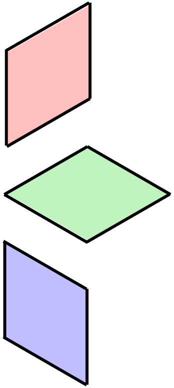
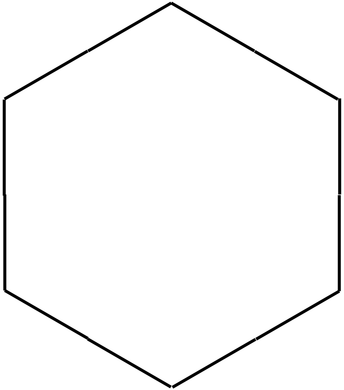
The problem of the calissons, by rewriting

Vincent van Oostrom
University of Sussex
vvo@sussex.ac.uk

The problem of the calissons (David & Tomei 89)

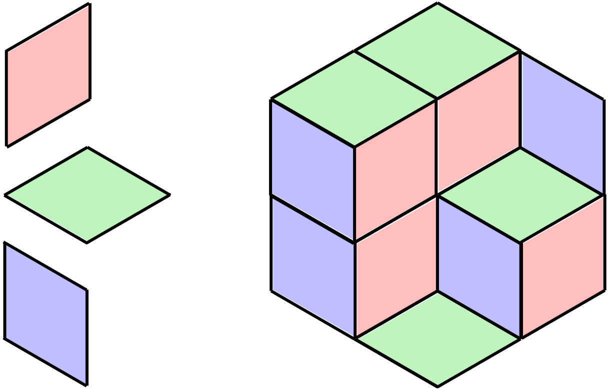


calissons

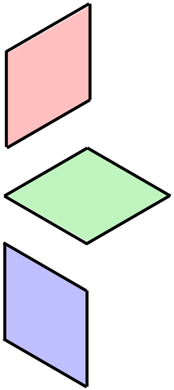


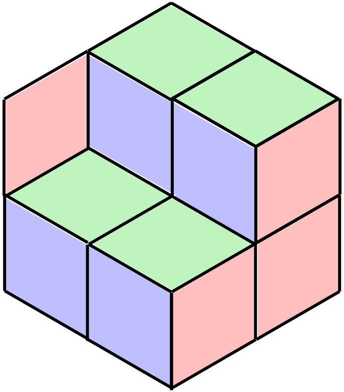
hexagonal box

The problem of the calissons



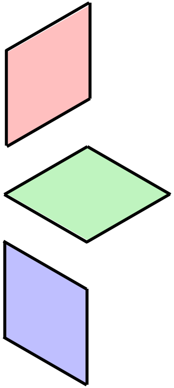
The problem of the calissons

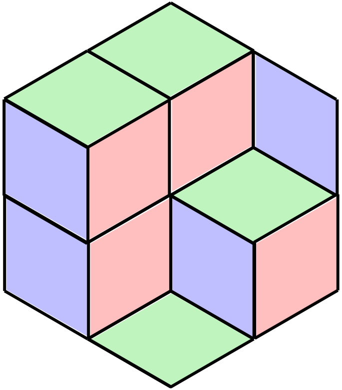




box random generator

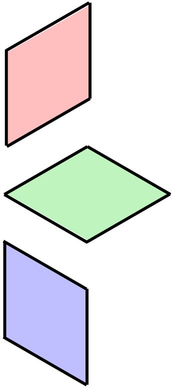
The problem of the calissons

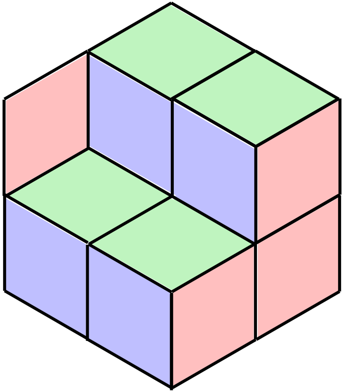




spectrum $(4, 4, 4)$

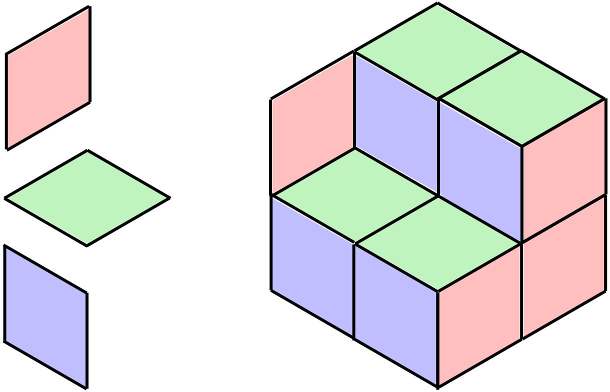
The problem of the calissons





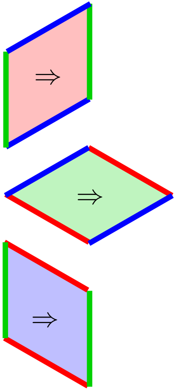
spectrum $(4, 4, 4)$

The problem of the calissons by 4 confluence techniques



same box $\Longrightarrow$ same spectrum

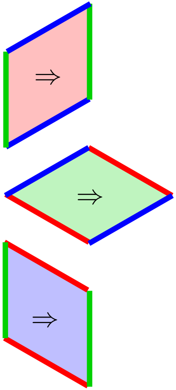
(1) random descent



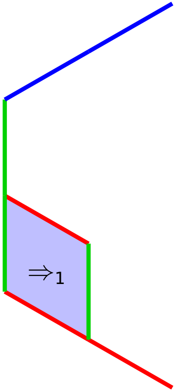
filling rules


filling

(1) random descent

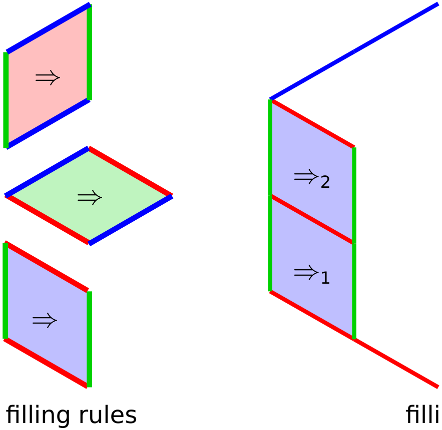


filling rules

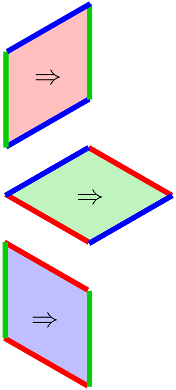


filling

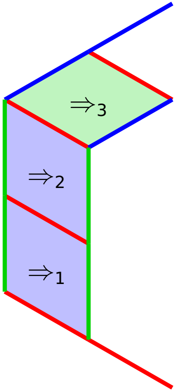
(1) random descent



(1) random descent

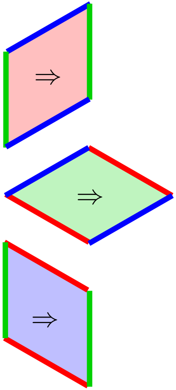


filling rules

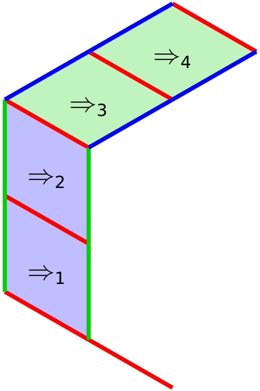


filling

(1) random descent

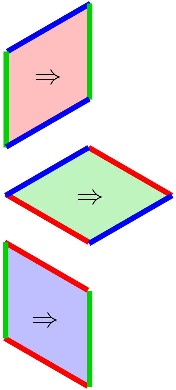


filling rules

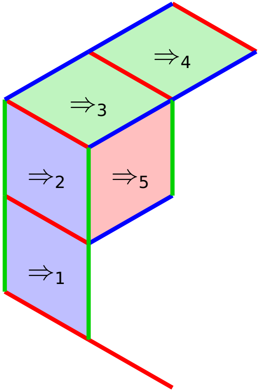


filling

(1) random descent

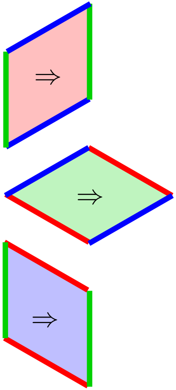


filling rules

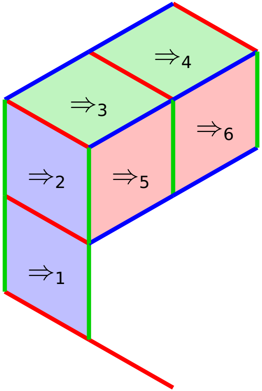


filling

(1) random descent

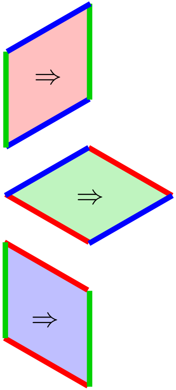


filling rules

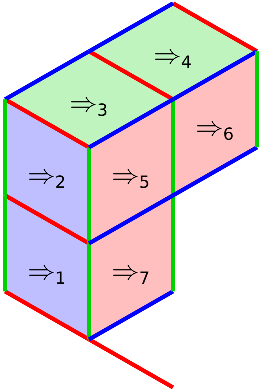


filling

(1) random descent

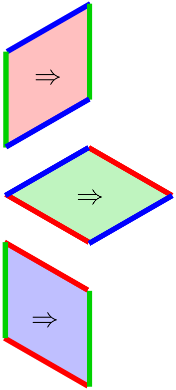


filling rules

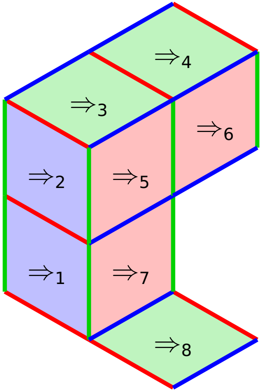


filling

(1) random descent

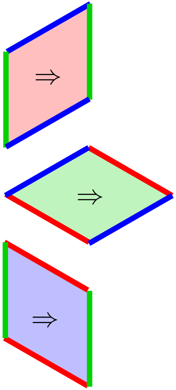


filling rules

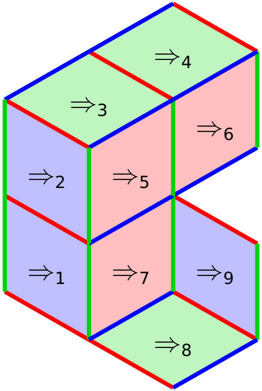


filling

(1) random descent

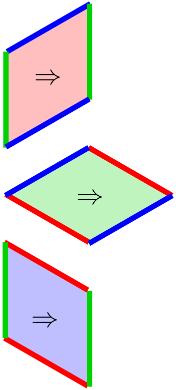


filling rules

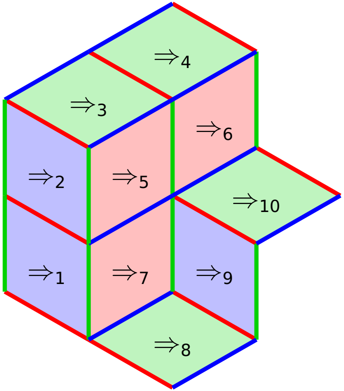


filling

(1) random descent

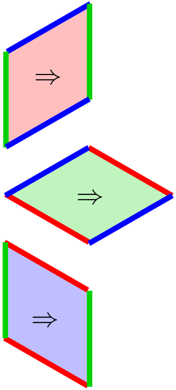


filling rules

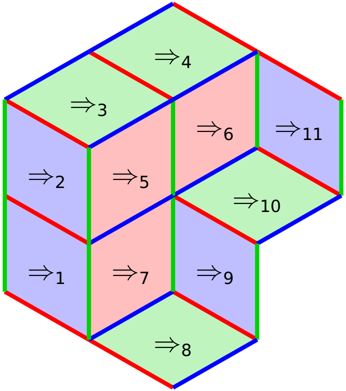


filling

(1) random descent

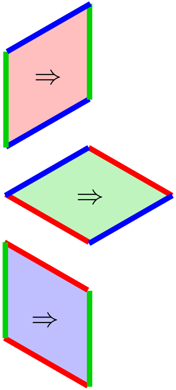


filling rules

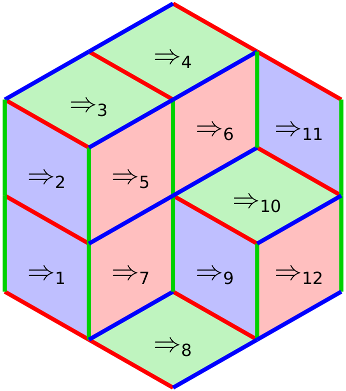


filling

(1) random descent

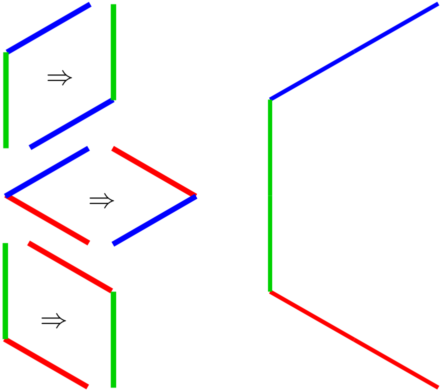


filling rules



filling

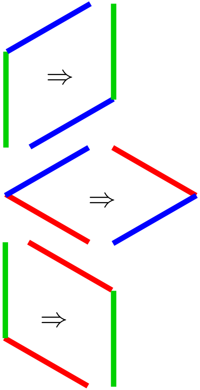
(1) random descent



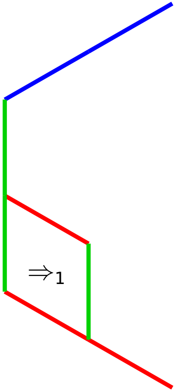
string rewrite rules

string rewrite

(1) random descent

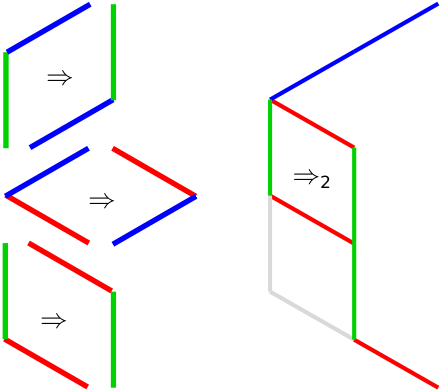


string rewrite rules



string rewrite

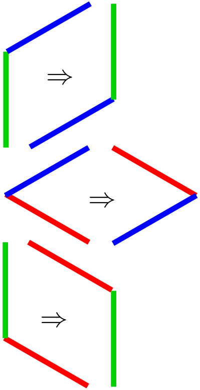
(1) random descent



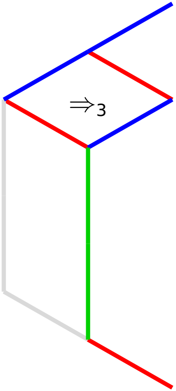
string rewrite rules

string rewrite

(1) random descent

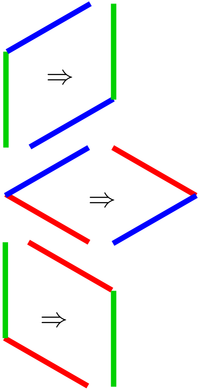


string rewrite rules

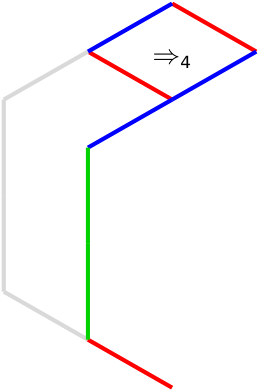


string rewrite

(1) random descent

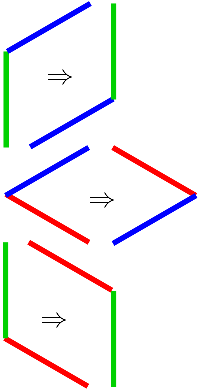


string rewrite rules

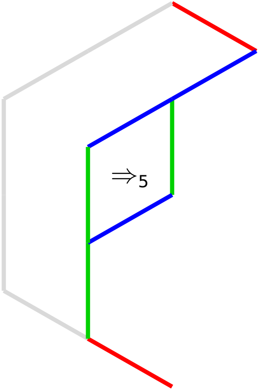


string rewrite

(1) random descent

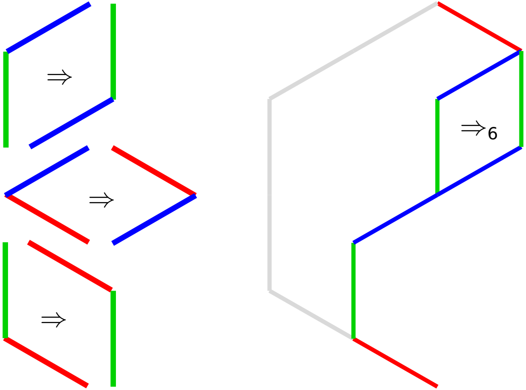


string rewrite rules



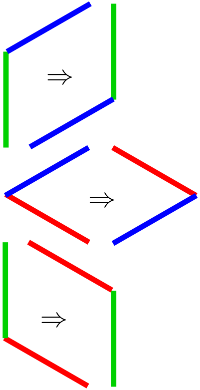
string rewrite

(1) random descent

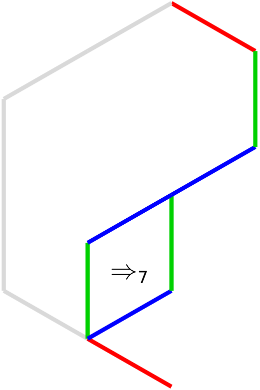


string rewrite rules string rewrite

(1) random descent

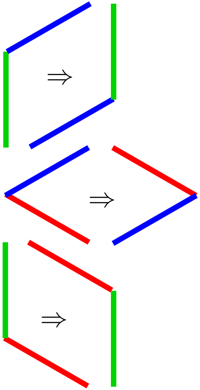


string rewrite rules

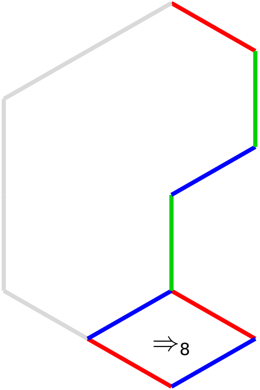


string rewrite

(1) random descent

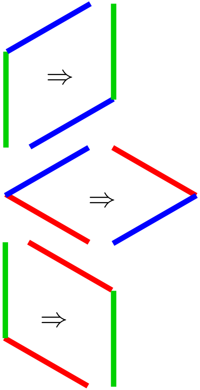


string rewrite rules

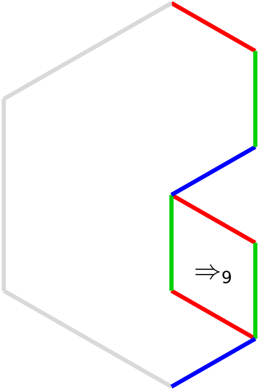


string rewrite

(1) random descent

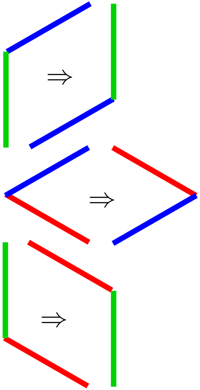


string rewrite rules

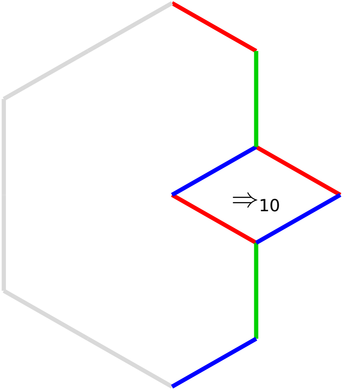


string rewrite

(1) random descent

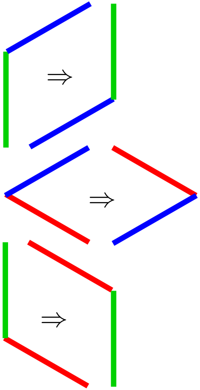


string rewrite rules

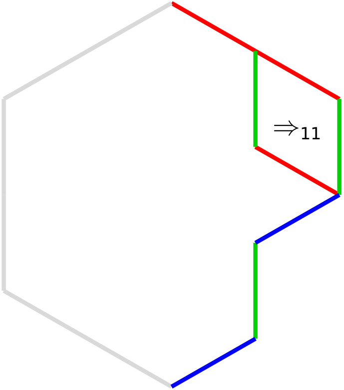


string rewrite

(1) random descent

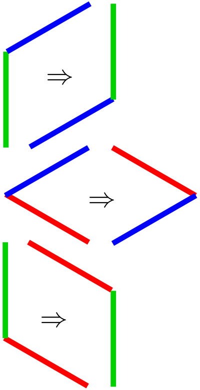


string rewrite rules

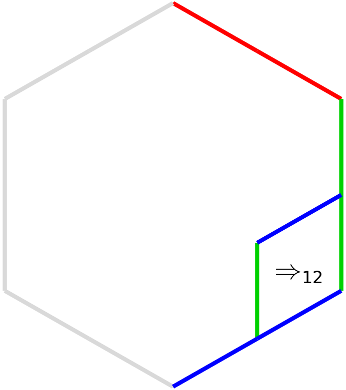


string rewrite

(1) random descent

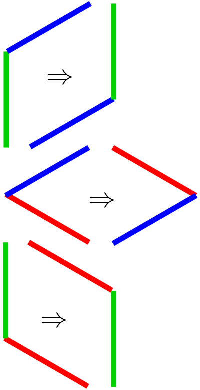


string rewrite rules

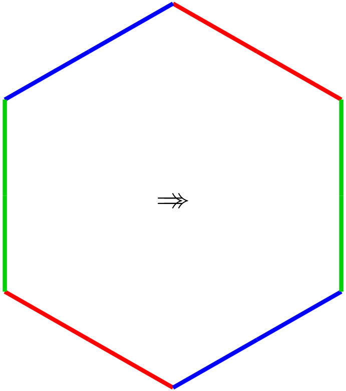


string rewrite

(1) random descent



string rewrite rules



string rewrite

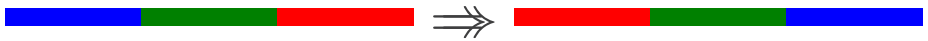
(1) random descent

- filling $\Rightarrow$ is string rewrite system over  with rules

(recover hexagonal shape from associating colours to angles of lines; Logo)

(1) random descent

- filling $\Rightarrow$ is string rewrite system over {} with rules

- filled box B iff exists  filling B
  (any partial filling allows some filling step toward that B)

(1) random descent

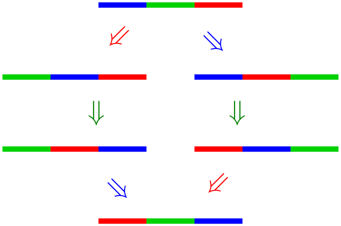
- filling $\Rightarrow$ is string rewrite system over $\{$, , $\}$ with rules

- filled box B iff exists $\Rrightarrow$ filling B
- filling $\Rightarrow$ is ordered weak Church–Rosser (OWCR) for measure on steps

$$\Rightarrow \mapsto (1,0,0) \qquad \Rightarrow \mapsto (0,1,0) \qquad \Rightarrow \mapsto (0,0,1)$$

(measure: mapping steps to (non-zero) elements of a derivation monoid)

critical peak



legs same measure (1,1,1)

(1) random descent

- filling $\Rightarrow$ is string rewrite system over $\{\textcolor{red}{\rule{2em}{0.5em}}, \textcolor{green}{\rule{2em}{0.5em}}, \textcolor{blue}{\rule{2em}{0.5em}}\}$ with rules

$$\textcolor{blue}{\rule{2em}{0.5em}}\textcolor{green}{\rule{2em}{0.5em}} \;\textcolor{red}{\Rightarrow}\; \textcolor{green}{\rule{2em}{0.5em}}\textcolor{blue}{\rule{2em}{0.5em}} \qquad \textcolor{blue}{\rule{2em}{0.5em}}\textcolor{red}{\rule{2em}{0.5em}} \;\textcolor{green}{\Rightarrow}\; \textcolor{red}{\rule{2em}{0.5em}}\textcolor{blue}{\rule{2em}{0.5em}} \qquad \textcolor{green}{\rule{2em}{0.5em}}\textcolor{red}{\rule{2em}{0.5em}} \;\textcolor{blue}{\Rightarrow}\; \textcolor{red}{\rule{2em}{0.5em}}\textcolor{green}{\rule{2em}{0.5em}}$$

- filled box B iff exists $\textcolor{blue}{\rule{4em}{0.5em}}\textcolor{green}{\rule{4em}{0.5em}}\textcolor{red}{\rule{4em}{0.5em}} \Rrightarrow \textcolor{red}{\rule{4em}{0.5em}}\textcolor{green}{\rule{4em}{0.5em}}\textcolor{blue}{\rule{4em}{0.5em}}$ filling B
- filling $\Rightarrow$ is ordered weak Church–Rosser (OWCR) for measure on steps

$$\textcolor{red}{\Rightarrow} \mapsto (\textcolor{red}{1}, \textcolor{green}{0}, \textcolor{blue}{0}) \qquad \textcolor{green}{\Rightarrow} \mapsto (\textcolor{red}{0}, \textcolor{green}{1}, \textcolor{blue}{0}) \qquad \textcolor{blue}{\Rightarrow} \mapsto (\textcolor{red}{0}, \textcolor{green}{0}, \textcolor{blue}{1})$$

- OWCR $\iff$ random descent (RD) so all fillings same spectrum ($=$ measure)
  (RD: if reduction ends in nf then all maximal such do with same measure;
  Newman 42, 07, & Toyama 16)

(1) random descent

- filling $\Rightarrow$ is string rewrite system over $\{\textcolor{red}{\blacksquare}, \textcolor{green}{\blacksquare}, \textcolor{blue}{\blacksquare}\}$ with rules

$$\textcolor{blue}{\blacksquare}\textcolor{green}{\blacksquare} \;\textcolor{red}{\Rightarrow}\; \textcolor{green}{\blacksquare}\textcolor{blue}{\blacksquare} \qquad \textcolor{blue}{\blacksquare}\textcolor{red}{\blacksquare} \;\textcolor{green}{\Rightarrow}\; \textcolor{red}{\blacksquare}\textcolor{blue}{\blacksquare} \qquad \textcolor{green}{\blacksquare}\textcolor{red}{\blacksquare} \;\textcolor{blue}{\Rightarrow}\; \textcolor{red}{\blacksquare}\textcolor{green}{\blacksquare}$$

- filled box B iff exists $\textcolor{blue}{\blacksquare}\textcolor{green}{\blacksquare}\textcolor{red}{\blacksquare} \Rightarrow\!\!\!\Rightarrow \textcolor{red}{\blacksquare}\textcolor{green}{\blacksquare}\textcolor{blue}{\blacksquare}$ filling B
- filling $\Rightarrow$ is ordered weak Church–Rosser (OWCR) for measure on steps

$$\textcolor{red}{\Rightarrow} \mapsto (\textcolor{red}{1}, \textcolor{green}{0}, \textcolor{blue}{0}) \qquad \textcolor{green}{\Rightarrow} \mapsto (\textcolor{red}{0}, \textcolor{green}{1}, \textcolor{blue}{0}) \qquad \textcolor{blue}{\Rightarrow} \mapsto (\textcolor{red}{0}, \textcolor{green}{0}, \textcolor{blue}{1})$$

- OWCR $\iff$ random descent (RD) so all fillings same spectrum
- filling $\Rightarrow$ is weakly normalising (WN) so filling fills
  ($\Rightarrow$ is sorting-by-swapping; termination of bubblesort shows WN)

(1) random descent

- filling $\Rightarrow$ is string rewrite system over $\{\ ,\ ,\ \}$ with rules

- filled box B iff exists  $\Rrightarrow$  filling B
- filling $\Rightarrow$ is ordered weak Church–Rosser (OWCR) for measure on steps

$$\Rightarrow \mapsto (1,0,0) \qquad \Rightarrow \mapsto (0,1,0) \qquad \Rightarrow \mapsto (0,0,1)$$

- OWCR $\iff$ random descent (RD) so all fillings same spectrum
- filling $\Rightarrow$ is weakly normalising (WN) so filling fills

remark

CR & SN $\iff$ OWCR & WN (22), measure on objects $\iff$ on steps
(answer of sorts to Barendregt–Geuvers–Klop conjecture; to when WN lifts to SN)

(1) random descent

- filling $\Rightarrow$ is string rewrite system over $\{$, , $\}$ with rules

- filled box B iff exists $\Rrightarrow$ filling B
- filling $\Rightarrow$ is ordered weak Church–Rosser (OWCR) for measure on steps

$$\Rightarrow \mapsto (1,0,0) \qquad \Rightarrow \mapsto (0,1,0) \qquad \Rightarrow \mapsto (0,0,1)$$

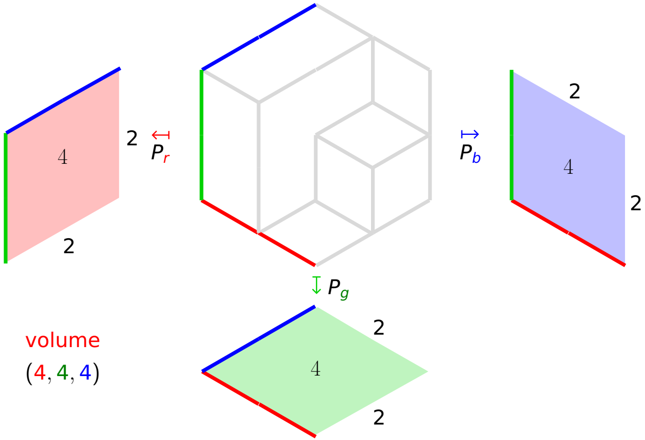
- OWCR $\iff$ random descent (RD) so all fillings same spectrum
- filling $\Rightarrow$ is weakly normalising (WN) so filling fills

remark

CR & SN $\iff$ OWCR & WN (✓ 22), measure on objects $\iff$ on steps

measure on objects decreasing for filling?

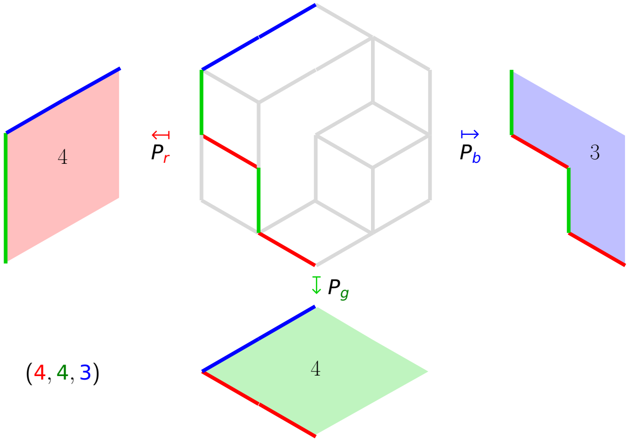
(2) proof order



volume

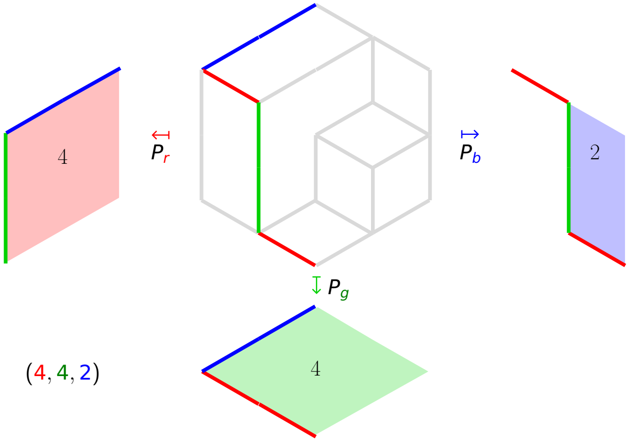
$(4, 4, 4)$

(2) proof order



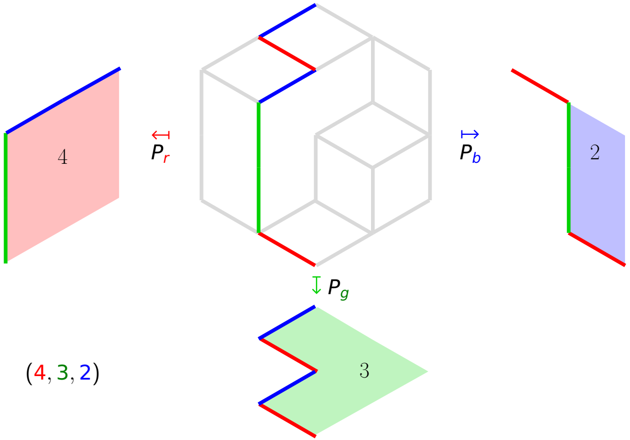
$(4, 4, 3)$

(2) proof order



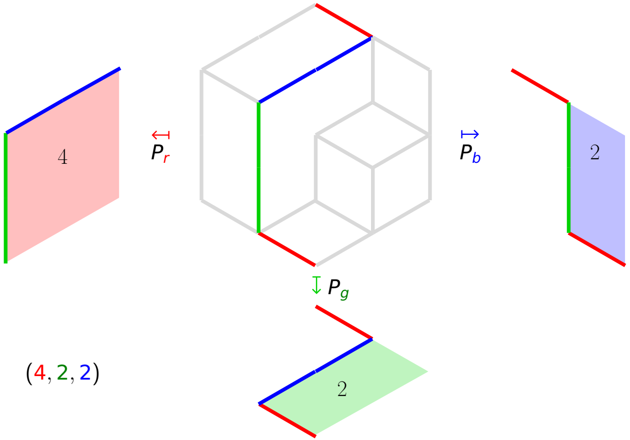
(4, 4, 2)

(2) proof order



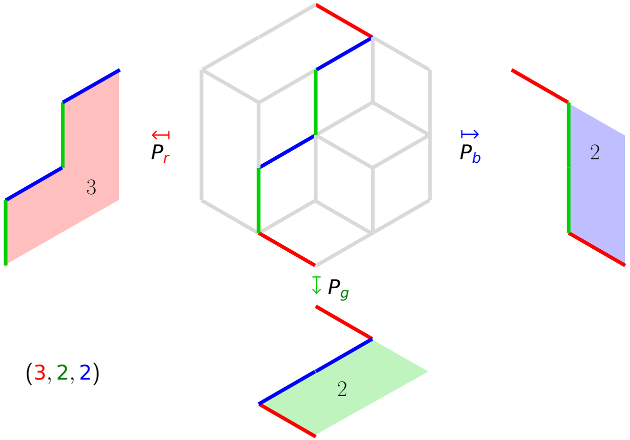
$(4, 3, 2)$

(2) proof order



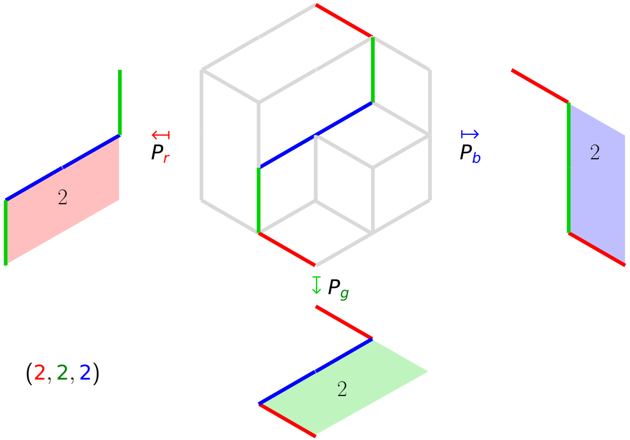
$(4, 2, 2)$

(2) proof order

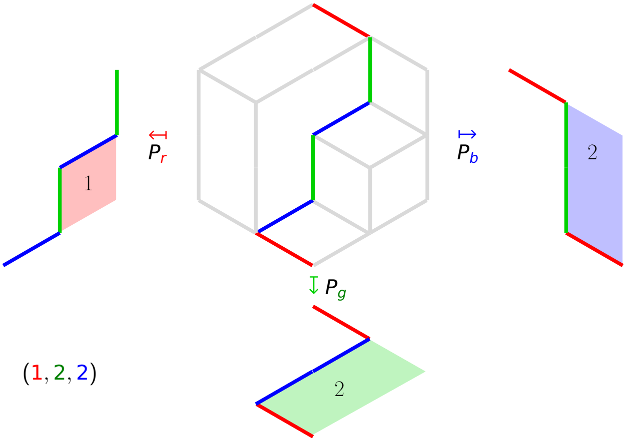


$(3, 2, 2)$

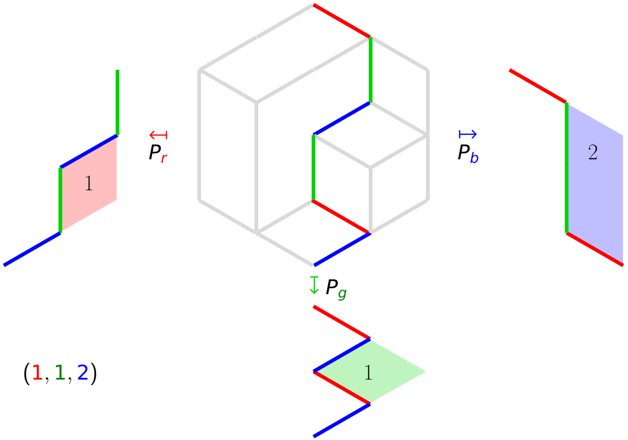
(2) proof order



(2) proof order

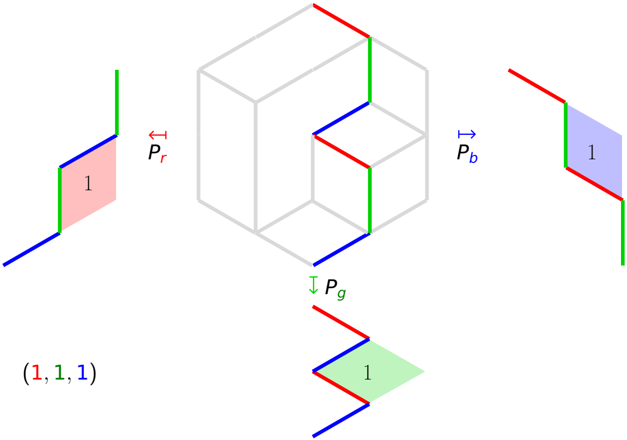


(2) proof order

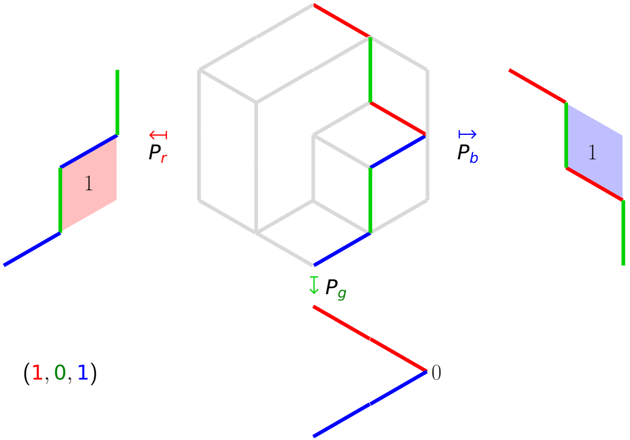


$(1, 1, 2)$

(2) proof order

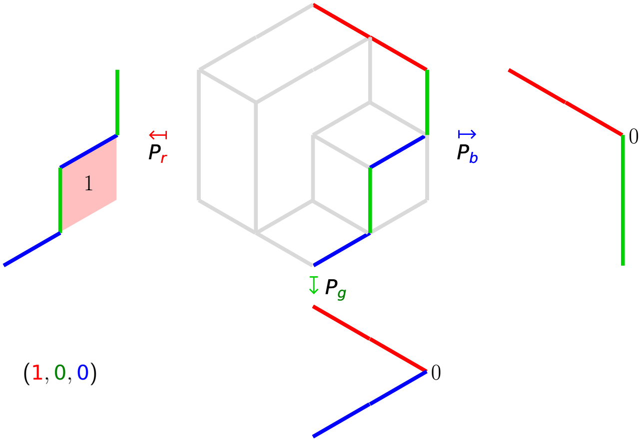


(2) proof order



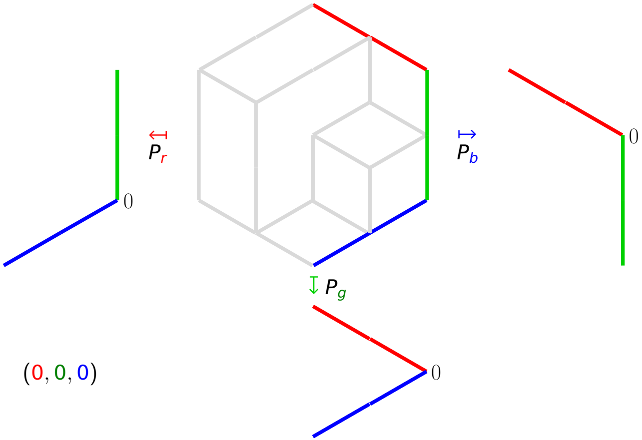
$(1, 0, 1)$

(2) proof order



$(1, 0, 0)$

(2) proof order



$(0, 0, 0)$

(2) proof order

- filling $\Rightarrow$ is string rewrite system over {} with rules

- filled box B iff exists  $\twoheadrightarrow$  filling B
- filling $\Rightarrow$ is WN so filling fills

(2) proof order

- filling $\Rightarrow$ is string rewrite system over  with rules

- filled box B iff exists  filling B
- filling $\Rightarrow$ is WN so filling fills
- filling $\Rightarrow$ decrements (one component of) volume (r, g, b) of path P
  (volume of trichrome path P: triple of areas of projections P_r,P_g,P_b
  area of dichrome path P: #missing calissons)

(2) proof order

- filling $\Rightarrow$ is string rewrite system over $\{$$\}$ with rules

- filled box B iff exists  $\Rrightarrow$  filling B
- filling $\Rightarrow$ is WN so filling fills
- filling $\Rightarrow$ decrements volume (r, g, b) of path P so SN
- volume of normal form path is $(0, 0, 0)$ so spectrum $=$ volume of initial path (initial path only depends on hexagon / box, not on filling / filled box)

(2) proof order

- filling $\Rightarrow$ is string rewrite system over $\{$$\}$ with rules

- filled box B iff exists  filling B
- filling $\Rightarrow$ is WN so filling fills
- filling $\Rightarrow$ decrements volume (r, g, b) of path P so SN
- volume of normal form path is $(0, 0, 0)$ so spectrum $=$ volume of initial path

remark

proof order (Bachmaier & Dershowitz 94) as involutive monoid homomorphism
area proof order to triple (ℓ, a, r) with #missing calissons a (Felgenhauer & 13)

(2) proof order

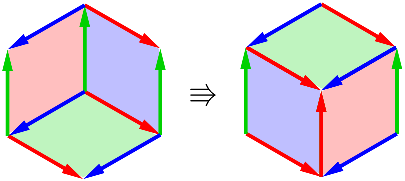
- filling $\Rightarrow$ is string rewrite system over $\{$$\}$ with rules

- filled box B iff exists  $\Rrightarrow$  filling B
- filling $\Rightarrow$ is WN so filling fills
- filling $\Rightarrow$ decrements volume (r, g, b) of path P so SN
- volume of normal form path is $(0, 0, 0)$ so spectrum $=$ volume of initial path

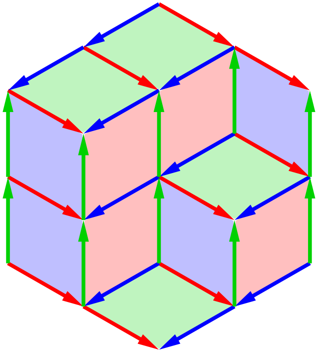
remark

proof order as involutive monoid homomorphism
area proof order to triple (ℓ, a, r) with #missing calissons a
proofs by random descent and proof order show spectrum independent of filling
but can different fillings be related?

(3) bricklaying

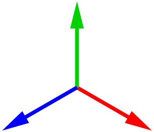


bricklaying rule

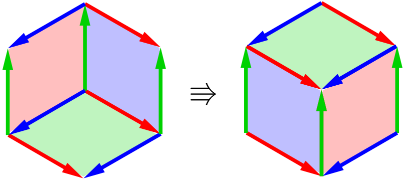


bricklaying step possible?

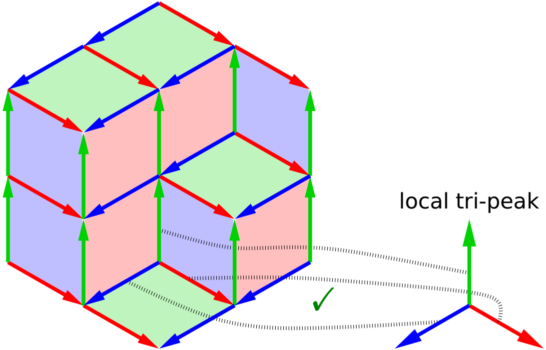
local tri-peak



(3) bricklaying

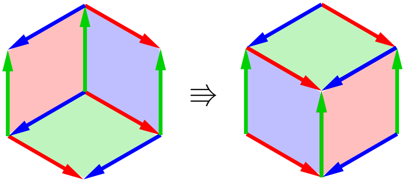


bricklaying rule

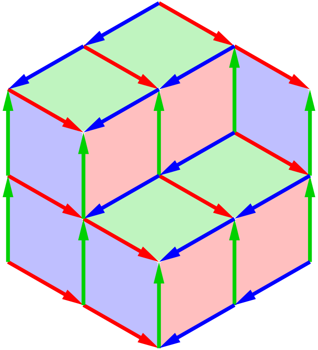


bricklaying step

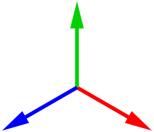
(3) bricklaying



bricklaying rule

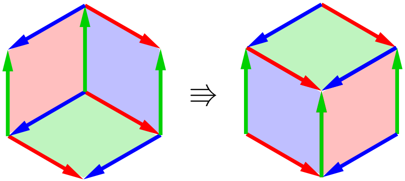


bricklaying step possible?

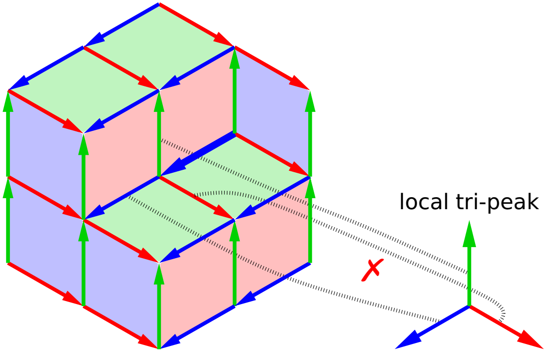


local tri-peak

(3) bricklaying

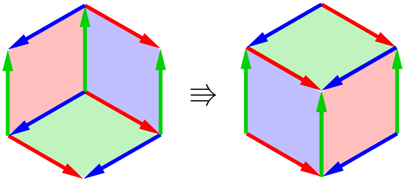


bricklaying rule

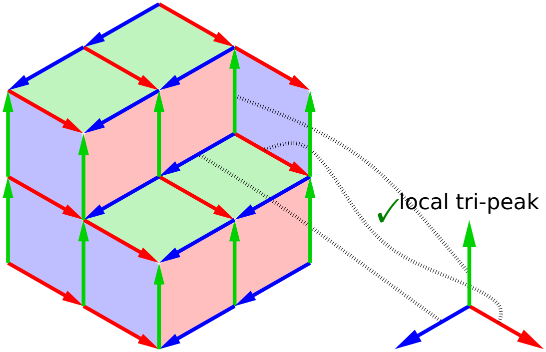


bricklaying step possible?

(3) bricklaying

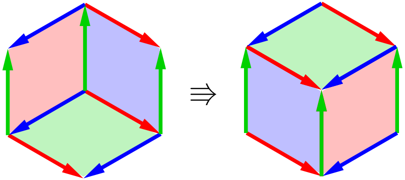


bricklaying rule

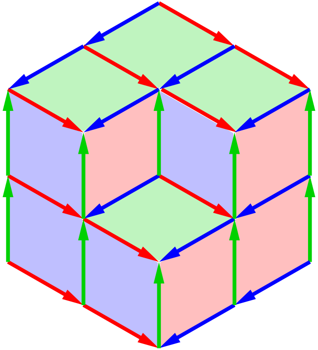


bricklaying step

(3) bricklaying

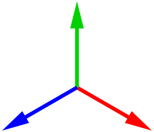


bricklaying rule

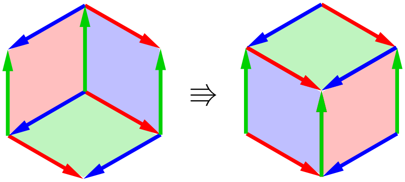


bricklaying step possible?

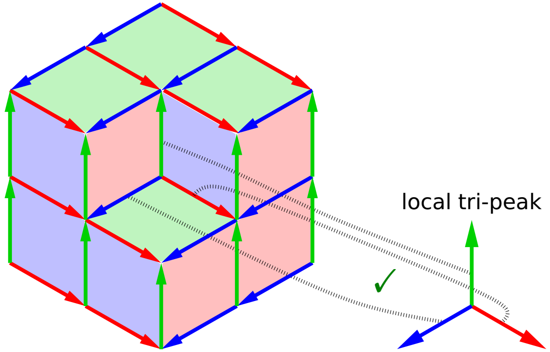
local tri-peak



(3) bricklaying

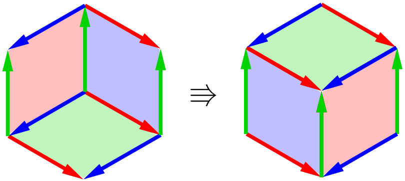


bricklaying rule

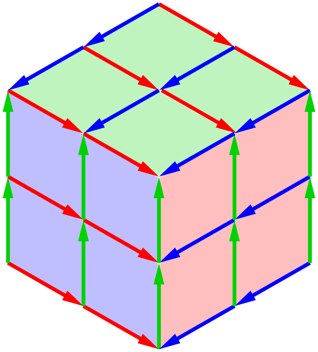


bricklaying step

(3) bricklaying

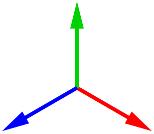


bricklaying rule

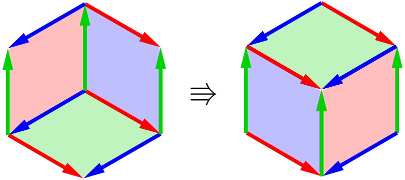


bricklaying step possible?

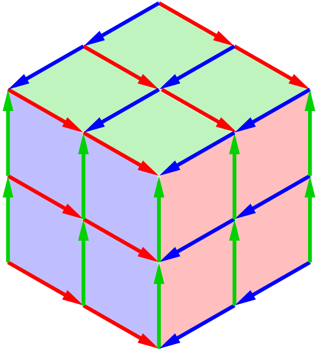
local tri-peak



(3) bricklaying

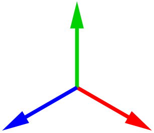


bricklaying rule



big brick (bricklayer)

local tri-peak



(3) bricklaying

- bricklaying $\Rrightarrow$ is graph rewrite system over beds
  (bed: plane bed-graph; bed-graph: dag obtained by tiling; ⩔ 22)

(3) bricklaying

- bricklaying $\Rrightarrow$ is graph rewrite system over beds
- spectrum per construction preserved by bricklaying $\Rrightarrow$ steps

(3) bricklaying

- bricklaying $\Rrightarrow$ is graph rewrite system over beds
- spectrum preserved by bricklaying $\Rrightarrow$ steps
- bricklaying $\Rrightarrow$ terminating
  (trivial; calissons closer to their origin)

(3) bricklaying

- bricklaying $\Rrightarrow$ is graph rewrite system over beds
- spectrum preserved by bricklaying $\Rrightarrow$ steps
- bricklaying $\Rrightarrow$ terminating
- bricklaying $\Rrightarrow$ normal form iff big brick
  (out-degree edges ≤ 3; if some tri-peak $\implies$ bricklaying step found by following back in-edges; if no tri-peaks $\implies$ big brick; holds for bed-graphs)

(3) bricklaying

- bricklaying $\Rrightarrow$ is graph rewrite system over beds
- spectrum preserved by bricklaying $\Rrightarrow$ steps
- bricklaying $\Rrightarrow$ terminating
- bricklaying $\Rrightarrow$ normal form iff big brick
- big brick unique for hexagon; filled boxes $\Rrightarrow$-convertible so same spectrum (4 calissons of each colour)

(3) bricklaying

- bricklaying $\Rrightarrow$ is graph rewrite system over beds
- spectrum preserved by bricklaying $\Rrightarrow$ steps
- bricklaying $\Rrightarrow$ terminating
- bricklaying $\Rrightarrow$ normal form iff big brick
- big brick unique for hexagon; filled boxes $\Rrightarrow$-convertible so same spectrum

remark

conversions : (2-dimensional) tiling = beds : (3-dimensional) bricklaying ; 22

(3) bricklaying

- bricklaying $\Rrightarrow$ is graph rewrite system over beds
- spectrum preserved by bricklaying $\Rrightarrow$ steps
- bricklaying $\Rrightarrow$ terminating
- bricklaying $\Rrightarrow$ normal form iff big brick
- big brick unique for hexagon; filled boxes $\Rrightarrow$-convertible so same spectrum

remark

conversions : tiling = beds : bricklaying

bricklaying reduces all fillings to $\Rrightarrow$-normal form, a big brick, unique for hexagon but characterisation of big bricks?

(3) bricklaying

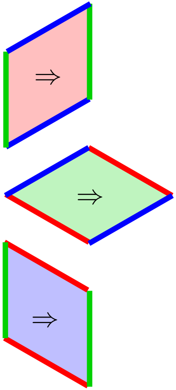
- bricklaying $\Rrightarrow$ is graph rewrite system over beds
- spectrum preserved by bricklaying $\Rrightarrow$ steps
- bricklaying $\Rrightarrow$ terminating
- bricklaying $\Rrightarrow$ normal form iff big brick
- big brick unique for hexagon; filled boxes $\Rrightarrow$-convertible so same spectrum

remark

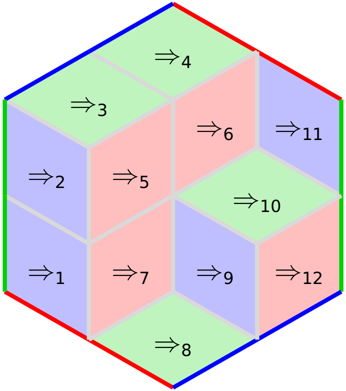
conversions : tiling = beds : bricklaying

bricklaying reduces all fillings to $\Rrightarrow$-normal form, a big brick, unique for hexagon
filling ($\Rightarrow$) equivalent iff projection ($\Downarrow$) equivalent; big brick least $\Downarrow$-upperbound

(4) local undercutting; from ⇛-filling to ⇓-projection

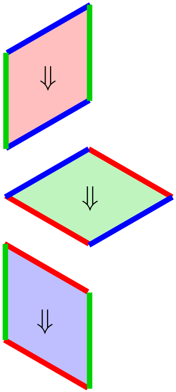


filling rules

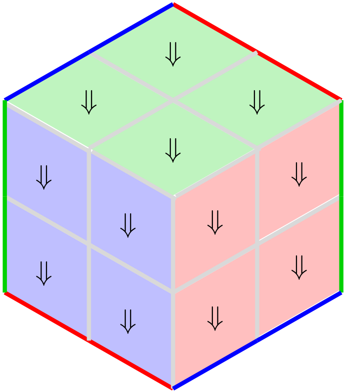


from ⇛-filling

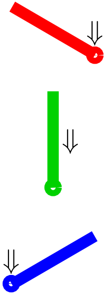
(4) local undercutting; from ⇒-filling to ⇓-projection



zap rules

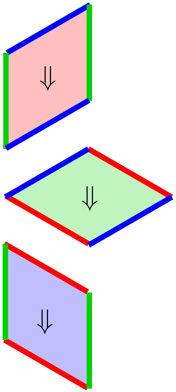


to ⇓-projection with same spectrum

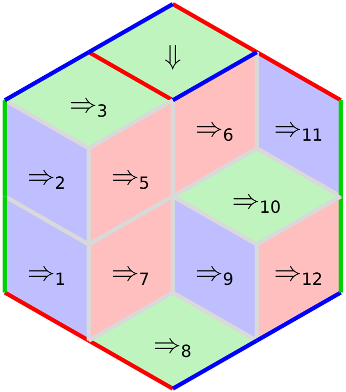


trivial zap rules

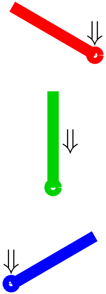
(4) local undercutting



zap rules

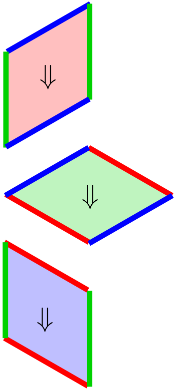


zap step

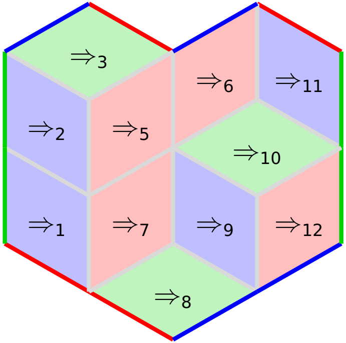


trivial zap rules

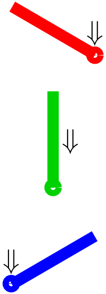
(4) local undercutting



zap rules

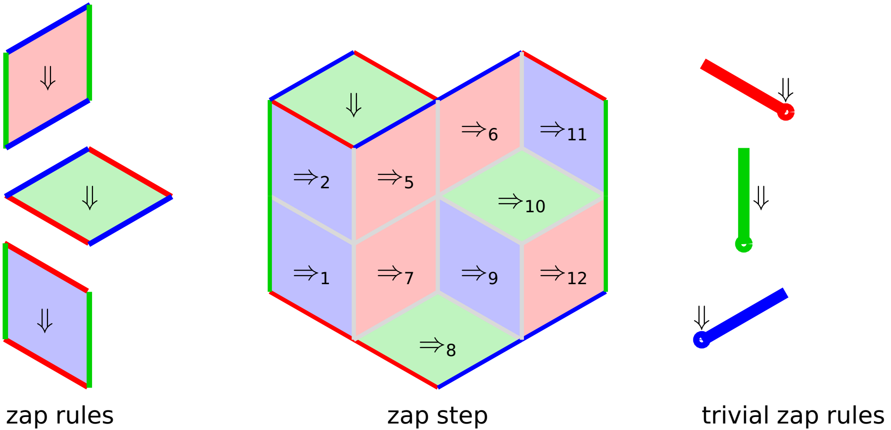


foliage (for cyclic conversion)

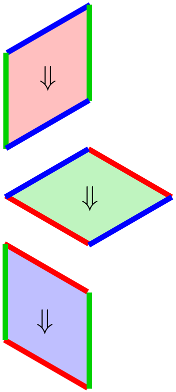


trivial zap rules

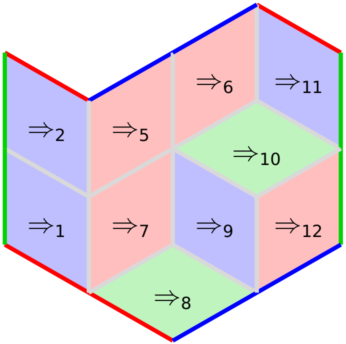
(4) local undercutting



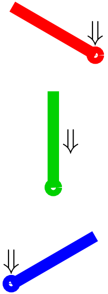
(4) local undercutting



zap rules

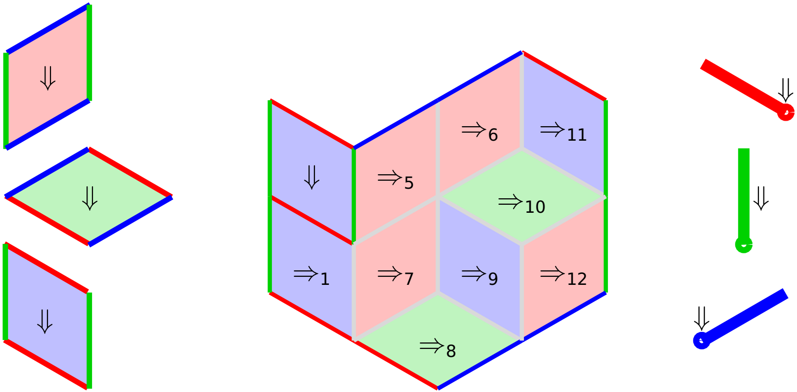


foliage



trivial zap rules

(4) local undercutting

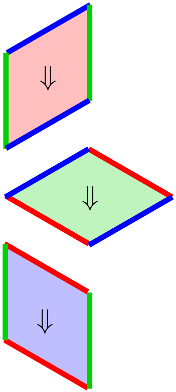


zap rules

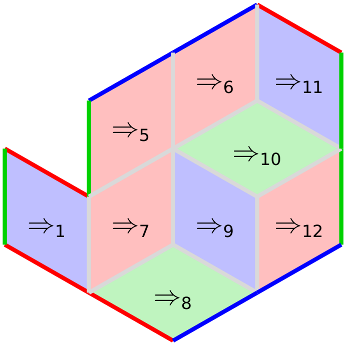
zap step

trivial zap rules

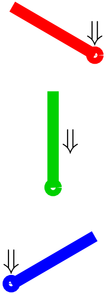
(4) local undercutting



zap rules

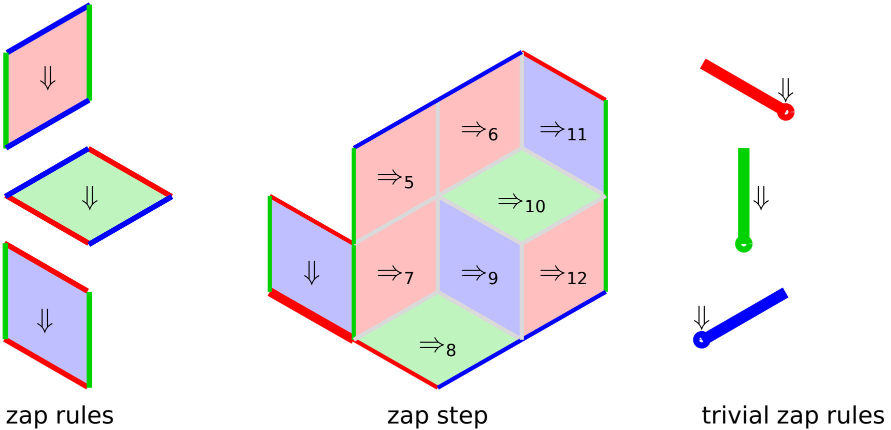


foliage

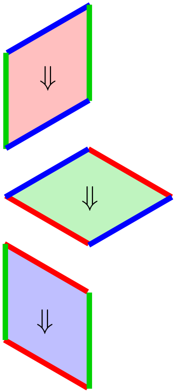


trivial zap rules

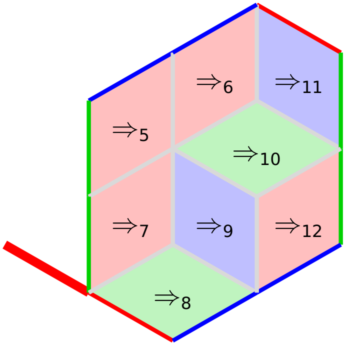
(4) local undercutting



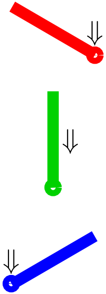
(4) local undercutting



zap rules

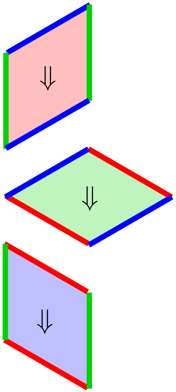


foliage

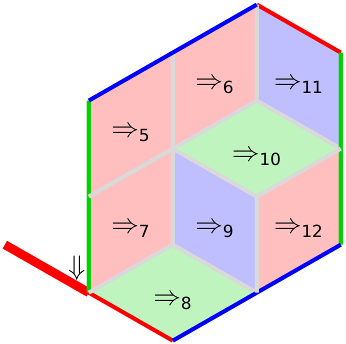


trivial zap rules

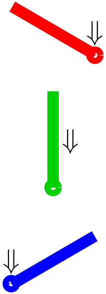
(4) local undercutting



zap rules

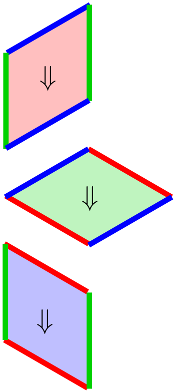


trivial zap step

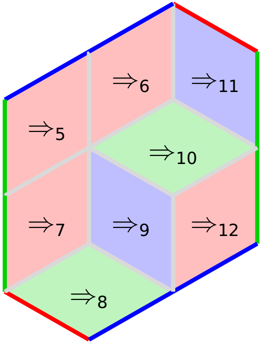


trivial zap rules

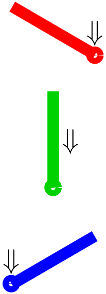
(4) local undercutting



zap rules

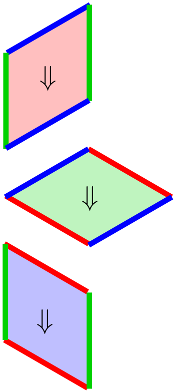


foliage

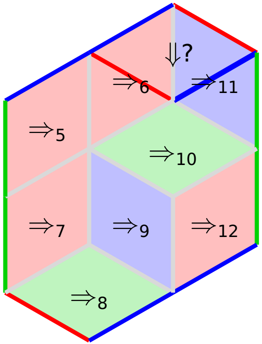


trivial zap rules

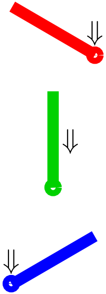
(4) local undercutting



zap rules

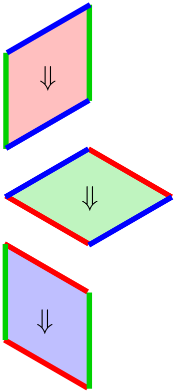


zap step incompatible with ⇒



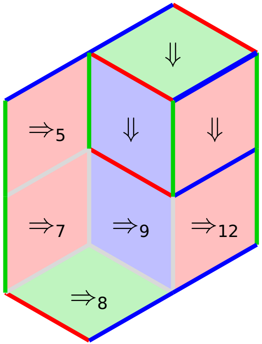
trivial zap rules

(4) local undercutting

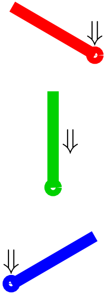


zap rules

preserves spectrum

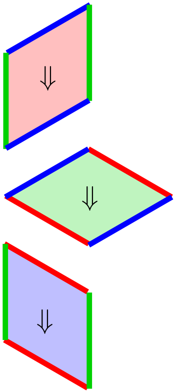


⇓ locally undercuts ⇒

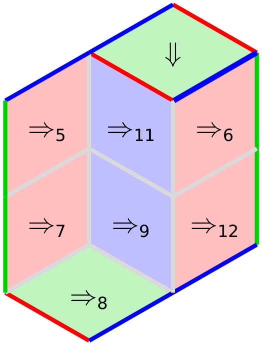


trivial zap rules

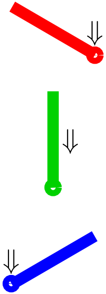
(4) local undercutting



zap rules

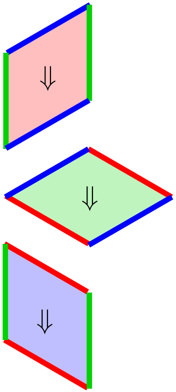


zap step compatible with ⇒

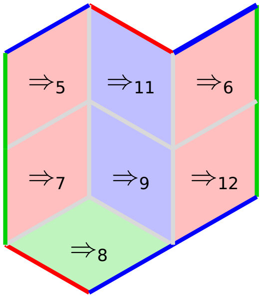


trivial zap rules

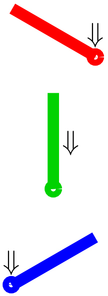
(4) local undercutting



zap rules

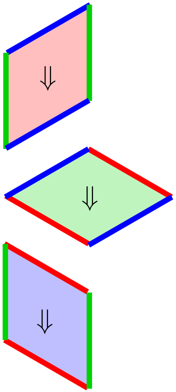


foliage

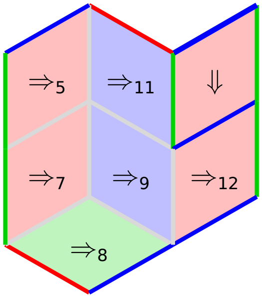


trivial zap rules

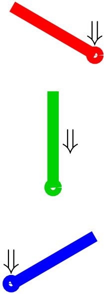
(4) local undercutting



zap rules

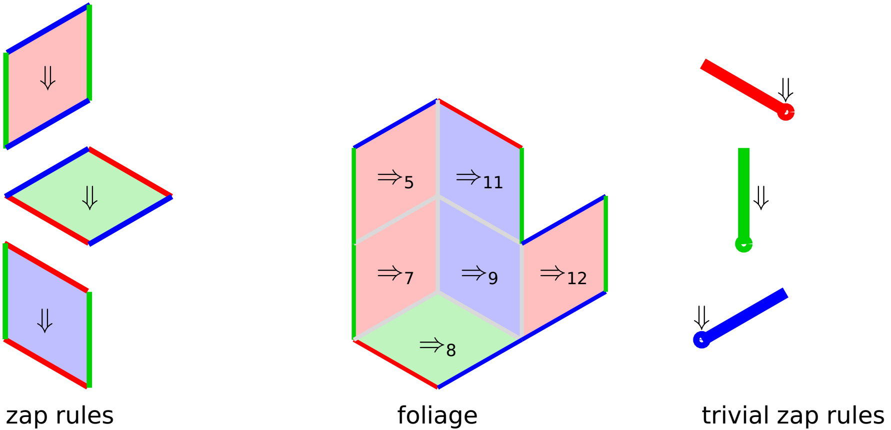


zap step

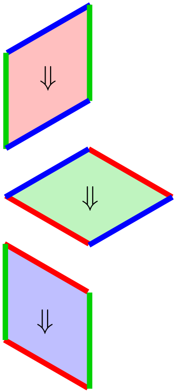


trivial zap rules

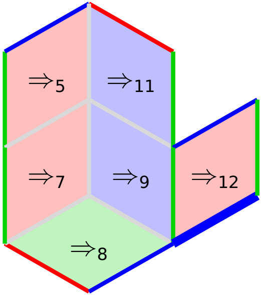
(4) local undercutting



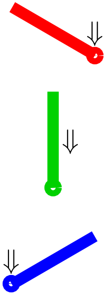
(4) local undercutting



zap rules

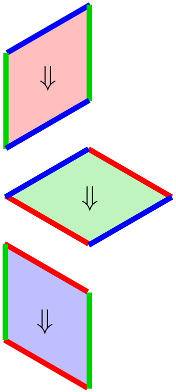


zap step

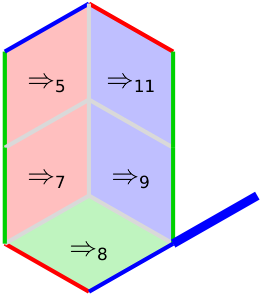


trivial zap rules

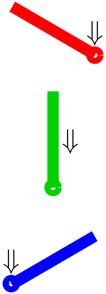
(4) local undercutting



zap rules

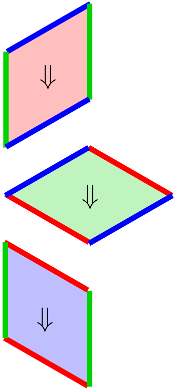


foliage

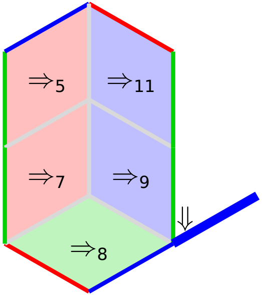


trivial zap rules

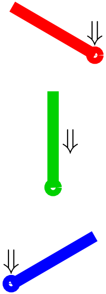
(4) local undercutting



zap rules

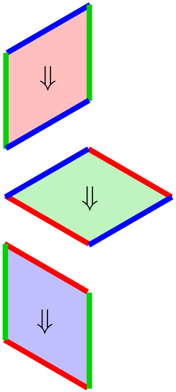


trivial zap step

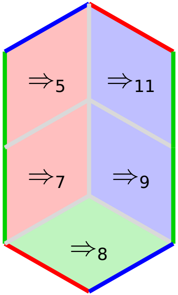


trivial zap rules

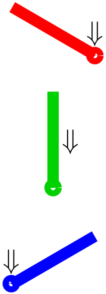
(4) local undercutting



zap rules

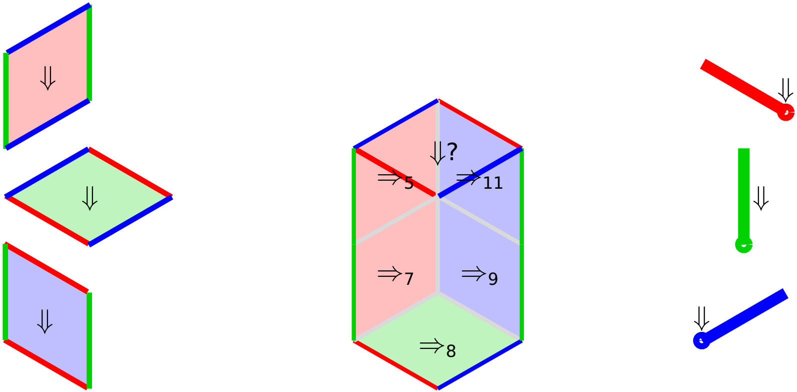


foliage



trivial zap rules

(4) local undercutting

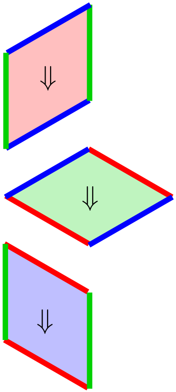


zap rules

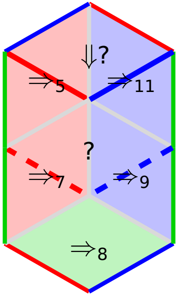
zap step incompatible with ⇒

trivial zap rules

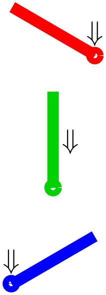
(4) local undercutting



zap rules

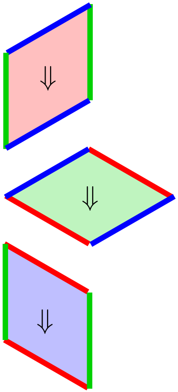


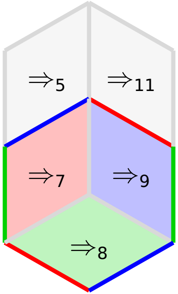
no local undercutting yet

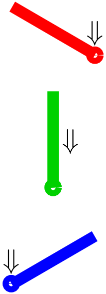


trivial zap rules

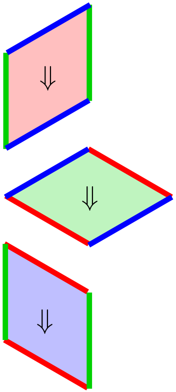
(4) local undercutting



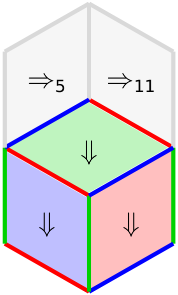




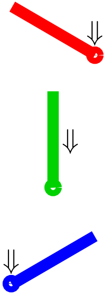
(4) local undercutting



zap rules

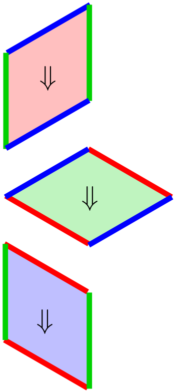


zap steps by IH for subfoliage

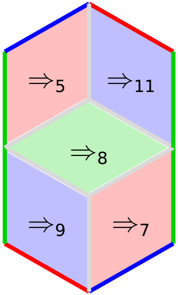


trivial zap rules

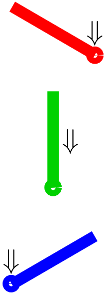
(4) local undercutting



zap rules

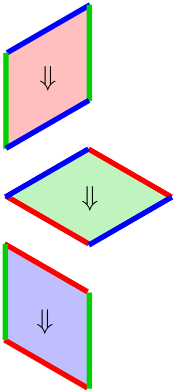


foliage

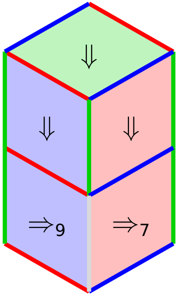


trivial zap rules

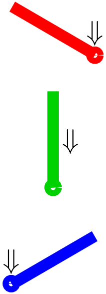
(4) local undercutting



zap rules

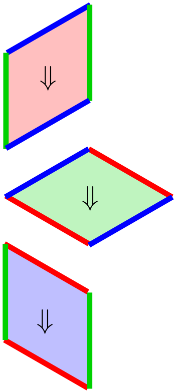


⇓ locally undercuts ⇒

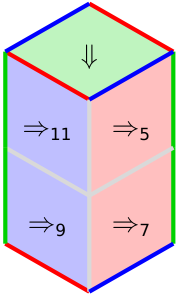


trivial zap rules

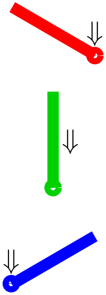
(4) local undercutting



zap rules

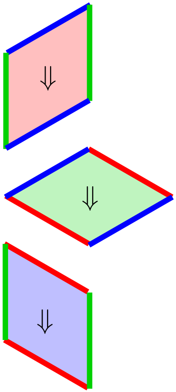


zap step compatible with ⇒

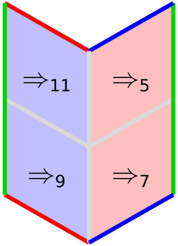


trivial zap rules

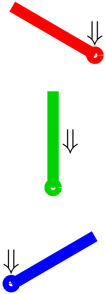
(4) local undercutting



zap rules

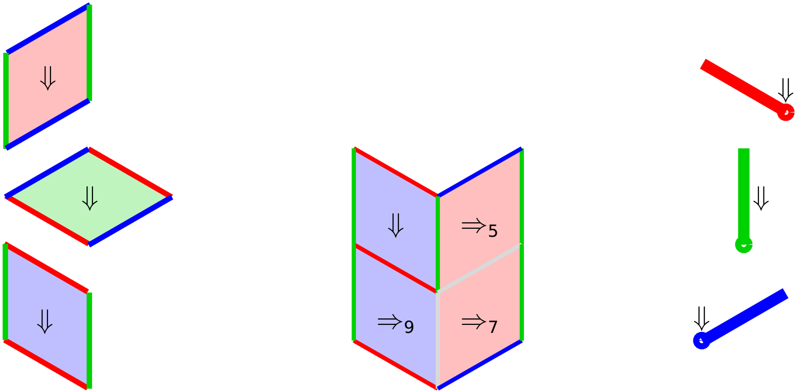


foliage



trivial zap rules

(4) local undercutting

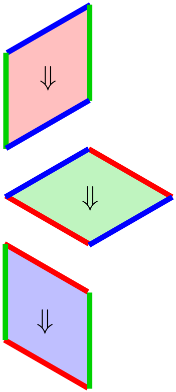


zap rules

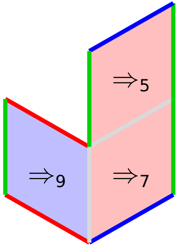
zap step

trivial zap rules

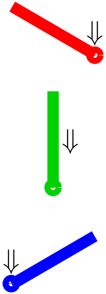
(4) local undercutting



zap rules

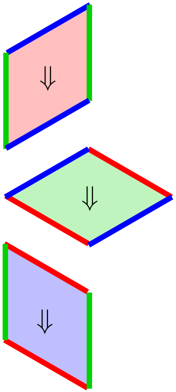


foliage

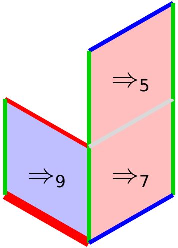


trivial zap rules

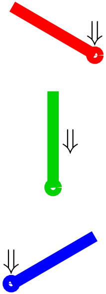
(4) local undercutting



zap rules

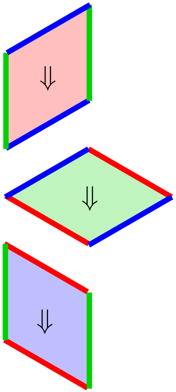


zap step

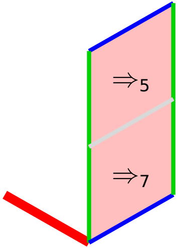


trivial zap rules

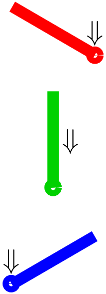
(4) local undercutting



zap rules

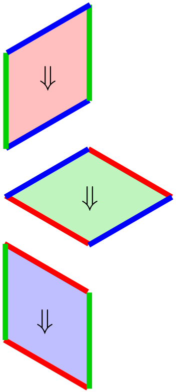


foliage

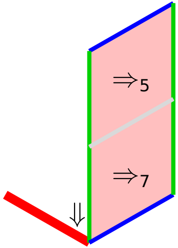


trivial zap rules

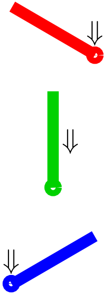
(4) local undercutting



zap rules

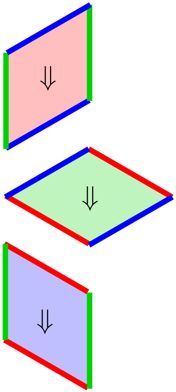


trivial zap step

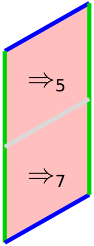


trivial zap rules

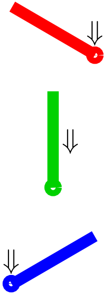
(4) local undercutting



zap rules

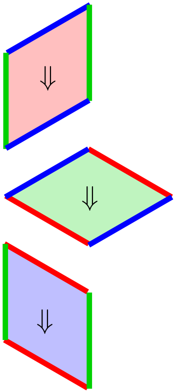


foliage

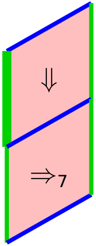


trivial zap rules

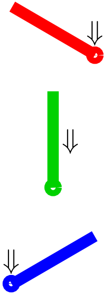
(4) local undercutting



zap rules

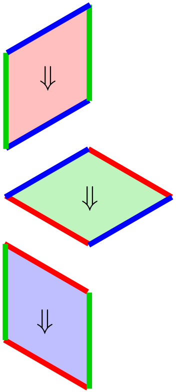


zap step

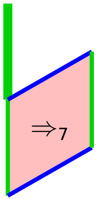


trivial zap rules

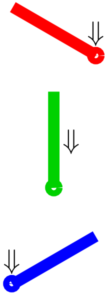
(4) local undercutting



zap rules

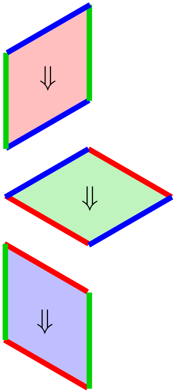


foliage

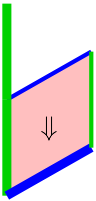


trivial zap rules

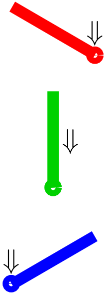
(4) local undercutting



zap rules

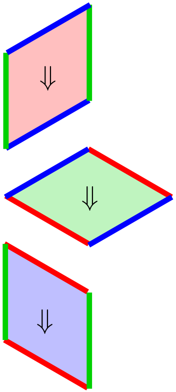


zap step

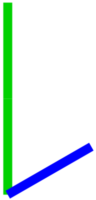


trivial zap rules

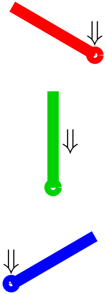
(4) local undercutting



zap rules

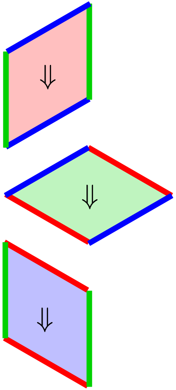


foliage

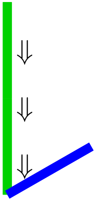


trivial zap rules

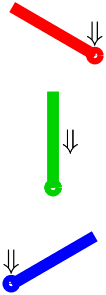
(4) local undercutting



zap rules

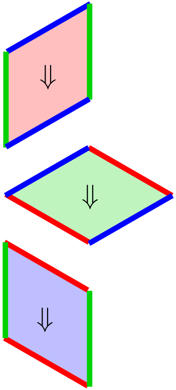


3 trivial zap steps



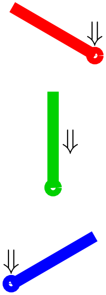
trivial zap rules

(4) local undercutting



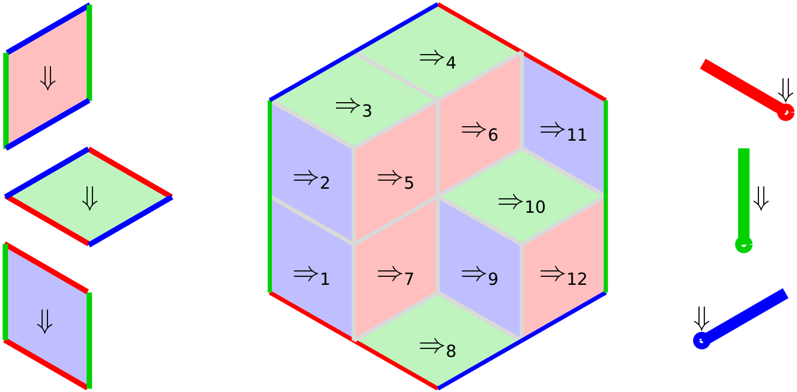
zap rules

empty foliage



trivial zap rules

(4) local undercutting; from ⇒-filling to ⇓-projection

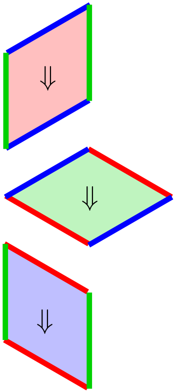


zap rules

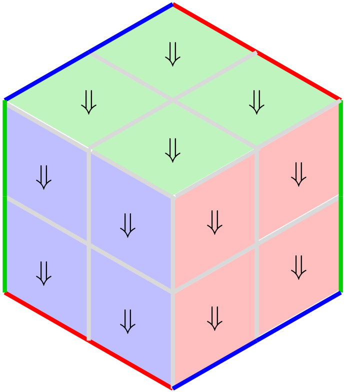
from ⇒-filling

trivial zap rules

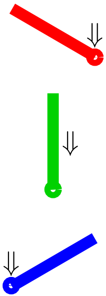
(4) local undercutting; from ⇛-filling to ⤋-projection



zap rules



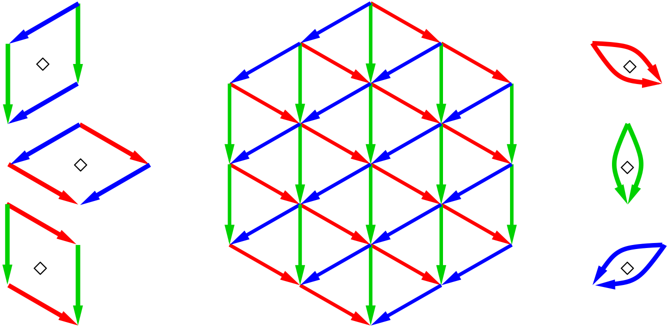
to ⤋-projection with same spectrum



trivial zap rules

(4) local undercutting

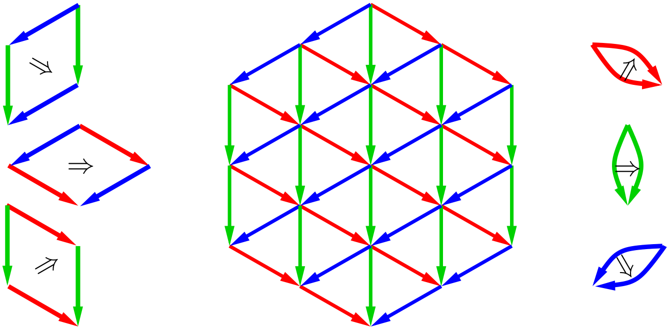
- calissons as diamonds ${}^{\phi}_{\chi}\diamond^{\psi}_{\upsilon}$ and ${}^{\phi}_{\varepsilon}\diamond^{\phi}_{\varepsilon}$ of grid rewrite system $\rightarrow$ for hexagon filling $\phi \cdot \chi \Rightarrow \psi \cdot \upsilon$ on reductions, projection $\phi^{-1} \cdot \psi \Downarrow \chi \cdot \upsilon^{-1}$ on conversions



diamonds | grid rewrite system $\rightarrow$ | trivial diamonds

(4) local undercutting

- calissons as diamonds ${}^{\phi}_{\chi}\diamond^{\psi}_{\upsilon}$ and ${}^{\phi}_{\varepsilon}\diamond^{\phi}_{\varepsilon}$ of grid rewrite system $\rightarrow$ for hexagon filling $\phi \cdot \chi \Rightarrow \psi \cdot \upsilon$ on reductions, projection $\phi^{-1} \cdot \psi \Downarrow \chi \cdot \upsilon^{-1}$ on conversions



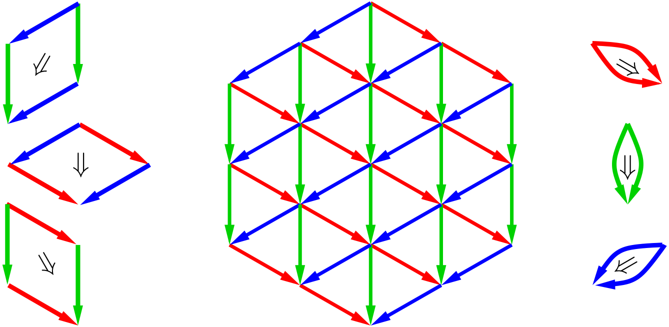
filling rules

grid rewrite system $\rightarrow$

trivial filling rules

(4) local undercutting

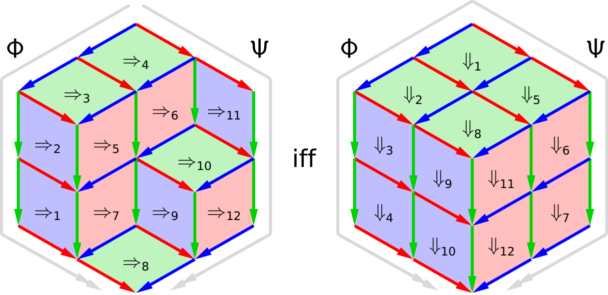
- calissons as diamonds ${}^{\phi}_{\chi}\diamond^{\psi}_{\upsilon}$ and ${}^{\phi}_{\varepsilon}\diamond^{\phi}_{\varepsilon}$ of grid rewrite system $\rightarrow$ for hexagon filling $\phi \cdot \chi \Rightarrow \psi \cdot \upsilon$ on reductions, projection $\phi^{-1} \cdot \psi \Downarrow \chi \cdot \upsilon^{-1}$ on conversions



projection rules grid rewrite system $\rightarrow$ trivial projection rules

(4) local undercutting; from $\Rightarrow$-filling to $\Downarrow$-projection

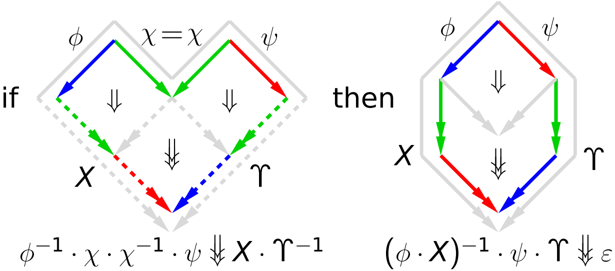
- calissons as diamonds ${}^{\phi}_{\chi}\diamond^{\psi}_{\upsilon}$ and ${}^{\phi}_{\varepsilon}\diamond^{\phi}_{\varepsilon}$ of grid rewrite system $\to$ for hexagon filling $\phi \cdot \chi \Rightarrow \psi \cdot \upsilon$ on reductions, projection $\phi^{-1} \cdot \psi \Downarrow \chi \cdot \upsilon^{-1}$ on conversions
- $\Phi \twoheadRightarrow \Psi$ iff $\Phi^{-1} \cdot \Psi \twoheadDownarrow \varepsilon$ for reductions Φ, Ψ (Lévy 78, & Klop & de Vrijer 98)



filling $\Phi \twoheadRightarrow \Psi$ projection $\Phi^{-1} \cdot \Psi \twoheadDownarrow \varepsilon$

(4) local undercutting; from $\Rightarrow$-filling to $\Downarrow$-projection

- calissons as diamonds ${}^{\phi}_{\chi}\diamond^{\psi}_{\upsilon}$ and ${}^{\phi}_{\varepsilon}\diamond^{\phi}_{\varepsilon}$ of grid rewrite system $\rightarrow$ for hexagon filling $\phi \cdot \chi \Rightarrow \psi \cdot \upsilon$ on reductions, projection $\phi^{-1} \cdot \psi \Downarrow \chi \cdot \upsilon^{-1}$ on conversions
- $\Phi \Rrightarrow \Psi$ iff $\Phi^{-1} \cdot \Psi \Downarrow\!\!\Downarrow \varepsilon$ for reductions Φ, Ψ
  if $\rightarrow$ terminating and projection $\Downarrow$ locally undercutting (LUC)
  local undercutting; novel, based on Dehornoy et al. 15:



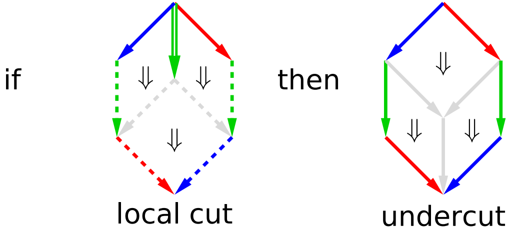
(4) local undercutting

- calissons as diamonds ${}^{\phi}_{\chi}\diamond^{\psi}_{\upsilon}$ and ${}^{\phi}_{\varepsilon}\diamond^{\phi}_{\varepsilon}$ of grid rewrite system $\rightarrow$ for hexagon filling $\phi \cdot \chi \Rightarrow \psi \cdot \upsilon$ on reductions, projection $\phi^{-1} \cdot \psi \Downarrow \chi \cdot \upsilon^{-1}$ on conversions
- $\Phi \Rrightarrow \Psi$ iff $\Phi^{-1} \cdot \Psi \Downarrow\!\!\!\!\Downarrow \varepsilon$ for reductions Φ, Ψ
  if $\rightarrow$ terminating and projection $\Downarrow$ locally undercutting (LUC)
- grid rewrite system $\rightarrow$ is terminating (trivial; $\rightarrow$ is a dag)

(4) local undercutting

- calissons as diamonds ${}^{\phi}_{\chi}\diamond^{\psi}_{\upsilon}$ and ${}^{\phi}_{\varepsilon}\diamond^{\phi}_{\varepsilon}$ of grid rewrite system $\rightarrow$ for hexagon filling $\phi \cdot \chi \Rightarrow \psi \cdot \upsilon$ on reductions, projection $\phi^{-1} \cdot \psi \Downarrow \chi \cdot \upsilon^{-1}$ on conversions
- $\Phi \Rrightarrow \Psi$ iff $\Phi^{-1} \cdot \Psi \Downarrow\!\!\Downarrow \varepsilon$ for reductions Φ, Ψ

  if $\rightarrow$ terminating and projection $\Downarrow$ locally undercutting (LUC)
- grid rewrite system $\rightarrow$ is terminating
- projection $\Downarrow$ is locally undercutting



(4) local undercutting

- calissons as diamonds ${}^{\phi}_{\chi}\diamond^{\psi}_{\upsilon}$ and ${}^{\phi}_{\varepsilon}\diamond^{\phi}_{\varepsilon}$ of grid rewrite system $\rightarrow$ for hexagon filling $\phi \cdot \chi \Rightarrow \psi \cdot \upsilon$ on reductions, projection $\phi^{-1} \cdot \psi \Downarrow \chi \cdot \upsilon^{-1}$ on conversions
- $\Phi \Rrightarrow \Psi$ iff $\Phi^{-1} \cdot \Psi \Downarrow\!\!\!\Downarrow \varepsilon$ for reductions Φ, Ψ
  if $\rightarrow$ terminating and projection $\Downarrow$ locally undercutting (LUC)
- grid rewrite system $\rightarrow$ is terminating
- projection $\Downarrow$ is locally undercutting
- undercutting preserves spectrum so spectrum of filling and projection same
  (spectrum of projection is unique by random descent; 07)

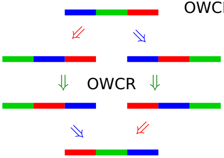
(4) local undercutting

- calissons as diamonds ${}^{\phi}_{\chi}\diamond^{\psi}_{\upsilon}$ and ${}^{\phi}_{\varepsilon}\diamond^{\phi}_{\varepsilon}$ of grid rewrite system $\rightarrow$ for hexagon filling $\phi \cdot \chi \Rightarrow \psi \cdot \upsilon$ on reductions, projection $\phi^{-1} \cdot \psi \Downarrow \chi \cdot \upsilon^{-1}$ on conversions
- $\Phi \Rrightarrow \Psi$ iff $\Phi^{-1} \cdot \Psi \Downarrow\!\!\Downarrow \varepsilon$ for reductions Φ, Ψ
  if $\rightarrow$ terminating and projection $\Downarrow$ locally undercutting (LUC)
- grid rewrite system $\rightarrow$ is terminating
- projection $\Downarrow$ is locally undercutting
- undercutting preserves spectrum so spectrum of filling and projection same

remark

zapping, contracting conversion cycles to a loop, goes back to Newman 42
it is a basic tool for e.g. Finite Derivation Types (Squier 87), Garside theory (Dehornoy et al. 15), homotopy type theory (Kraus & von Raumer 23), and polygraphs (the ‘polybook’, Ara et al. 25)

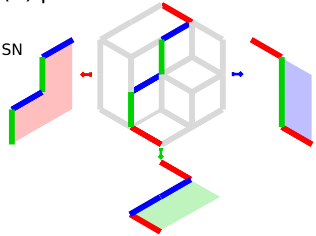
(1) random descent



OWCR & WN $\Longleftrightarrow$ WCR & SN

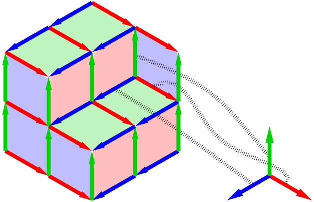
quantitative commutation

(2) proof order



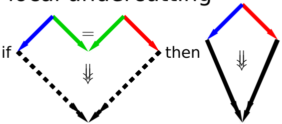
typed involutive monoids for conversions

(3) bricklaying



big brick as unique normal form of beds

(4) local undercutting



common multiple $\Longrightarrow$ least common multiple

upperbound $\Longrightarrow$ least upperbound / emph

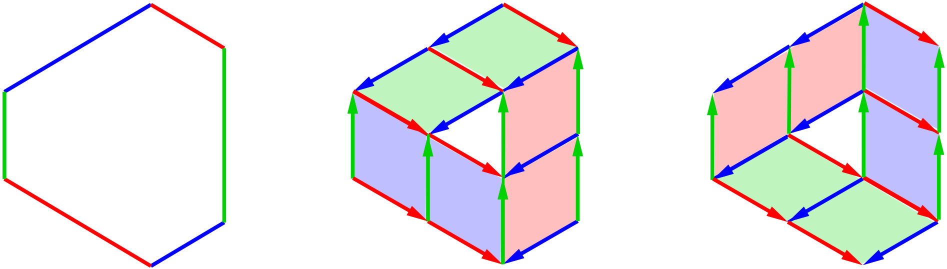
confluent $\Longrightarrow$ orthogonal

Conclusions

- modern confluence techniques powerful; 4 solve problem of the calissons (for all zonogonal hexagons; non-convex boxes? Dijkstra 89)

Conclusions

- modern confluence techniques powerful; solve problem of the calissons (for all zonogonal hexagons; non-convex boxes? Dijkstra 89)



Conclusions

- modern confluence techniques powerful; solve problem of the calissons
- local semi-lattice (LSL = LUC with ${}_{X}^{\phi}\diamond_{\Upsilon}^{\psi}$ iff ${}_{\Upsilon}^{\psi}\diamond_{X}^{\phi}$) $\Longrightarrow$ filling iff projection for term rewriting and positive braids; extends Dehornoy et al. 15 (Projection Theorem: permutation iff projection equivalence (Terese 03) entails cube-property; Lévy 78)

Conclusions

- modern confluence techniques powerful; solve problem of the calissons
- local semi-lattice (LSL = LUC with ${}^{\phi}_{\chi}\diamond^{\psi}_{\Upsilon}$ iff ${}^{\psi}_{\Upsilon}\diamond^{\phi}_{\chi}$) $\implies$ filling iff projection for term rewriting and positive braids
- productivity instead of termination of $\rightarrow$ for filling iff projection (given LSL)? (coinduction instead of induction?)

Conclusions

- modern confluence techniques powerful; solve problem of the calissons
- local semi-lattice (LSL = LUC with ${}_{X}^{\phi}\diamond_{\Upsilon}^{\psi}$ iff ${}_{\Upsilon}^{\psi}\diamond_{X}^{\phi}$) $\implies$ filling iff projection for term rewriting and positive braids
- productivity instead of termination of $\rightarrow$ for filling iff projection (given LSL)?
- quantitative tiling methods (combinatorics) for quantitative rewriting?

Conclusions

- modern confluence techniques powerful; solve problem of the calissons
- local semi-lattice (LSL = LUC with ${}^{\phi}_{X}\diamond^{\psi}_{\Upsilon}$ iff ${}^{\psi}_{\Upsilon}\diamond^{\phi}_{X}$) $\Longrightarrow$ filling iff projection for term rewriting and positive braids
- productivity instead of termination of $\to$ for filling iff projection (given LSL)?
- quantitative tiling methods for quantitative rewriting?
- contrapositive of LSL for non-confluence? Dehornoy et al. 15; Klop 24

Conclusions

- modern confluence techniques powerful; solve problem of the calissons
- local semi-lattice (LSL = LUC with ${}_{X}^{\phi}\diamond_{\Upsilon}^{\psi}$ iff ${}_{\Upsilon}^{\psi}\diamond_{X}^{\phi}$) $\implies$ filling iff projection for term rewriting and positive braids
- productivity instead of termination of $\rightarrow$ for filling iff projection (given LSL)?
- quantitative tiling methods for quantitative rewriting?
- contrapositive of LSL for non-confluence?

Thanks to

Jan Willem Klop for suggesting to model calissons by rewriting as in (1),(2)

Conclusions

- modern confluence techniques powerful; solve problem of the calissons
- local semi-lattice (LSL = LUC with ${}_{X}^{\phi}\diamond_{\Upsilon}^{\psi}$ iff ${}_{\Upsilon}^{\psi}\diamond_{X}^{\phi}$) $\implies$ filling iff projection for term rewriting and positive braids
- productivity instead of termination of $\rightarrow$ for filling iff projection (given LSL)?
- quantitative tiling methods for quantitative rewriting?
- contrapositive of LSL for non-confluence?

Thanks to

Jan Willem Klop for suggesting to model the problem of the calissons by rewriting
Nicolai Kraus, Yves Guiraud for discussion on Newman's II-Lemma (see paper)

Conclusions

- modern confluence techniques powerful; solve problem of the calissons
- local semi-lattice (LSL = LUC with ${}^{\phi}_{\chi}\diamond^{\psi}_{\Upsilon}$ iff ${}^{\psi}_{\Upsilon}\diamond^{\phi}_{\chi}$) $\implies$ filling iff projection for term rewriting and positive braids
- productivity instead of termination of $\to$ for filling iff projection (given LSL)?
- quantitative tiling methods for quantitative rewriting?
- contrapositive of LSL for non-confluence?

Thanks to

Jan Willem Klop for suggesting to model calissons by rewriting
Nicolai Kraus, Yves Guiraud for discussion on Newman's II-Lemma
Nils for example generator and filler (app needs WebGL)

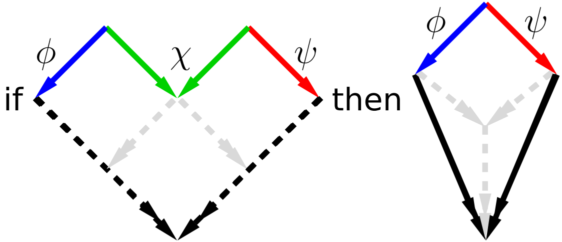
Conclusions

- modern confluence techniques powerful; solve problem of the calissons
- local semi-lattice (LSL = LUC with ${}^{\phi}_{\chi}\diamond^{\psi}_{\Upsilon}$ iff ${}^{\psi}_{\Upsilon}\diamond^{\phi}_{\chi}$) $\implies$ filling iff projection for term rewriting and positive braids
- productivity instead of termination of $\rightarrow$ for filling iff projection (given LSL)?
- quantitative tiling methods for quantitative rewriting?
- contrapositive of LSL for non-confluence?

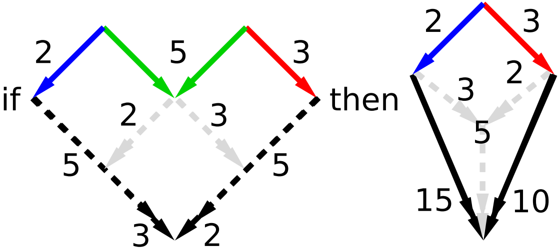
Thanks to

Jan Willem Klop for suggesting to model calissons by rewriting
Nicolai Kraus, Yves Guiraud for discussion on Newman's II-Lemma
Nils for example generator and filler
you for your interest

Local undercutting / semi-lattice



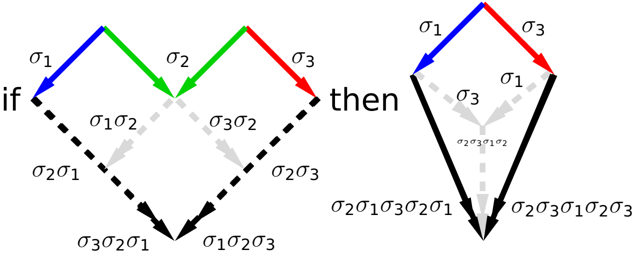
LSL for least upperbounds



Example (positive natural numbers with multiplication)

30 is an upperbound of $\text{lcm}(2,5)$ and $\text{lcm}(5,3)$
and is so too of $\text{lcm}(2,3)$ obtained by cutting 5
($\text{lcm}(2,3)$ undercuts the upperbound 30 of $\text{lcm}(2,5)$ and $\text{lcm}(5,3)$)

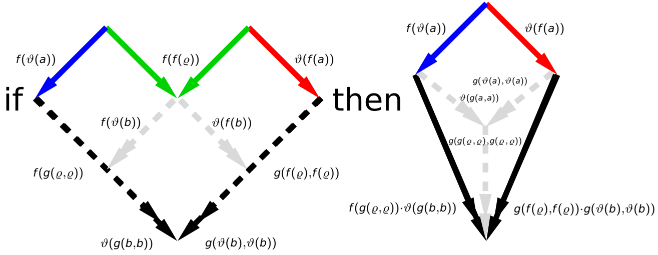
LSL for least common multiples



Example (positive braids; Dehornoy et al. 15, Example II.4.20)

$\sigma_1\sigma_2\sigma_1\sigma_3\sigma_2\sigma_1$ is a common multiple of $\mathrm{lcm}(\sigma_1, \sigma_2)$ and $\mathrm{lcm}(\sigma_2, \sigma_3)$
and is so too of $\mathrm{lcm}(\sigma_1, \sigma_3)$ obtained by cutting σ_2
(with $\sigma_1\sigma_2\sigma_1 = \sigma_2\sigma_1\sigma_2$, $\sigma_1\sigma_3 = \sigma_3\sigma_1$, $\sigma_1\sigma_2\sigma_1 = \sigma_2\sigma_1\sigma_2$ on Artin generators σ_i)

LSL for orthogonality



Example (orthogonal TRSs; Terese 03, Figure 8.53)

$g(g(b,b),g(b,b))$ is a common reduct of $f(\vartheta(a))^{-1}\cdot f(f(\varrho))$ and $f(f(\varrho))^{-1}\cdot\vartheta(f(a))$ is so too of $f(\vartheta(a))^{-1}\cdot\vartheta(f(a))$ obtained by cutting $f(f(\varrho))$
(for OTRS rules $\varrho : a \to b$ and $\vartheta : f(x) \to g(x)$)