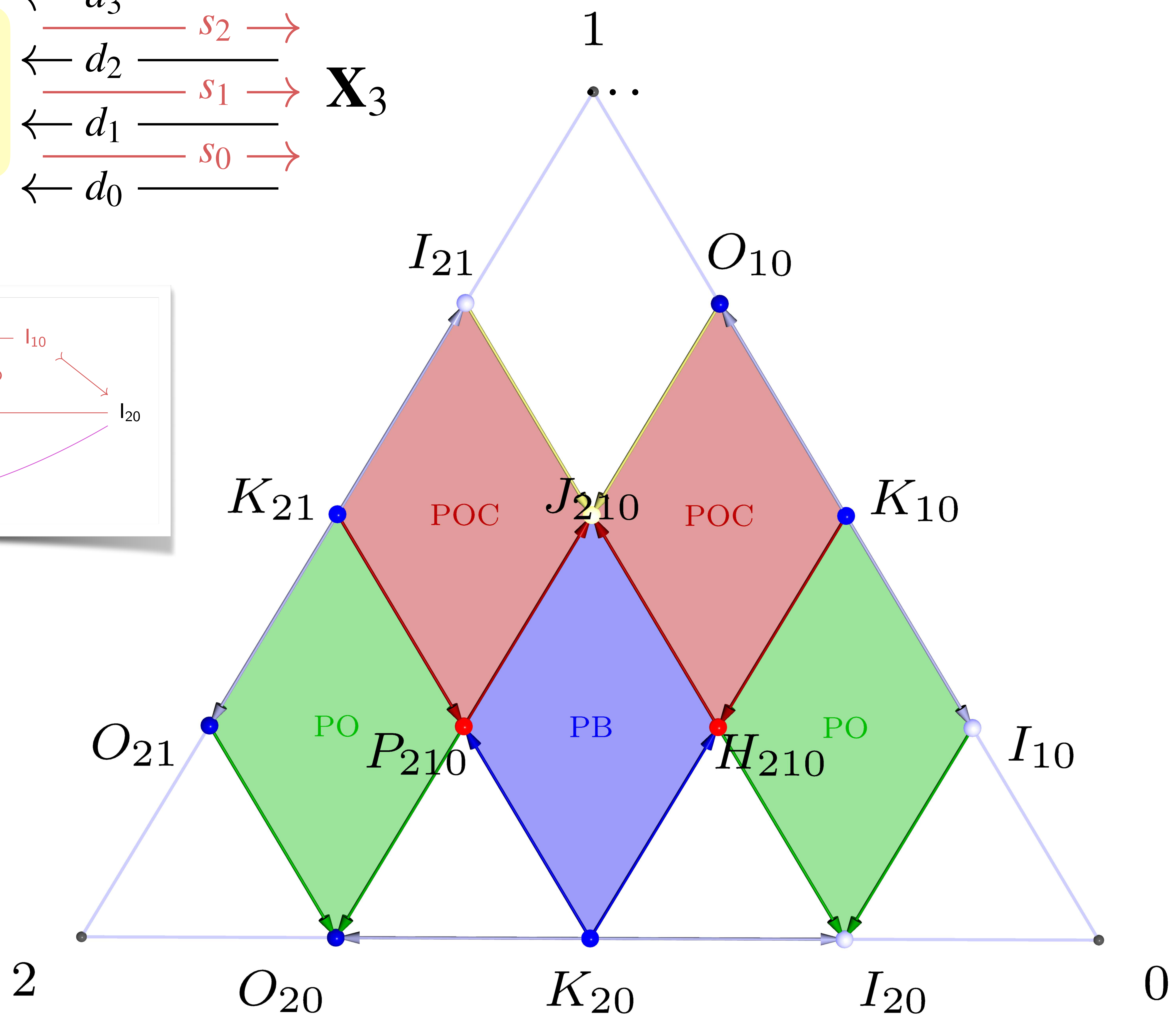


Construction of a suitable **decomposition space**

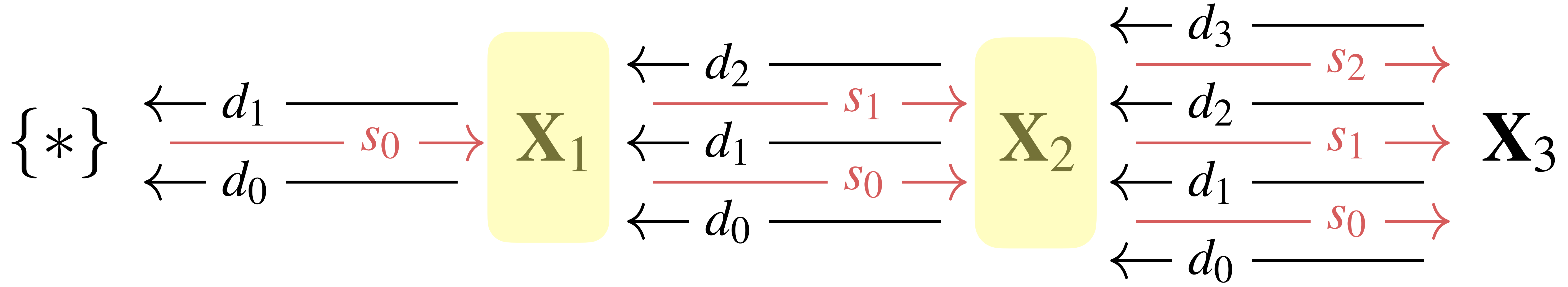


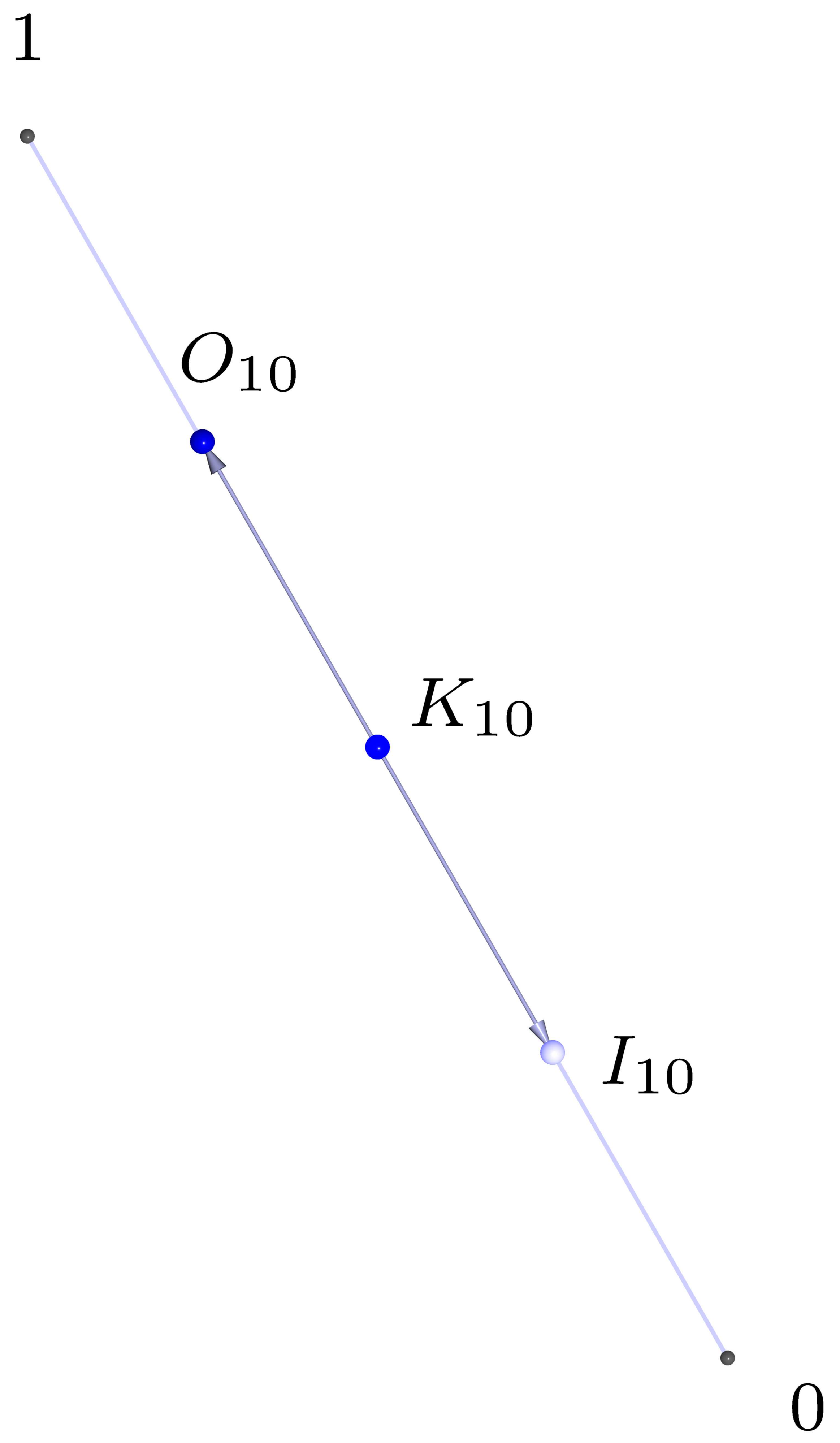


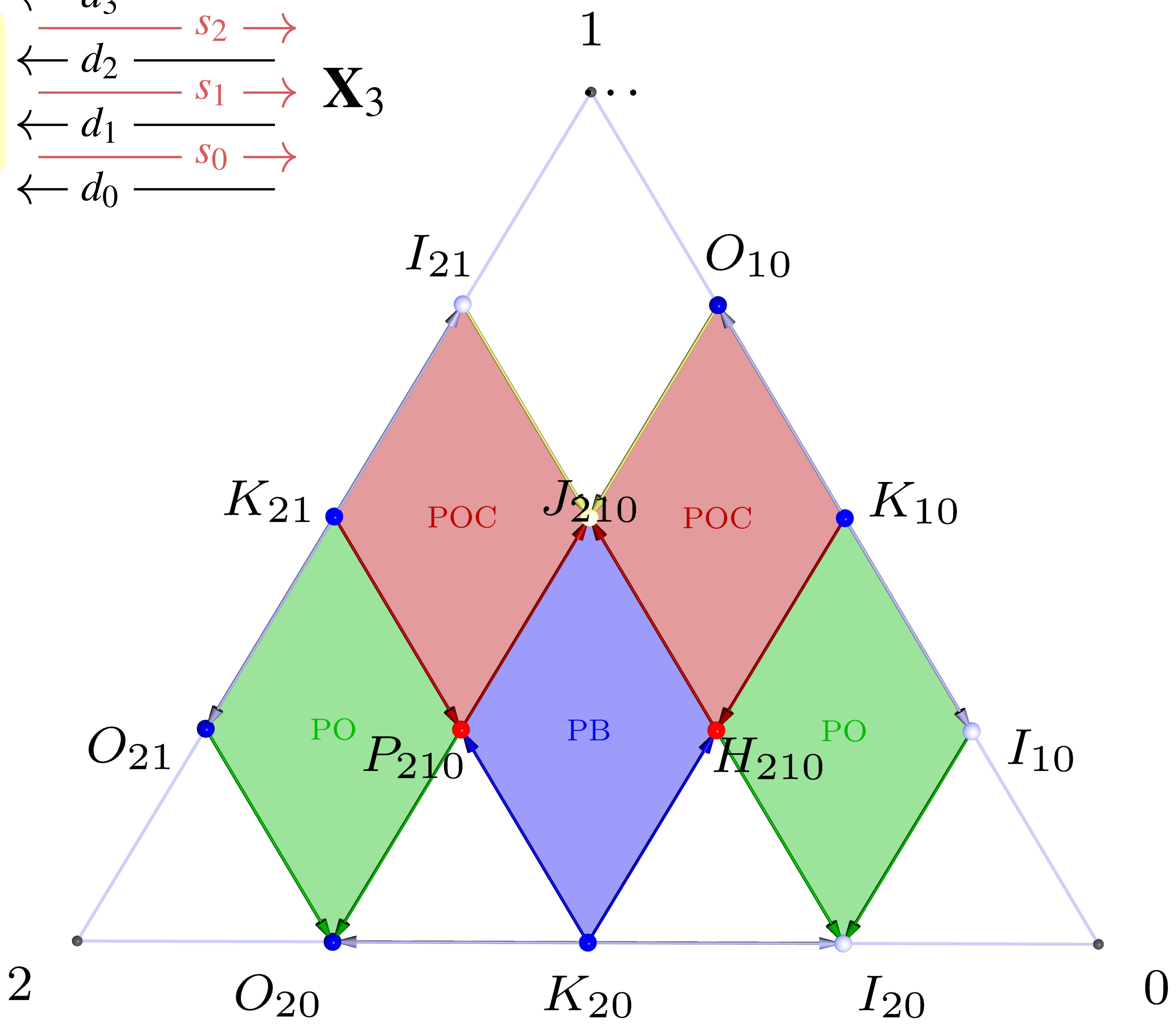




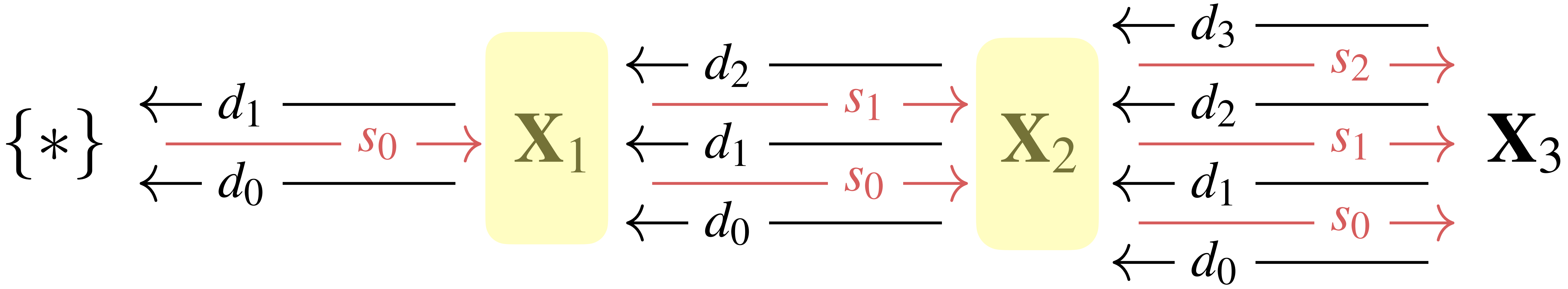
Construction of a suitable **decomposition space**



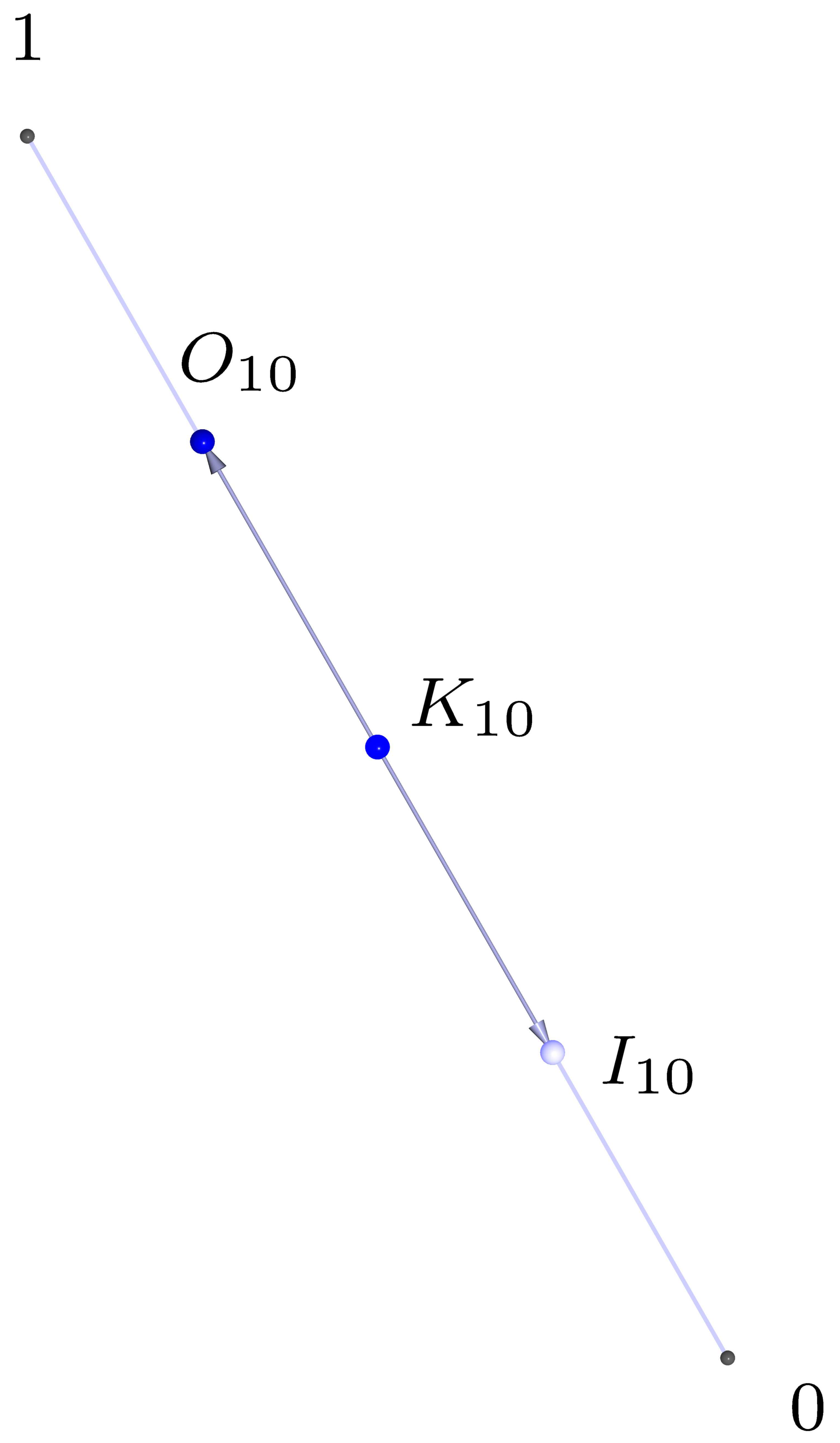


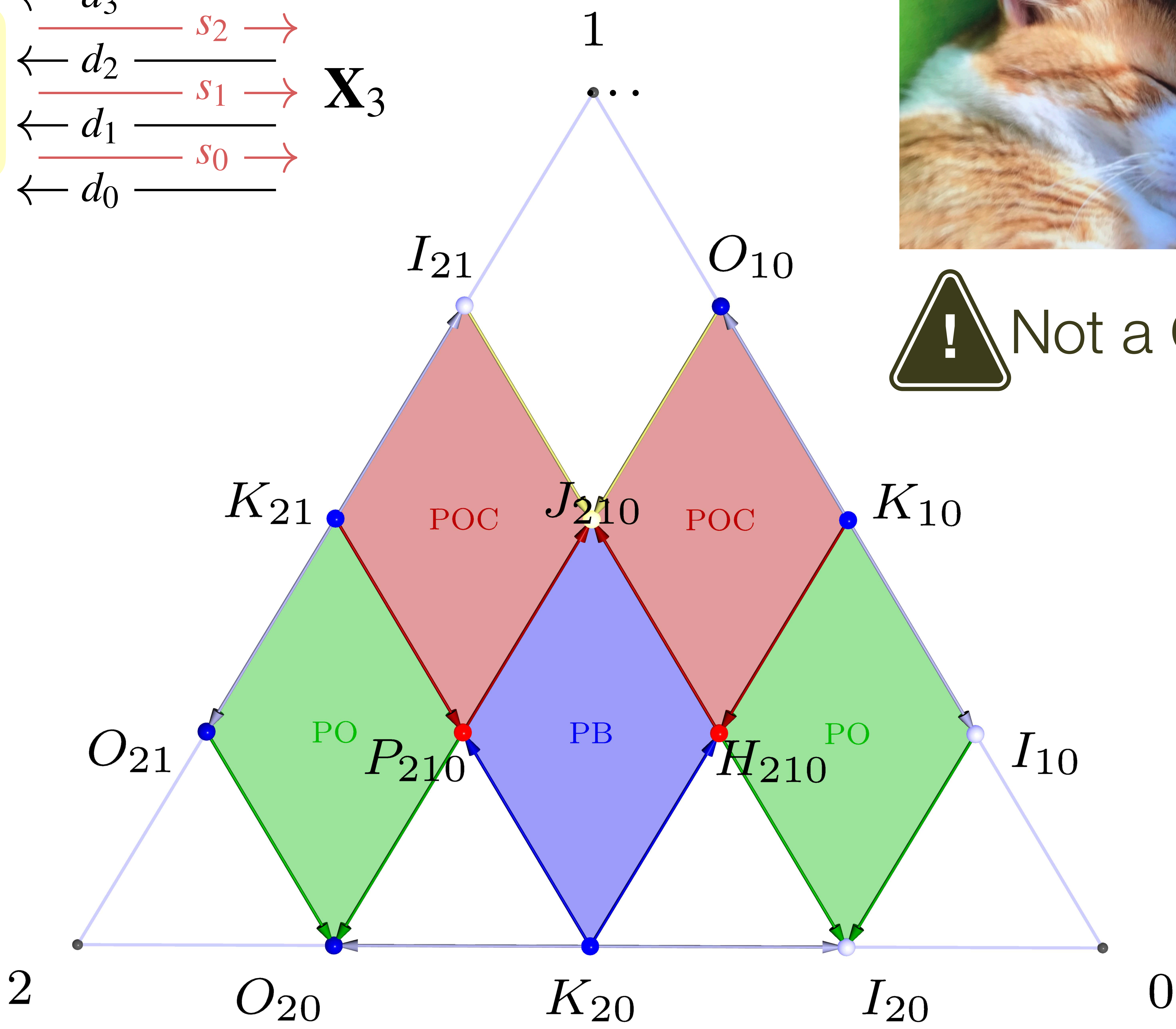


Construction of a suitable **decomposition space**

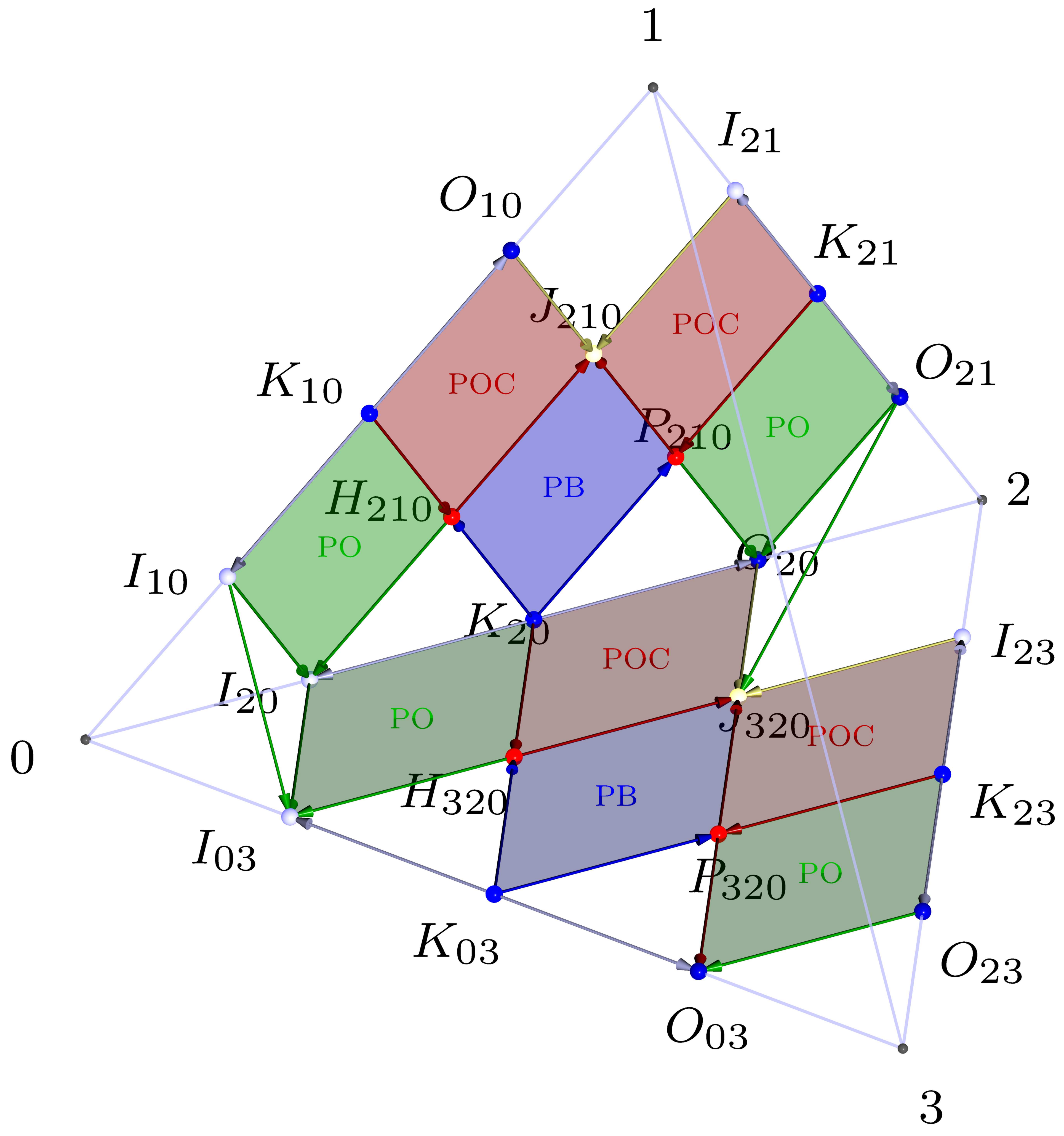


Not a **CAT**!

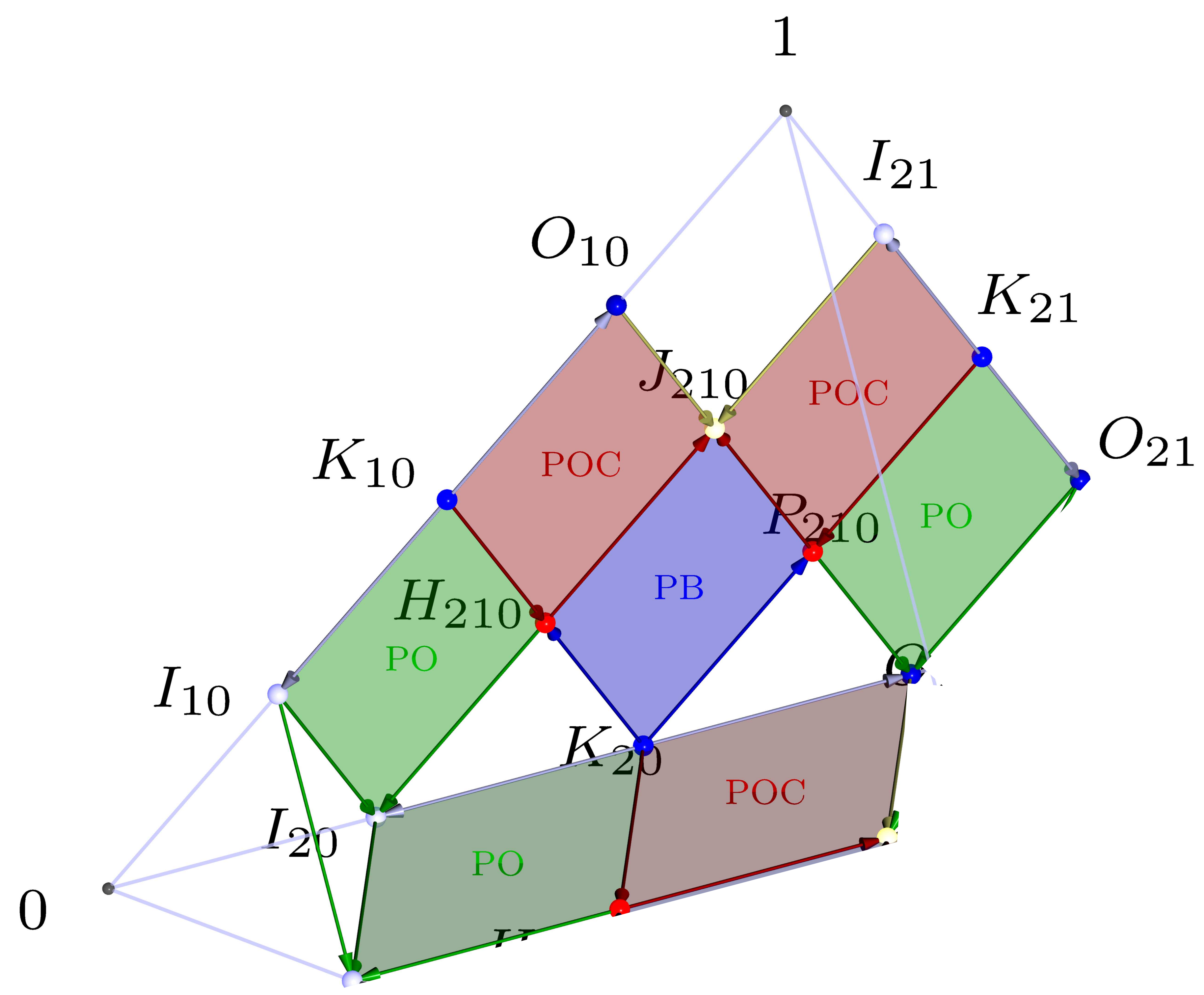




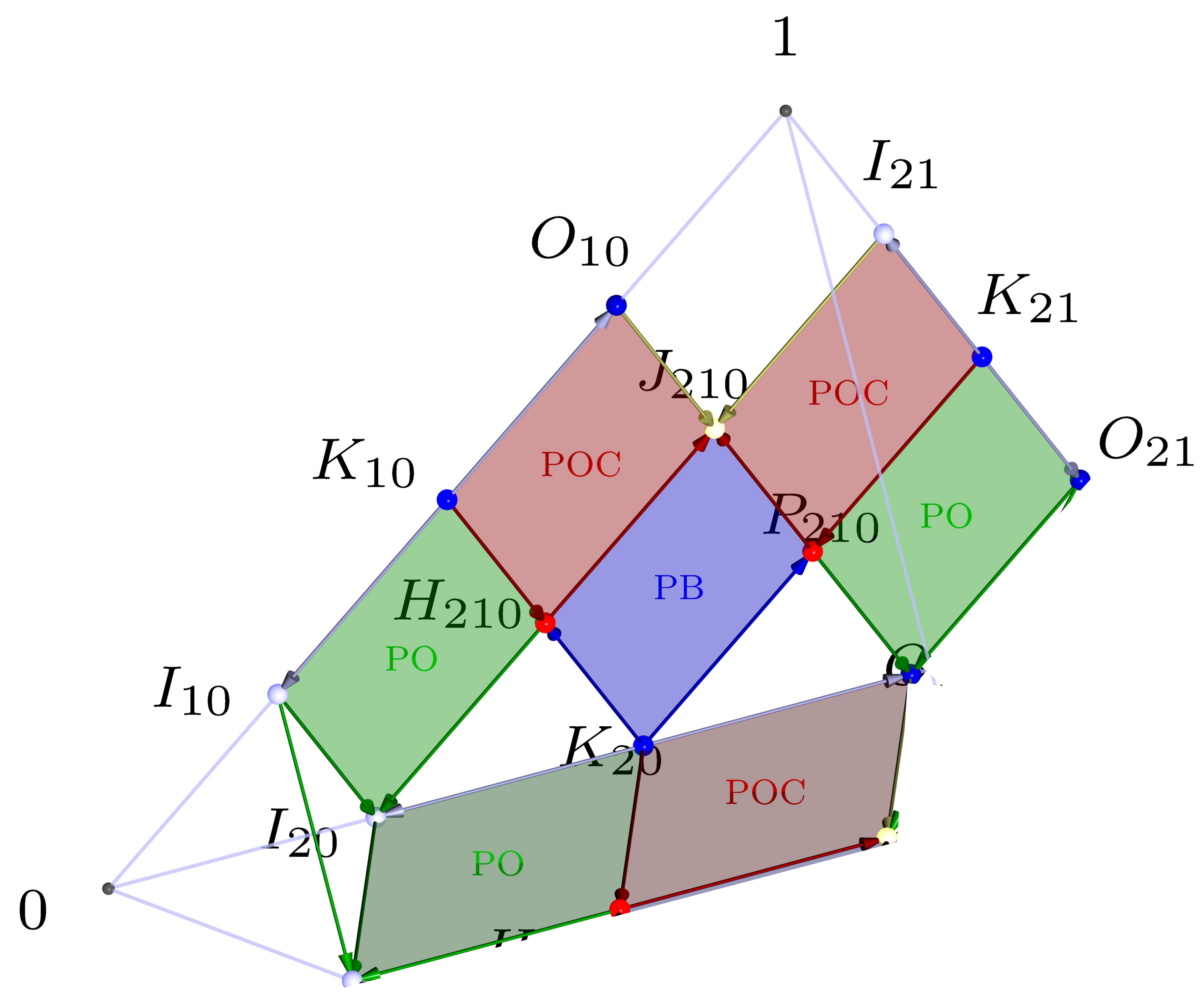
Construction of a suitable **decomposition space**

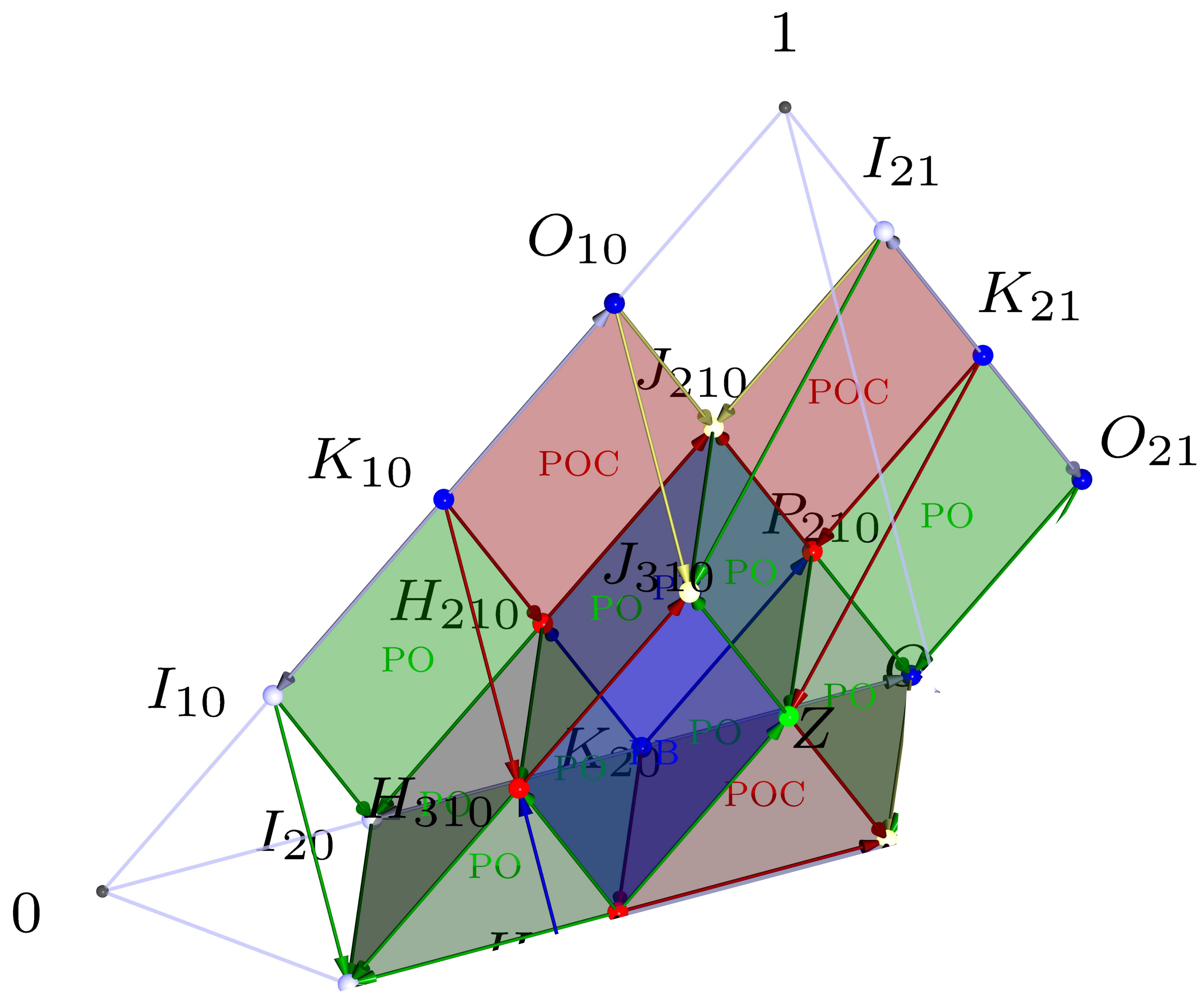


Construction of a suitable **decomposition space**

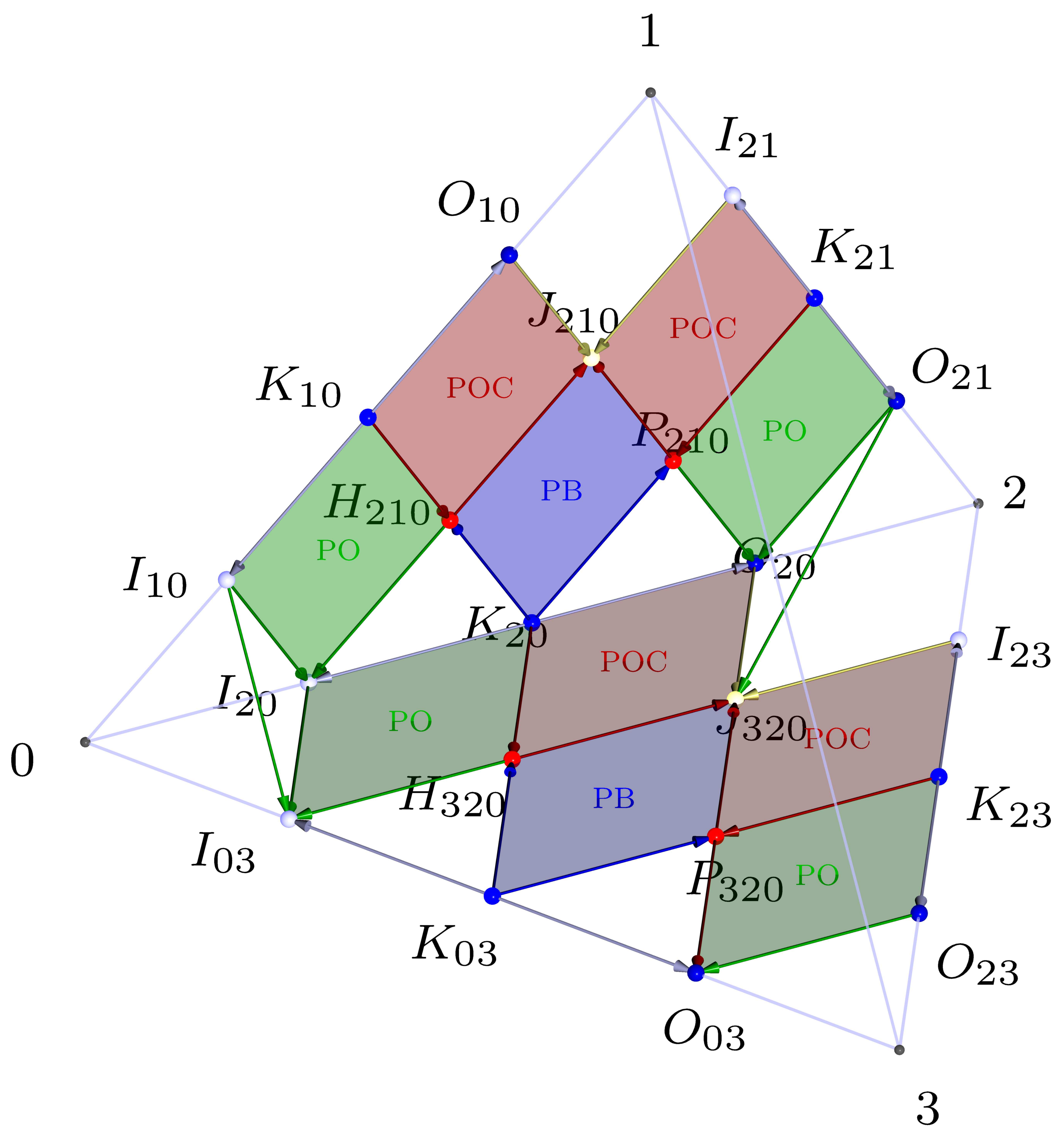


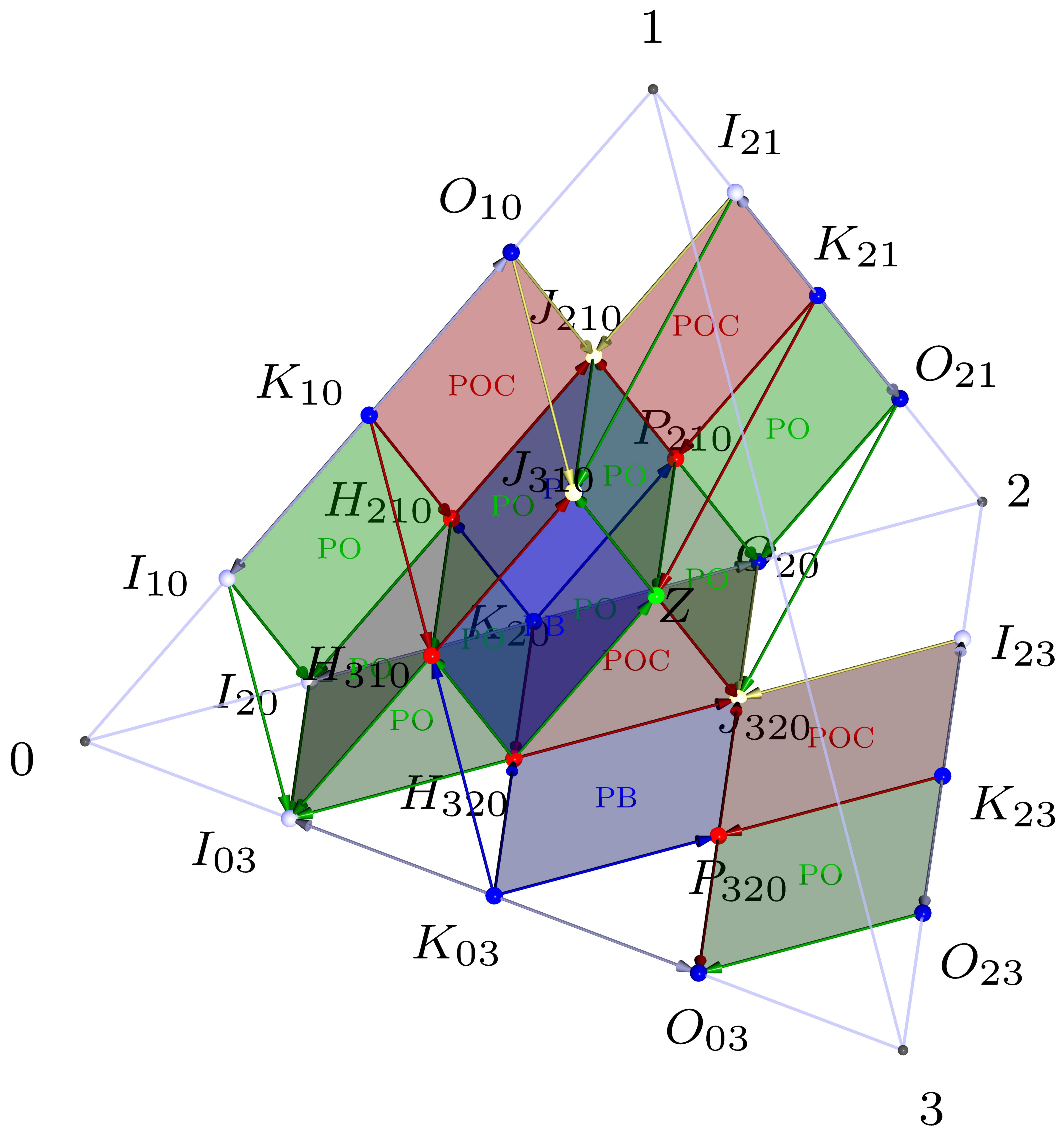
Construction of a suitable **decomposition space**



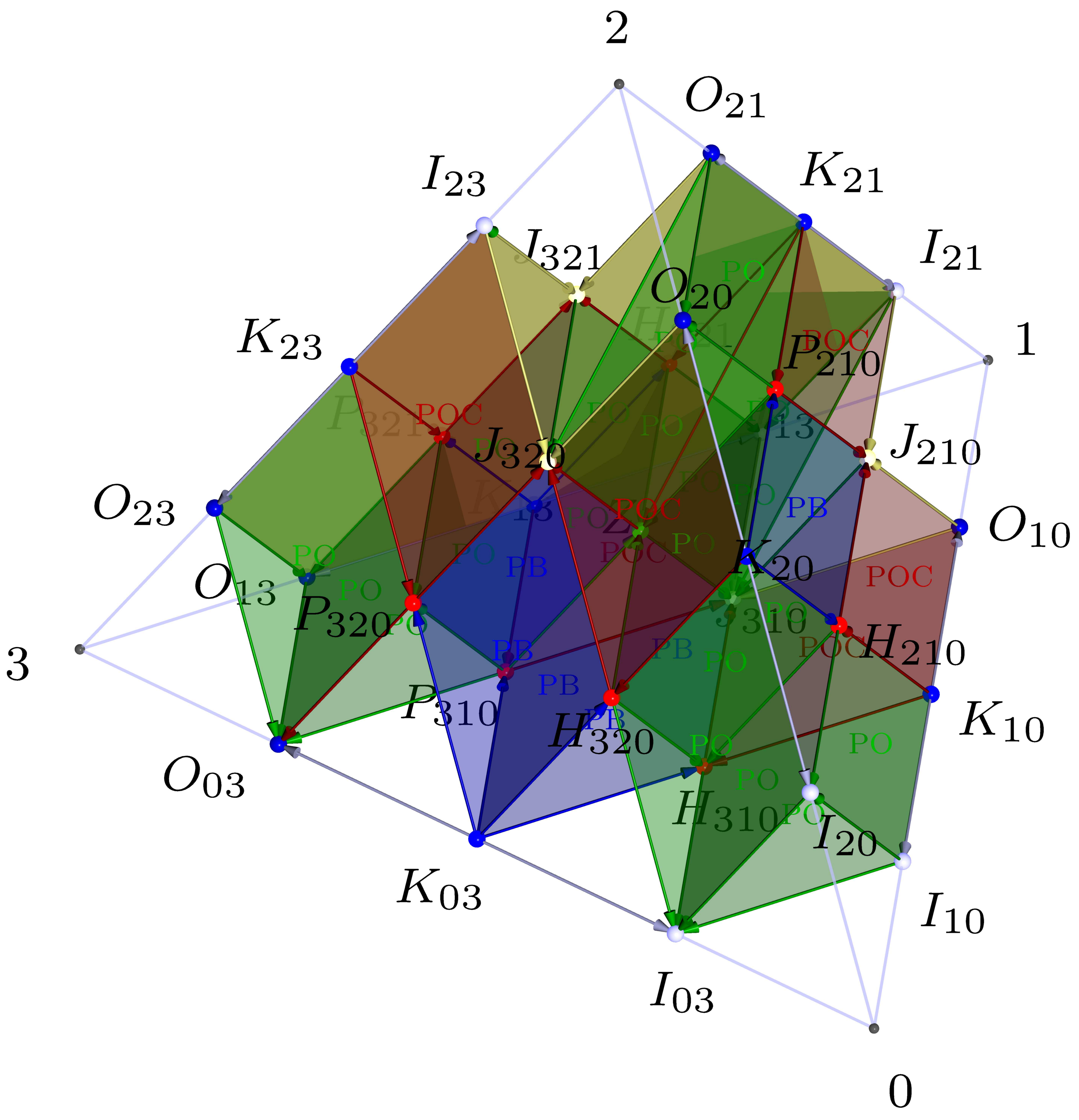


Construction of a suitable **decomposition space**

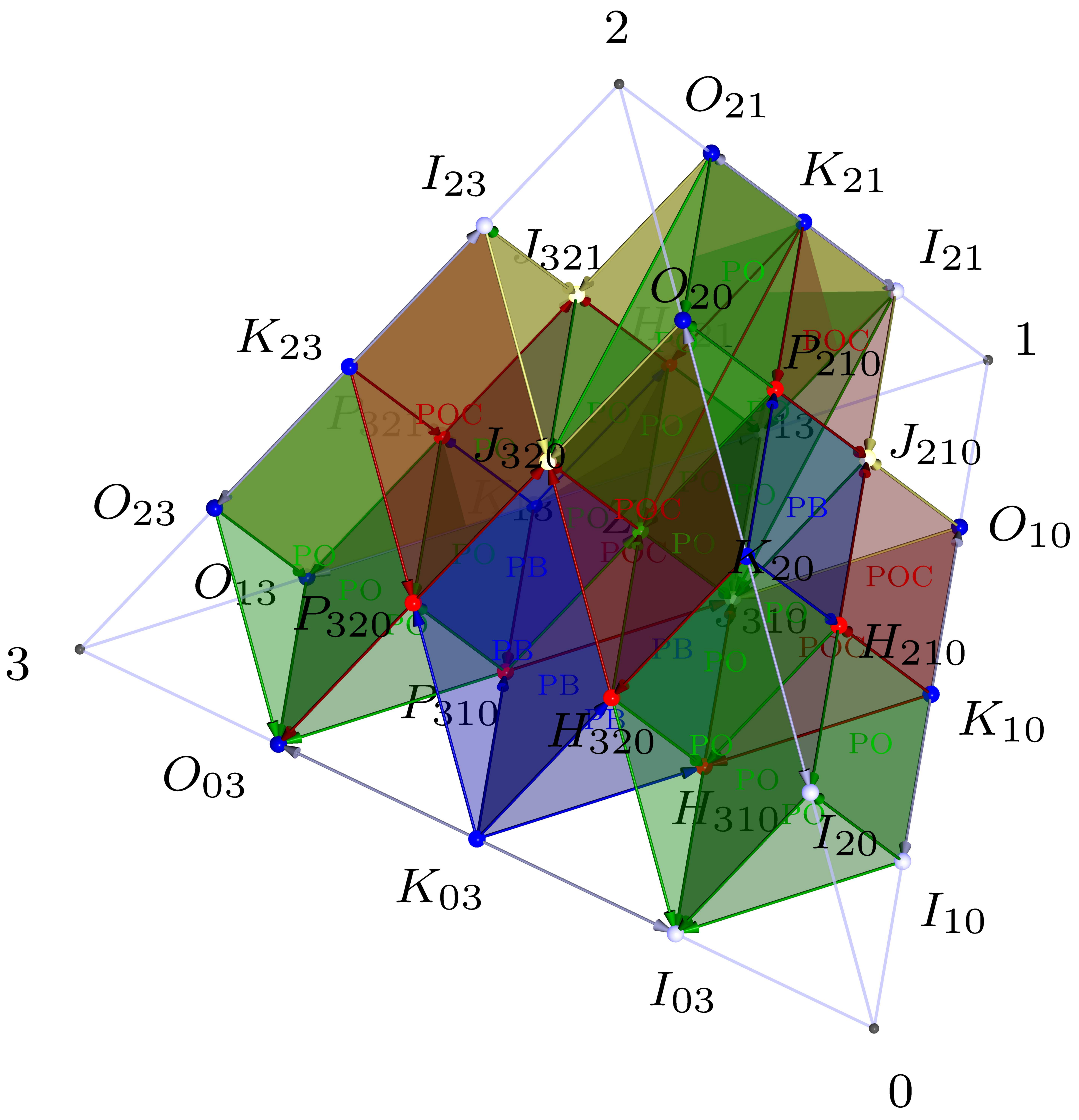


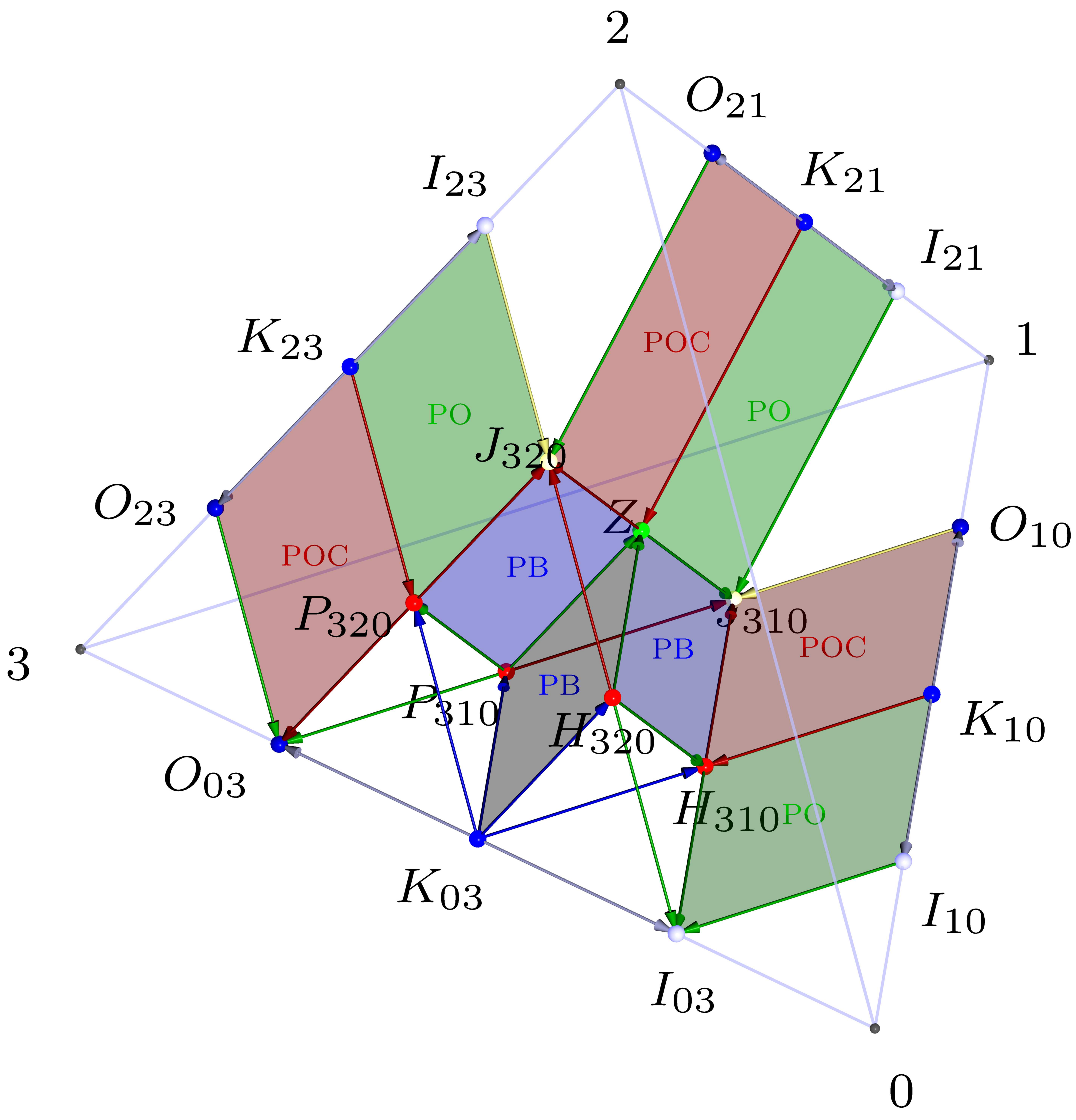


Construction of a suitable **decomposition space**

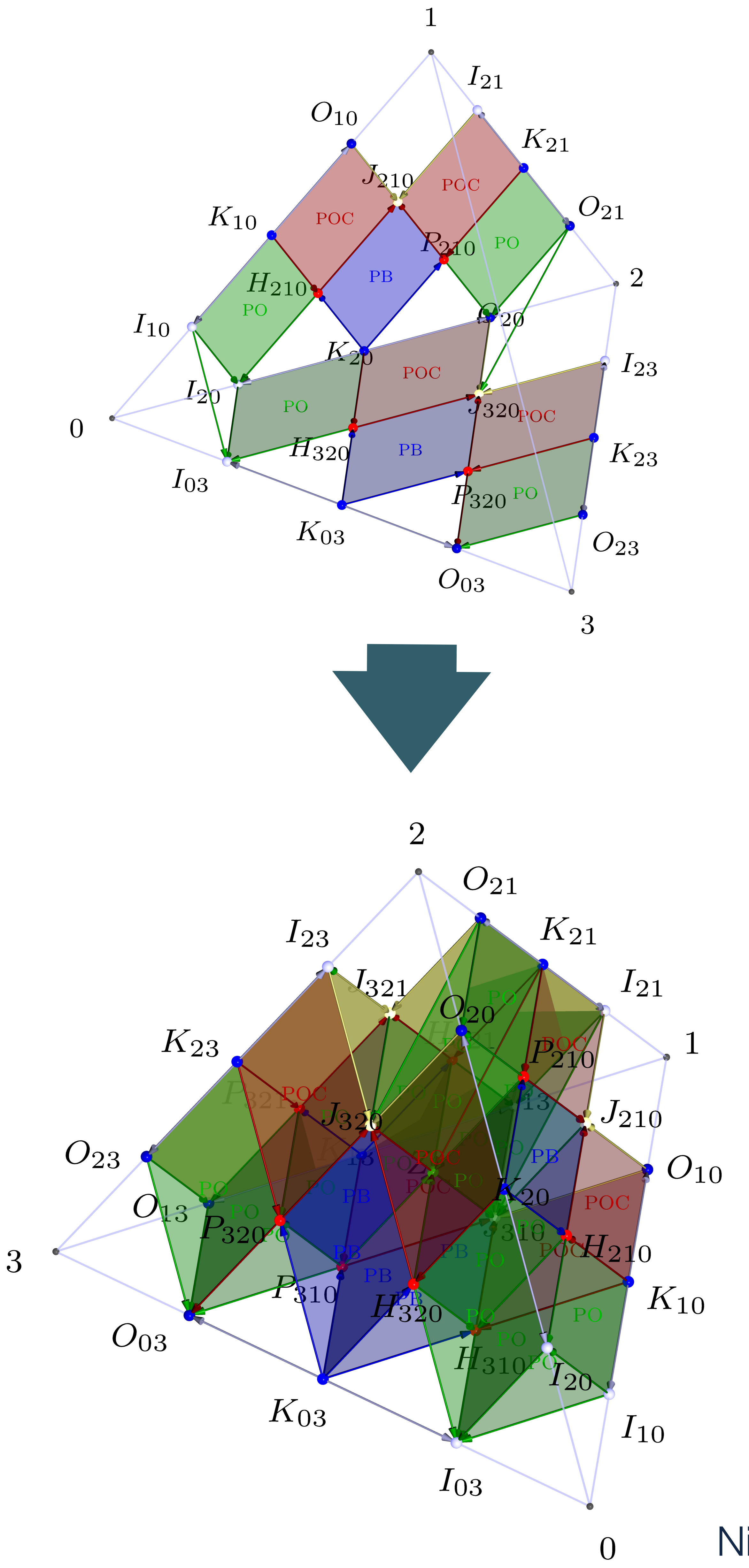


Construction of a suitable **decomposition space**



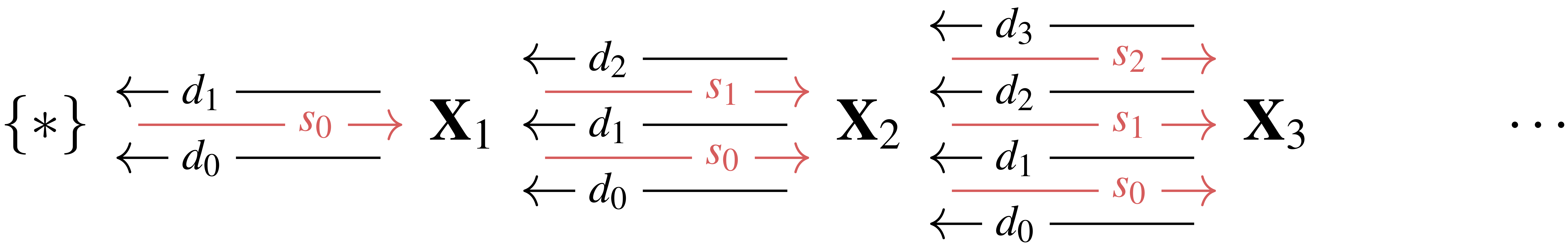


Construction of a suitable **decomposition space**



$$
\begin{array}{ccc}
X_3 & \xrightarrow{\;d_3\;} & X_2 \\
{\scriptstyle d_1}\downarrow & \lrcorner & \downarrow{\scriptstyle d_1} \\
X_2 & \xrightarrow[\;d_2\;]{} & X_1
\end{array}
$$

Construction of a suitable **decomposition space**



Theorem

$\mathbf{X}_\bullet$ is a decomposition space. This means that for all $0 < i < n$ the two squares

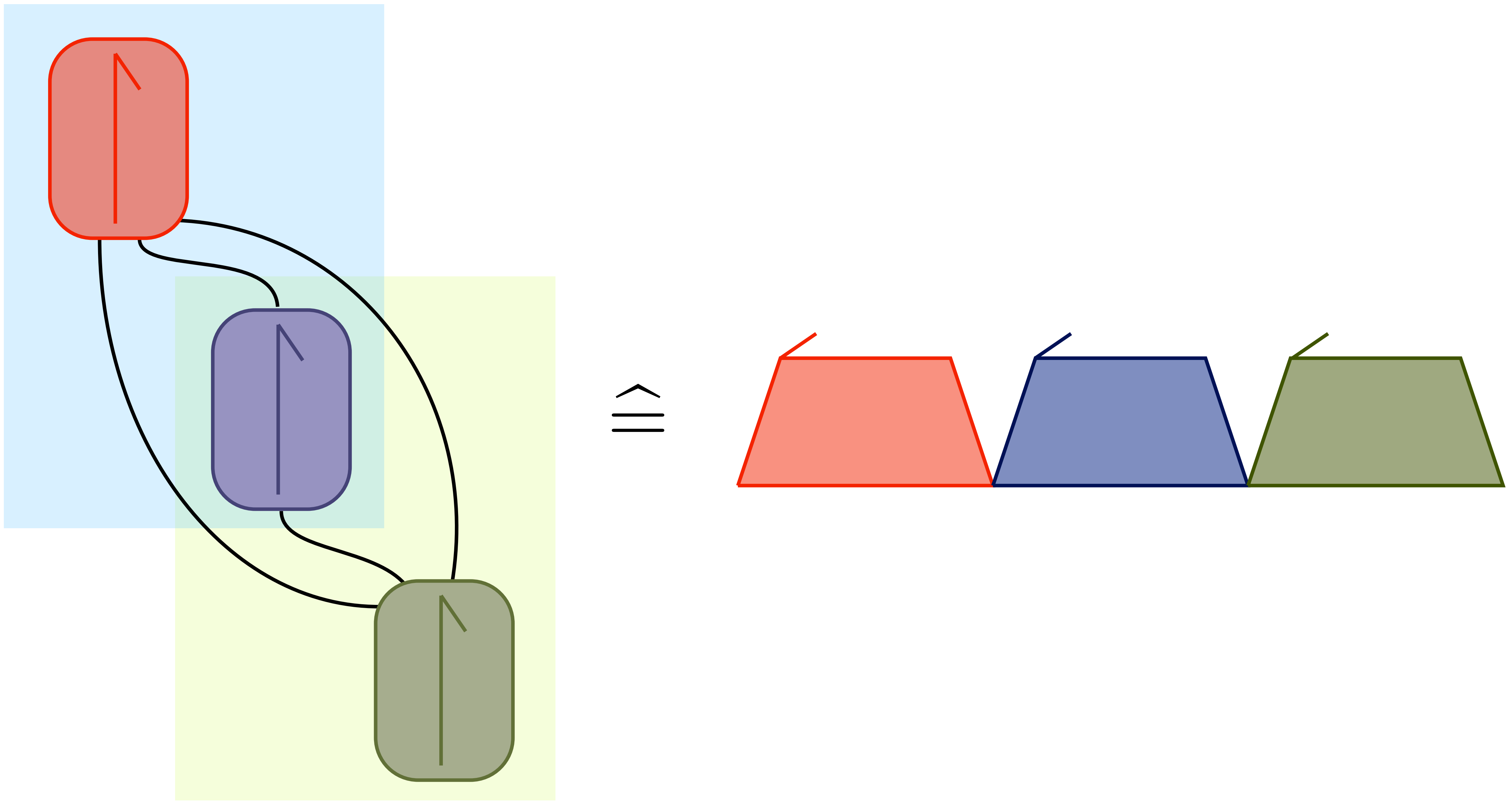
$$\begin{array}{ccc} \mathbf{X}_{n+1} & \xrightarrow{d_{n+1}} & \mathbf{X}_n \\ {\scriptstyle d_i}\downarrow & & \downarrow{\scriptstyle d_i} \\ \mathbf{X}_n & \xrightarrow[d_n]{} & \mathbf{X}_{n-1} \end{array} \qquad\qquad \begin{array}{ccc} \mathbf{X}_{n+1} & \xrightarrow{d_0} & \mathbf{X}_n \\ {\scriptstyle d_{i+1}}\downarrow & & \downarrow{\scriptstyle d_i} \\ \mathbf{X}_n & \xrightarrow[d_0]{} & \mathbf{X}_{n-1} \end{array}$$

are (homotopy) pullbacks.

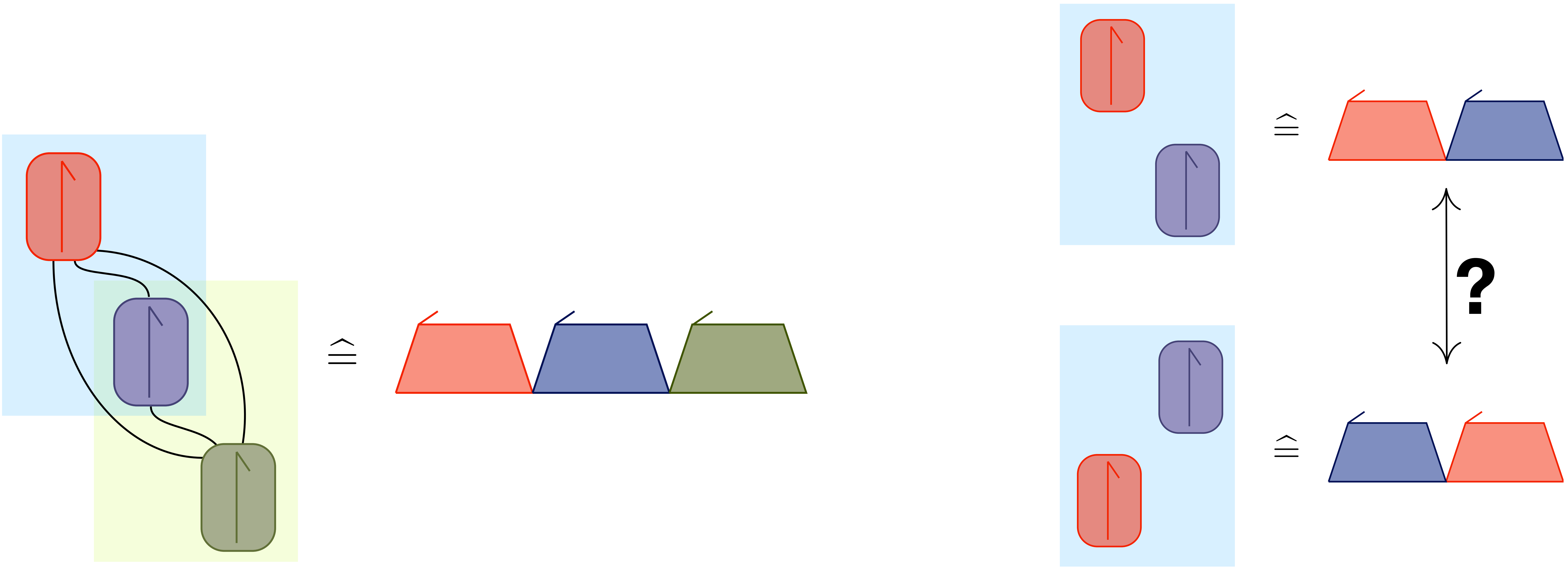
Plan of the talk

1. Discrete rewriting and diagram Hopf Algebras
2. Categorical rewriting theory
3. From rewriting to tracelets
4. Tracelet decomposition spaces
5. Tracelet Hopf algebras

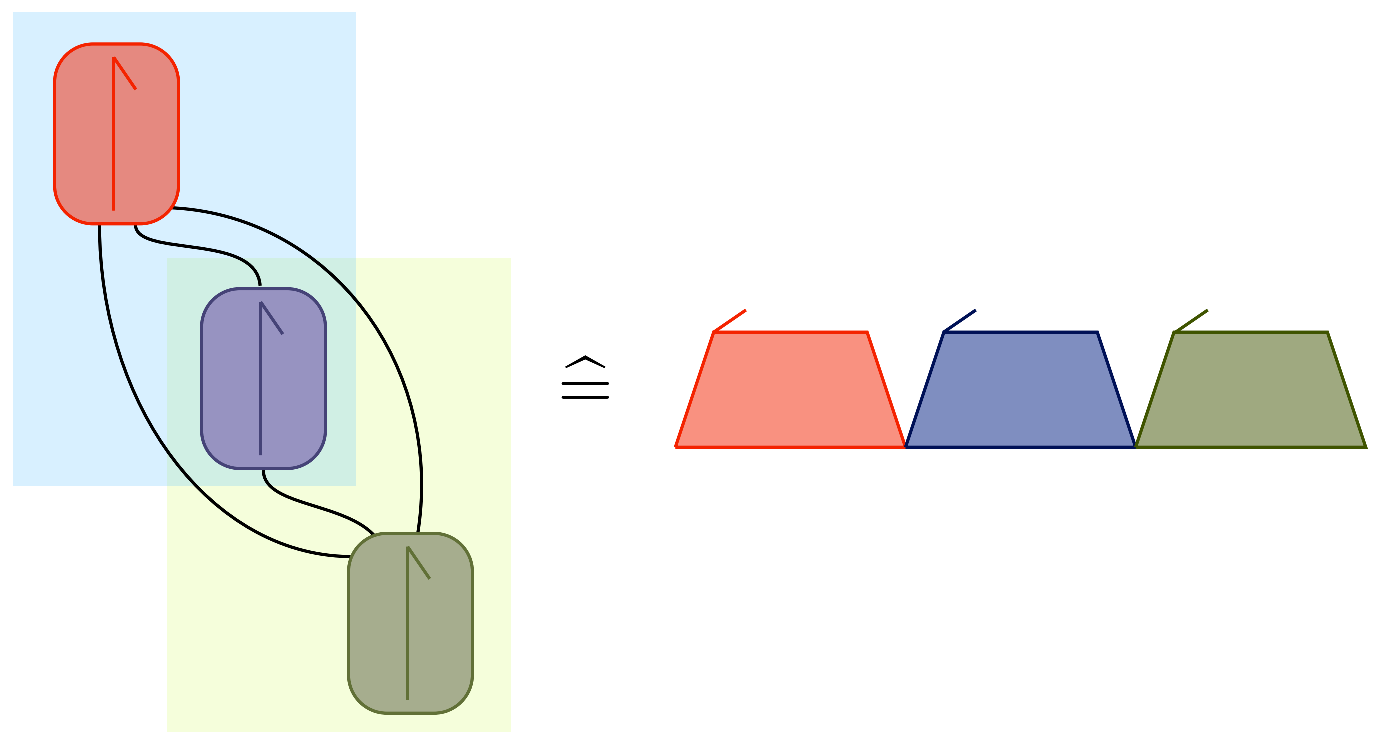
"Sequential" vs. "diagrammatic" interpretation of tracelets

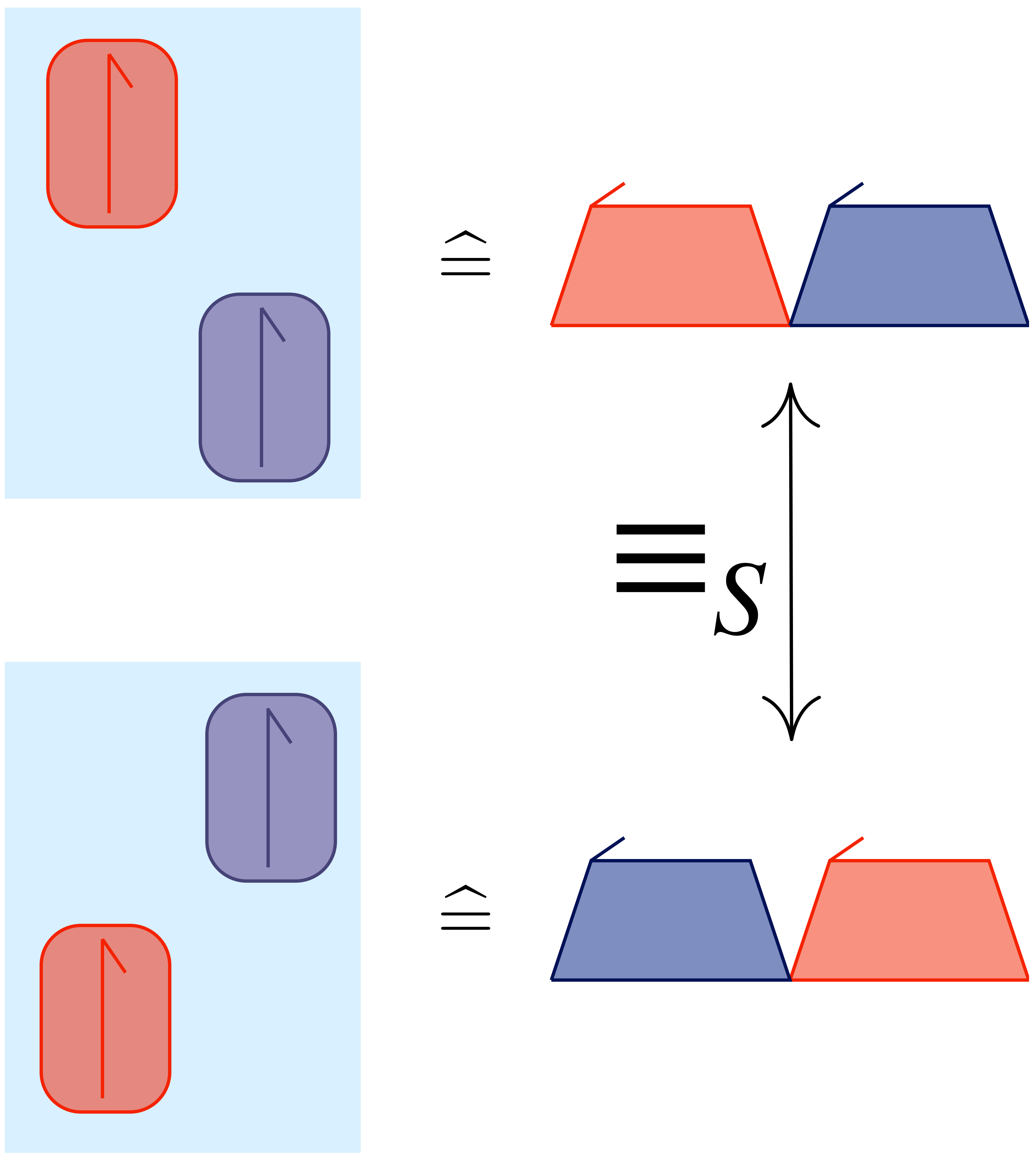


“Sequential” vs. “diagrammatic” interpretation of tracelets



"Sequential" vs. "diagrammatic" interpretation of tracelets



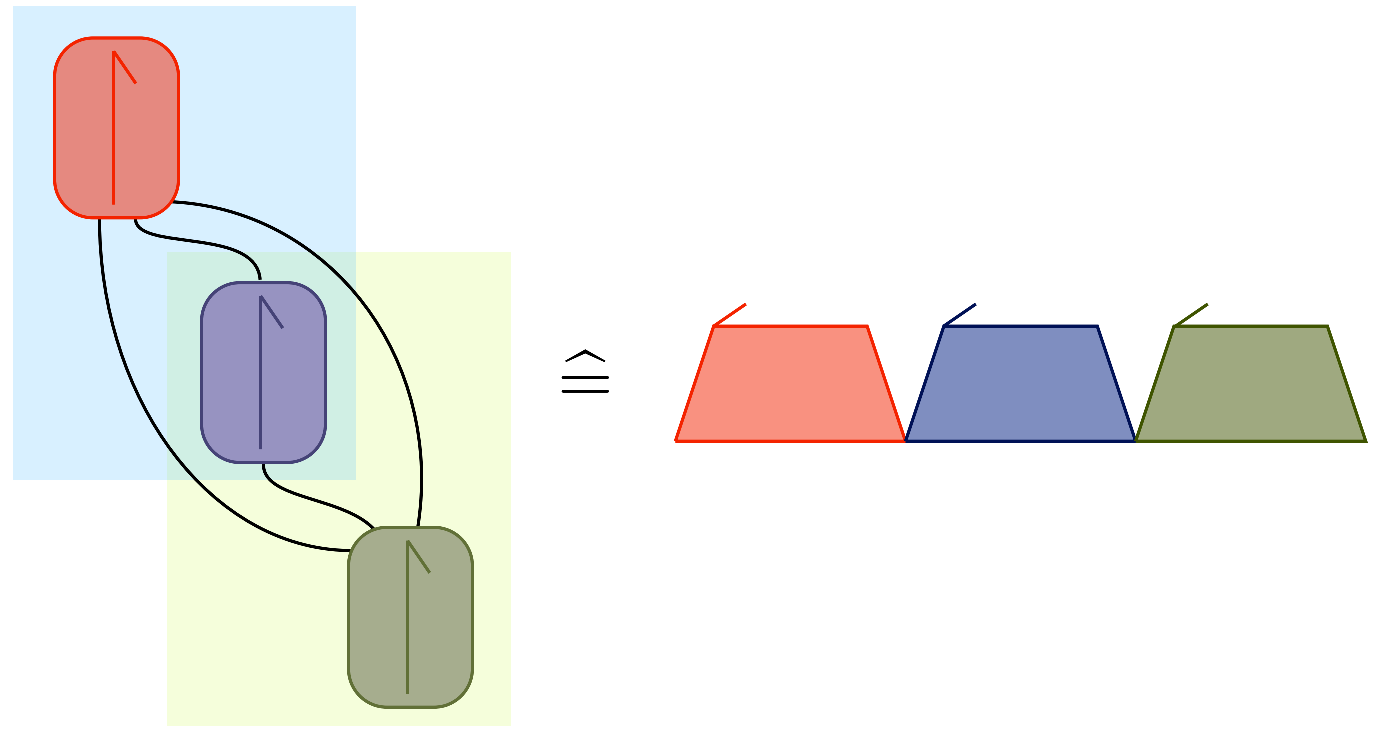


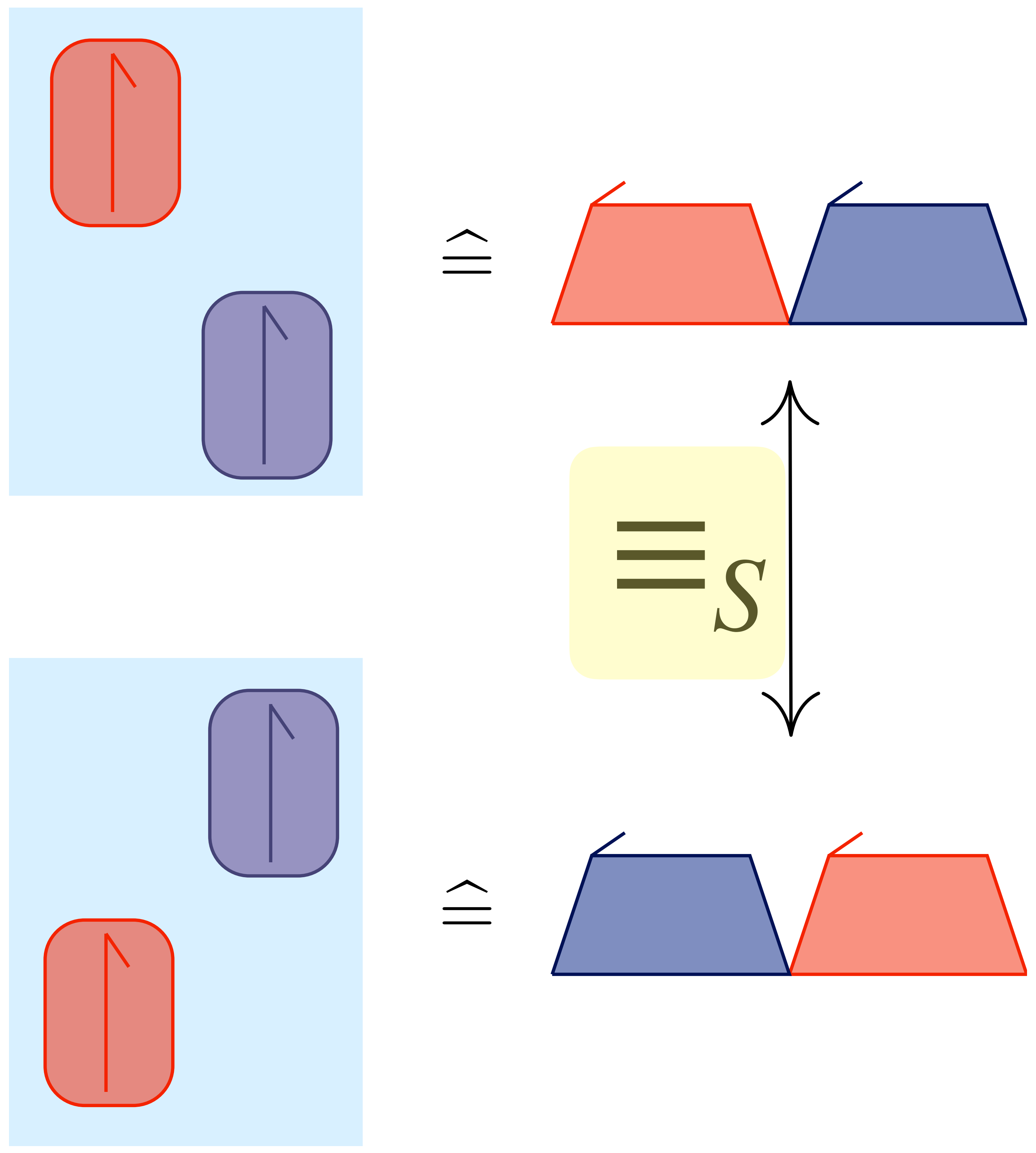
Normal form equivalence [B. 2019]

$$\equiv_N \;:=\; rst(\;\equiv_A \cup \equiv_S \cup \equiv_T\;)$$

- $\equiv_A$ — **abstraction equivalence** (= point-wise isos)
- $\equiv_S$ — **shift equivalence** (= "*sequential commutativity*")
- $\equiv_T$ — **equivalence up to trivial tracelets**, i.e., for any tracelet T, we define $T \;\equiv_T\; T^{\mu_\varnothing}\angle T_\varnothing \;\equiv_T\; T_\varnothing{}^{\mu_\varnothing}\angle T$

"Sequential" vs. "diagrammatic" interpretation of tracelets



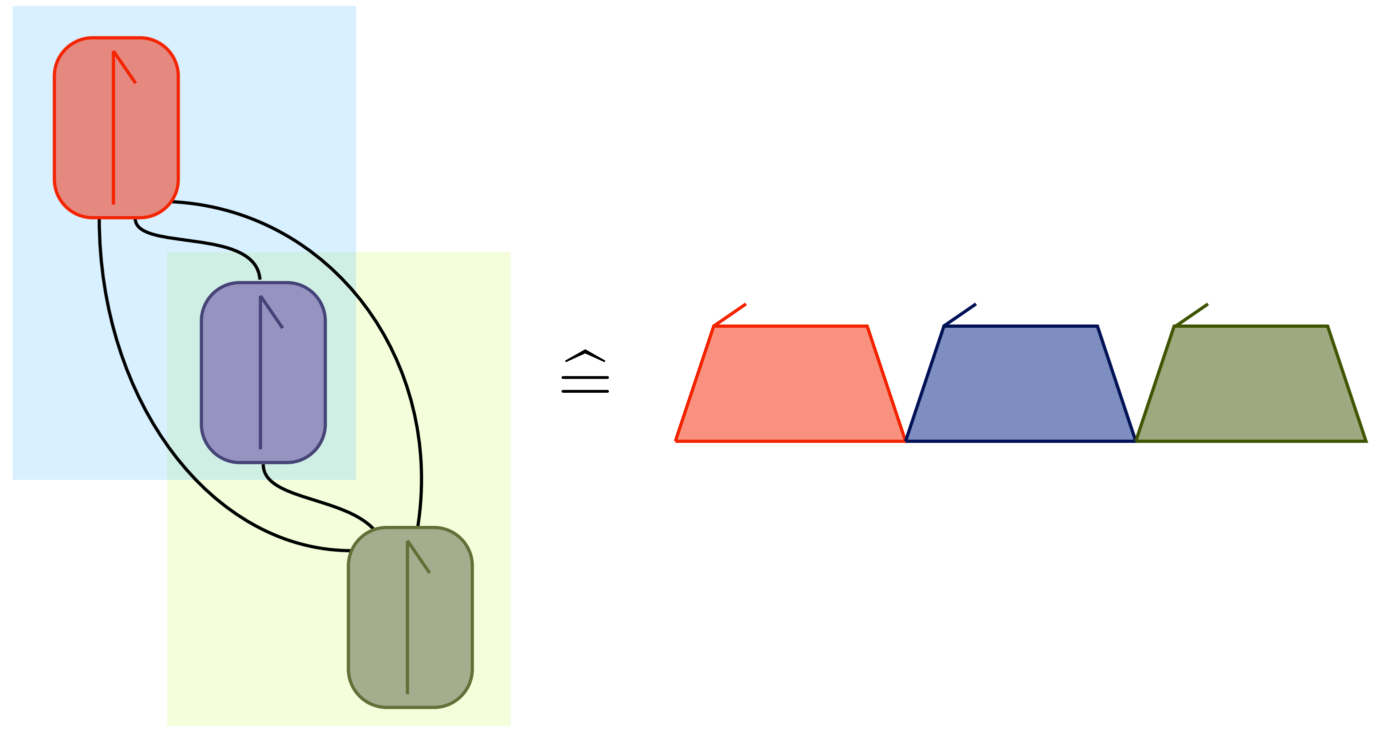


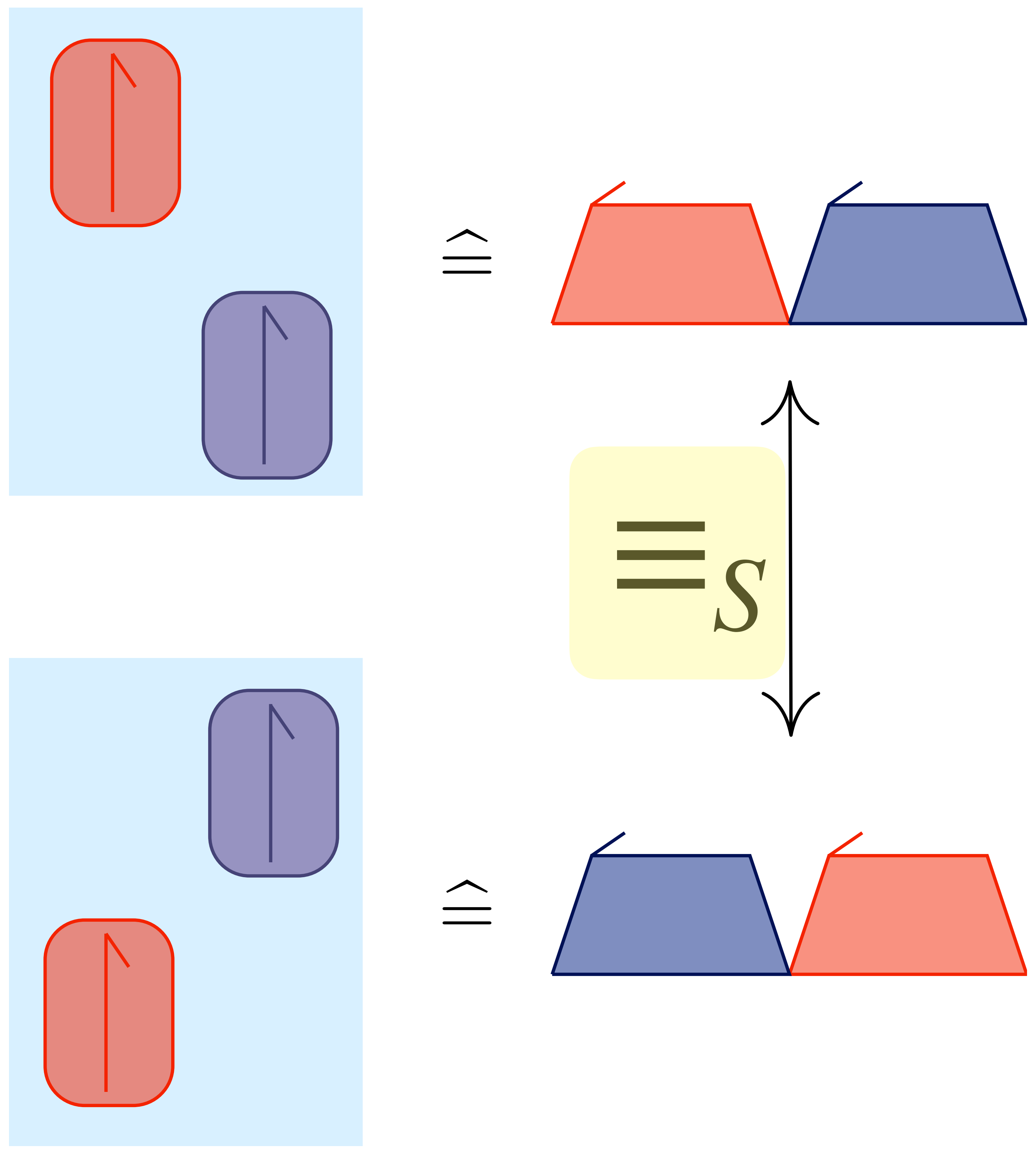
Normal form equivalence [B. 2019]

$$\equiv_N \;:= rst(\equiv_A \cup \equiv_S \cup \equiv_T)$$

- $\equiv_A$ — **abstraction equivalence** (= point-wise isos)
- $\equiv_S$ — **shift equivalence** (= "*sequential commutativity*")
- $\equiv_T$ — **equivalence up to trivial tracelets**, i.e., for any tracelet T, we define $T \;\equiv_T\; T^{\mu_\varnothing}\angle T_\varnothing \;\equiv_T\; T_\varnothing{}^{\mu_\varnothing}\angle T$

"Sequential" vs. "diagrammatic" interpretation of tracelets





Normal form equivalence [B. 2019]

$$\equiv_N \;:=\; rst(\,\equiv_A \cup \equiv_S \cup \equiv_T\,)$$

- $\equiv_A$ — **abstraction equivalence** (= point-wise isos)
- $\equiv_S$ — **shift equivalence** (= "*sequential commutativity*")
- $\equiv_T$ — **equivalence up to trivial tracelets**, i.e., for any tracelet T, we define $T \;\equiv_T\; T^{\mu_\varnothing}\angle T_\varnothing \;\equiv_T\; T_\varnothing{}^{\mu_\varnothing}\angle T$

$$T_A \uplus T_B := [T_A{}^{\mu_\varnothing}\angle T_B]_{\equiv_N} = [T_B{}^{\mu_\varnothing}\angle T_A]_{\equiv_N}$$

Tracelet **algebra structure** [B. & Kock, 2021]

Definition: Primitive tracelets

Let $\mathcal{T}_N := \mathcal{T}\diagup_{\equiv_N}$ denote the set of $\equiv_N$-equivalence classes of tracelets. Then $\mathfrak{Prim}(\mathcal{T}_N)$, the set of primitive tracelets, is defined as

$$\mathfrak{Prim}(\mathcal{T}_N) := \{[T]_{\equiv_N} | T \neq T_{\varnothing} \wedge \nexists T_A, T_B \neq T_{\varnothing} : T \equiv_N T_A \uplus T_B\} \,.$$

Tracelet **algebra structure** [B. & Kock, 2021]

Definition: Primitive tracelets

Let $\mathcal{T}_N := \mathcal{T}/_{\equiv_N}$ denote the set of $\equiv_N$-equivalence classes of tracelets. Then $\mathfrak{Prim}(\mathcal{T}_N)$, the set of primitive tracelets, is defined as

$$\mathfrak{Prim}(\mathcal{T}_N) := \{[T]_{\equiv_N} | T \neq T_{\varnothing} \wedge \nexists T_A, T_B \neq T_{\varnothing} : T \equiv_N T_A \uplus T_B\} .$$

Definition: Tracelet $\mathbb{K}$-vector space $\widehat{\mathcal{T}}$

Let $\widehat{\mathcal{T}}$ be the $\mathbb{K}$-vector space spanned by a basis indexed by $\equiv_N$-equivalence classes, in the sense that there exists an isomorphism $\delta : \mathcal{T}_N \xrightarrow{\sim} \mathsf{basis}(\widehat{\mathcal{T}})$. We will use the notation $\widehat{T} := \delta(T)$ for the basis vector associated to some class $T \in \mathcal{T}_N$. We denote by $\mathsf{Prim}(\widehat{\mathcal{T}}) \subset \widehat{\mathcal{T}}$ the sub-vector space of $\widehat{\mathcal{T}}$ spanned by basis vectors indexed by primitive tracelets.

Tracelet **algebra structure**

[B. & Kock, 2021]

Definition: Primitive tracelets

Let $\mathcal{T}_N := \mathcal{T}/_{\equiv_N}$ denote the set of $\equiv_N$-equivalence classes of tracelets. Then $\mathfrak{Prim}(\mathcal{T}_N)$, the set of primitive tracelets, is defined as

$$\mathfrak{Prim}(\mathcal{T}_N) := \{[T]_{\equiv_N} | T \neq T_\varnothing \wedge \nexists T_A, T_B \neq T_\varnothing : T \equiv_N T_A \uplus T_B\} .$$

Definition: Tracelet $\mathbb{K}$-vector space $\widehat{\mathcal{T}}$

Let $\widehat{\mathcal{T}}$ be the $\mathbb{K}$-vector space spanned by a basis indexed by $\equiv_N$-equivalence classes, in the sense that there exists an isomorphism $\delta : \mathcal{T}_N \xrightarrow{\cong} \text{basis}(\widehat{\mathcal{T}})$. We will use the notation $\widehat{T} := \delta(T)$ for the basis vector associated to some class $T \in \mathcal{T}_N$. We denote by $\text{Prim}(\widehat{\mathcal{T}}) \subset \widehat{\mathcal{T}}$ the sub-vector space of $\widehat{\mathcal{T}}$ spanned by basis vectors indexed by primitive tracelets.

Definition: Tracelet algebra product and unit

Let $\otimes \equiv \otimes_{\mathbb{K}}$ be the tensor product operation on the $\mathbb{K}$-vector space $\widehat{\mathcal{T}}$. Then the multiplication map μ and the unit map $\eta : \mathbb{K} \to \widehat{\mathcal{T}}$ are defined via their action on basis vectors of $\widehat{\mathcal{T}}$ as follows:

$$\mu : \widehat{\mathcal{T}} \otimes \widehat{\mathcal{T}} \to \widehat{\mathcal{T}} : \widehat{T} \otimes \widehat{T'} \mapsto \widehat{T} \diamond \widehat{T'} , \qquad \widehat{T} \diamond \widehat{T'} := \sum_{\mu \in MT_T(T')} \delta\left([T \,{}^{\mu}\!\angle\, T']_{\equiv_N}\right)$$

$$\eta : \mathbb{K} \to \widehat{\mathcal{T}} : k \mapsto k \cdot \widehat{T}_\varnothing .$$

Both definitions are suitably extended by (bi-)linearity to generic (pairs of) elements of $\widehat{\mathcal{T}}$.

Tracelet **coalgebra**, **bialgebra** and **filtration** [B. & Kock, 2021]

Definition: Tracelet coproduct and counit

Fixing the notational convention $\uplus_{i\in\emptyset}T_i := T_{\varnothing}$ for later convenience, let $T \equiv_N \uplus_{i\in I} T_i$ be the tracelet normal form for a given tracelet $T \in \mathfrak{T}$ (where $T_i \in \mathfrak{Prim}(\mathfrak{T}_N)$ for all $i \in I$ if $T \neq T_{\varnothing}$). Then the tracelet coproduct Δ and tracelet counit ε are defined via their action on basis vectors $\widehat{T} = \delta(T)$ of $\widehat{\mathfrak{T}}$ as

$$\Delta : \widehat{\mathfrak{T}} \to \widehat{\mathfrak{T}} \otimes \widehat{\mathfrak{T}} : \widehat{T} \mapsto \Delta(\widehat{T}) := \sum_{X \subset I} \delta\left(\left[\biguplus_{x\in X} T_x\right]_{\equiv_N}\right) \otimes \delta\left(\left[\biguplus_{y\in I\setminus X} T_y\right]_{\equiv_N}\right)$$

and $\varepsilon : \widehat{\mathfrak{T}} \to \mathbb{K} : \widehat{T} \mapsto \mathrm{coeff}_{\widehat{T}_{\varnothing}}(\widehat{T})$. Both definitions are extended by linearity to generic elements of $\widehat{\mathfrak{T}}$.

Tracelet **coalgebra**, **bialgebra** and **filtration** [B. & Kock, 2021]

Definition: Tracelet coproduct and counit

Fixing the notational convention $\uplus_{i\in\emptyset} T_i := T_{\varnothing}$ for later convenience, let $T \equiv_N \uplus_{i\in I} T_i$ be the tracelet normal form for a given tracelet $T \in \mathfrak{T}$ (where $T_i \in \mathfrak{Prim}(\mathfrak{T}_N)$ for all $i \in I$ if $T \neq T_{\varnothing}$). Then the tracelet coproduct Δ and tracelet counit ε are defined via their action on basis vectors $\widehat{T} = \delta(T)$ of $\widehat{\mathfrak{T}}$ as

$$\Delta : \widehat{\mathfrak{T}} \to \widehat{\mathfrak{T}} \otimes \widehat{\mathfrak{T}} : \widehat{T} \mapsto \Delta(\widehat{T}) := \sum_{X \subset I} \delta\left(\left[\biguplus_{x\in X} T_x\right]_{\equiv_N}\right) \otimes \delta\left(\left[\biguplus_{y\in I\setminus X} T_y\right]_{\equiv_N}\right)$$

and $\varepsilon : \widehat{\mathfrak{T}} \to \mathbb{K} : \widehat{T} \mapsto \mathsf{coeff}_{\widehat{T}_{\varnothing}}(\widehat{T})$. Both definitions are extended by linearity to generic elements of $\widehat{\mathfrak{T}}$.

Theorem:

The data $(\widehat{\mathfrak{T}}, \mu, \eta, \Delta, \varepsilon)$ defines a bialgebra.

Tracelet **coalgebra**, **bialgebra** and **filtration**

[B. & Kock, 2021]

Definition: Tracelet coproduct and counit

Fixing the notational convention $\uplus_{i\in\emptyset} T_i := T_\varnothing$ for later convenience, let $T \equiv_N \uplus_{i\in I} T_i$ be the tracelet normal form for a given tracelet $T \in \mathfrak{T}$ (where $T_i \in \mathfrak{Prim}(\mathfrak{T}_N)$ for all $i \in I$ if $T \neq T_\varnothing$). Then the tracelet coproduct Δ and tracelet counit ε are defined via their action on basis vectors $\widehat{T} = \delta(T)$ of $\widehat{\mathfrak{T}}$ as

$$\Delta : \widehat{\mathfrak{T}} \to \widehat{\mathfrak{T}} \otimes \widehat{\mathfrak{T}} : \widehat{T} \mapsto \Delta(\widehat{T}) := \sum_{X \subset I} \delta\left(\left[\biguplus_{x\in X} T_x\right]_{\equiv_N}\right) \otimes \delta\left(\left[\biguplus_{y\in I\setminus X} T_y\right]_{\equiv_N}\right)$$

and $\varepsilon : \widehat{\mathfrak{T}} \to \mathbb{K} : \widehat{T} \mapsto \mathrm{coeff}_{\widehat{T}_\varnothing}(\widehat{T})$. Both definitions are extended by linearity to generic elements of $\widehat{\mathfrak{T}}$.

Theorem:

The data $(\widehat{\mathfrak{T}}, \mu, \eta, \Delta, \varepsilon)$ defines a bialgebra.

Theorem:

The tracelet bialgebra $(\widehat{\mathfrak{T}}, \mu, \eta, \Delta, \varepsilon)$ is connected and filtered, with connected component $\widehat{\mathfrak{T}}^{(0)} := \mathrm{span}_{\mathbb{K}}\{\widehat{T}_\varnothing\}$, and with the higher components of the filtration given by the subspaces

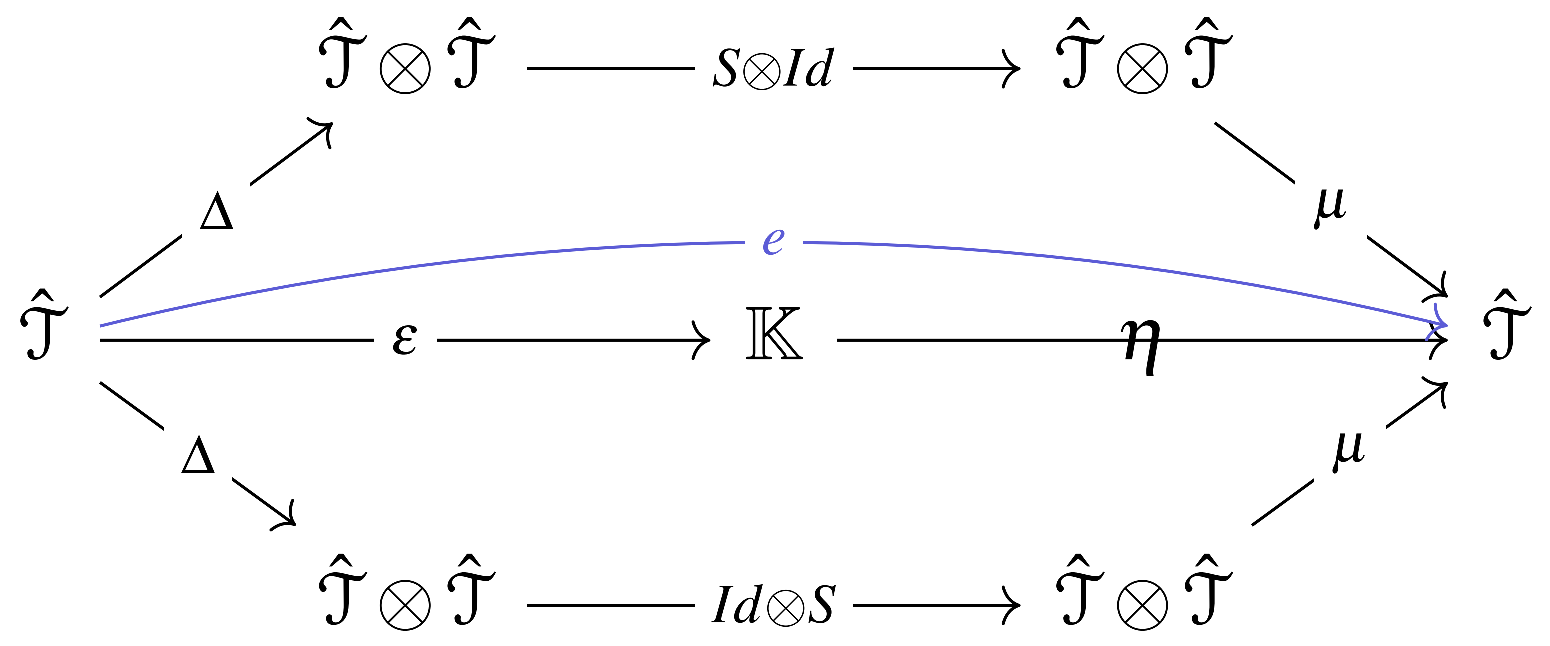
$$\forall n > 0 : \quad \widehat{\mathfrak{T}}^{(n)} := span_{\mathbb{K}} \left\{ \widehat{T}_1 \uplus \ldots \uplus \widehat{T}_n \,\middle|\, \widehat{T}_1, \ldots, \widehat{T}_n \in Prim(\widehat{\mathfrak{T}}) \right\},$$

where in a slight abuse of notations $\widehat{T}_1 \uplus \ldots \uplus \widehat{T}_n := \delta(T_1 \uplus \ldots \uplus T_n)$.

Tracelet **Hopf algebra structure** [B. & Kock, 2021]

Theorem

The tracelet bialgebra $(\widehat{\mathfrak{T}}, \mu, \eta, \Delta, \varepsilon)$ admits the structure of a Hopf algebra, where the antipode S, which is to say the endomorphism of $\widehat{\mathfrak{T}}$ that makes the diagram below commute,



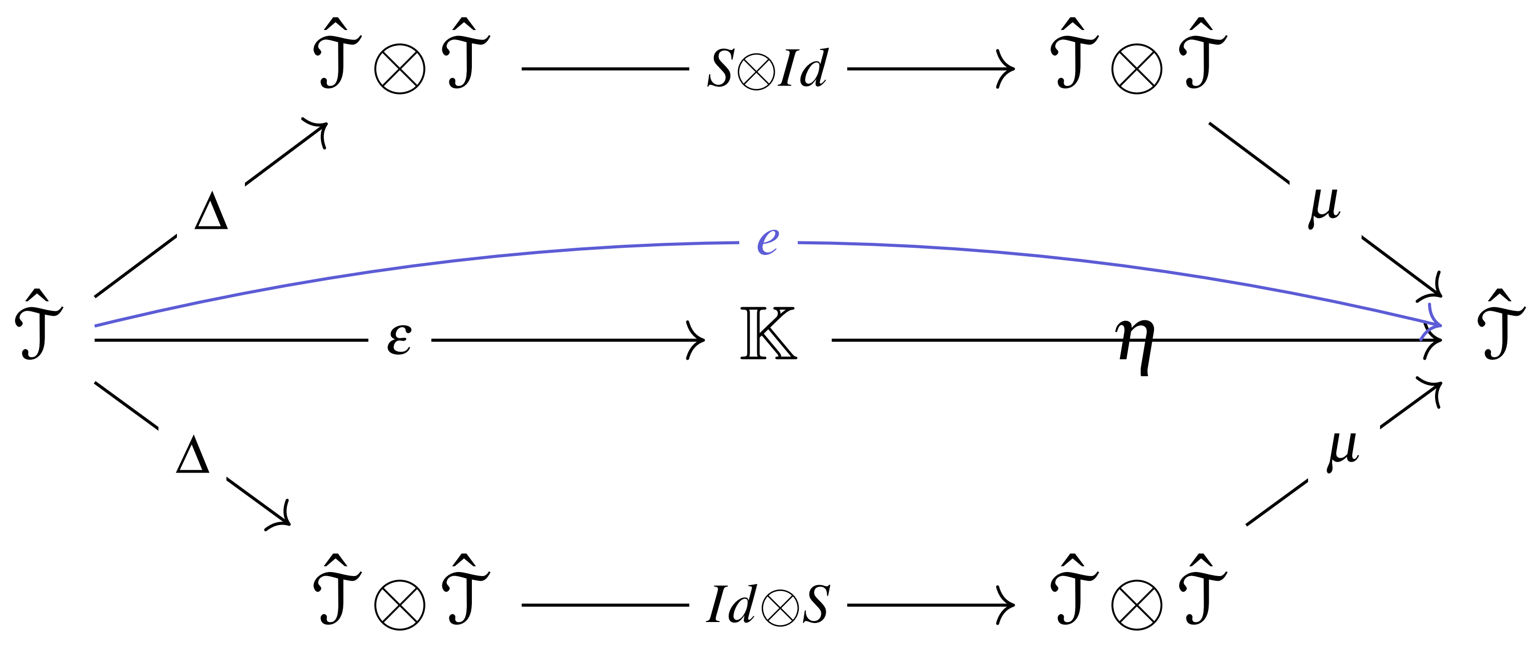
is given by $\mathsf{S} := \mathsf{Id}^{\star^{-1}}$. The latter denotes the inverse of the identity morphism $\mathsf{Id} : \widehat{\mathfrak{T}} \to \widehat{\mathfrak{T}}$ under the ***convolution product*** $\star$ of linear endomorphisms on $\widehat{\mathfrak{T}}$. More concretely, letting $\mathsf{e} := \eta \circ \varepsilon$ denote the unit for the convolution product $\star$,

$$S(\widehat{T}) = Id^{\star^{-1}}(\widehat{T}) = (e - (e - Id))^{\star^{-1}} = e(\widehat{T}) + \sum_{k\geq 1}(e - Id)^{\star\, k}(\widehat{T}).$$

Tracelet **Hopf algebra structure** [B. & Kock, 2021]

Theorem

The tracelet bialgebra $(\widehat{\mathcal{T}}, \mu, \eta, \Delta, \varepsilon)$ admits the structure of a Hopf algebra, where the antipode S, which is to say the endomorphism of $\widehat{\mathcal{T}}$ that makes the diagram below commute,



is given by $\mathsf{S} := \mathsf{Id}^{\star^{-1}}$. The latter denotes the inverse of the identity morphism $\mathsf{Id} : \widehat{\mathcal{T}} \to \widehat{\mathcal{T}}$ under the ***convolution product*** $\star$ of linear endomorphisms on $\widehat{\mathcal{T}}$. More concretely, letting $\mathsf{e} := \eta \circ \varepsilon$ denote the unit for the convolution product $\star$,

$$S(\widehat{T}) = Id^{\star^{-1}}(\widehat{T}) = (e - (e - Id))^{\star^{-1}} = e(\widehat{T}) + \sum_{k \geq 1} (e - Id)^{\star\, k}(\widehat{T}).$$

Theorem

Let $\mathcal{L}_{\mathcal{T}} := (\mathsf{Prim}(\widehat{\mathcal{T}}), [.,.]_{\diamond})$ denote the ***tracelet Lie algebra***, with $[\widehat{\mathsf{T}}, \widehat{\mathsf{T}}']_{\diamond} := \widehat{\mathsf{T}} \diamond \widehat{\mathsf{T}}' - \widehat{\mathsf{T}}' \diamond \widehat{\mathsf{T}}$ (commutator operation w.r.t. $\diamond$). Then the tracelet Hopf algebra is isomorphic (in the sense of Hopf algebra isomorphisms) to the universal enveloping algebra of $\mathcal{L}_{\mathcal{T}}$.

On Stochastic Rewriting and Combinatorics via Rule-Algebraic Methods*

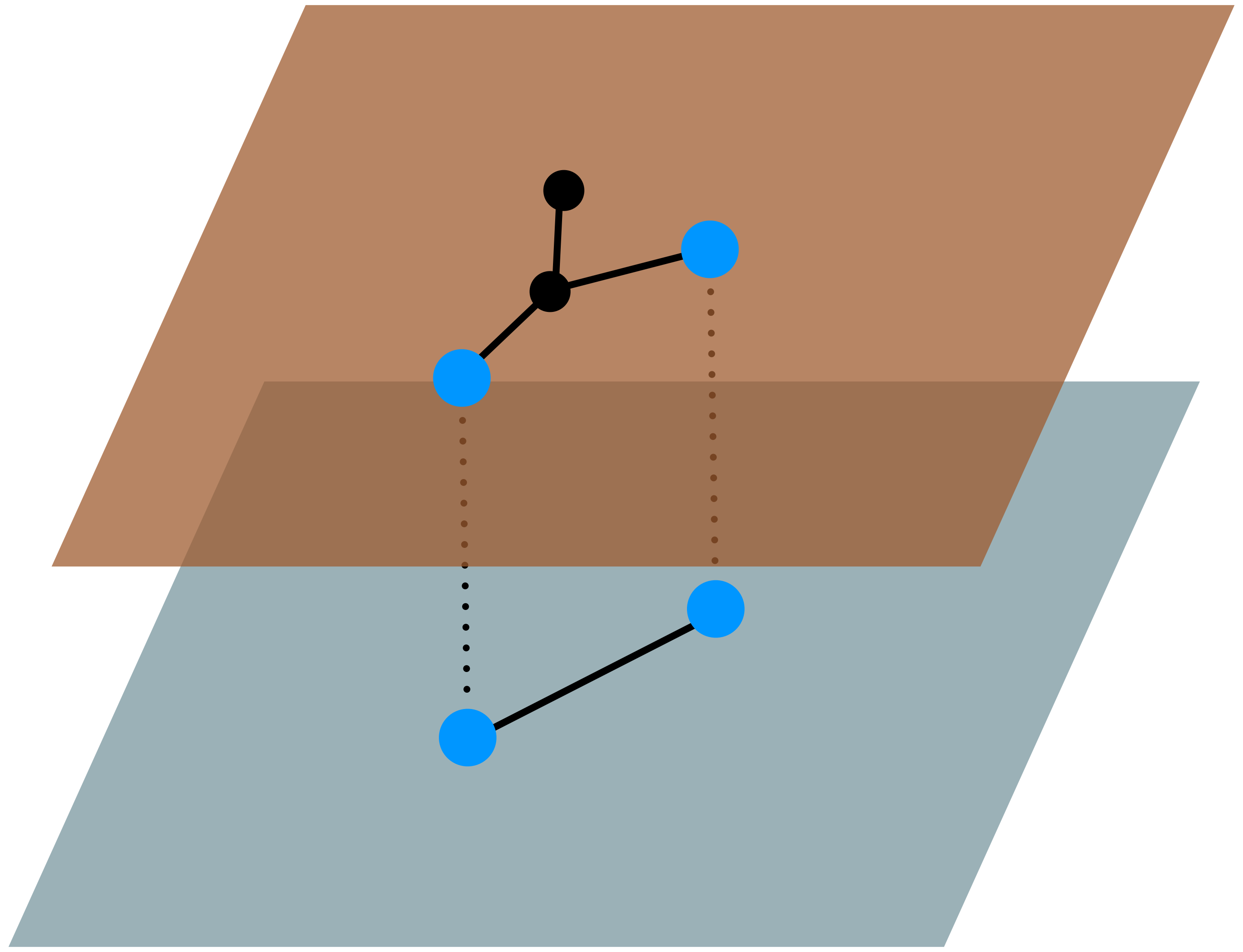
Nicolas Behr
Université de Paris, CNRS, IRIF
F-75006, Paris, France
nicolas.behr@irif.fr

Building upon the rule-algebraic stochastic mechanics framework, we present new results on the relationship of stochastic rewriting systems described in terms of continuous-time Markov chains, their embedded discrete-time Markov chains and certain types of generating function expressions in combinatorics. We introduce a number of generating function techniques that permit a novel form of static analysis for rewriting systems based upon marginalizing distributions over the states of the rewriting systems via pattern-counting observables.

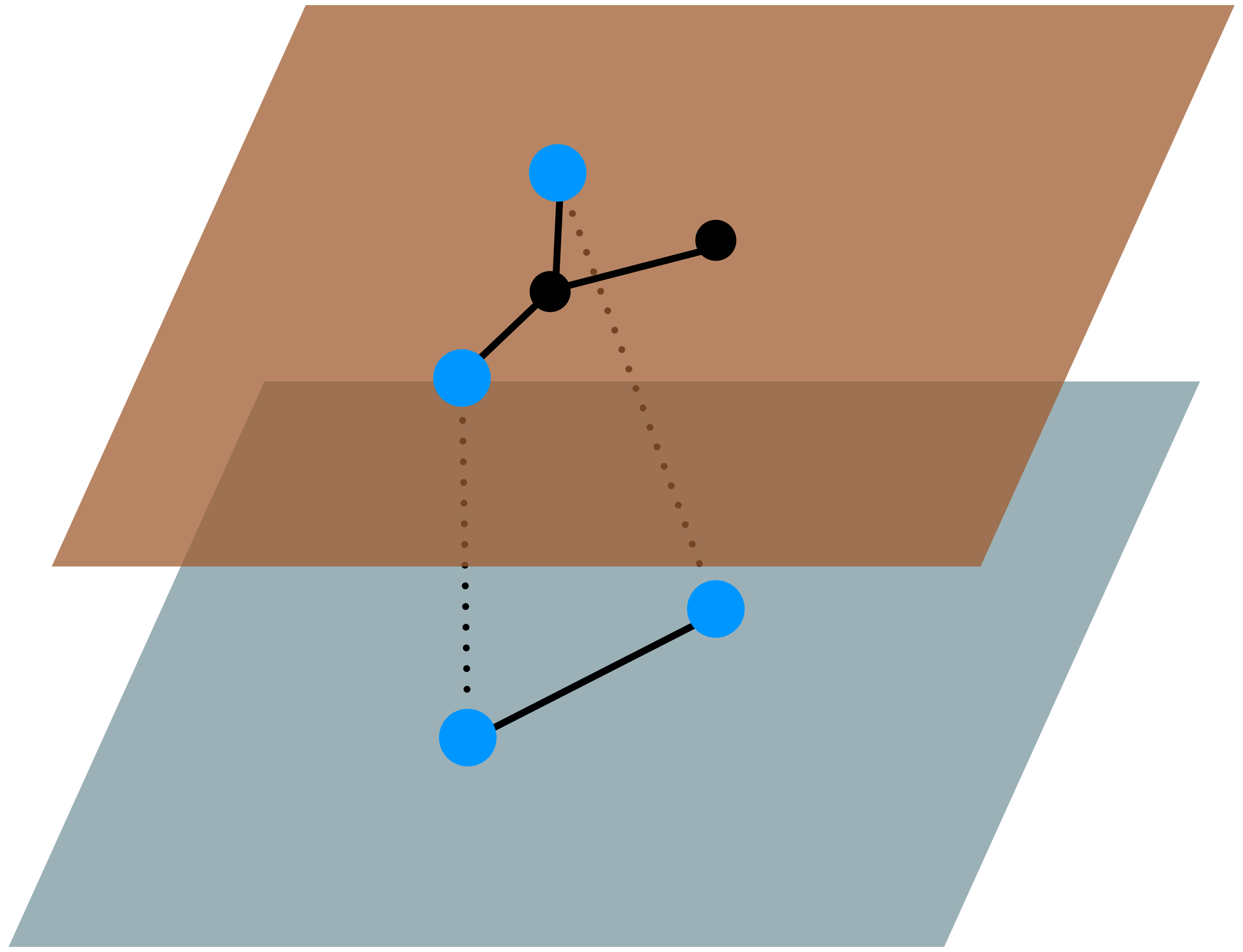
TERMGRAPH 2020

Example: generating **planar rooted binary trees (PRBTs)** uniformly

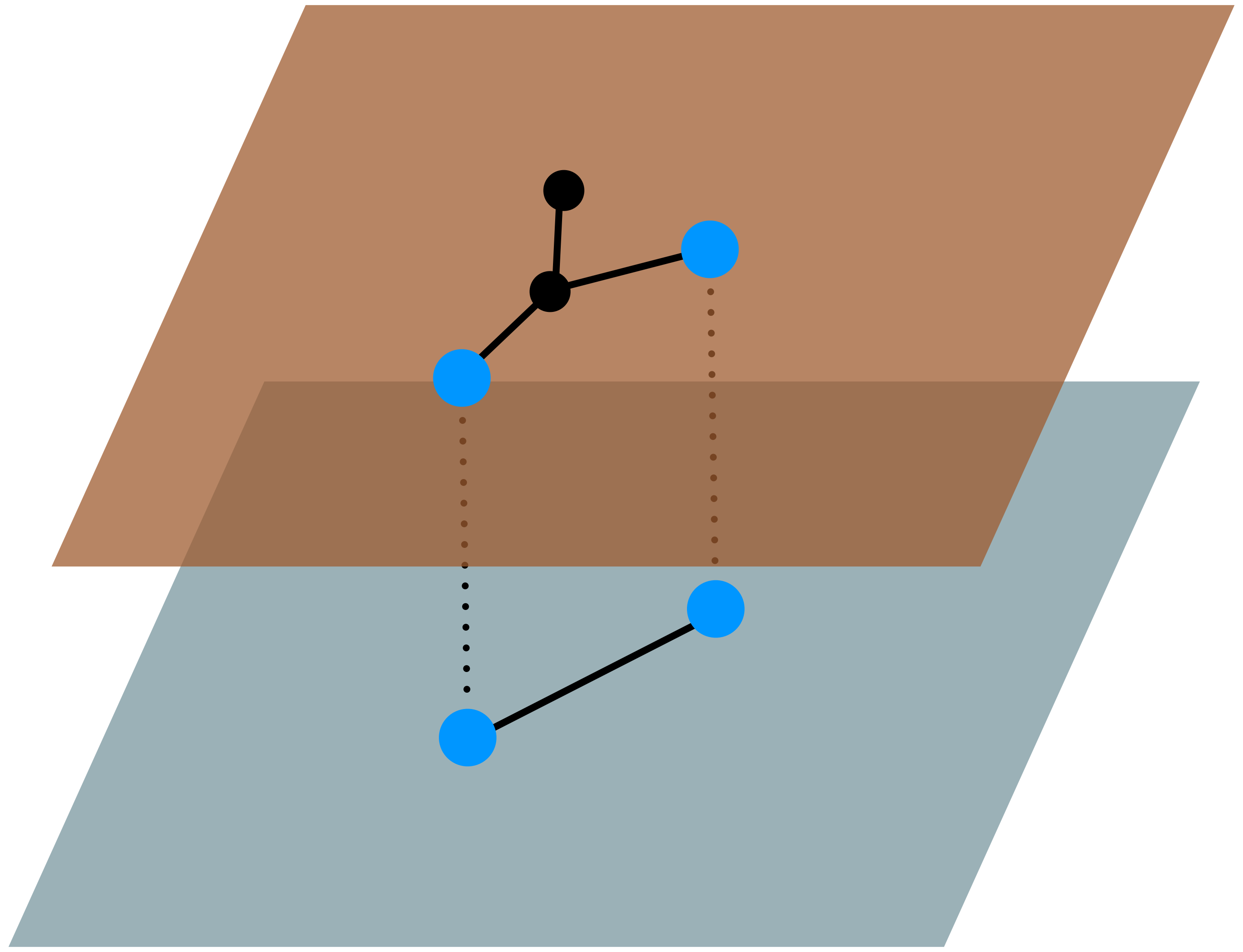
The Rémy uniform generator (heuristics)



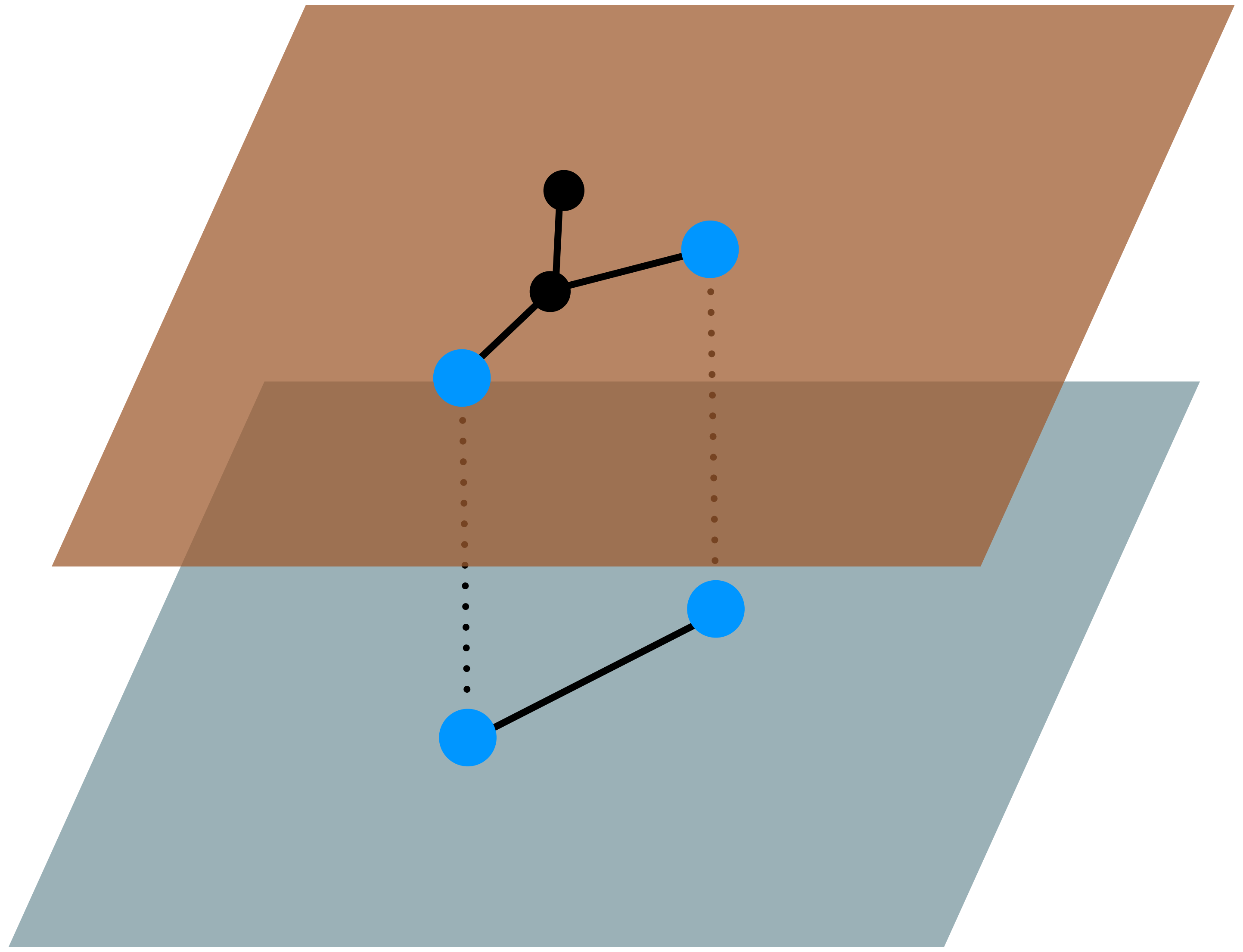
or



Example: generating **planar rooted binary trees (PRBTs)** uniformly

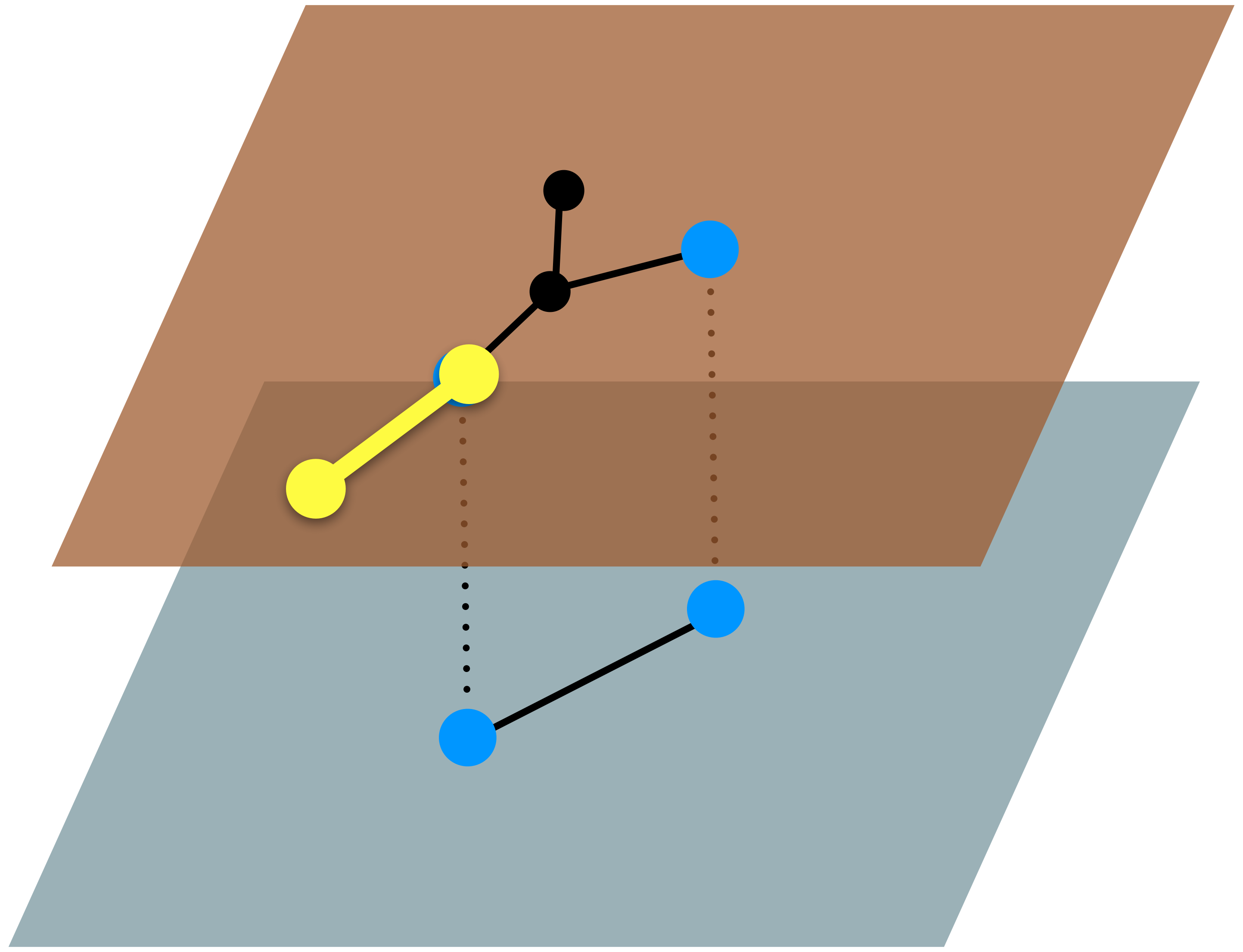


"counting" **after** rewriting

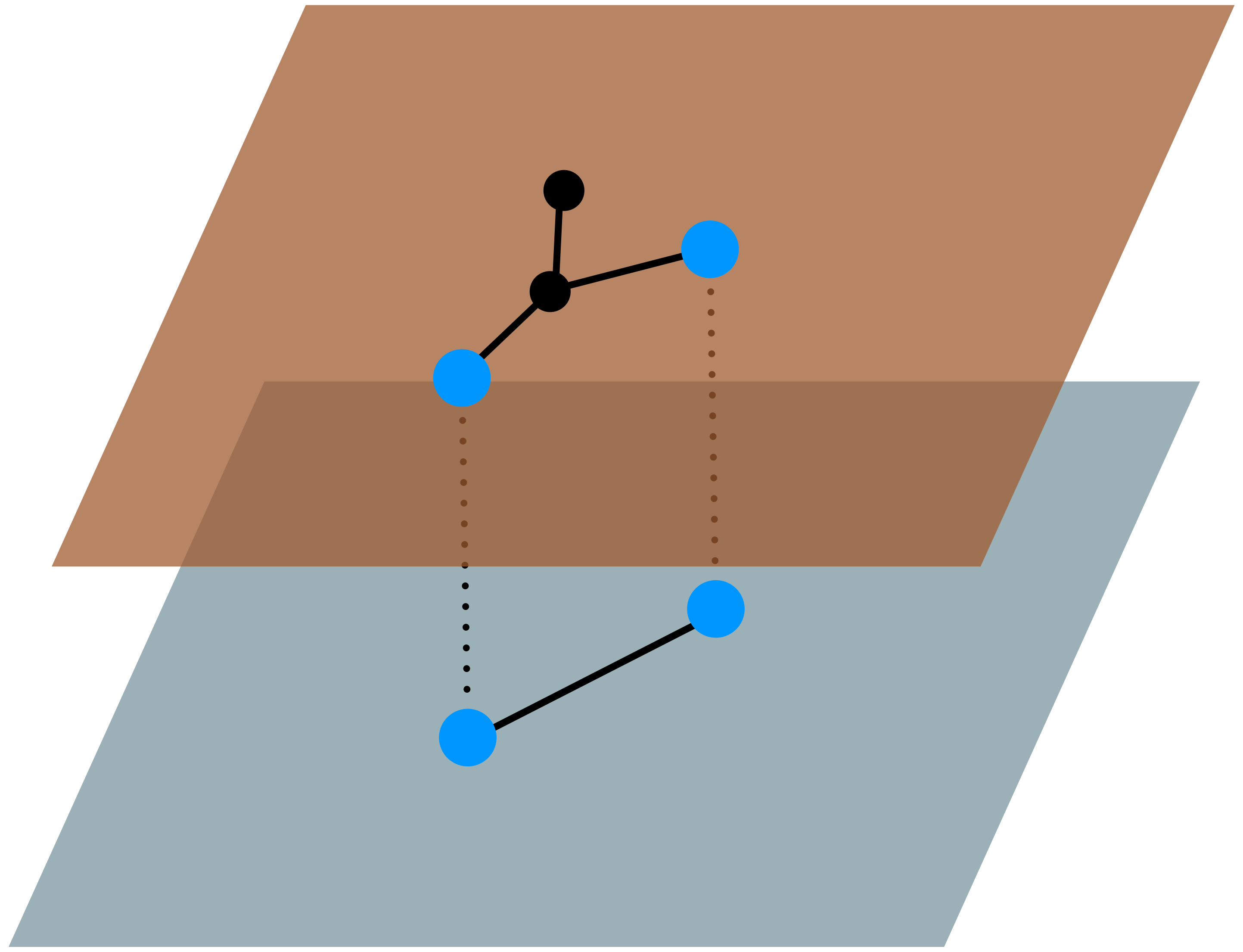


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

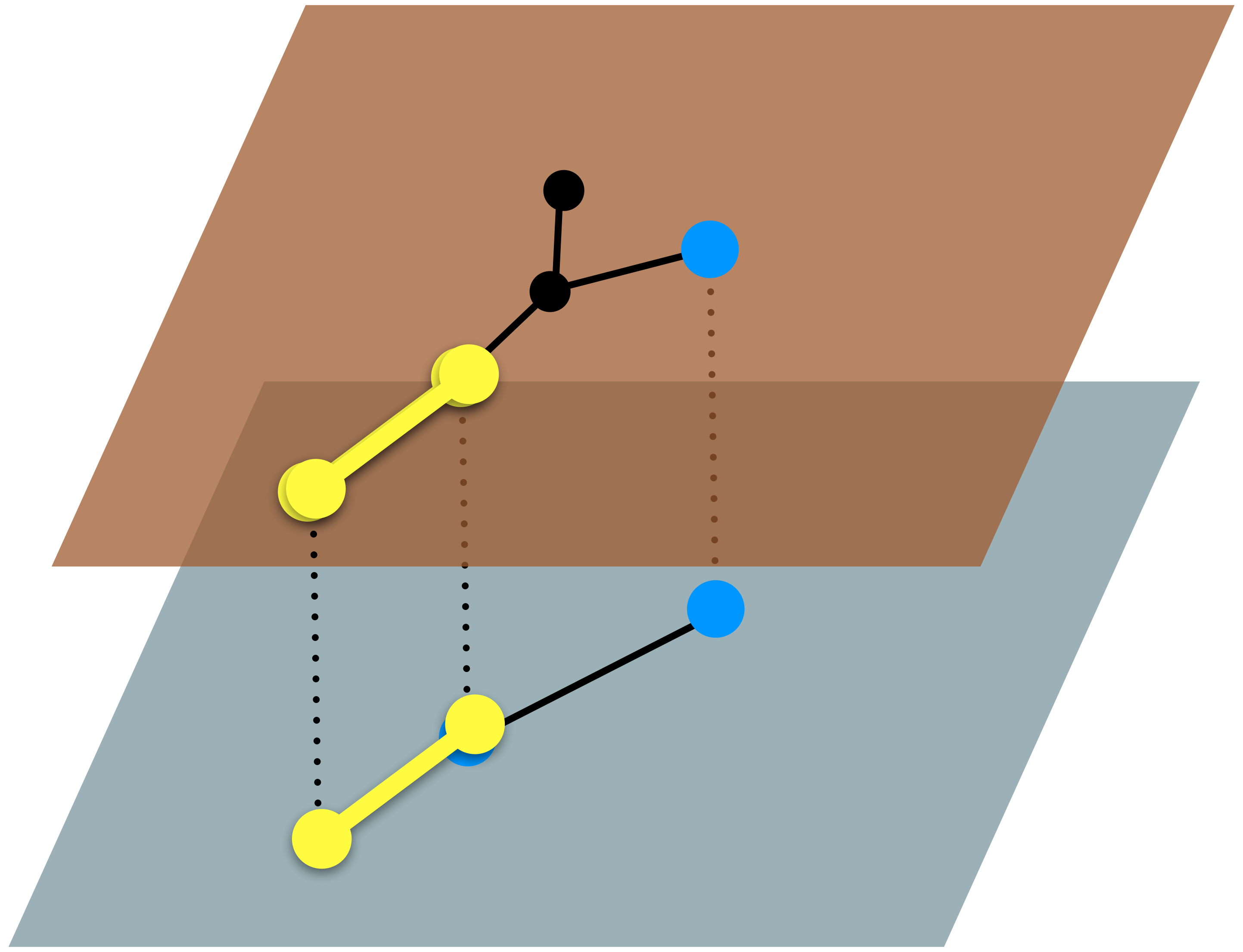


"counting" **after** rewriting

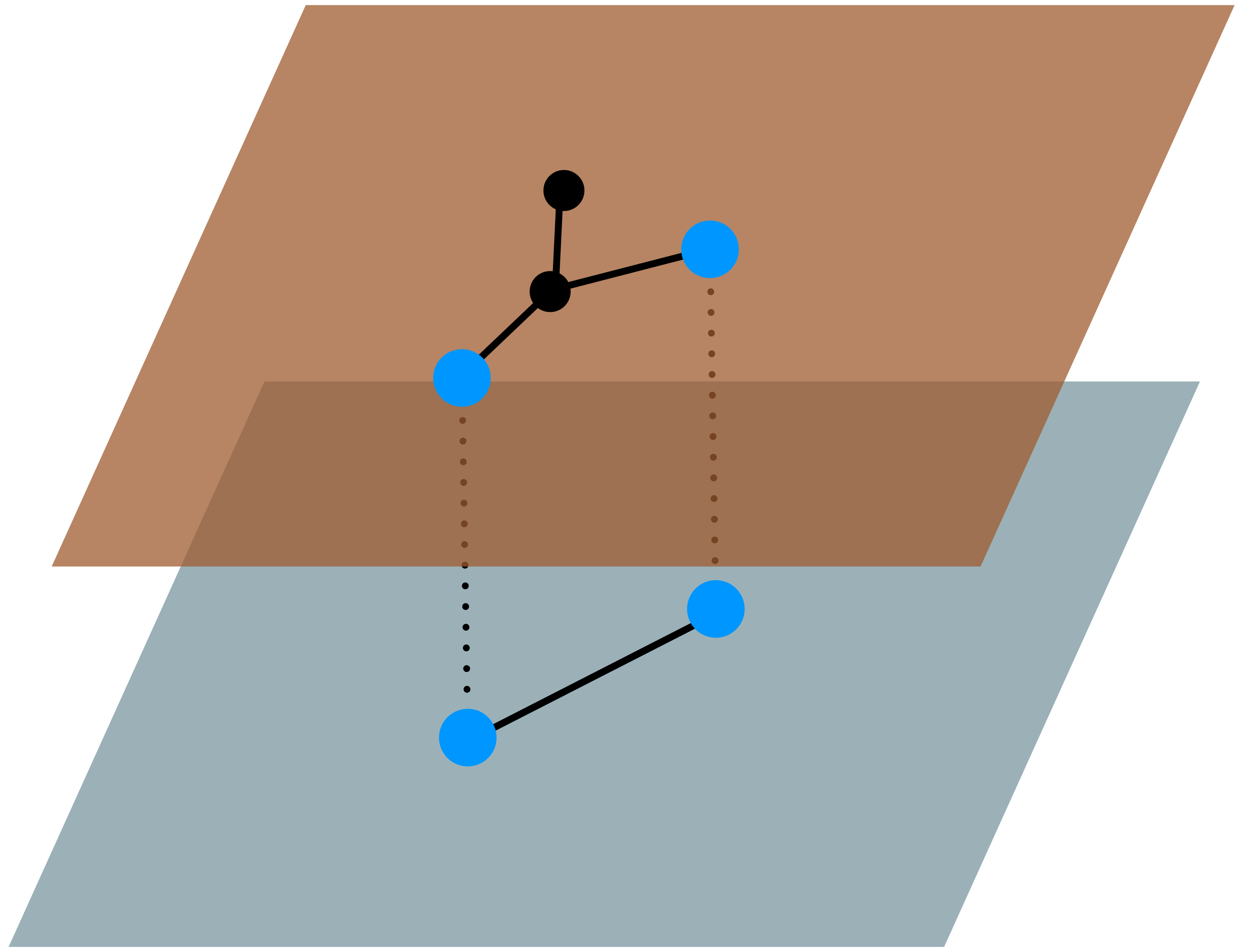


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

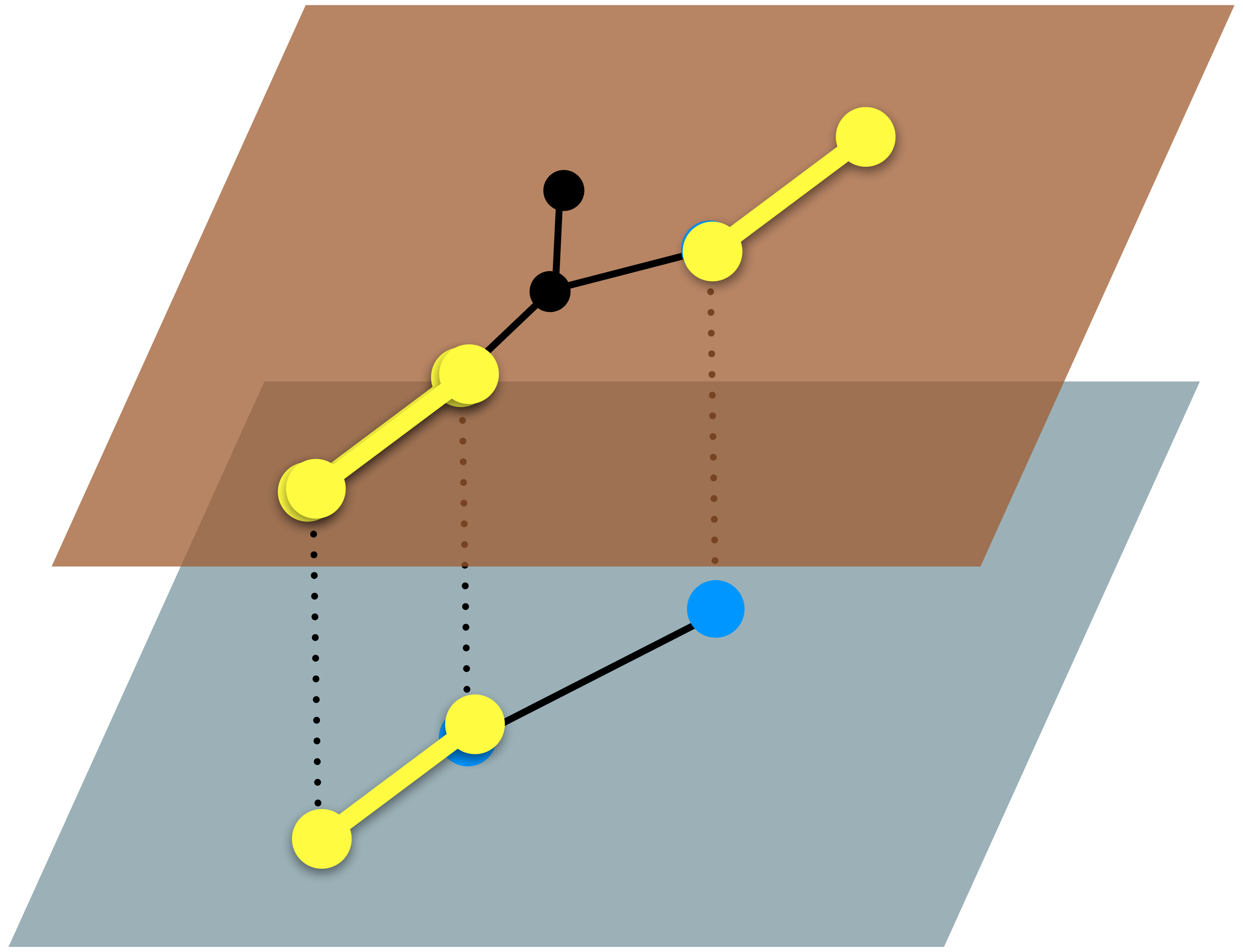


"counting" **after** rewriting

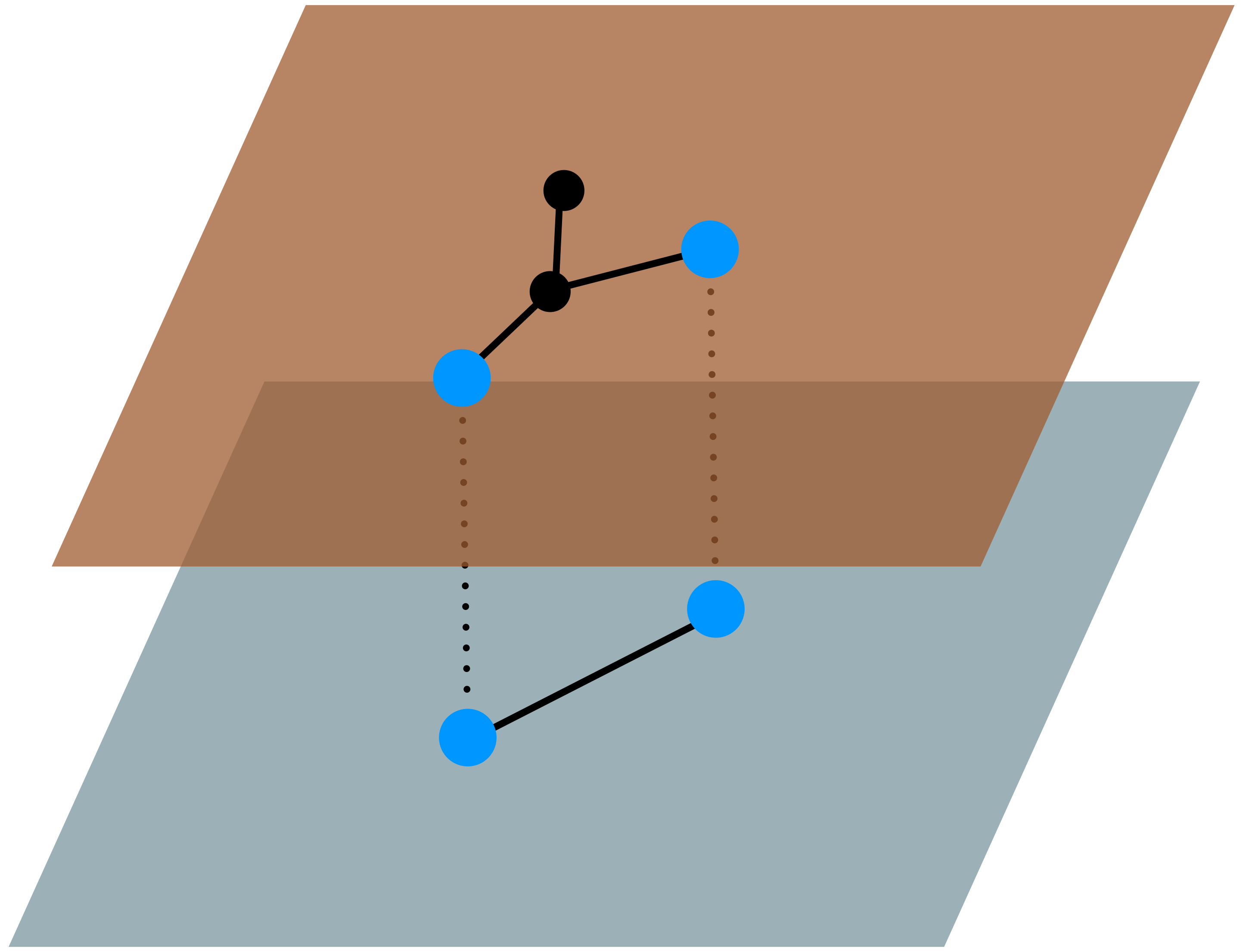


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

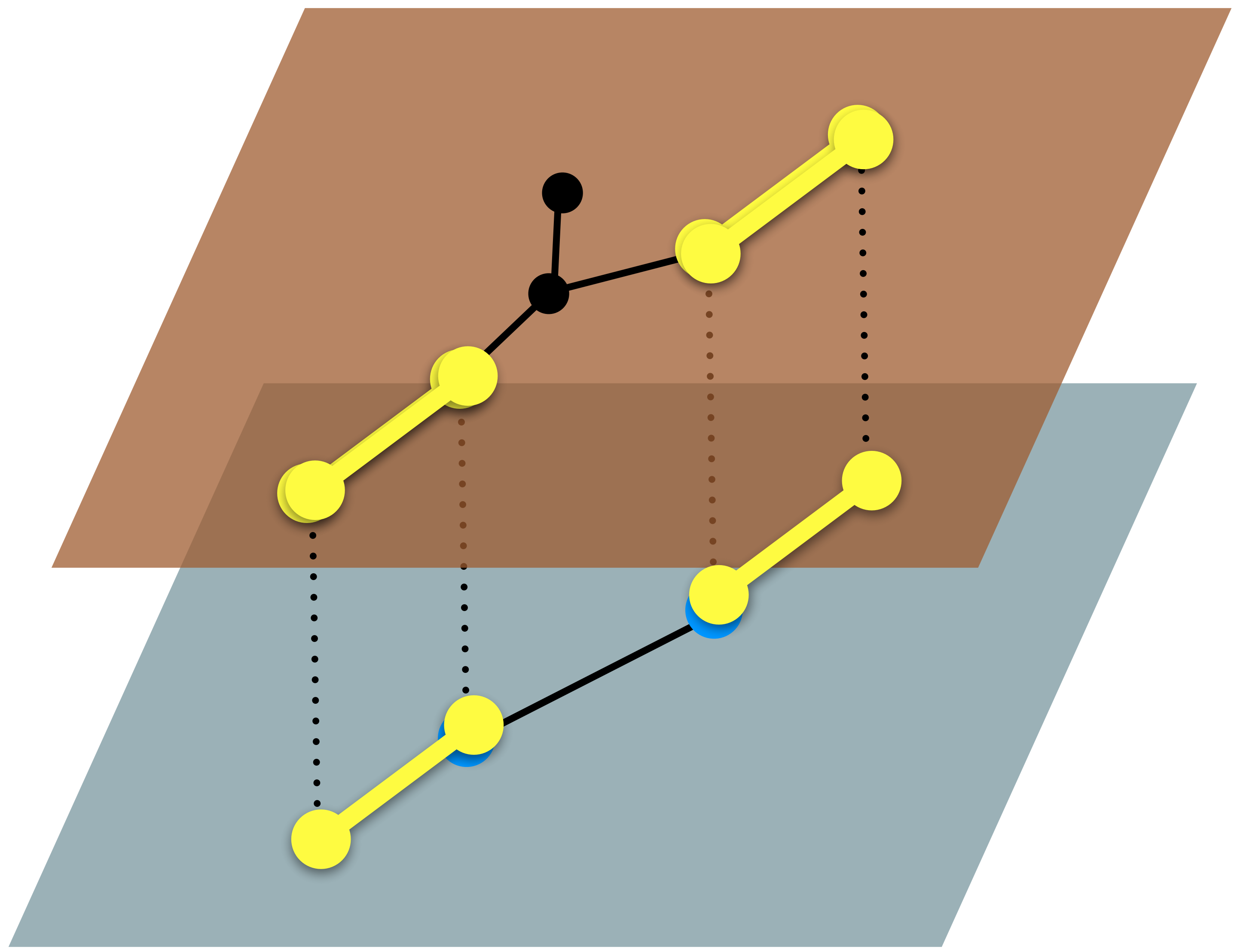


"counting" **after** rewriting

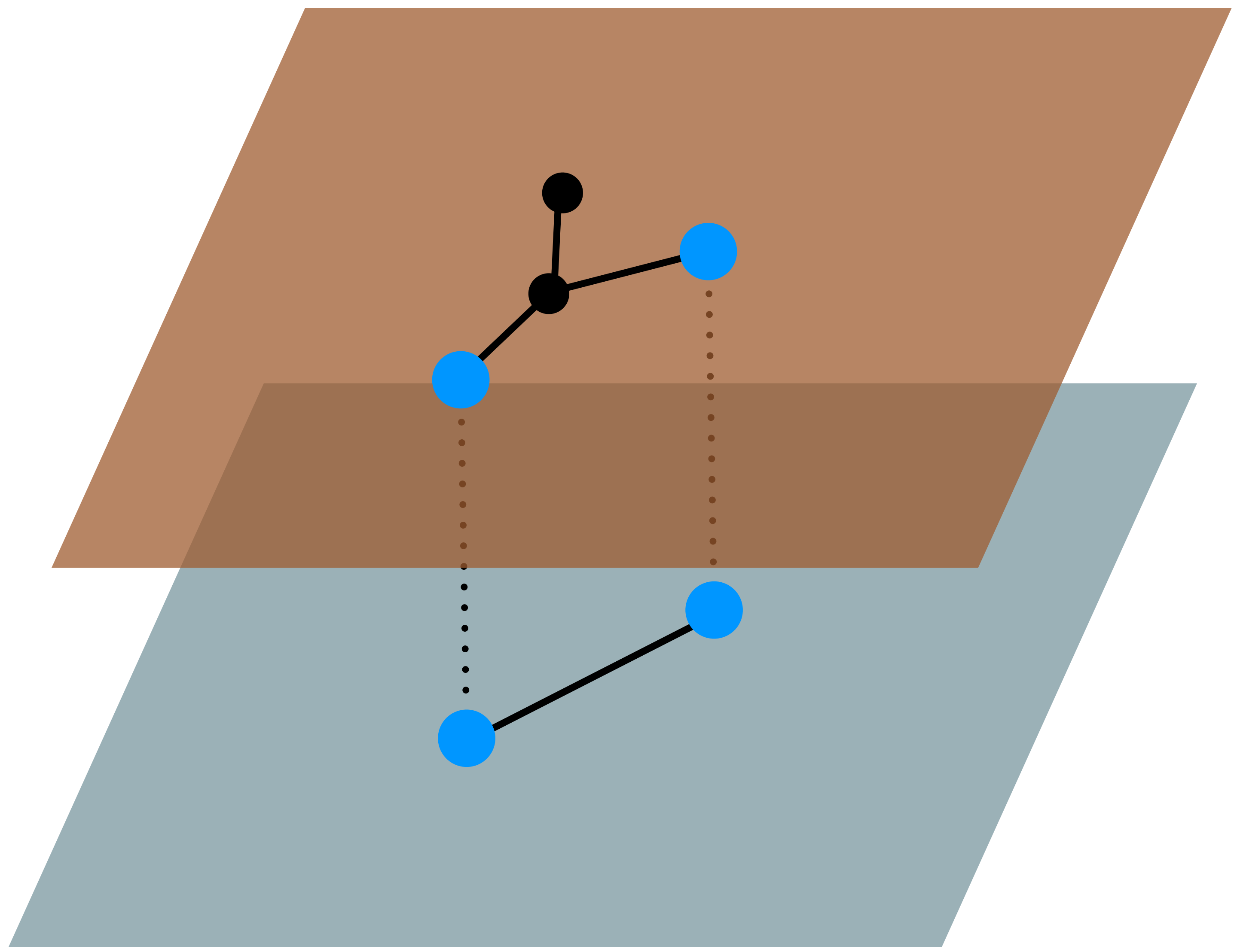


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

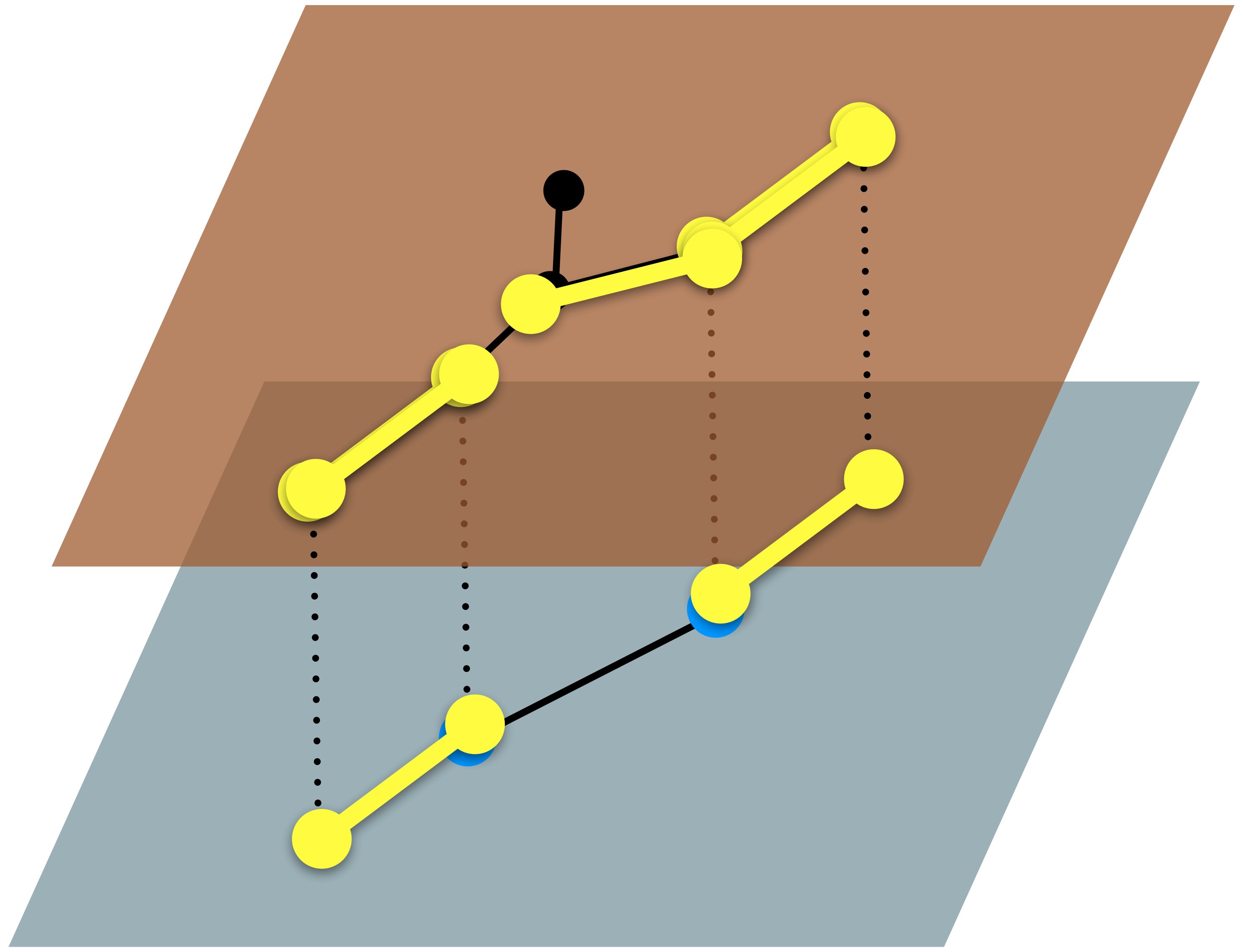


"counting" **after** rewriting

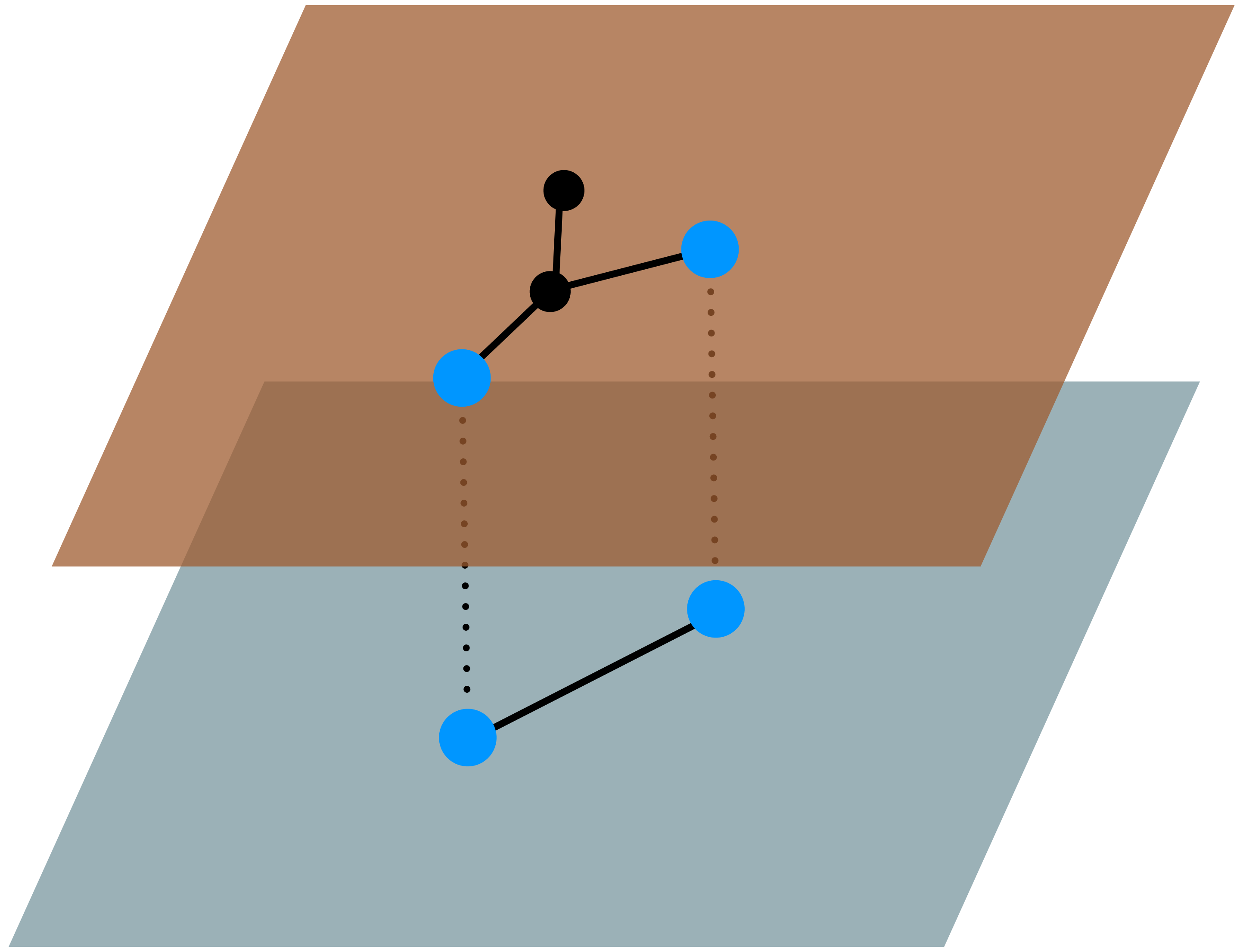


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

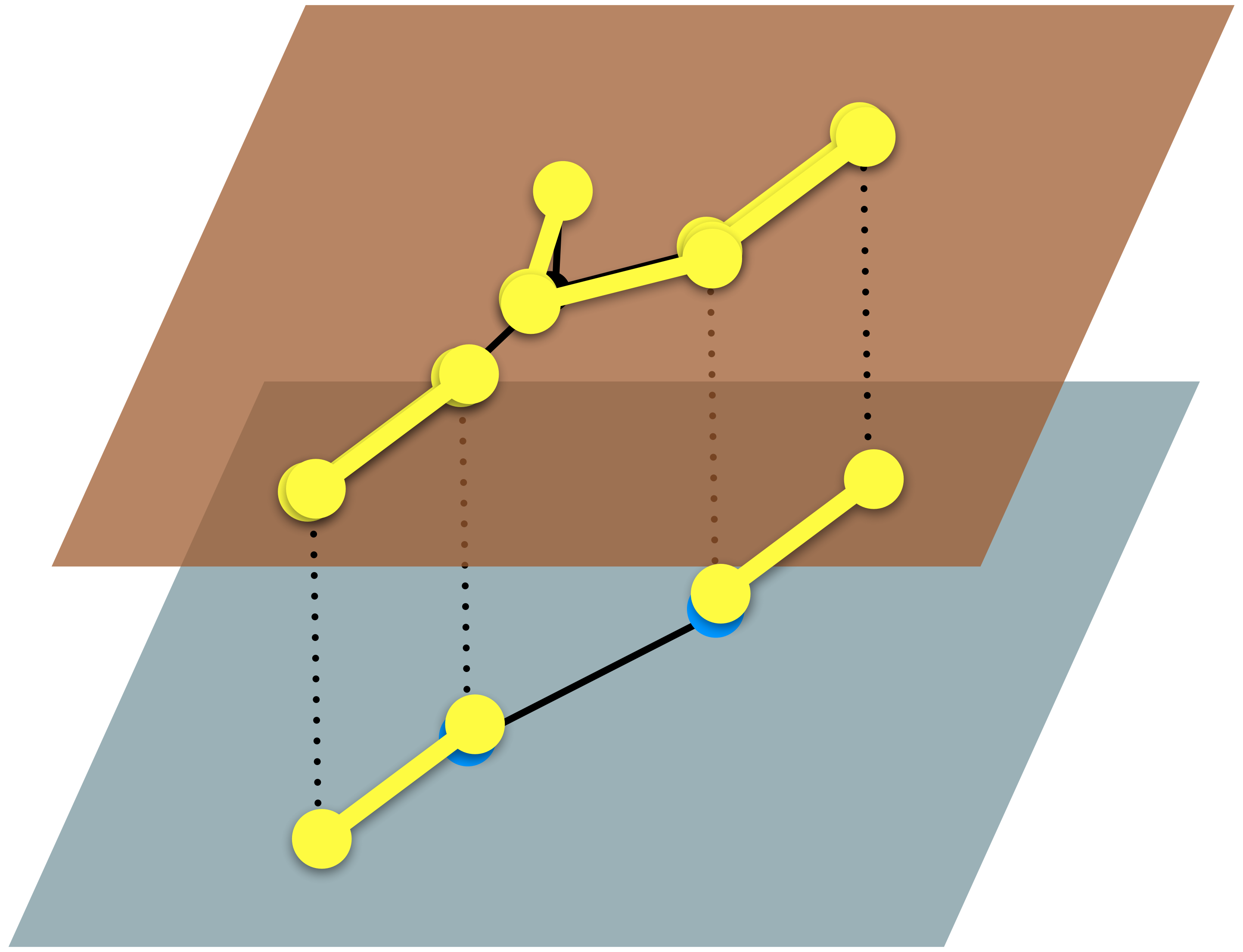


"counting" **after** rewriting

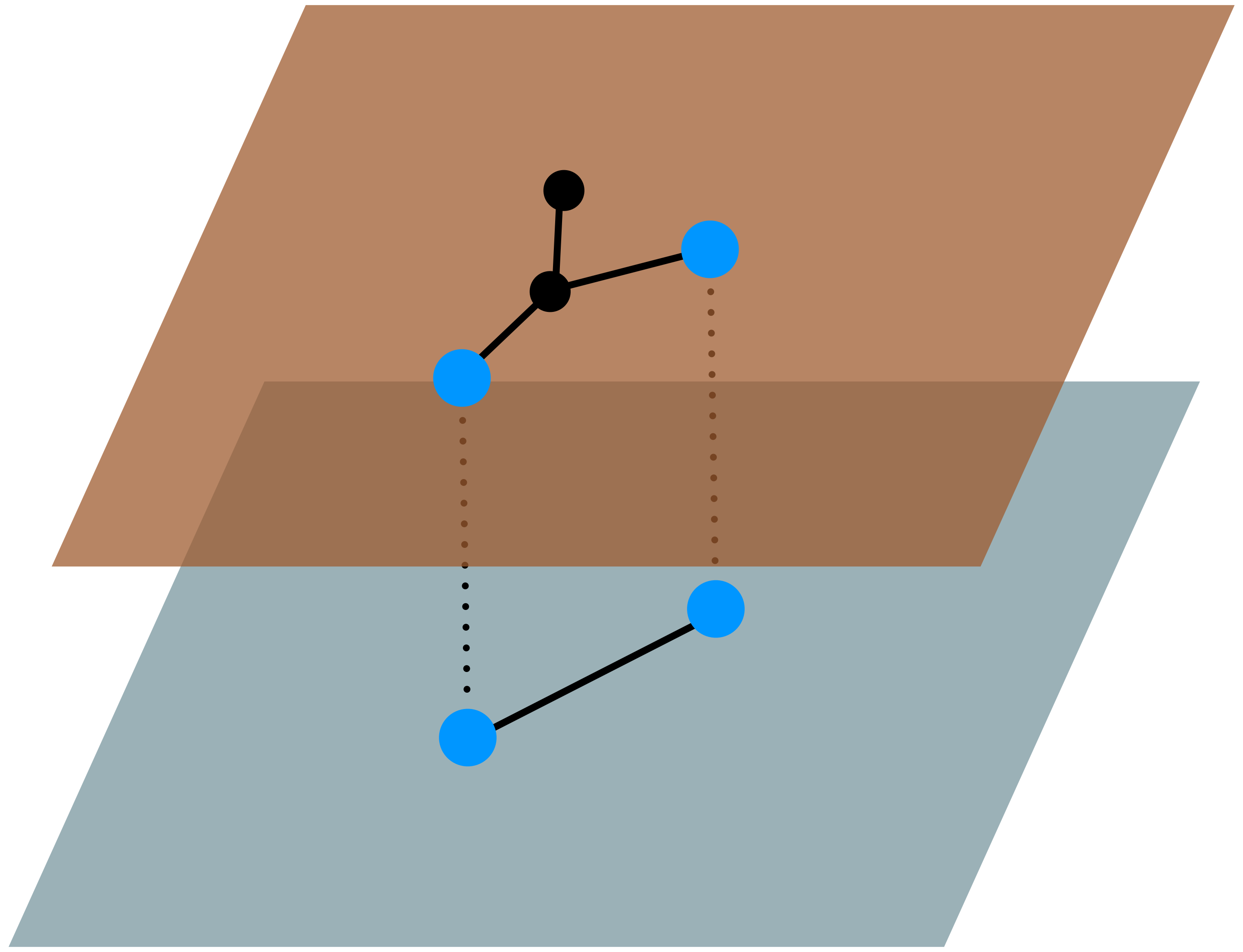


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

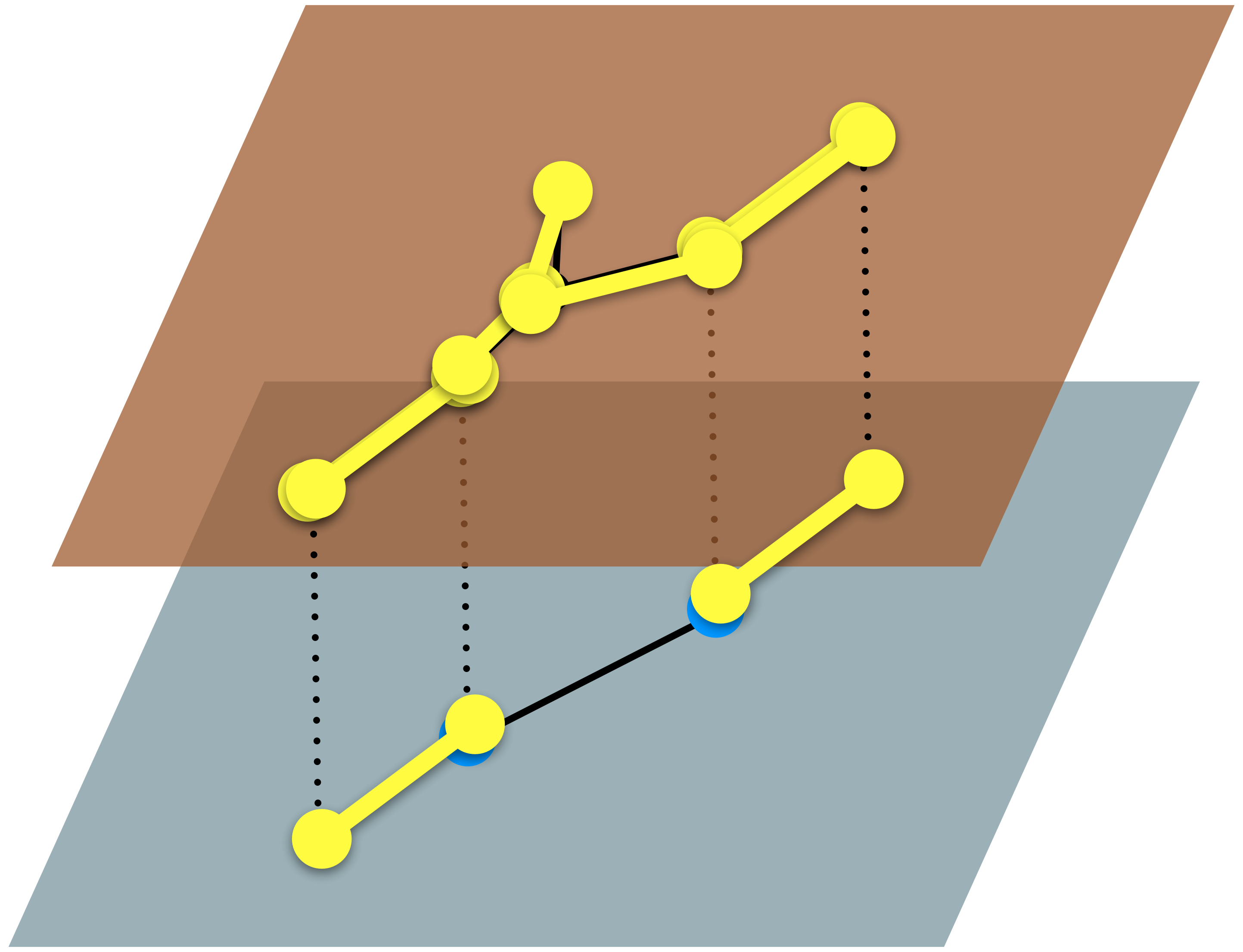


"counting" **after** rewriting

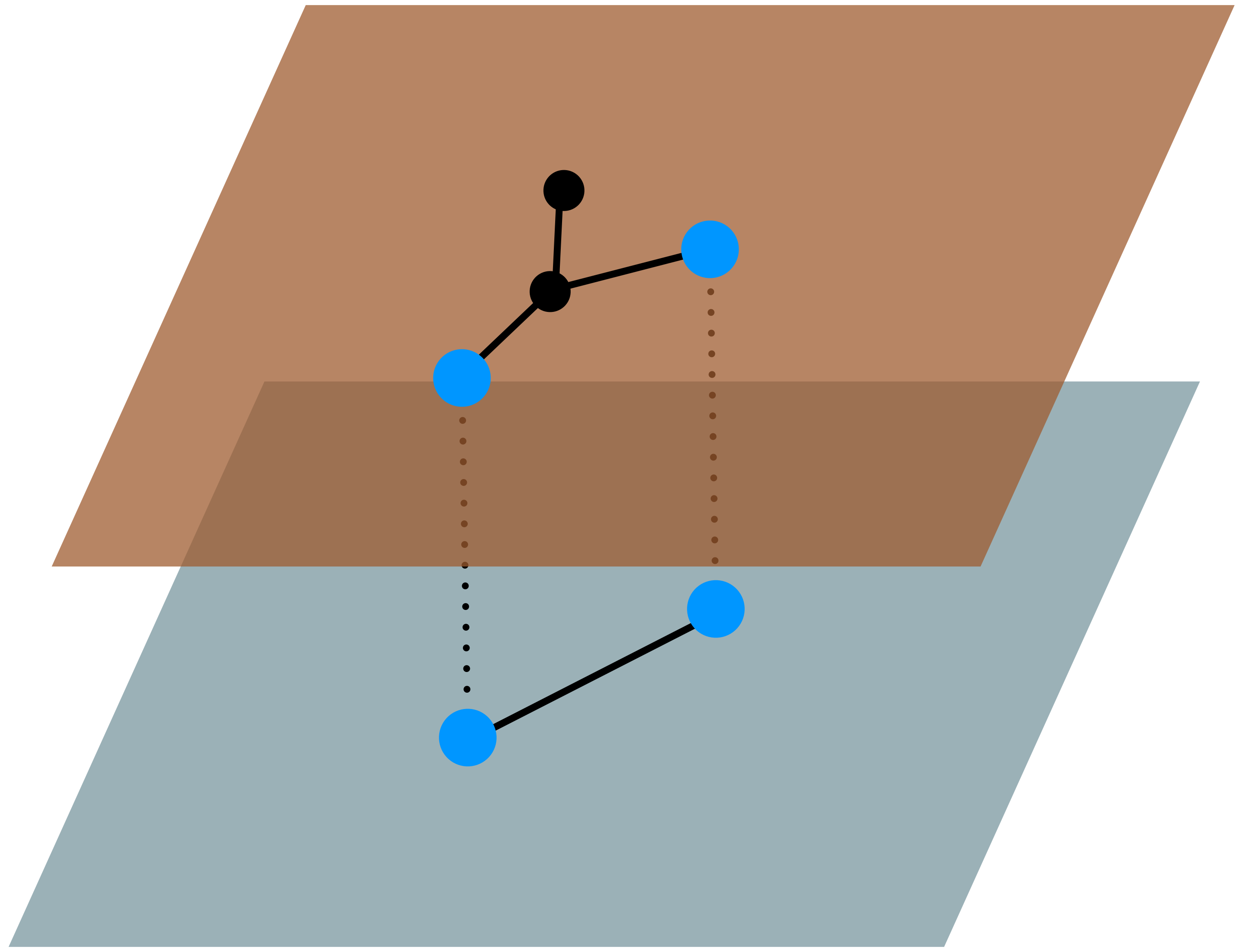


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

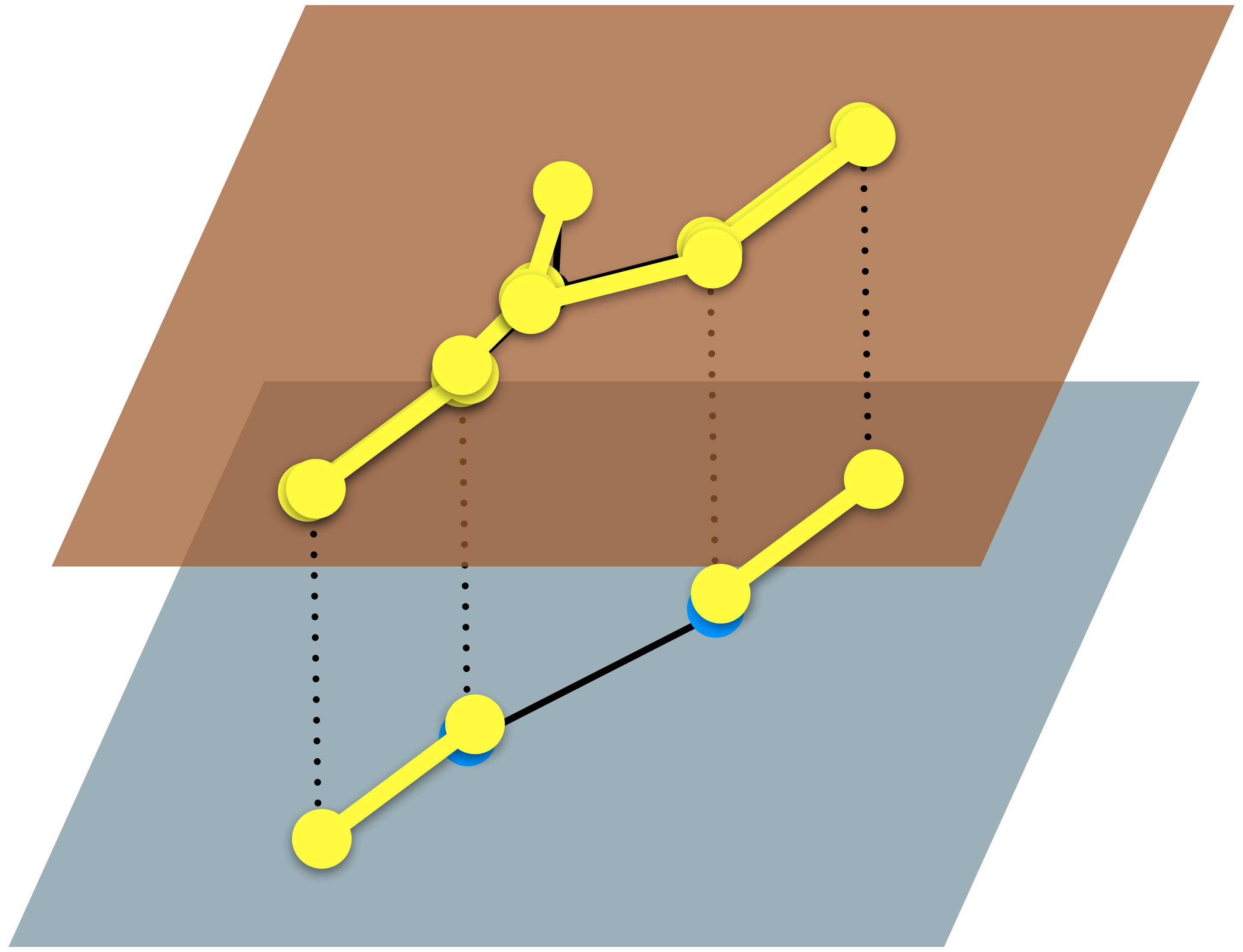


"counting" **after** rewriting

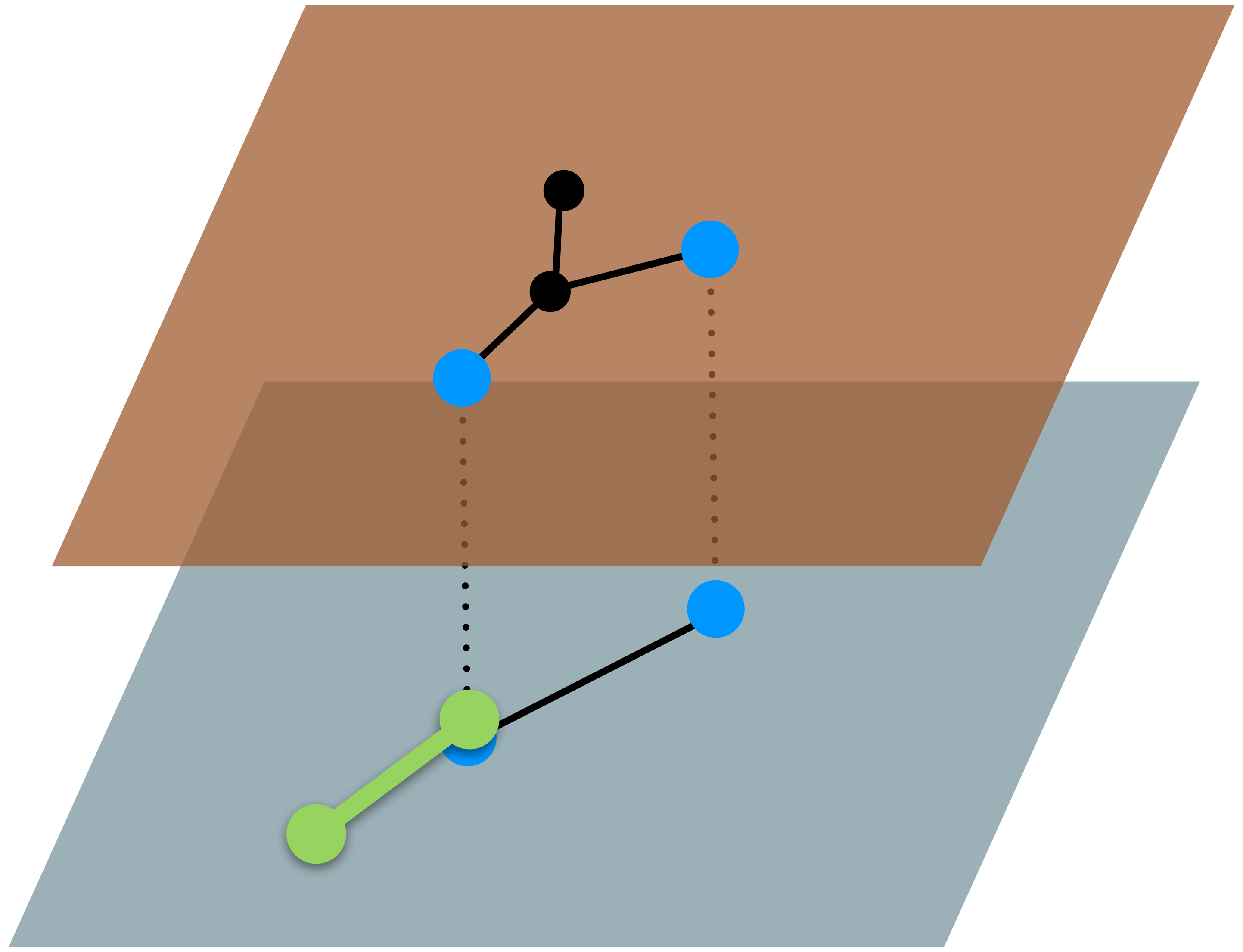


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

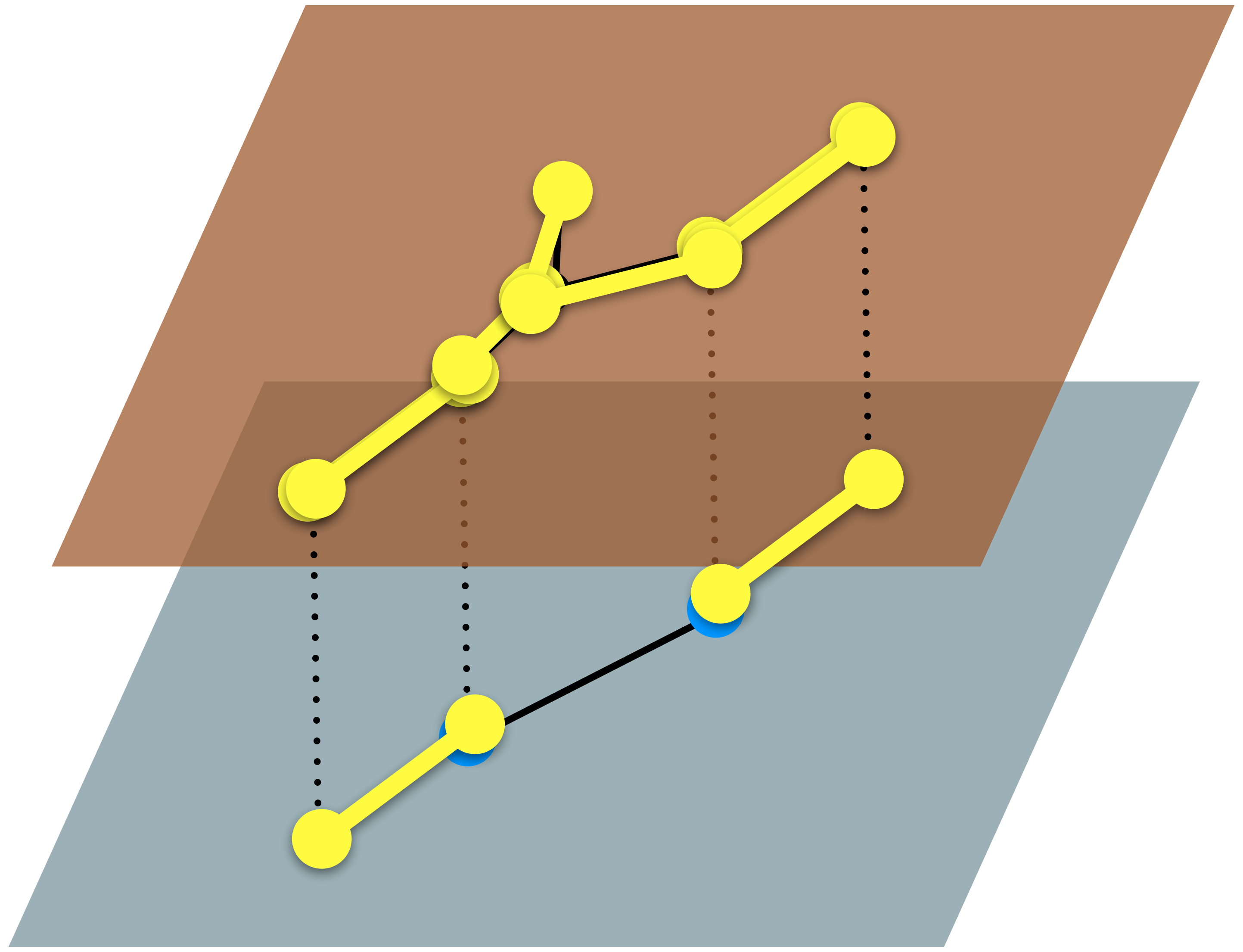


"counting" **after** rewriting

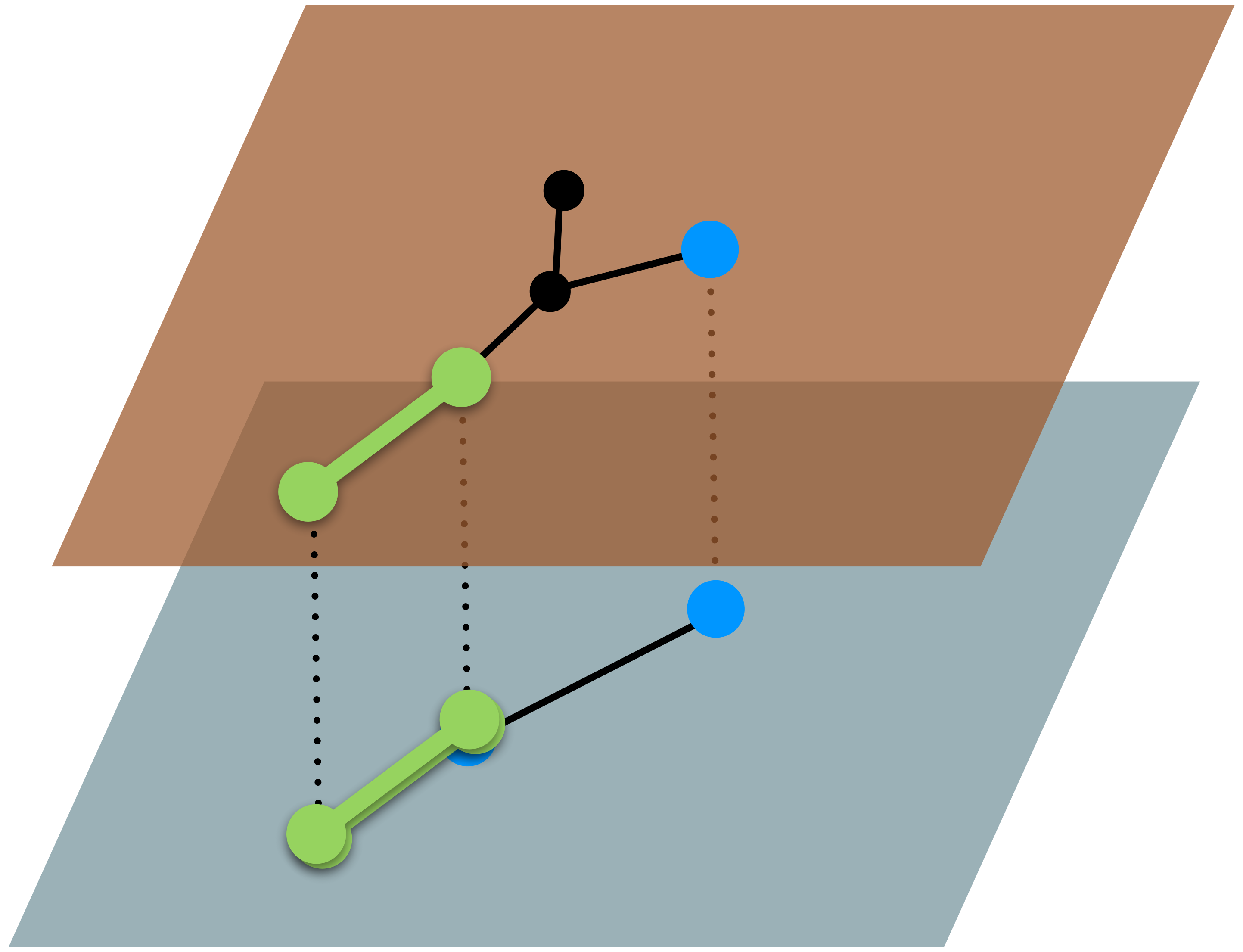


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

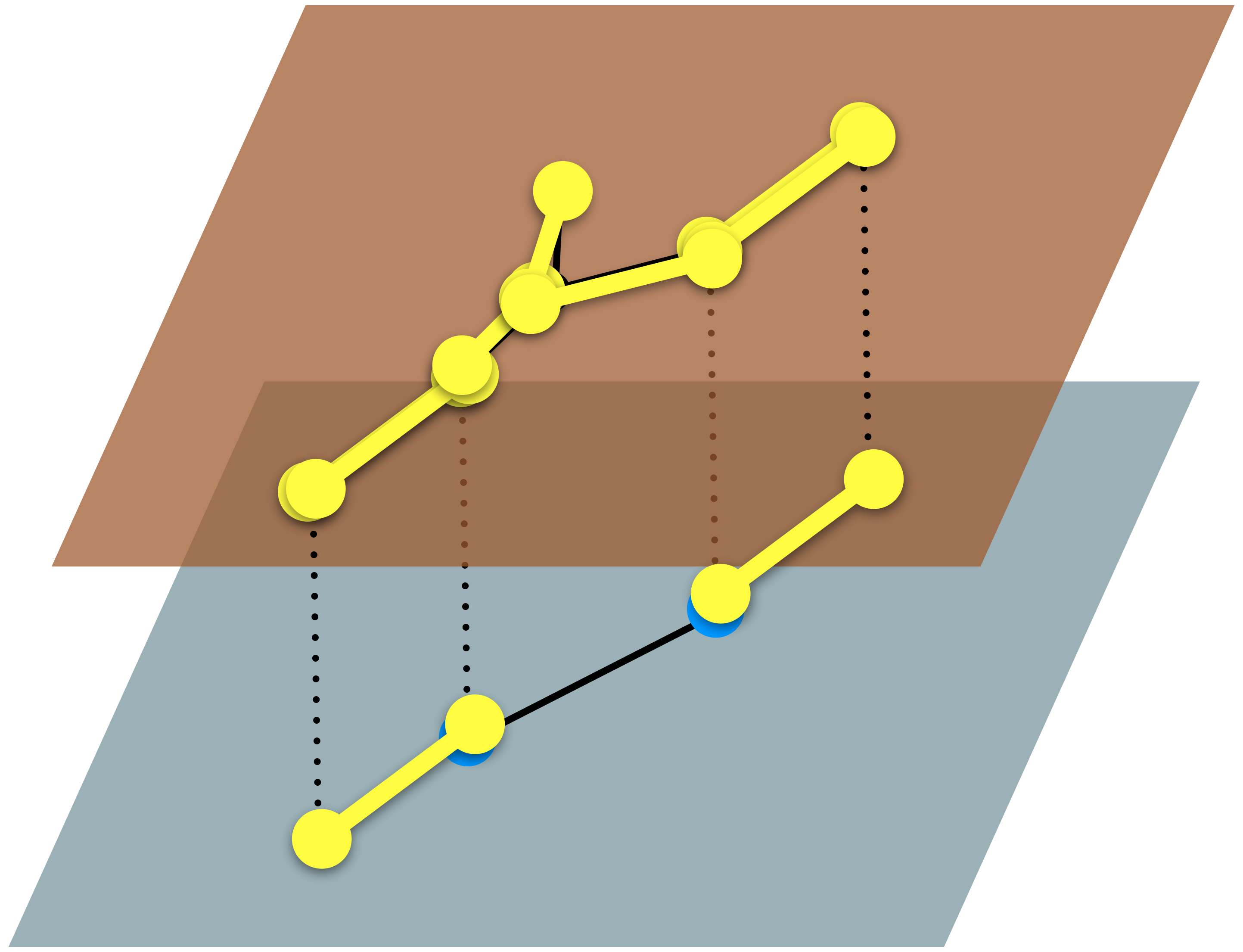


"counting" **after** rewriting

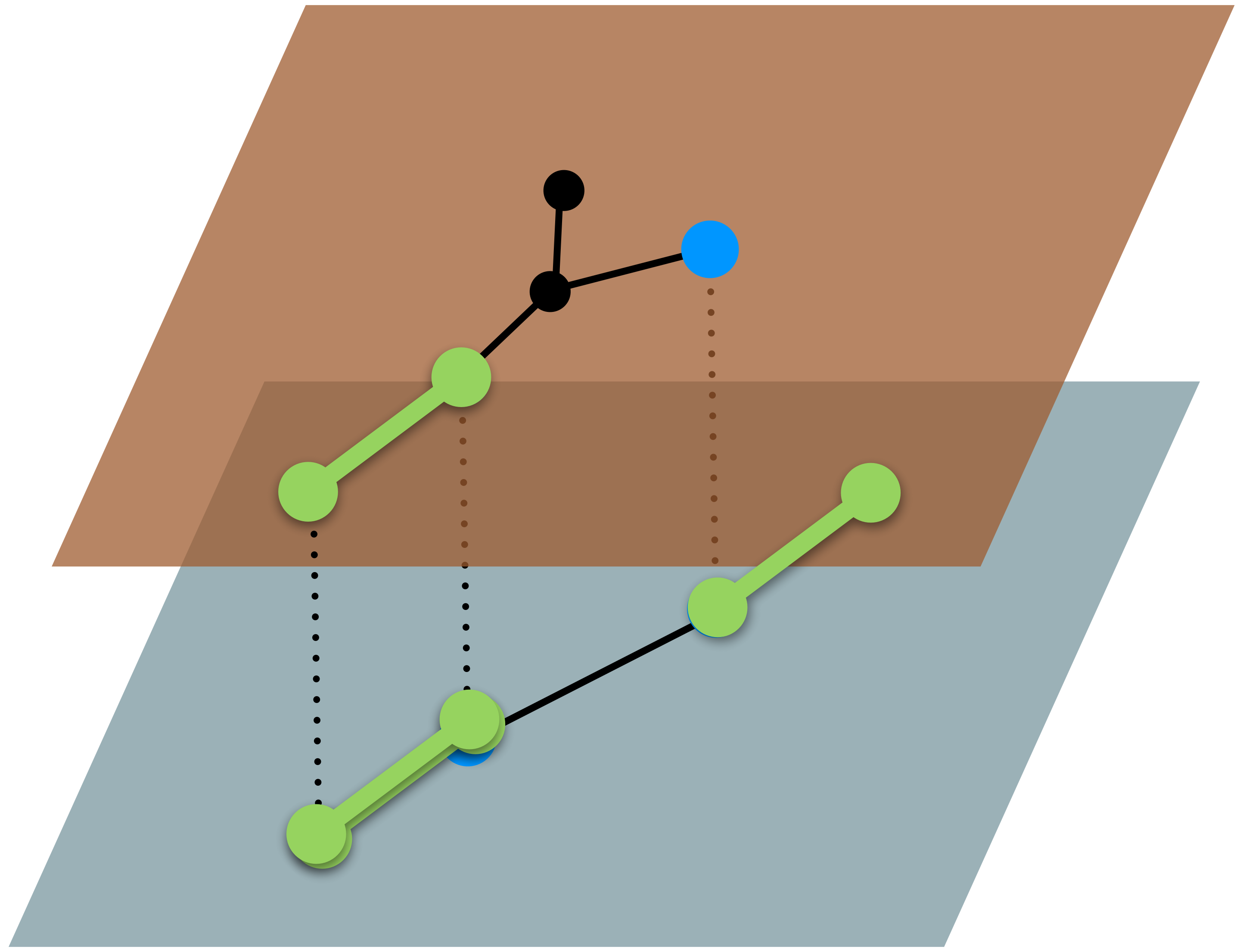


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

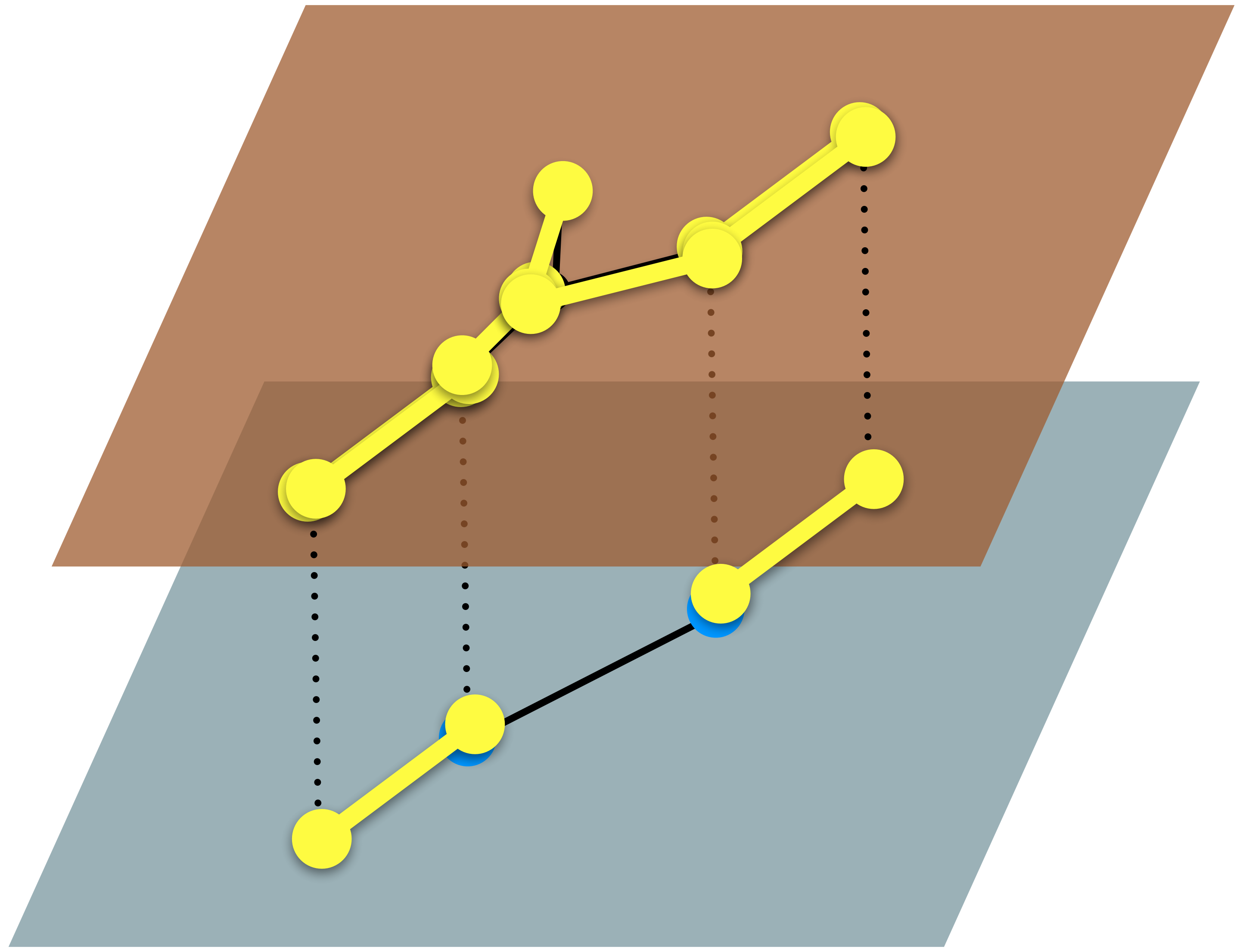


"counting" **after** rewriting

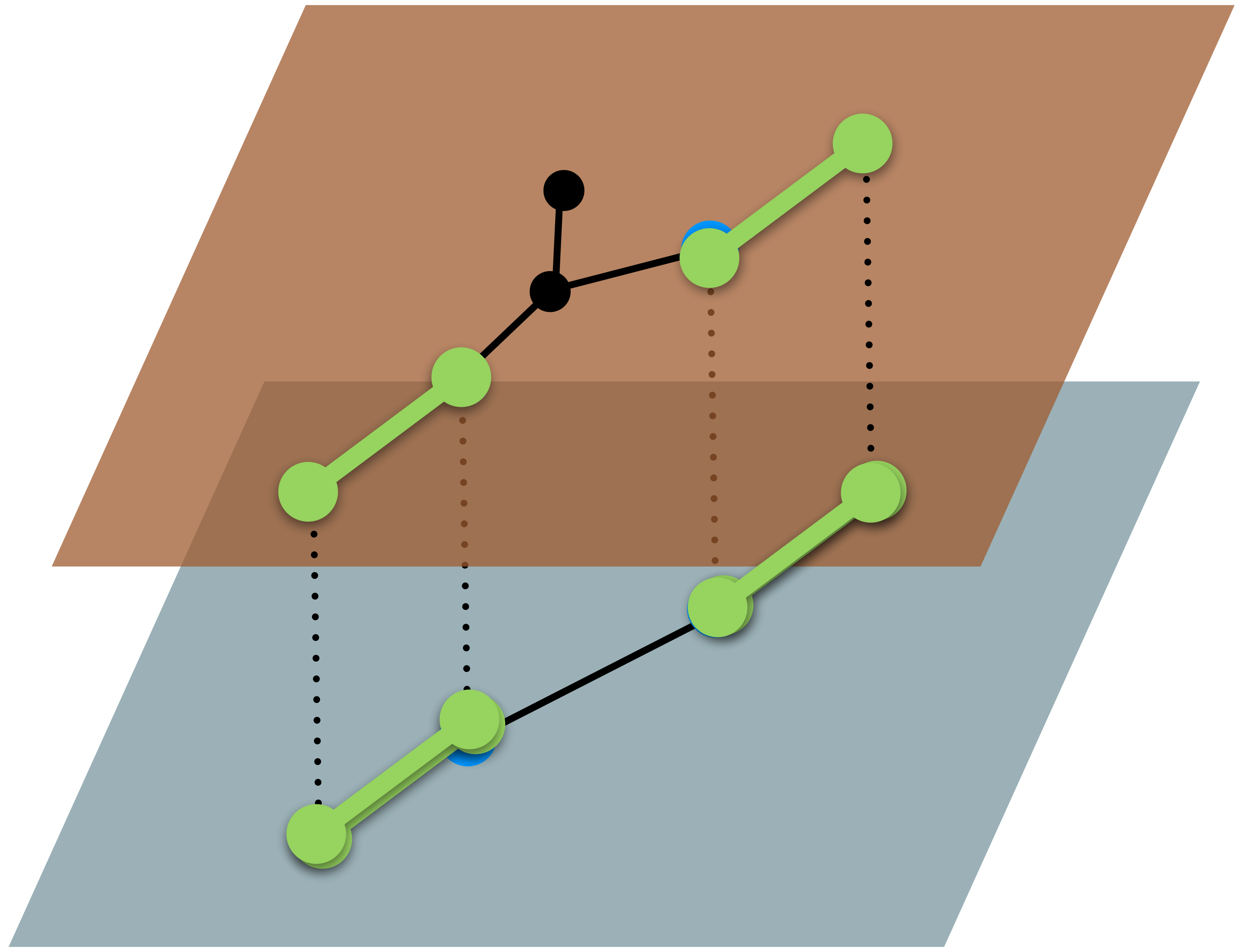


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

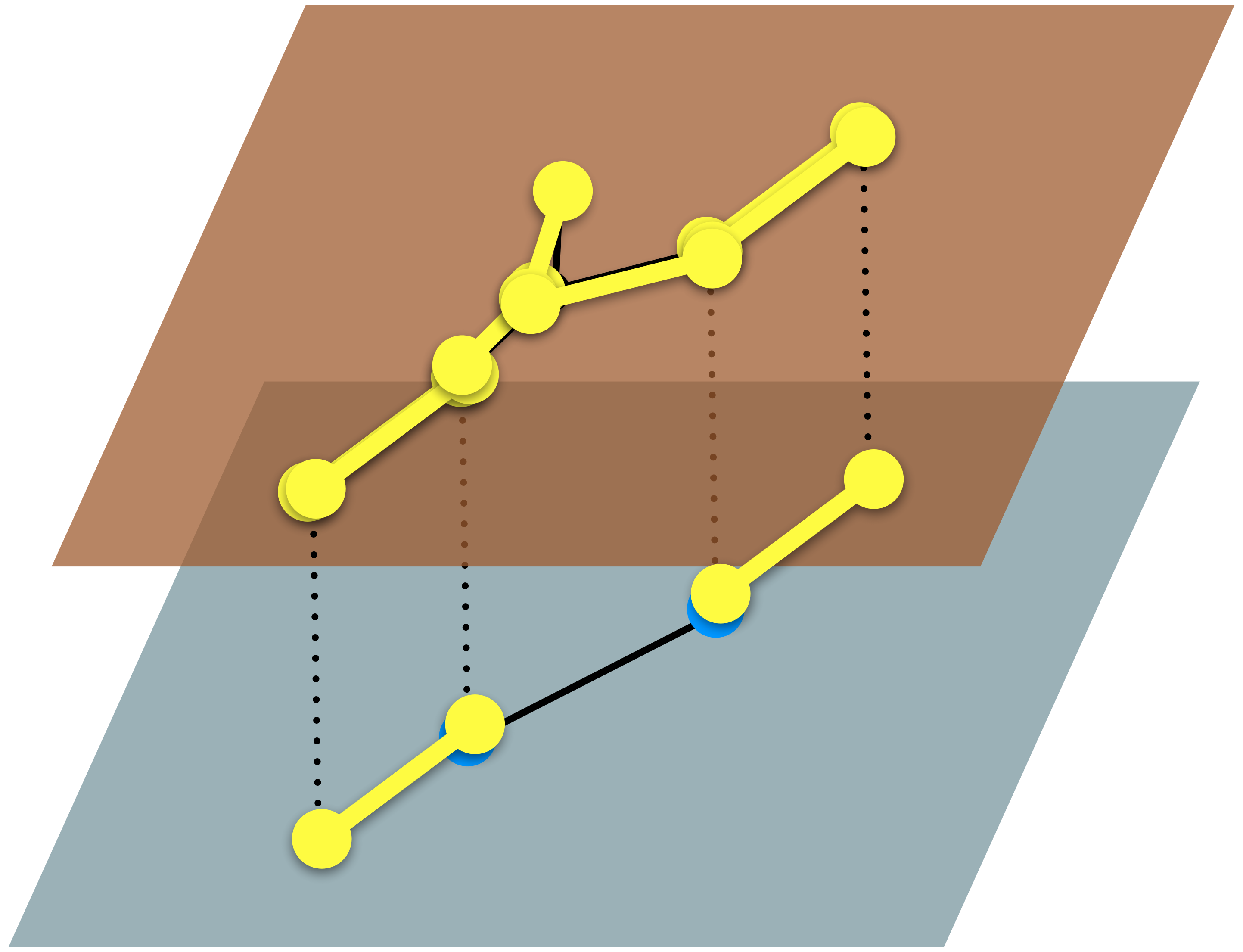


"counting" **after** rewriting

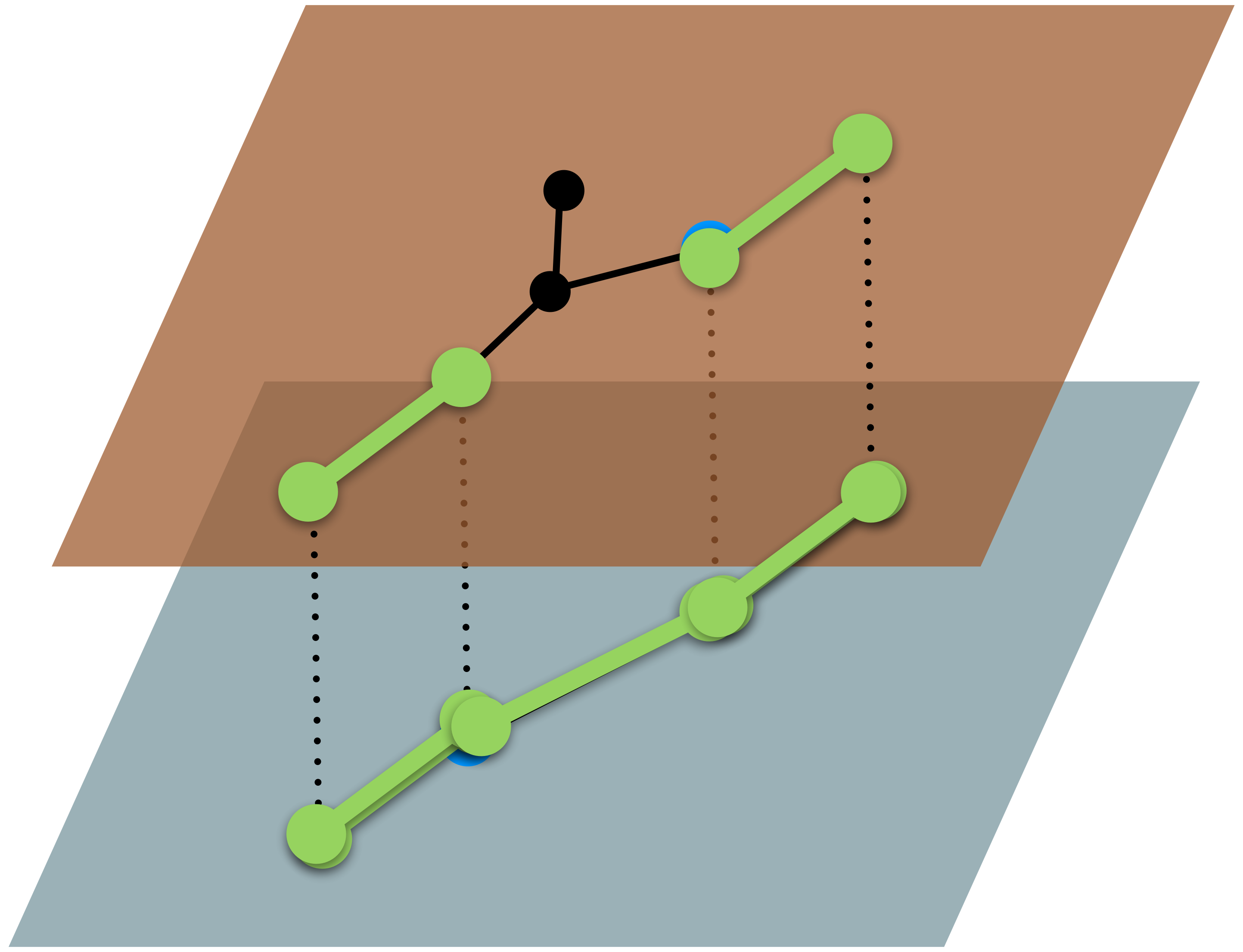


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

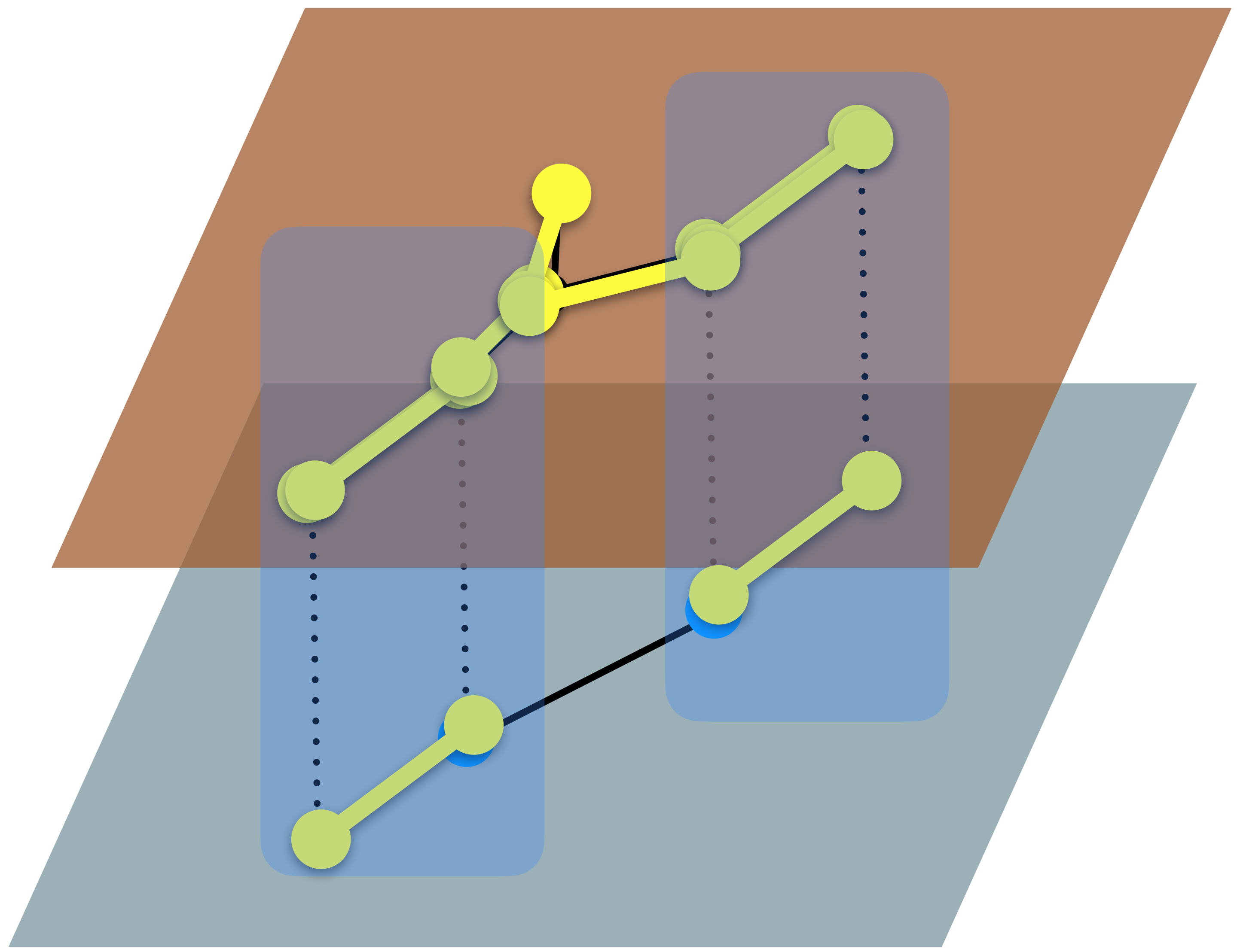


"counting" **after** rewriting

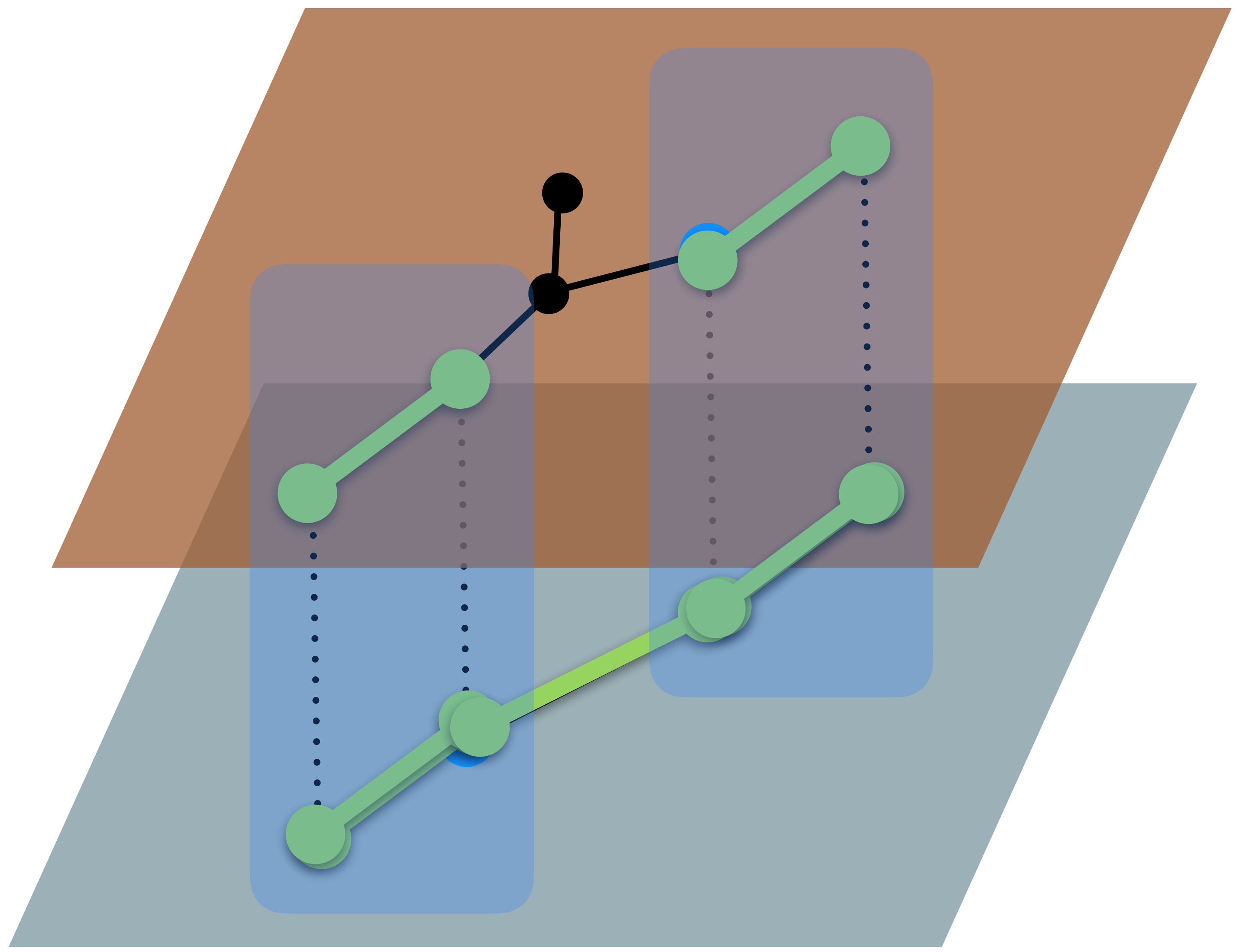


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

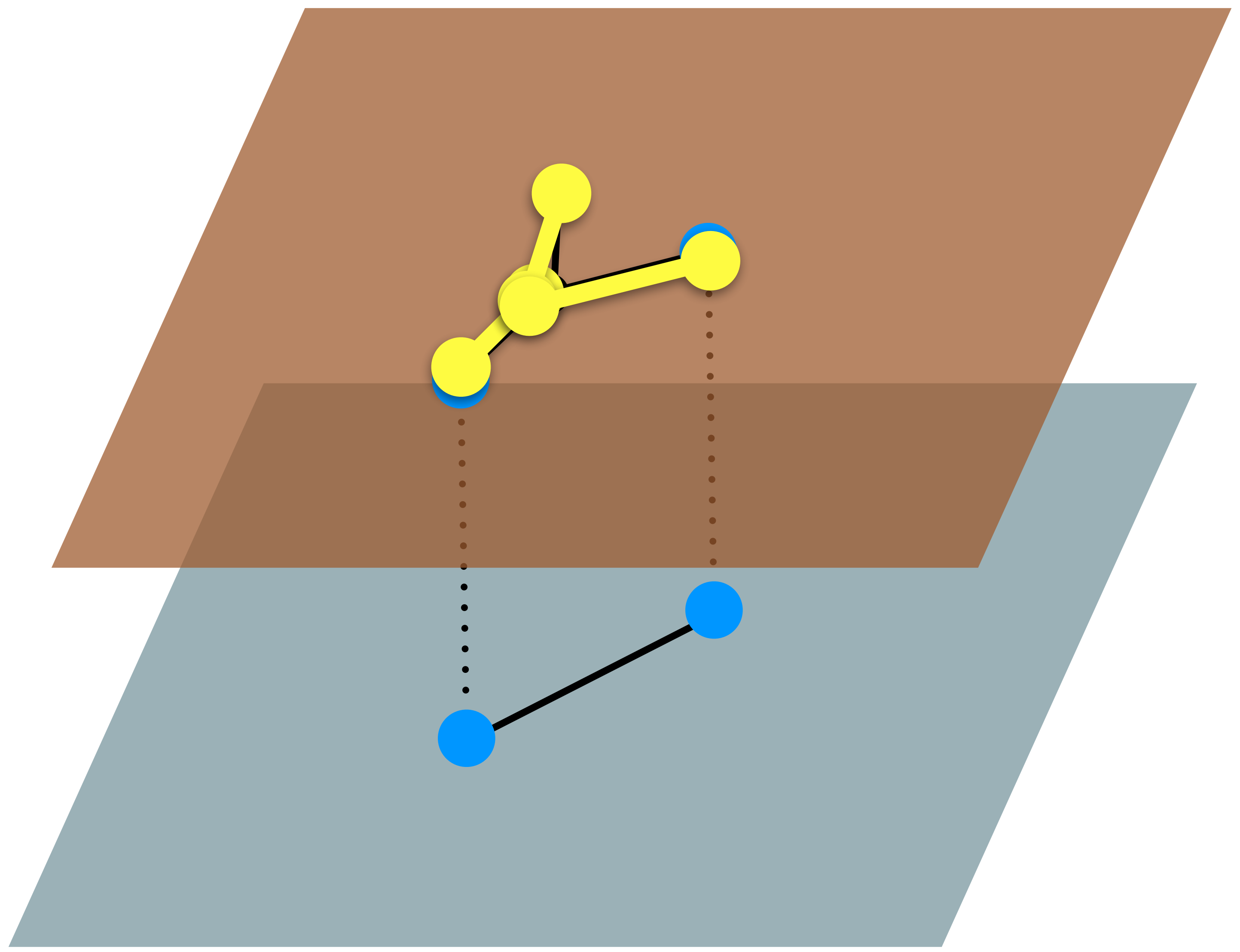


"counting" **after** rewriting

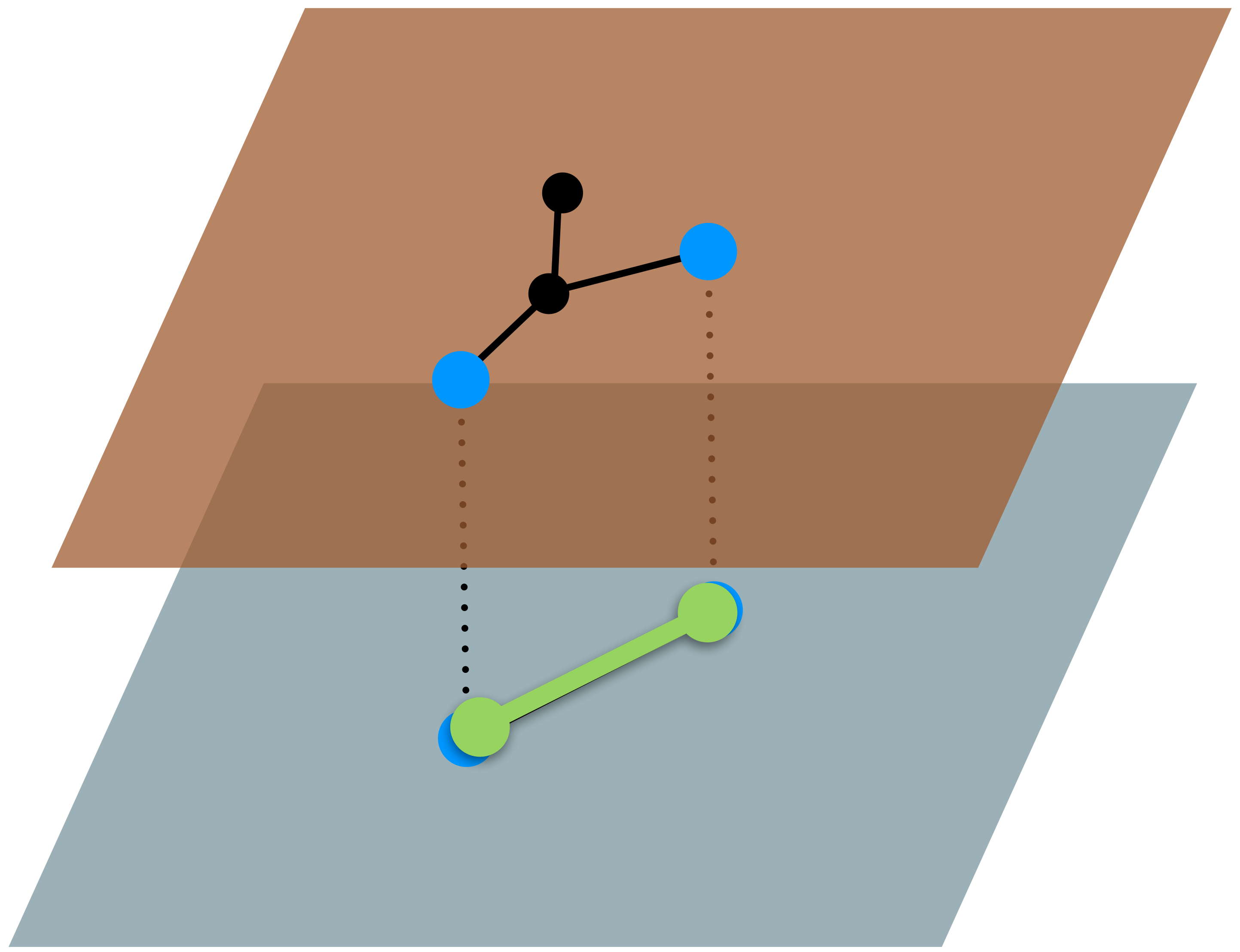


"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly



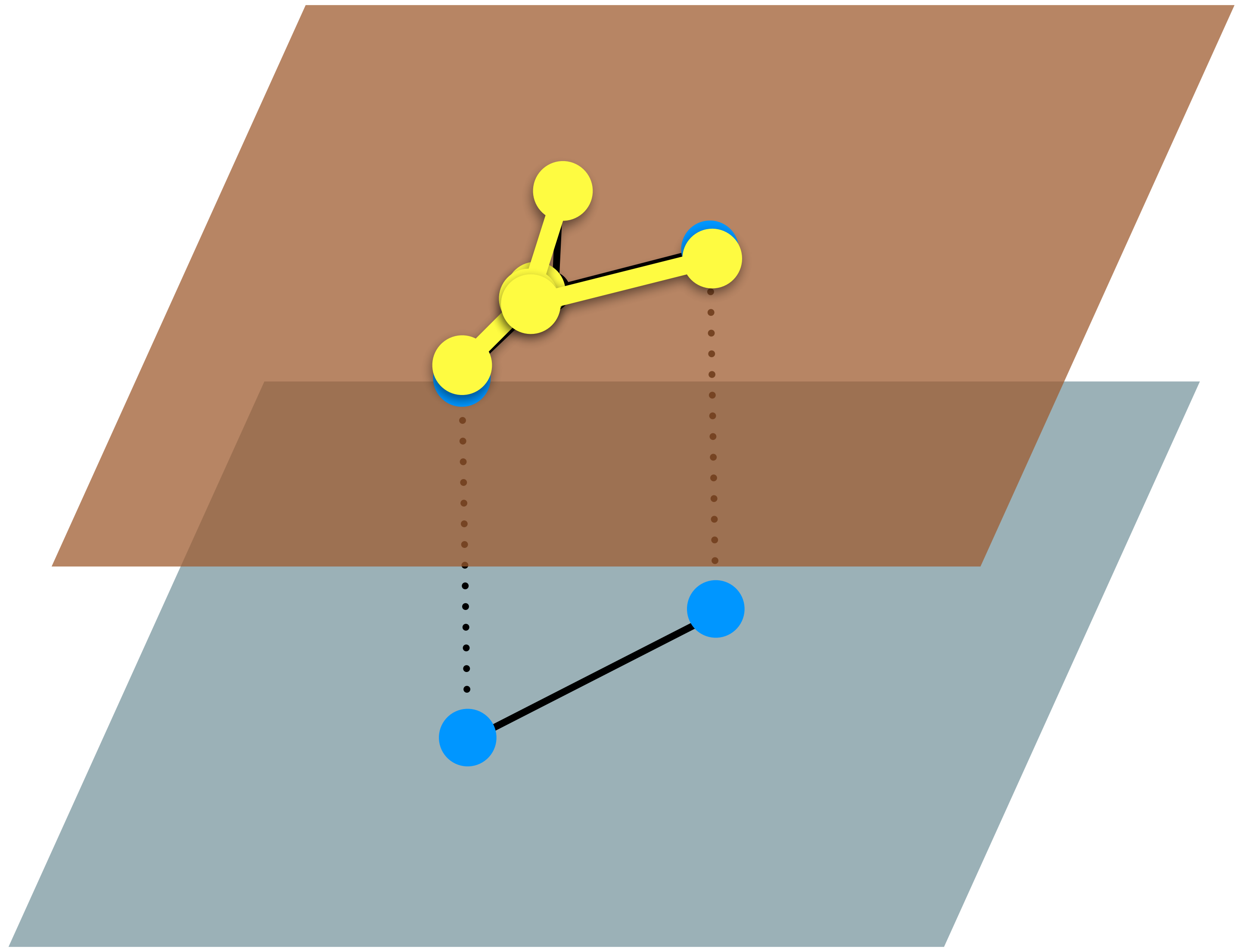
"counting" **after** rewriting



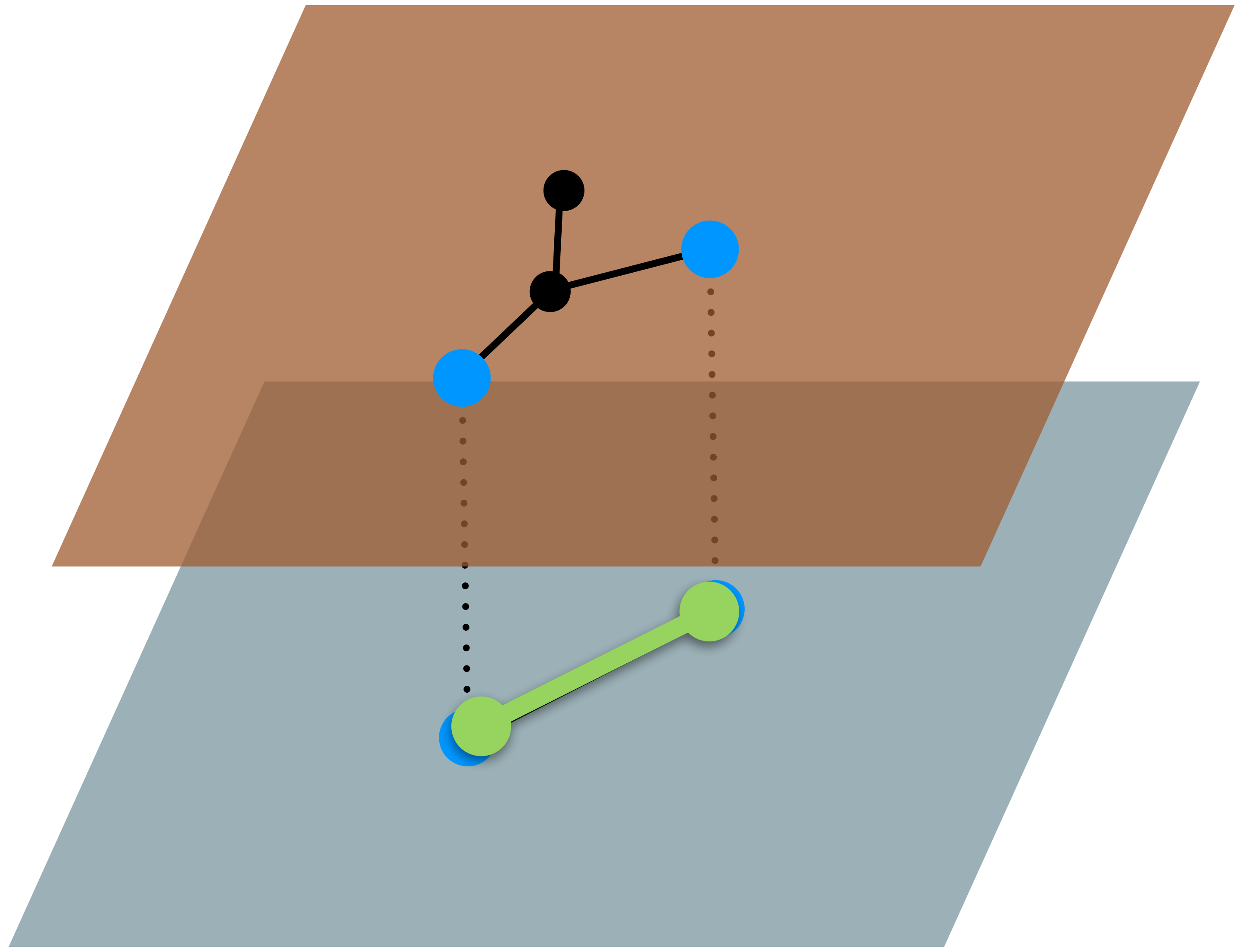
"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

3 non-trivial options



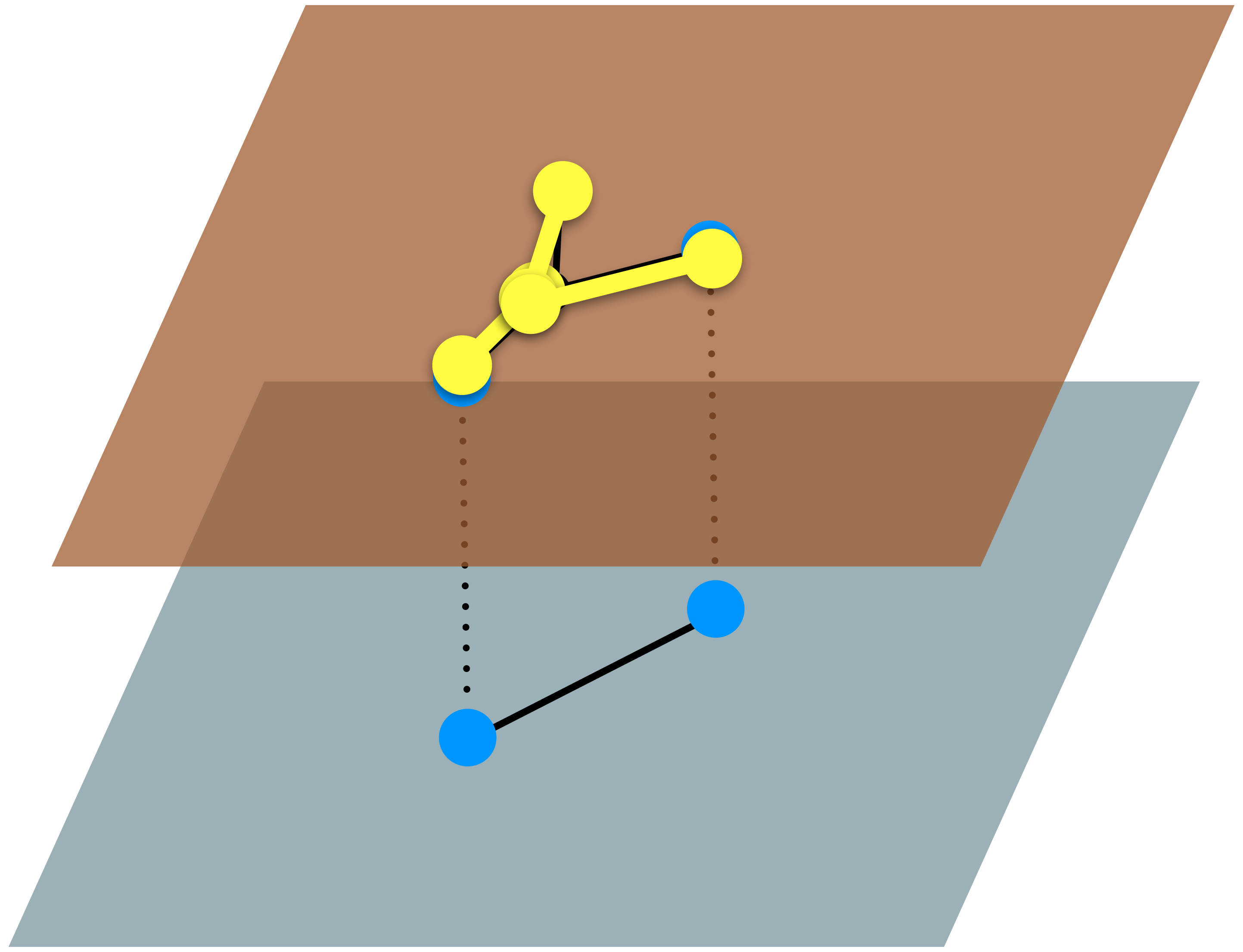
"counting" **after** rewriting



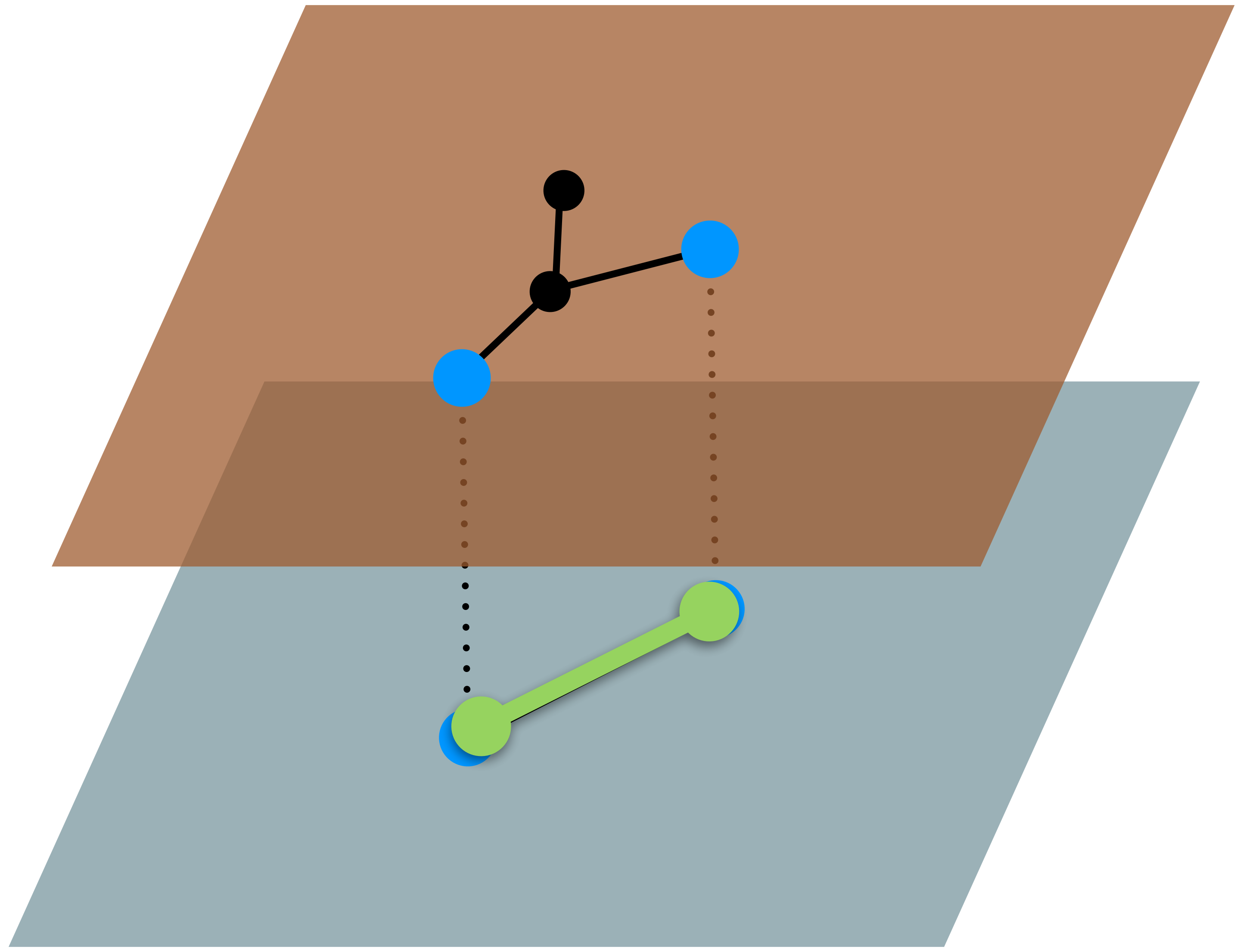
"counting" **before** rewriting

Example: generating **planar rooted binary trees (PRBTs)** uniformly

3 non-trivial options



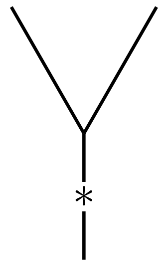
"counting" **after** rewriting

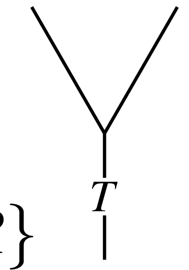


1 non-trivial option

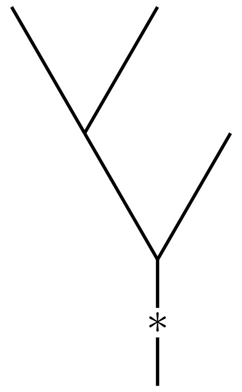
"counting" **before** rewriting

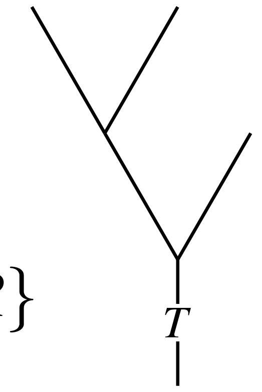
Example: generating **planar rooted binary trees (PRBTs)** uniformly

$\hat{O}_{P1} :=$ 

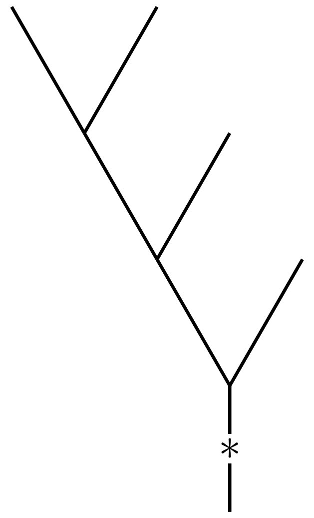
$\equiv \sum_{T\in\{I,L,R\}}$ 

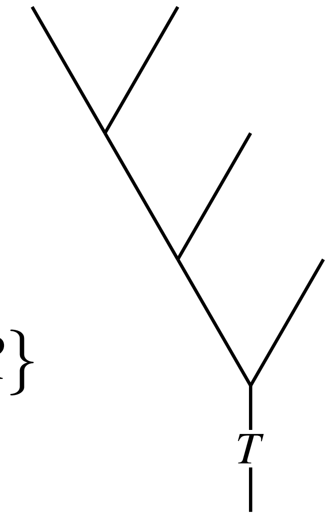
,

$\hat{O}_{P2} :=$ 

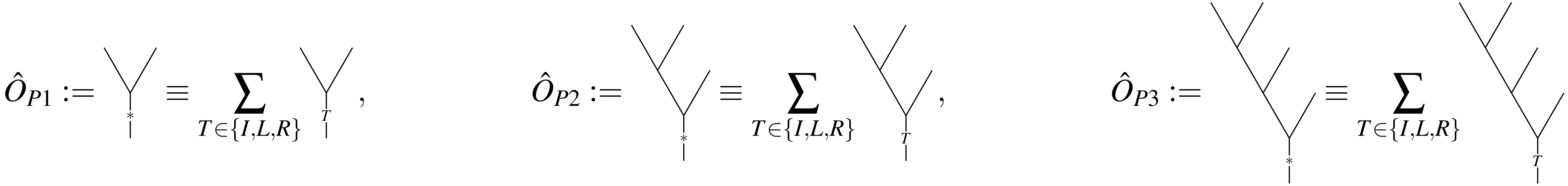
$\equiv \sum_{T\in\{I,L,R\}}$ 

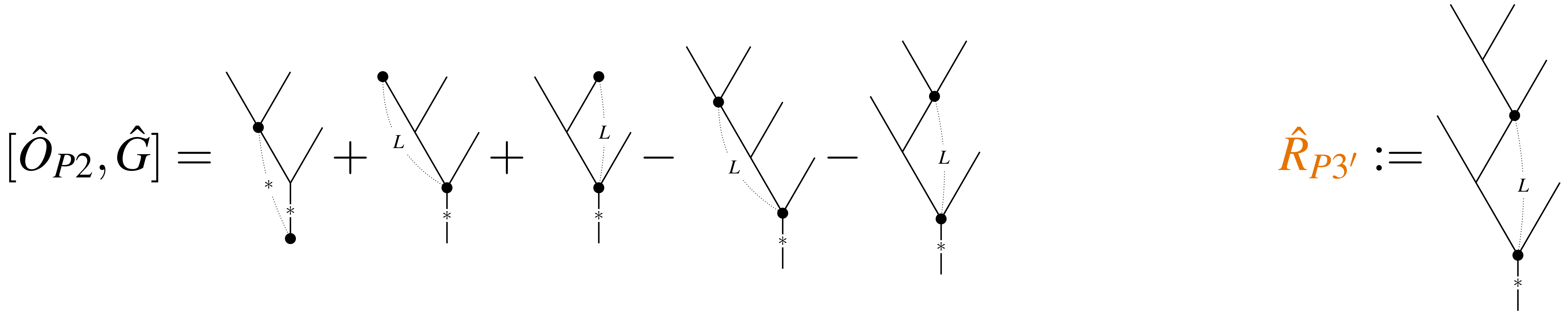
,

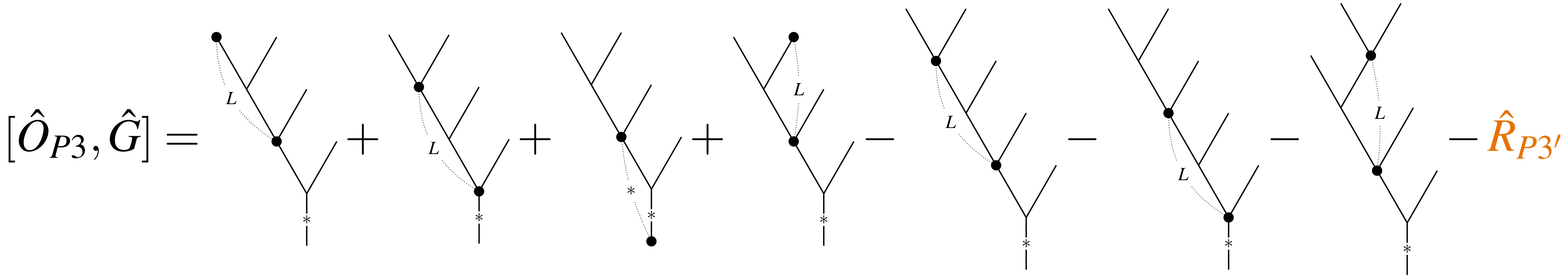
$\hat{O}_{P3} :=$ 

$\equiv \sum_{T\in\{I,L,R\}}$ 

Example: generating **planar rooted binary trees (PRBTs)** uniformly





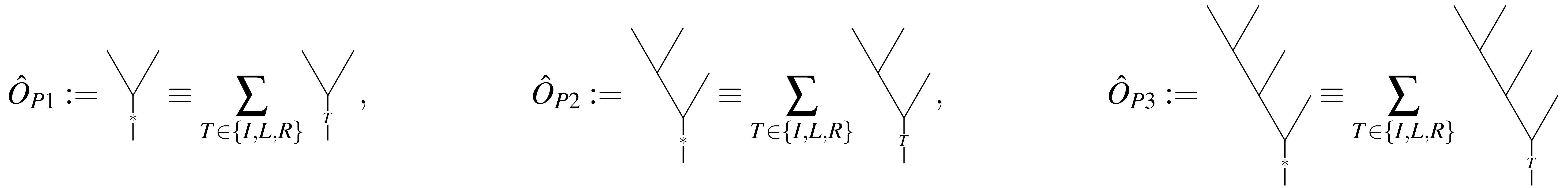


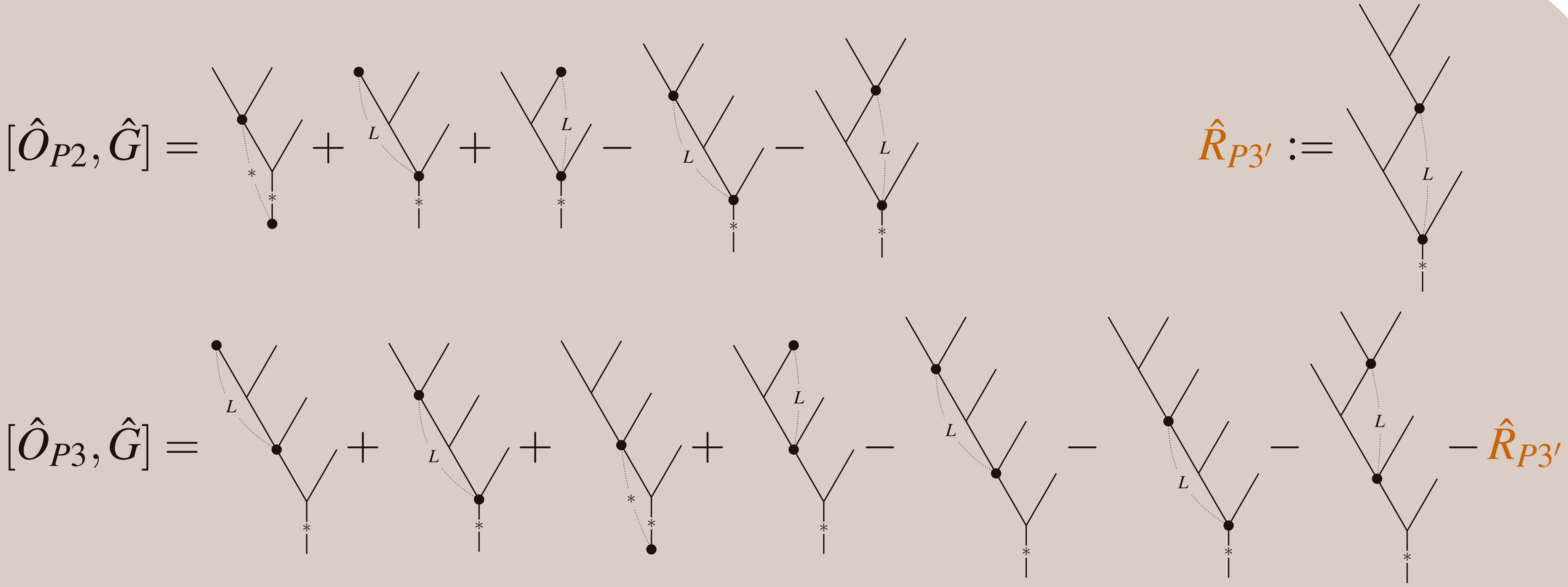
$$[\hat{O}_{P2}, [\hat{O}_{P2}, \hat{G}]] = [\hat{O}_{P2}, \hat{G}], \quad [\hat{O}_{P2}, [\hat{O}_{P3}, \hat{G}]] = [\hat{O}_{P3}, \hat{G}] + \hat{R}_{P3}$$

$$[\hat{O}_{P3}, [\hat{O}_{P3}, \hat{G}]] = [\hat{O}_{P3}, \hat{G}] + 2\hat{R}_{P3'}, \quad [\hat{O}_{P2}, \hat{R}_{P3'}] = 0, \quad [\hat{O}_{P3}, \hat{R}_{P3'}] = -\hat{R}_{P3'}$$

$$\langle| [\hat{O}_{P2}, \hat{G}] = \langle| (3\hat{O}_{P1} - 2\hat{O}_{P2}), \quad \langle| [\hat{O}_{P3}, \hat{G}] = \langle| (4\hat{O}_{P2} - 3\hat{O}_{P3}), \quad \langle| \hat{R}_{P3'} = \langle| \hat{O}_{P3}$$

Example: generating **planar rooted binary trees (PRBTs)** uniformly





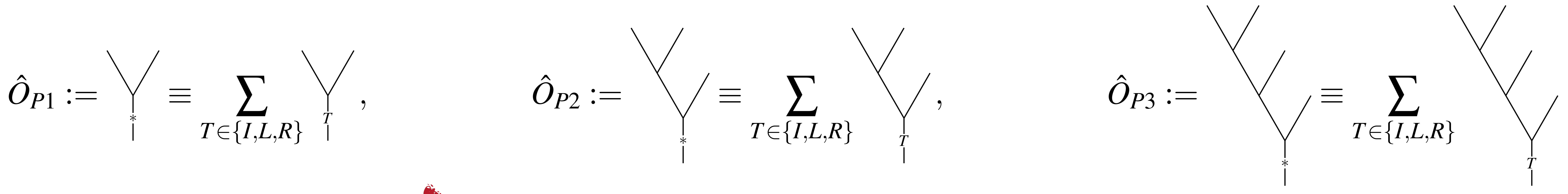
$$[\hat{O}_{P2}, [\hat{O}_{P2}, \hat{G}]] = [\hat{O}_{P2}, \hat{G}]\,, \quad [\hat{O}_{P2}, [\hat{O}_{P3}, \hat{G}]] = [\hat{O}_{P3}, \hat{G}] + \hat{R}_{P3}$$

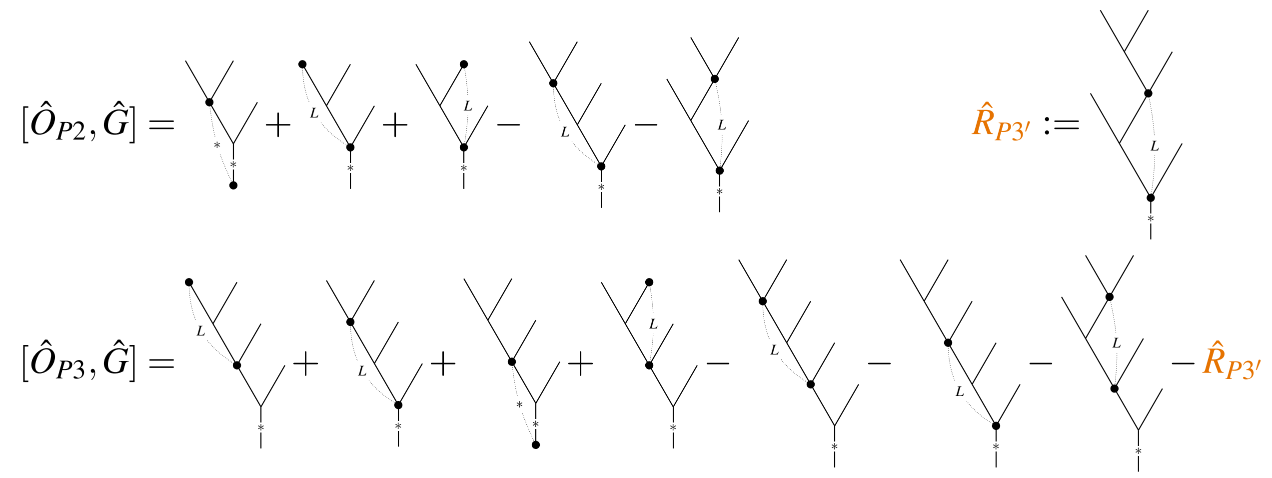
$$[\hat{O}_{P3}, [\hat{O}_{P3}, \hat{G}]] = [\hat{O}_{P3}, \hat{G}] + 2\hat{R}_{P3'}\,, \quad [\hat{O}_{P2}, \hat{R}_{P3'}] = 0\,, \quad [\hat{O}_{P3}, \hat{R}_{P3'}] = -\hat{R}_{P3'}$$

$$\langle| [\hat{O}_{P2}, \hat{G}] = \langle| (3\hat{O}_{P1} - 2\hat{O}_{P2})\,, \quad \langle| [\hat{O}_{P3}, \hat{G}] = \langle| (4\hat{O}_{P2} - 3\hat{O}_{P3})\,, \quad \langle| \hat{R}_{P3'} = \langle| \hat{O}_{P3}$$

?

Example: generating **planar rooted binary trees (PRBTs)** uniformly





$$[\hat{O}_{P2},[\hat{O}_{P2},\hat{G}]] = [\hat{O}_{P2},\hat{G}]\,,\quad [\hat{O}_{P2},[\hat{O}_{P3},\hat{G}]] = [\hat{O}_{P3},\hat{G}] + \hat{R}_{P3}$$

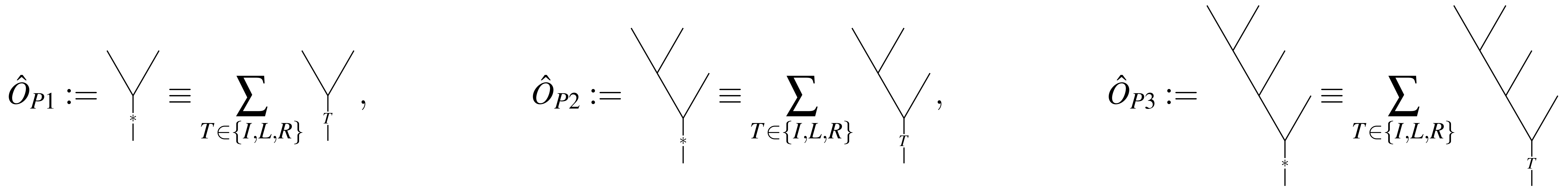
$$[\hat{O}_{P3},[\hat{O}_{P3},\hat{G}]] = [\hat{O}_{P3},\hat{G}] + 2\hat{R}_{P3'}\,,\quad [\hat{O}_{P2},\hat{R}_{P3'}] = 0\,,\quad [\hat{O}_{P3},\hat{R}_{P3'}] = -\hat{R}_{P3'}$$

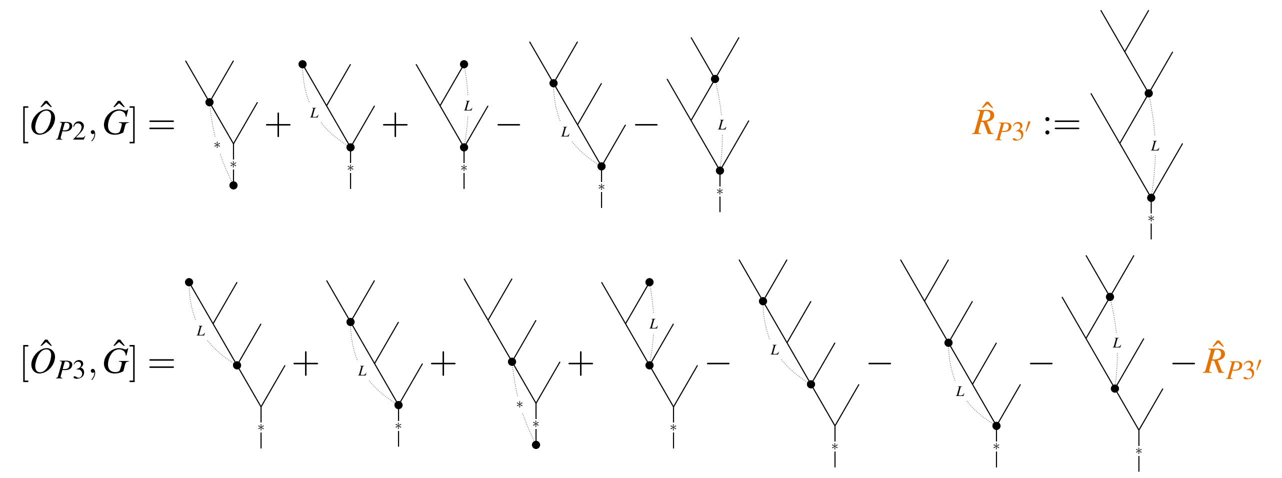
$$\langle|\,[\hat{O}_{P2},\hat{G}] = \langle|\,(3\hat{O}_{P1} - 2\hat{O}_{P2})\,,\quad \langle|\,[\hat{O}_{P3},\hat{G}] = \langle|\,(4\hat{O}_{P2} - 3\hat{O}_{P3})\,,\quad \langle|\,\hat{R}_{P3'} = \langle|\,\hat{O}_{P3}$$

$$\mathscr{G}(\lambda;\underline{\omega}) := \langle|\, e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}\,|\,\rangle\,,\quad \underline{\omega}\cdot\underline{\hat{O}} := \varepsilon\hat{O}_E + \gamma\hat{O}_{P1} + \mu\hat{O}_{P2} + \nu\hat{O}_{P3}$$

$$\begin{aligned}
\frac{\partial}{\partial\lambda}\mathscr{G}(\lambda;\underline{\omega}) &= \langle|\left(e^{ad_{\underline{\omega}\cdot\underline{\hat{O}}}}(\hat{G})\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}\,|\,\rangle \overset{(*)}{=} \langle|\left(e^{ad_{\nu\hat{O}_{P3}}}\left(e^{ad_{\mu\hat{O}_{P2}}}\left(e^{ad_{\varepsilon\hat{O}_E+\gamma\hat{O}_{P1}}}(\hat{G})\right)\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}\,|\,\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\left(e^{ad_{\nu\hat{O}_{P3}}}\left(e^{ad_{\mu\hat{O}_{P2}}}(\hat{G})\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}\,|\,\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\left(e^{ad_{\nu\hat{O}_{P3}}}\left(\hat{G} + (e^{\mu}-1)[\hat{O}_{P2},\hat{G}]\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}\,|\,\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\,\big(\hat{G} + (e^{\mu}-1)[\hat{O}_{P2},\hat{G}]\\
&\qquad + e^{\mu}(e^{\nu}-1)[\hat{O}_{P3},\hat{G}] + (e^{\nu}-1)(e^{\mu}-e^{-\nu})\hat{R}_{P3'}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}\,|\,\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\,\big(2\hat{O}_E + 3(e^{\mu}-1)\hat{O}_{P1} + (4e^{\mu+\nu} - 6e^{\mu} + 2)\hat{O}_{P2}\\
&\qquad + (3e^{\mu} + e^{-\nu} - 3e^{\mu+\nu} - 1)\hat{O}_{P3}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}\,|\,\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\,\big(2\tfrac{\partial}{\partial\varepsilon} + 3(e^{\mu}-1)\tfrac{\partial}{\partial\gamma} + (4e^{\mu+\nu} - 6e^{\mu} + 2)\tfrac{\partial}{\partial\mu}\\
&\qquad + (3e^{\mu} + e^{-\nu} - 3e^{\mu+\nu} - 1)\tfrac{\partial}{\partial\nu}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}\,|\,\rangle
\end{aligned}$$

Example: generating **planar rooted binary trees (PRBTs)** uniformly





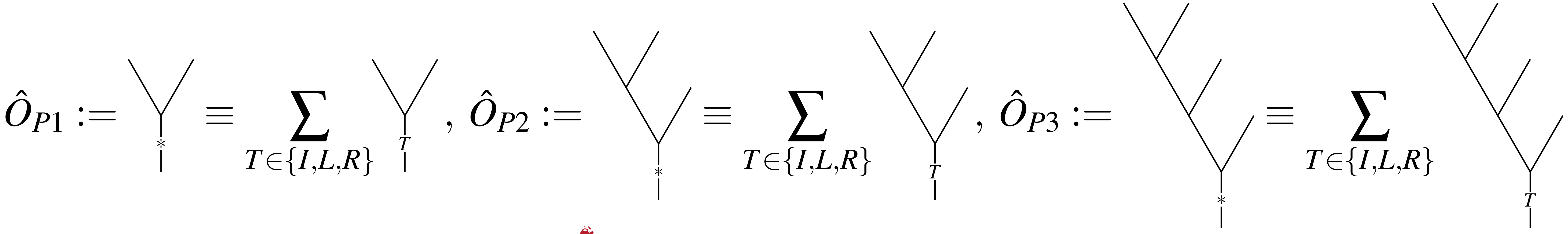
$$[\hat{O}_{P2},[\hat{O}_{P2},\hat{G}]] = [\hat{O}_{P2},\hat{G}]\,,\quad [\hat{O}_{P2},[\hat{O}_{P3},\hat{G}]] = [\hat{O}_{P3},\hat{G}] + \hat{R}_{P3}$$
$$[\hat{O}_{P3},[\hat{O}_{P3},\hat{G}]] = [\hat{O}_{P3},\hat{G}] + 2\hat{R}_{P3'}\,,\quad [\hat{O}_{P2},\hat{R}_{P3'}] = 0\,,\quad [\hat{O}_{P3},\hat{R}_{P3'}] = -\hat{R}_{P3'}$$
$$\langle| [\hat{O}_{P2},\hat{G}] = \langle|(3\hat{O}_{P1} - 2\hat{O}_{P2})\,,\quad \langle| [\hat{O}_{P3},\hat{G}] = \langle|(4\hat{O}_{P2} - 3\hat{O}_{P3})\,,\quad \langle|\hat{R}_{P3'} = \langle|\hat{O}_{P3}$$

$$\begin{aligned}
\mathscr{G}(\lambda;\underline{\omega}) &:= \langle| e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} ||\rangle\,,\quad \underline{\omega}\cdot\underline{\hat{O}} := \varepsilon\hat{O}_E + \gamma\hat{O}_{P1} + \mu\hat{O}_{P2} + \nu\hat{O}_{P3}\\
\tfrac{\partial}{\partial\lambda}\mathscr{G}(\lambda;\underline{\omega}) &= \langle|\left(e^{ad_{\underline{\omega}\cdot\underline{\hat{O}}}}(\hat{G})\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}||\rangle \overset{(*)}{=} \langle|\left(e^{ad_{\nu\hat{O}_{P3}}}\left(e^{ad_{\mu\hat{O}_{P2}}}\left(e^{ad_{\varepsilon\hat{O}_E+\gamma\hat{O}_{P1}}}(\hat{G})\right)\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}||\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\left(e^{ad_{\nu\hat{O}_{P3}}}\left(e^{ad_{\mu\hat{O}_{P2}}}(\hat{G})\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}||\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\left(e^{ad_{\nu\hat{O}_{P3}}}\left(\hat{G} + (e^{\mu}-1)[\hat{O}_{P2},\hat{G}]\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}||\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\big(\hat{G} + (e^{\mu}-1)[\hat{O}_{P2},\hat{G}]\\
&\qquad + e^{\mu}(e^{\nu}-1)[\hat{O}_{P3},\hat{G}] + (e^{\nu}-1)(e^{\mu}-e^{-\nu})\hat{R}_{P3'}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}||\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\big(2\hat{O}_E + 3(e^{\mu}-1)\hat{O}_{P1} + (4e^{\mu+\nu} - 6e^{\mu} + 2)\hat{O}_{P2}\\
&\qquad + (3e^{\mu} + e^{-\nu} - 3e^{\mu+\nu} - 1)\hat{O}_{P3}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}||\rangle\\
&= e^{2\varepsilon+\gamma}\langle|\big(2\tfrac{\partial}{\partial\varepsilon} + 3(e^{\mu}-1)\tfrac{\partial}{\partial\gamma} + (4e^{\mu+\nu} - 6e^{\mu} + 2)\tfrac{\partial}{\partial\mu}\\
&\qquad + (3e^{\mu} + e^{-\nu} - 3e^{\mu+\nu} - 1)\tfrac{\partial}{\partial\nu}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}}||\rangle
\end{aligned}$$

Granted that the derivation of the evolution equation for $\mathscr{G}(\lambda;\underline{\omega})$ is somewhat involved, one may extract from it a very interesting insight via a transformation of variables $\omega_i \to \ln x_i$ (which entails that $\frac{\partial}{\partial\omega_i} \to x_i\frac{\partial}{\partial x_i}$), and collecting coefficients for the operators $\hat{n}_i := x_i\frac{\partial}{\partial x_i}$:

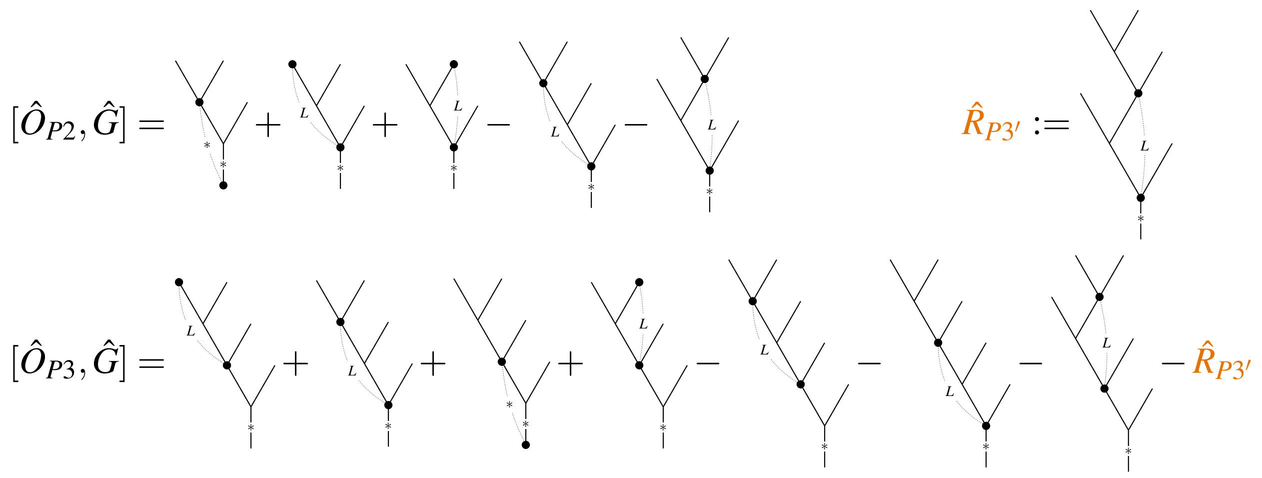
$$\begin{aligned}
&\tfrac{\partial}{\partial\lambda}\mathscr{G}(\lambda;\underline{\ln x}) = \hat{\mathsf{D}}\mathscr{G}(\lambda;\underline{\ln x})\\
&\hat{\mathsf{D}} = x_{\varepsilon}^2 x_{\nu}(2\hat{n}_{\varepsilon} - 3\hat{n}_{\gamma} + 2\hat{n}_{\mu} - \hat{n}_{\nu}) + x_{\varepsilon}^2 x_{\nu} x_{\mu}(3\hat{n}_{\gamma} - 6\hat{n}_{\mu} + 3\hat{n}_{\nu}) + x_{\varepsilon}^2 x_{\nu} x_{\mu}^2(4\hat{n}_{\mu} - 3\hat{n}_{\nu}) + x_{\varepsilon}^2\hat{n}_{\nu}
\end{aligned}\tag{58}$$

Example: generating **planar rooted binary trees (PRBTs)** uniformly



$$\mathscr{G}(\lambda;\underline{\omega}) := \langle| e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle\,, \quad \underline{\omega}\cdot\underline{\hat{O}} := \varepsilon\hat{O}_E + \gamma\hat{O}_{P1} + \mu\hat{O}_{P2} + \nu\hat{O}_{P3}$$

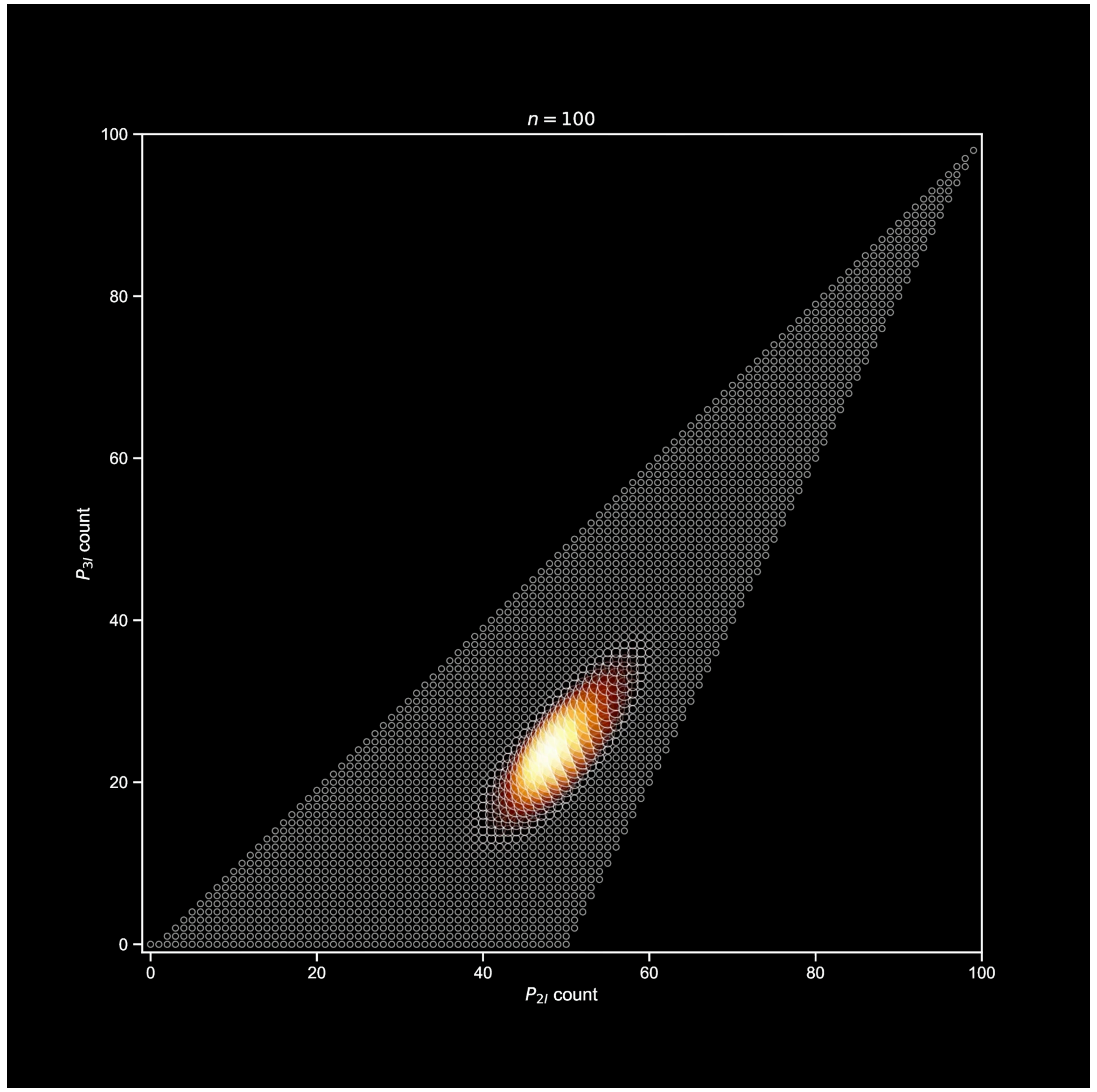
$$\begin{aligned}
\tfrac{\partial}{\partial\lambda}\mathscr{G}(\lambda;\underline{\omega}) &= \langle| \left(e^{ad_{\underline{\omega}\cdot\underline{\hat{O}}}}(\hat{G})\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \overset{(*)}{=} \langle| \left(e^{ad_{\nu\hat{O}_{P3}}}\left(e^{ad_{\mu\hat{O}_{P2}}}\left(e^{ad_{\varepsilon\hat{O}_E+\gamma\hat{O}_{P1}}}(\hat{G})\right)\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma} \langle| \left(e^{ad_{\nu\hat{O}_{P3}}}\left(e^{ad_{\mu\hat{O}_{P2}}}(\hat{G})\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma} \langle| \left(e^{ad_{\nu\hat{O}_{P3}}}\left(\hat{G} + (e^{\mu}-1)[\hat{O}_{P2},\hat{G}]\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma} \langle| \big(\hat{G} + (e^{\mu}-1)[\hat{O}_{P2},\hat{G}] \\
&\qquad + e^{\mu}(e^{\nu}-1)[\hat{O}_{P3},\hat{G}] + (e^{\nu}-1)(e^{\mu}-e^{-\nu})\hat{R}_{P3'}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma} \langle| \big(2\hat{O}_E + 3(e^{\mu}-1)\hat{O}_{P1} + (4e^{\mu+\nu} - 6e^{\mu} + 2)\hat{O}_{P2} \\
&\qquad + (3e^{\mu} + e^{-\nu} - 3e^{\mu+\nu} - 1)\hat{O}_{P3}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma} \langle| \big(2\tfrac{\partial}{\partial\varepsilon} + 3(e^{\mu}-1)\tfrac{\partial}{\partial\gamma} + (4e^{\mu+\nu} - 6e^{\mu} + 2)\tfrac{\partial}{\partial\mu} \\
&\qquad + (3e^{\mu} + e^{-\nu} - 3e^{\mu+\nu} - 1)\tfrac{\partial}{\partial\nu}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle
\end{aligned}$$



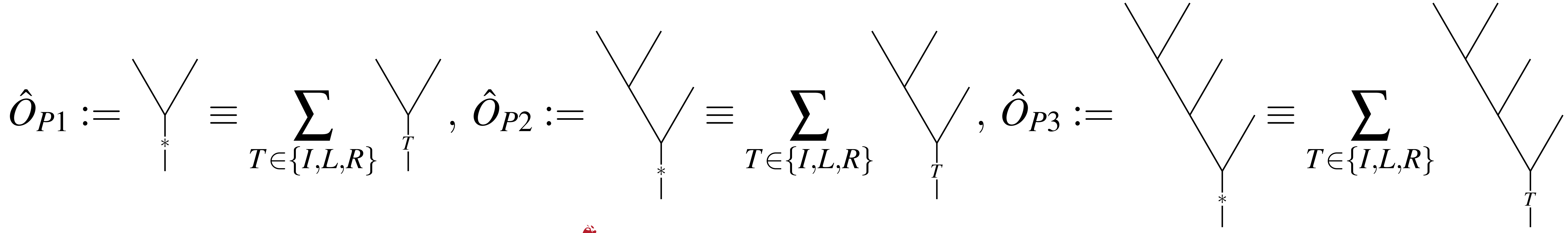
$$[\hat{O}_{P2},[\hat{O}_{P2},\hat{G}]] = [\hat{O}_{P2},\hat{G}]\,, \quad [\hat{O}_{P2},[\hat{O}_{P3},\hat{G}]] = [\hat{O}_{P3},\hat{G}] + \hat{R}_{P3}$$

$$[\hat{O}_{P3},[\hat{O}_{P3},\hat{G}]] = [\hat{O}_{P3},\hat{G}] + 2\hat{R}_{P3'}\,, \quad [\hat{O}_{P2},\hat{R}_{P3'}] = 0\,, \quad [\hat{O}_{P3},\hat{R}_{P3'}] = -\hat{R}_{P3'}$$

$$\langle|\,[\hat{O}_{P2},\hat{G}] = \langle|\,(3\hat{O}_{P1} - 2\hat{O}_{P2})\,, \quad \langle|\,[\hat{O}_{P3},\hat{G}] = \langle|\,(4\hat{O}_{P2} - 3\hat{O}_{P3})\,, \quad \langle|\,\hat{R}_{P3'} = \langle|\,\hat{O}_{P3}$$

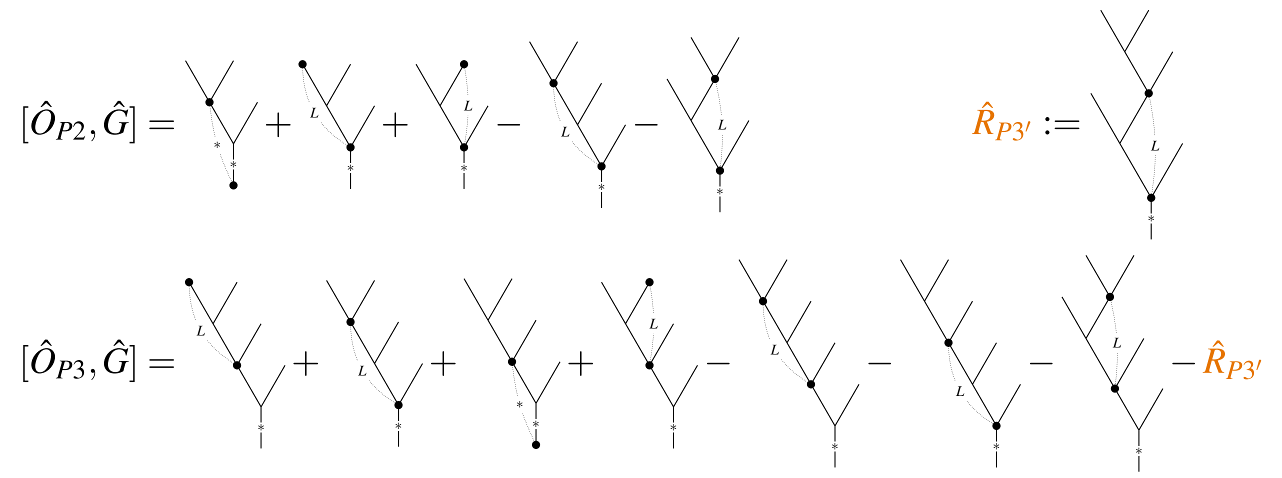


Example: generating **planar rooted binary trees (PRBTs)** uniformly



$$\mathscr{G}(\lambda;\underline{\omega}) := \langle| e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle\,, \quad \underline{\omega}\cdot\underline{\hat{O}} := \varepsilon\hat{O}_E + \gamma\hat{O}_{P1} + \mu\hat{O}_{P2} + \nu\hat{O}_{P3}$$

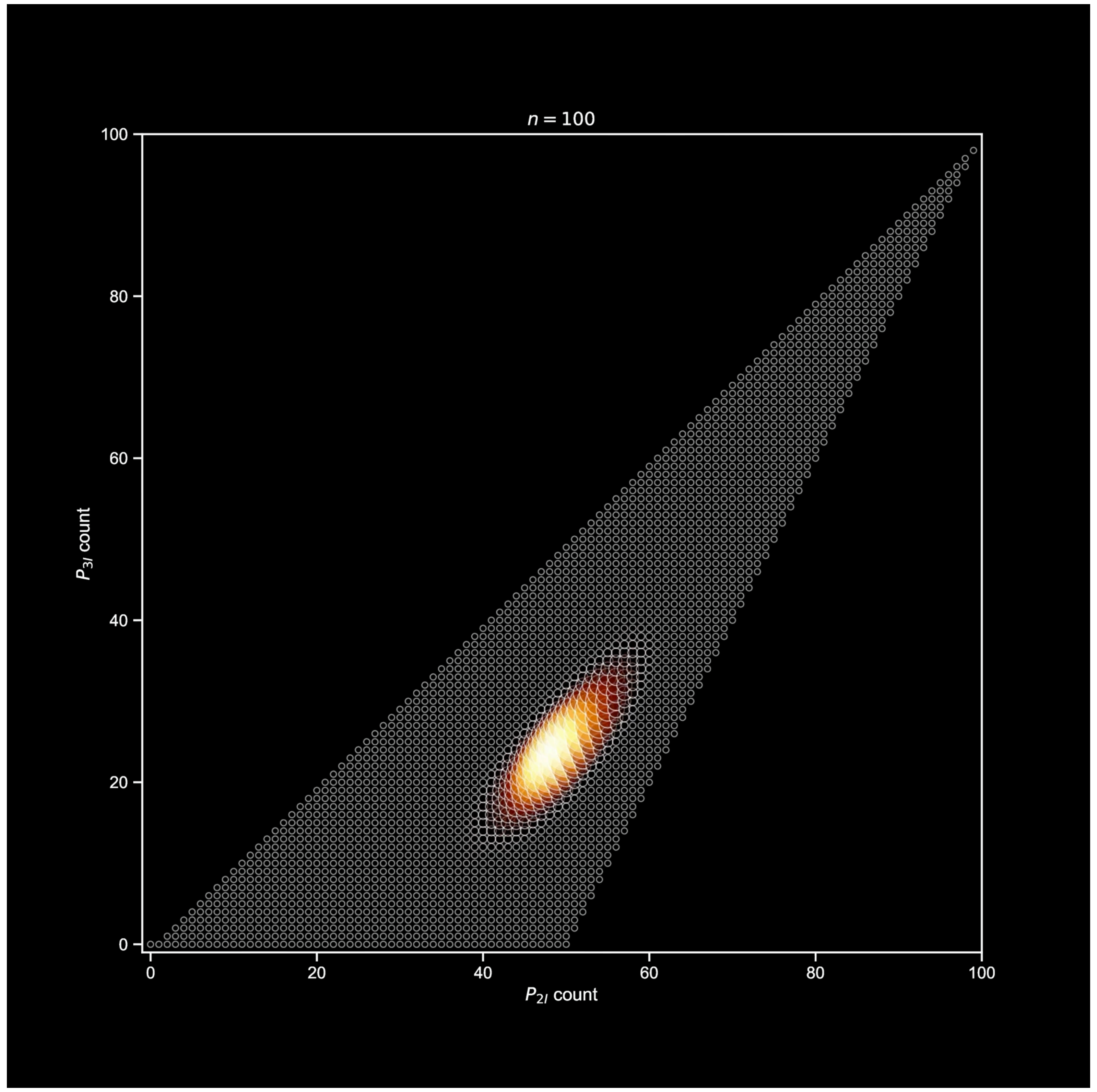
$$\begin{aligned}
\tfrac{\partial}{\partial\lambda}\mathscr{G}(\lambda;\underline{\omega}) &= \langle| \left(e^{ad_{\underline{\omega}\cdot\underline{\hat{O}}}}(\hat{G})\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \overset{(*)}{=} \langle| \left(e^{ad_{\nu\hat{O}_{P3}}}\left(e^{ad_{\mu\hat{O}_{P2}}}\left(e^{ad_{\varepsilon\hat{O}_E+\gamma\hat{O}_{P1}}}(\hat{G})\right)\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma}\langle| \left(e^{ad_{\nu\hat{O}_{P3}}}\left(e^{ad_{\mu\hat{O}_{P2}}}(\hat{G})\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma}\langle| \left(e^{ad_{\nu\hat{O}_{P3}}}\left(\hat{G} + (e^{\mu}-1)[\hat{O}_{P2},\hat{G}]\right)\right) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma}\langle| \big(\hat{G} + (e^{\mu}-1)[\hat{O}_{P2},\hat{G}] \\
&\qquad + e^{\mu}(e^{\nu}-1)[\hat{O}_{P3},\hat{G}] + (e^{\nu}-1)(e^{\mu}-e^{-\nu})\hat{R}_{P3'}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma}\langle| \big(2\hat{O}_E + 3(e^{\mu}-1)\hat{O}_{P1} + (4e^{\mu+\nu} - 6e^{\mu} + 2)\hat{O}_{P2} \\
&\qquad + (3e^{\mu} + e^{-\nu} - 3e^{\mu+\nu} - 1)\hat{O}_{P3}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle \\
&= e^{2\varepsilon+\gamma}\langle| \big(2\tfrac{\partial}{\partial\varepsilon} + 3(e^{\mu}-1)\tfrac{\partial}{\partial\gamma} + (4e^{\mu+\nu} - 6e^{\mu} + 2)\tfrac{\partial}{\partial\mu} \\
&\qquad + (3e^{\mu} + e^{-\nu} - 3e^{\mu+\nu} - 1)\tfrac{\partial}{\partial\nu}\big) e^{\underline{\omega}\cdot\underline{\hat{O}}} e^{\lambda\hat{G}} |\,|\rangle
\end{aligned}$$



$$[\hat{O}_{P2},[\hat{O}_{P2},\hat{G}]] = [\hat{O}_{P2},\hat{G}]\,, \quad [\hat{O}_{P2},[\hat{O}_{P3},\hat{G}]] = [\hat{O}_{P3},\hat{G}] + \hat{R}_{P3}$$

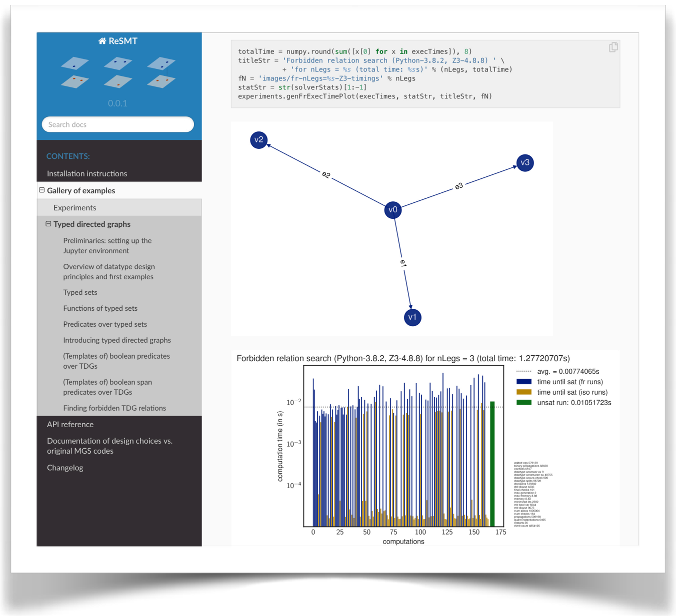
$$[\hat{O}_{P3},[\hat{O}_{P3},\hat{G}]] = [\hat{O}_{P3},\hat{G}] + 2\hat{R}_{P3'}\,, \quad [\hat{O}_{P2},\hat{R}_{P3'}] = 0\,, \quad [\hat{O}_{P3},\hat{R}_{P3'}] = -\hat{R}_{P3'}$$

$$\langle|\,[\hat{O}_{P2},\hat{G}] = \langle|\,(3\hat{O}_{P1} - 2\hat{O}_{P2})\,, \quad \langle|\,[\hat{O}_{P3},\hat{G}] = \langle|\,(4\hat{O}_{P2} - 3\hat{O}_{P3})\,, \quad \langle|\,\hat{R}_{P3'} = \langle|\,\hat{O}_{P3}$$



Outlook

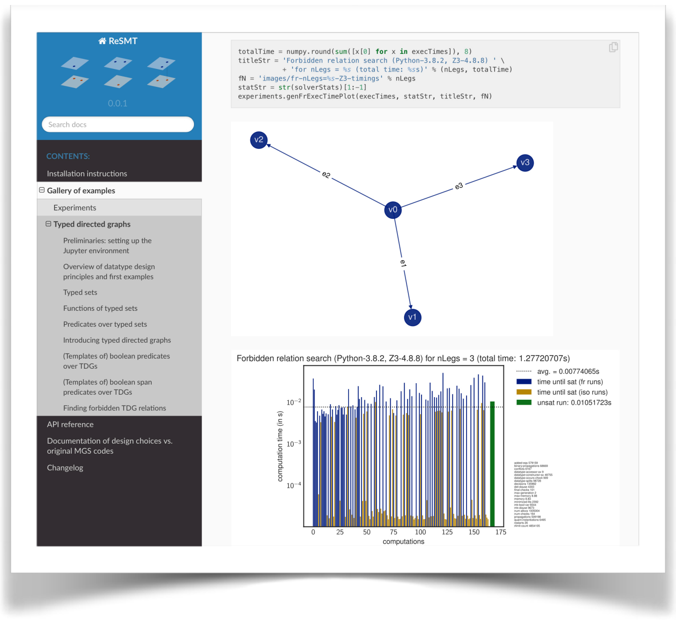
- Development of tracelet theory for analyzing **continuous-time Markov chains**
- **Algorithmic implementations** of tracelet generators and analysis methods (→**ReSMT**)
- Applications of **tracelet Hopf algebras** to **combinatorics**?



https://gitlab.com/nicolasbehr/ReSMT

Outlook

- Development of tracelet theory for analyzing **continuous-time Markov chains**
- **Algorithmic implementations** of tracelet generators and analysis methods (→**ReSMT**)
- Applications of **tracelet Hopf algebras** to **combinatorics**?



https://gitlab.com/nicolasbehr/ReSMT

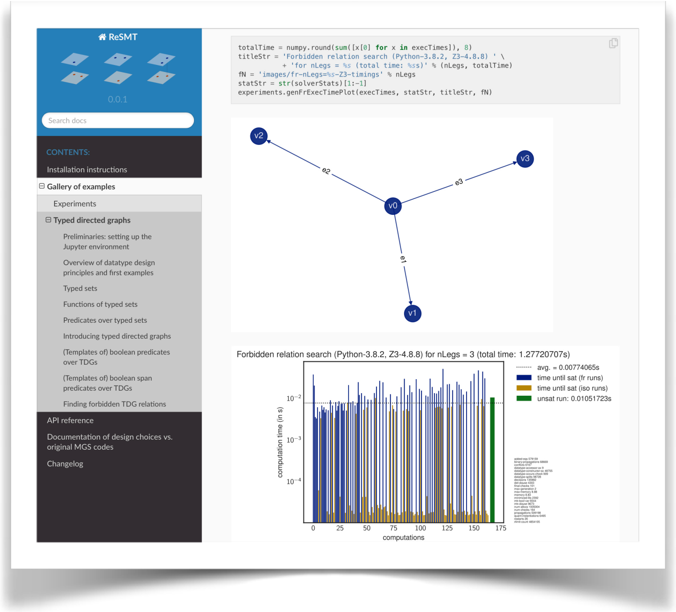
- **Long-term** perspectives**:**
  - Formalization of **categorical rewriting theory (CRT)** via proof assistants (**Coq!**)
  - **GReTA-ACT working group** on CRT starting this fall

    ⇒ *Please contact me for details if you are interested!*
  - Applications of **Grothendiek fibrations** and related concepts to CRT
  - "The 🐘 in the room": **chemical rewriting theory!**

https://www.irif.fr/~greta/

Outlook

- Development of tracelet theory for analyzing **continuous-time Markov chains**
- **Algorithmic implementations** of tracelet generators and analysis methods (→**ReSMT**)
- Applications of **tracelet Hopf algebras** to **combinatorics**?



https://gitlab.com/nicolasbehr/ReSMT

Thank you!

- **Long-term** perspectives**:**
  - Formalization of **categorical rewriting theory (CRT)** via proof assistants (**Coq!**)
  - **GReTA-ACT working group** on CRT starting this fall

    ⇒ *Please contact me for details if you are interested!*
  - Applications of **Grothendiek fibrations** and related concepts to CRT
  - "The 🐘 in the room": **chemical rewriting theory!**

https://www.irif.fr/~greta/