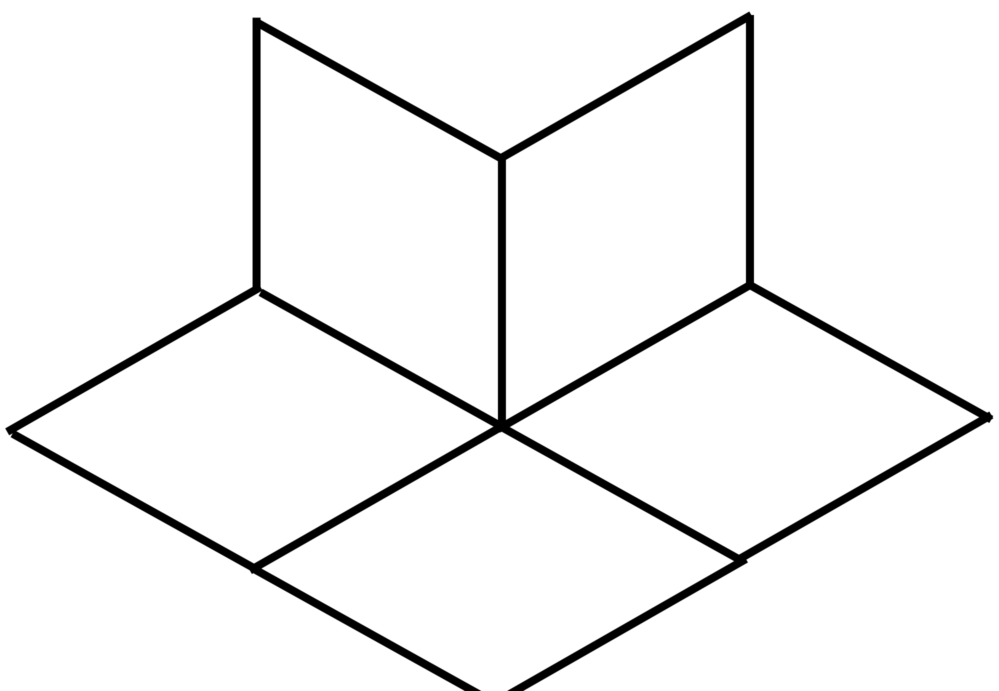
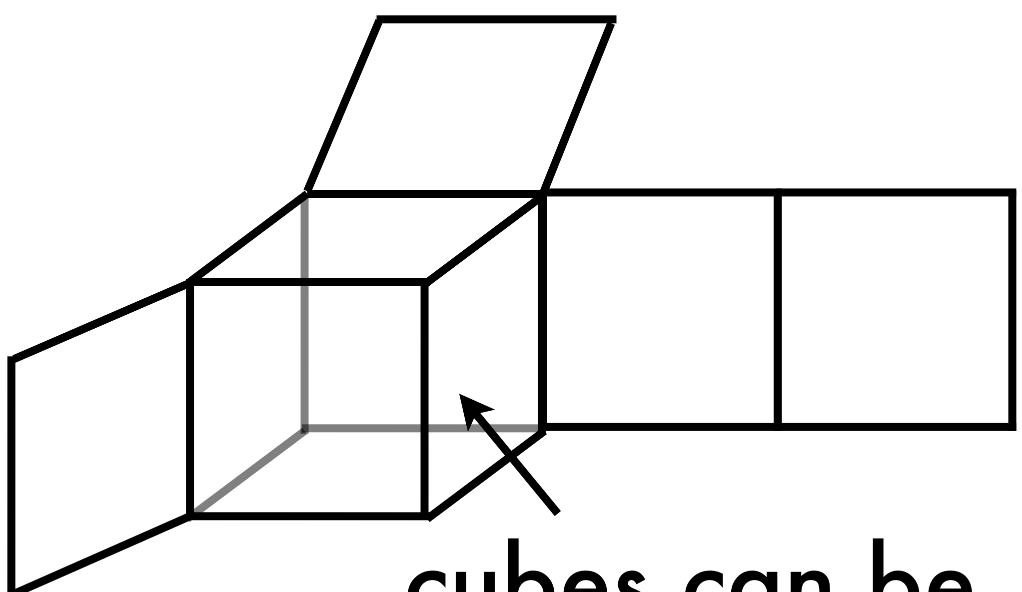
Geodesics in CAT(0) Cubical Complexes

Megan Owen (Fields Institute)
joint with F. Ardila (SFSU), S. Sullivant (NCSU)

Cubical Complexes

- cubical complex = polyhedral complex of unit cubes + all attaching maps are injective
- metric on cubical complex induced by Euclidean L^2 metric on each cube

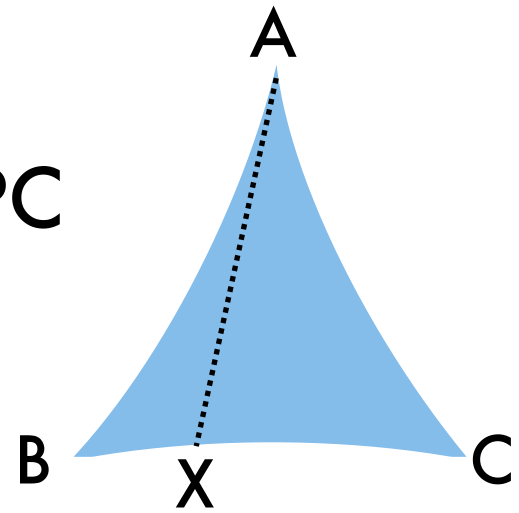




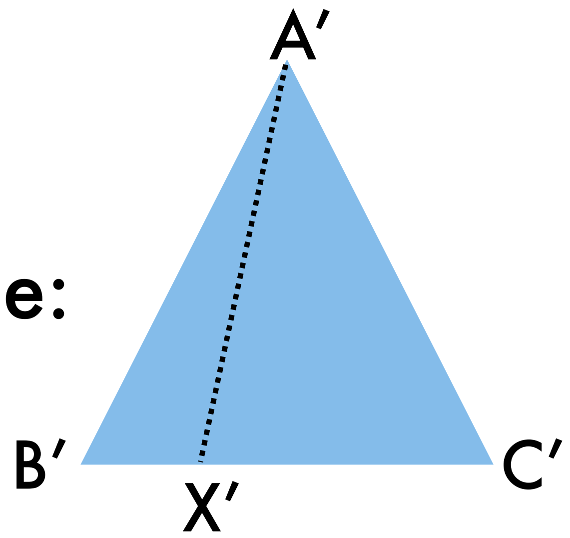
CAT(0)

- *non-positive curvature (NPC)* = triangles are at least as thin as in Euclidean space
- *global non-positive curvature* = all triangles are at least as thin as in Euclidean space = CAT(0)

triangle in a NPC space:



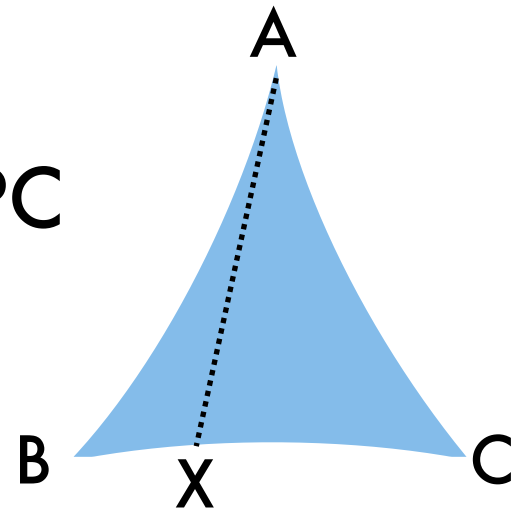
Euclidean comparison triangle:



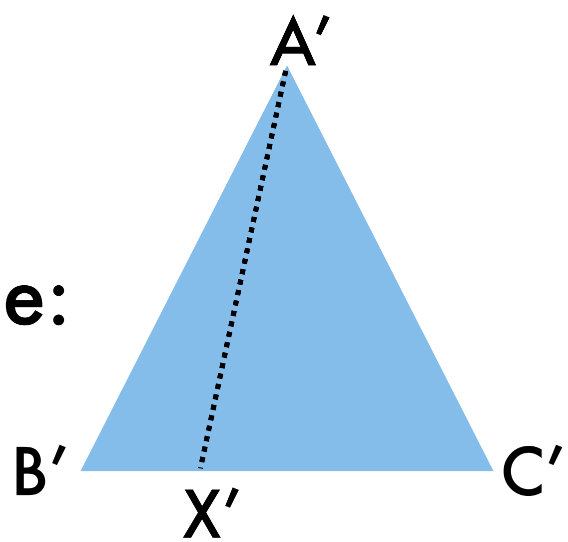
CAT(0)

- *non-positive curvature (NPC)* = triangles are at least as thin as in Euclidean space
- *global non-positive curvature* = all triangles are at least as thin as in Euclidean space = CAT(0)

triangle in a NPC space:



Euclidean comparison triangle:

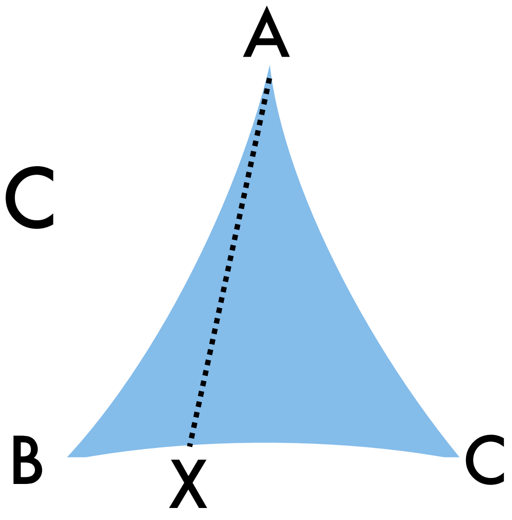


$d(A,X) \leq d'(A',X')$

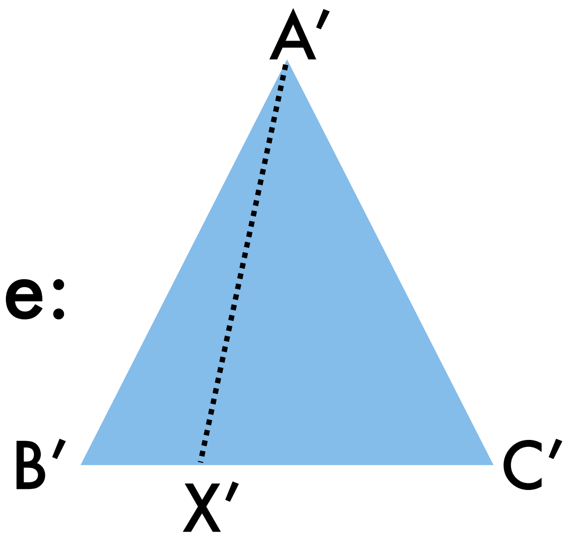
CAT(0)

- *non-positive curvature (NPC)* = triangles are at least as thin as in Euclidean space
- *global non-positive curvature* = all triangles are at least as thin as in Euclidean space = CAT(0)

triangle in a NPC space:



Euclidean comparison triangle:



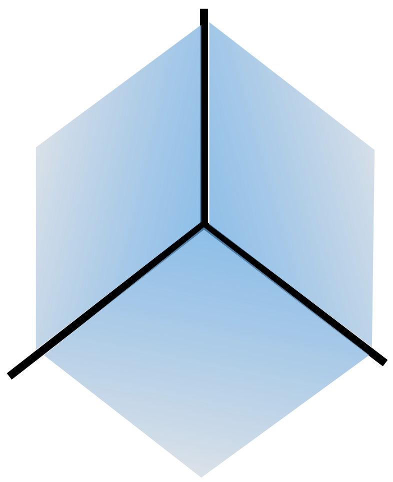
$d(A,X) \leq d'(A',X')$

- CAT(0) ⇒ unique shortest paths (*geodesics*)

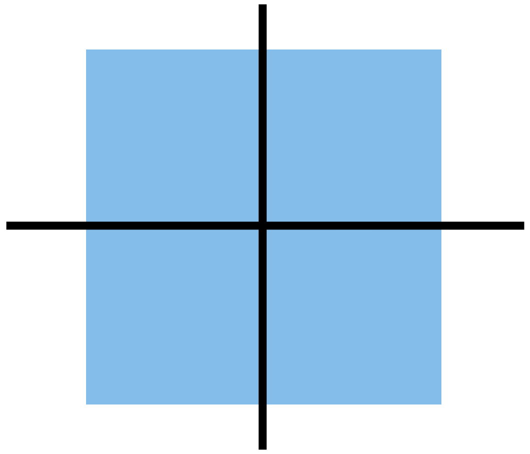
CAT(0) Cubical Complexes

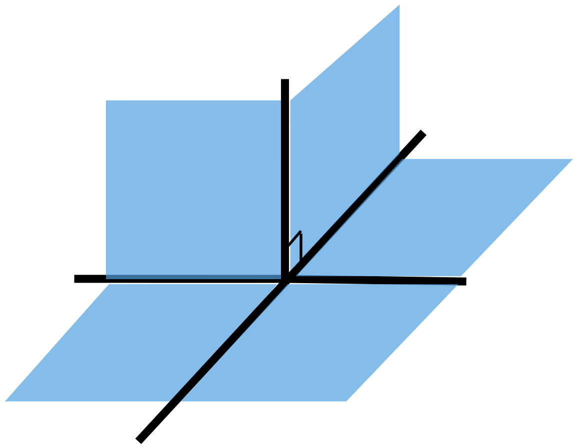
Theorem (Gromov, 1987):
A cubical complex is CAT(0)
$\Leftrightarrow$ it is simply connected and the link of any vertex is a flag simplicial complex

not CAT(0):



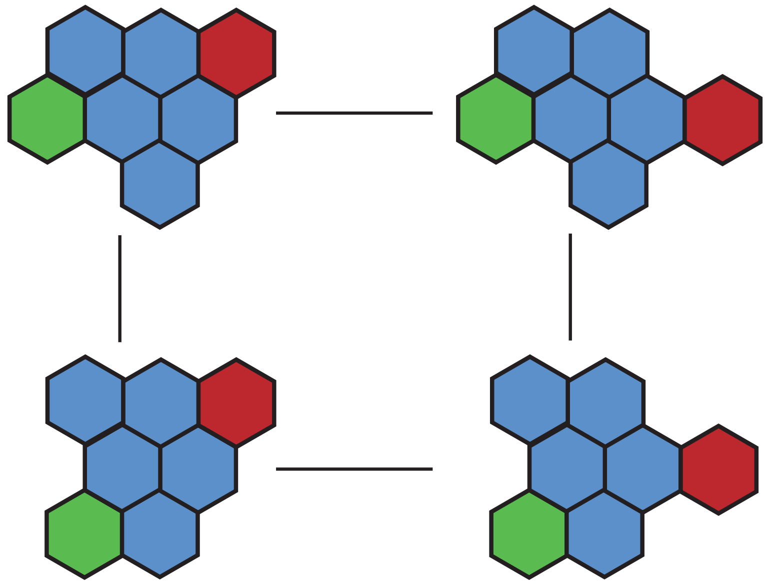
CAT(0):





Applications

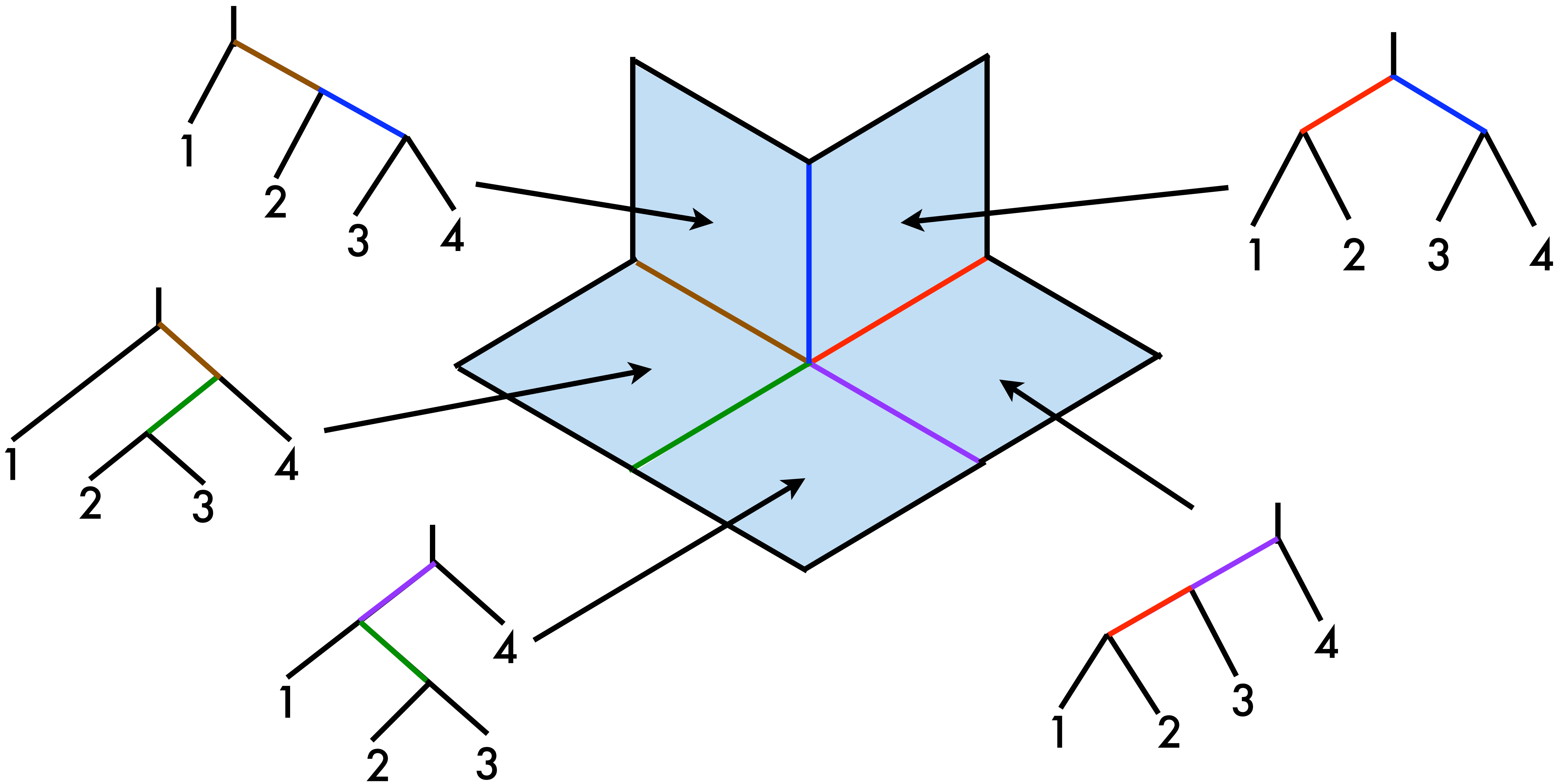
- CAT(0) cubical complexes appear in:
  - geometric group theory
  - reconfigurable systems:
    - robots perform discrete, reversible moves
    - moves represented as edges in the complex



Ghrist and Peterson, 2007

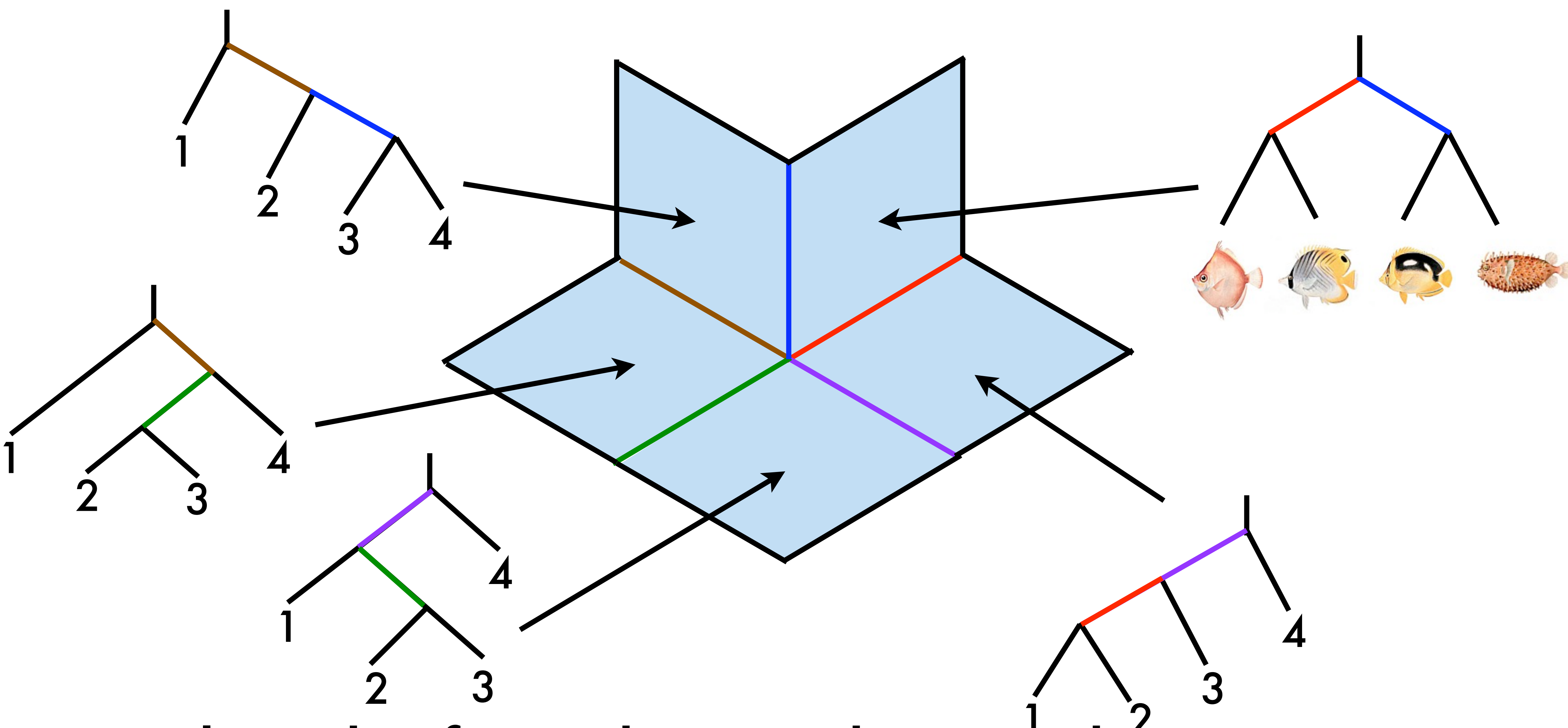
Application

Theorem (Billera, Holmes, Vogtmann, 2001):
The space of metric trees is a CAT(0) cubical complex.



Application: Phylogenetics

Theorem (Billera, Holmes, Vogtmann, 2001):
The space of metric trees is a CAT(0) cubical complex.

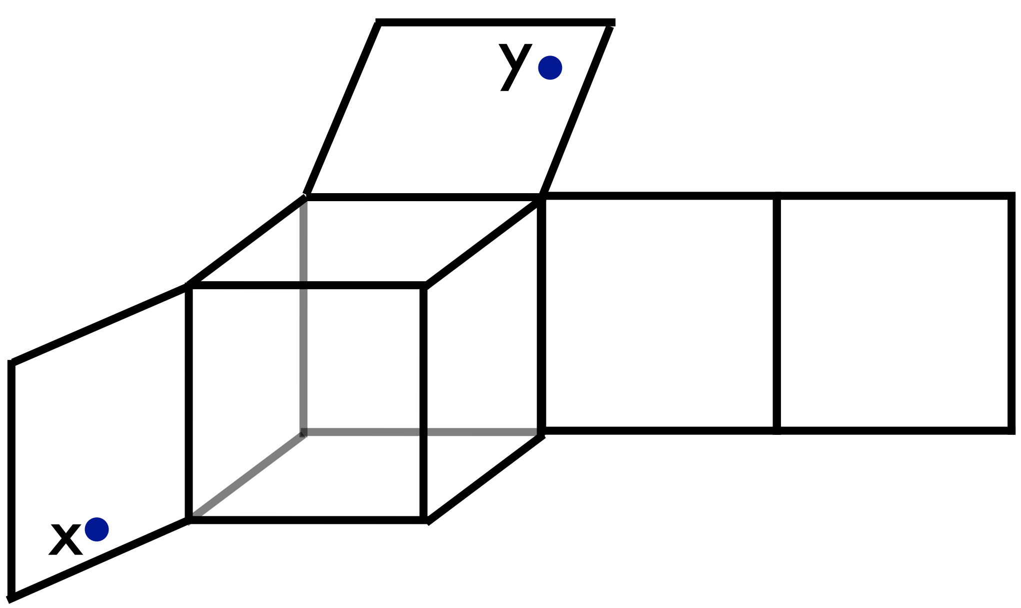


- length of geodesic = distance between trees

Problem

Problem:

Given a CAT(0) cubical complex and two points x and y, find the geodesic from x to y.



Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements
2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.
3. Find geodesic through this cube sequence.
4. If possible, improve cube sequence and repeat from 3.

Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

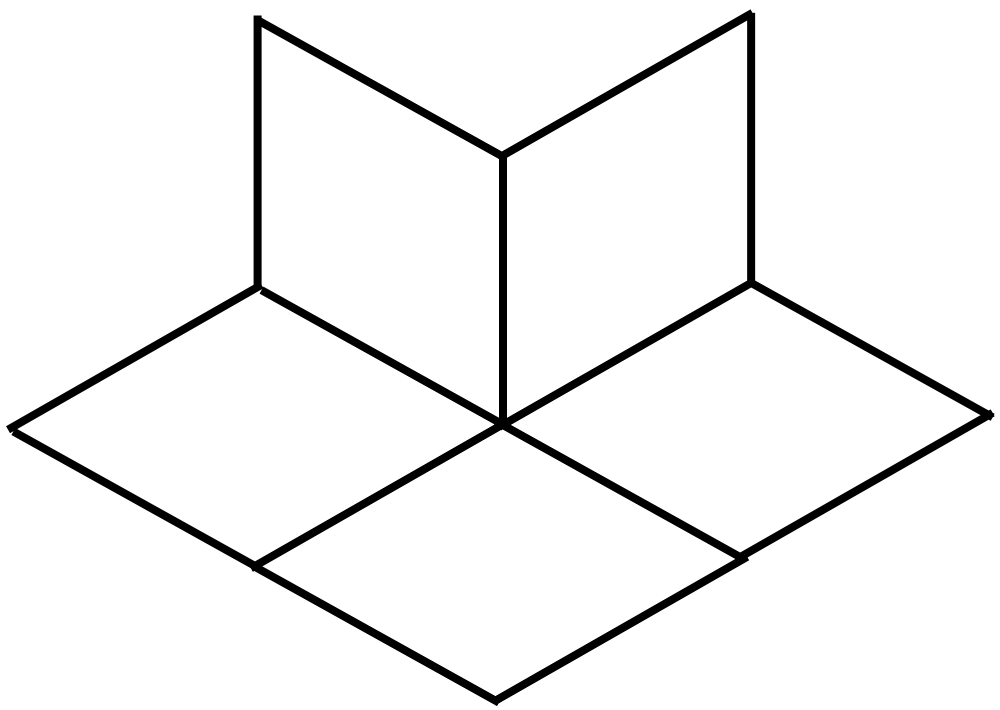
2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

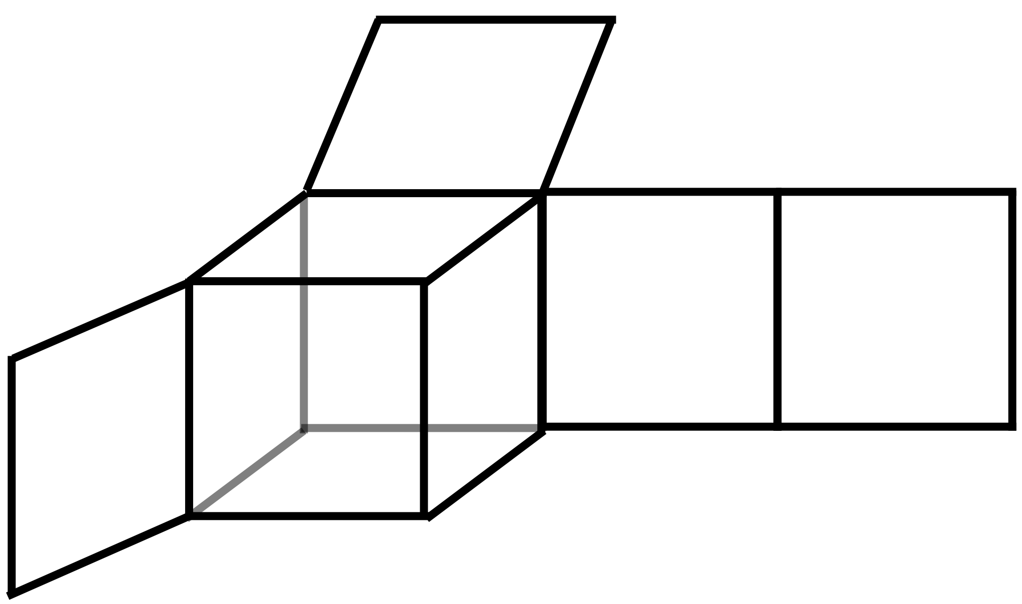
3. Find geodesic through this cube sequence.

4. If possible, improve cube sequence and repeat from 3.

1. Poset Representation

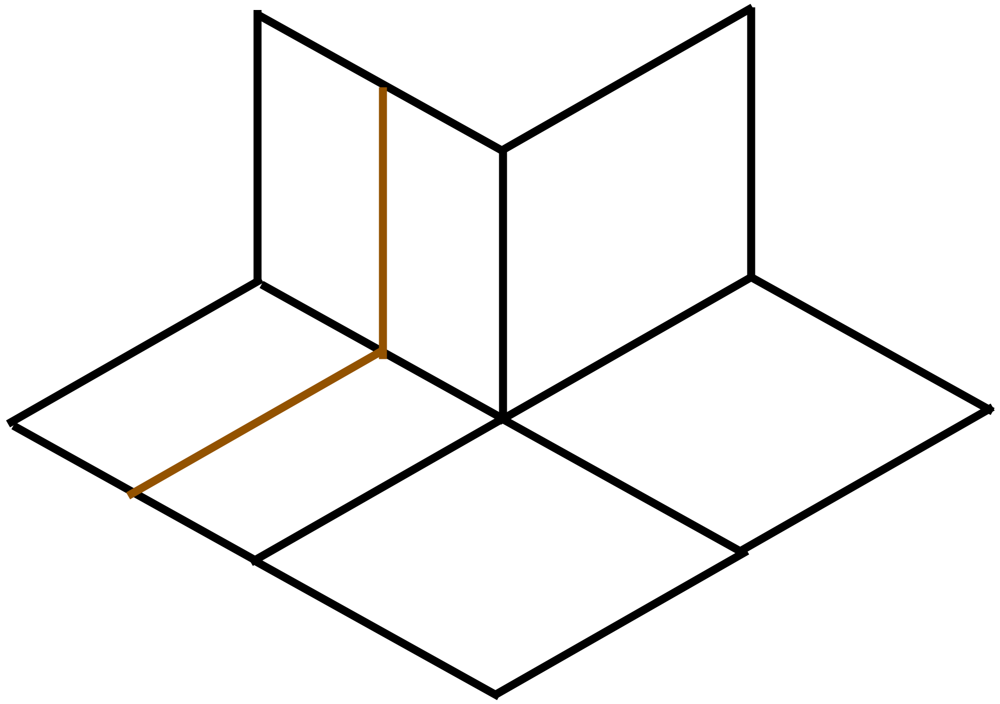
- goal: represent cube complex as a poset to induce coordinate system
- associate each cube edge with the perpendicular "hyperplane" that bisects it
- hyperplanes act as coordinates

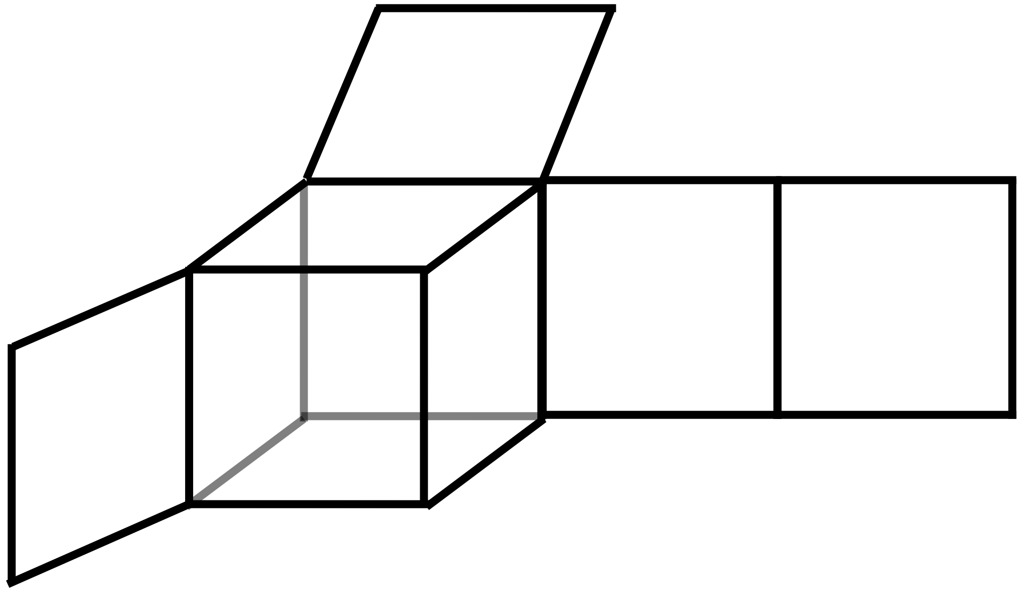




1. Poset Representation

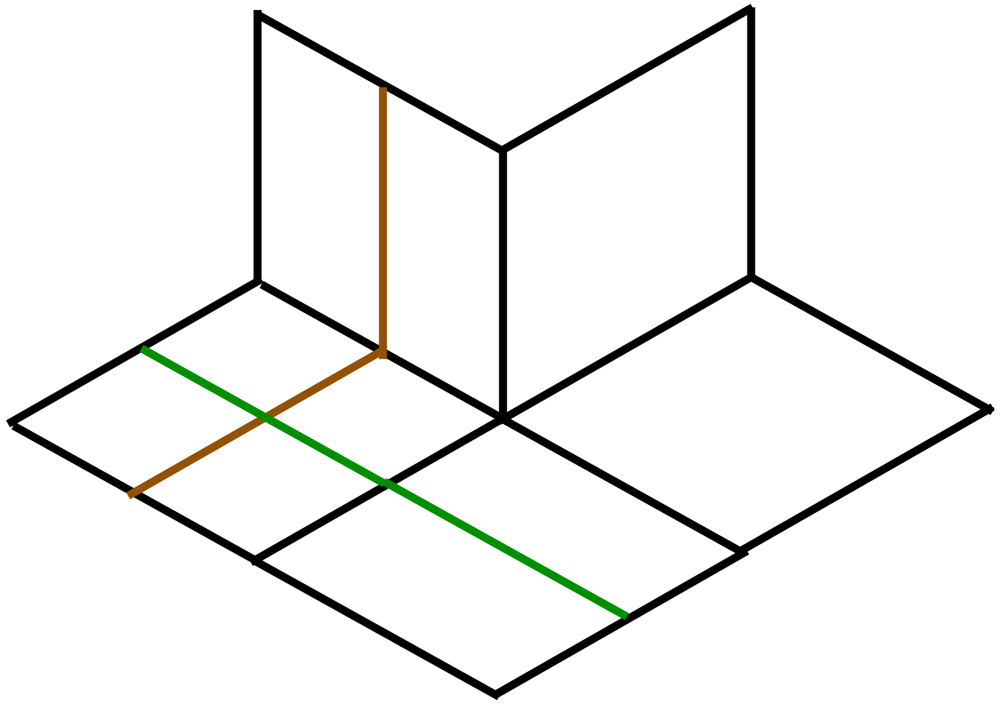
- goal: represent cube complex as a poset to induce coordinate system
- associate each cube edge with the perpendicular "hyperplane" that bisects it
- hyperplanes act as coordinates

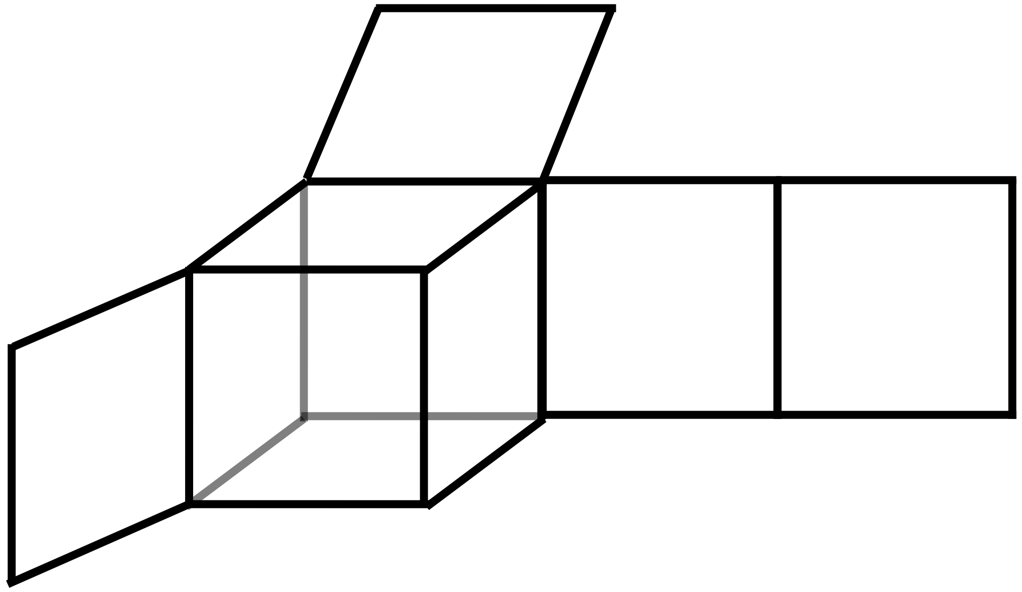




1. Poset Representation

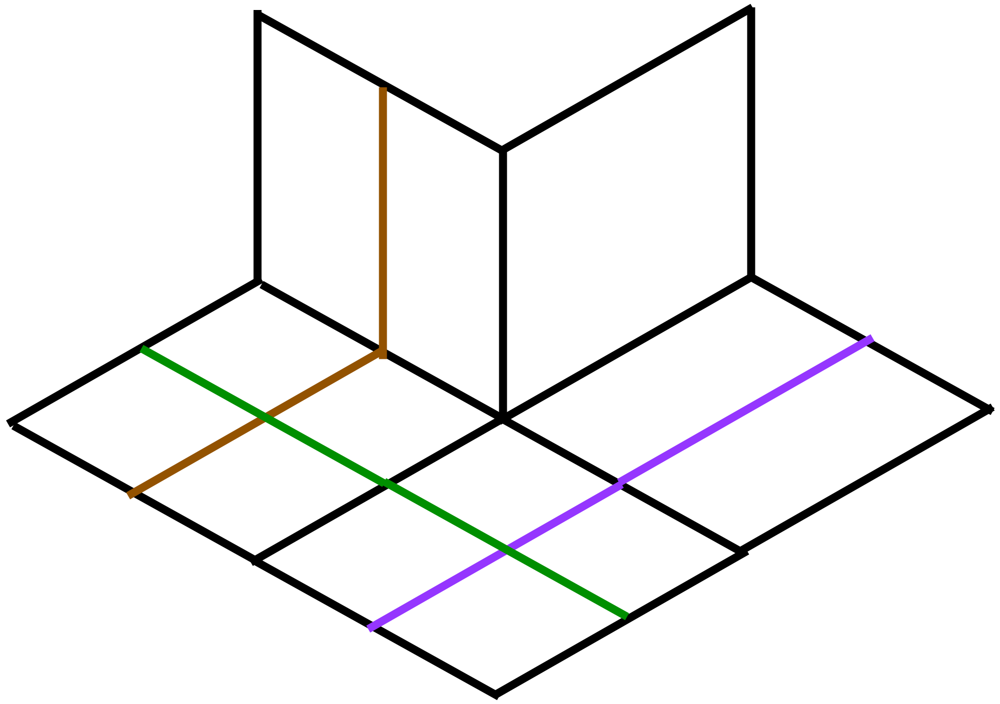
- goal: represent cube complex as a poset to induce coordinate system
- associate each cube edge with the perpendicular "hyperplane" that bisects it
- hyperplanes act as coordinates

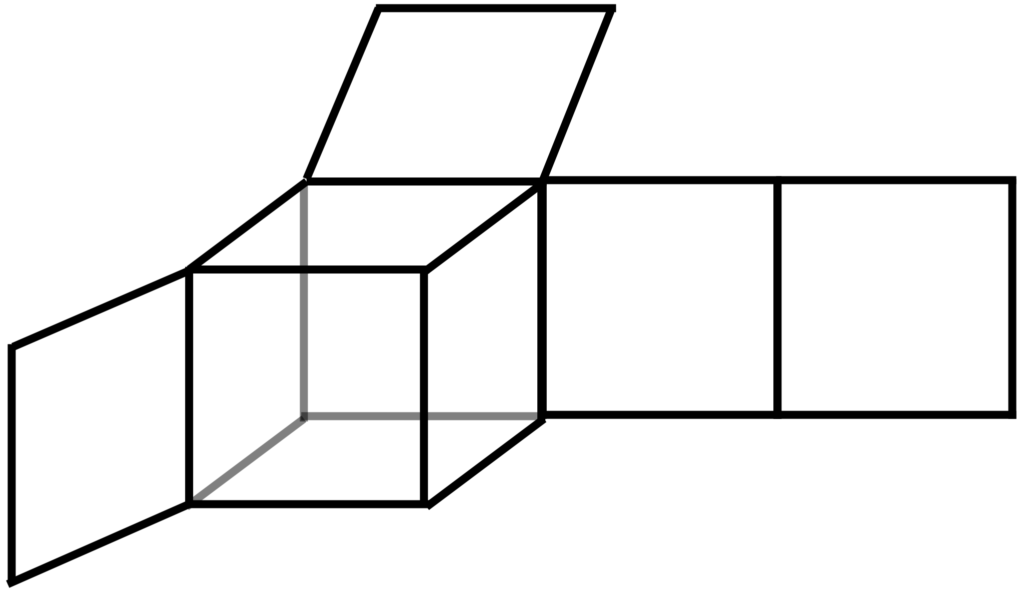




1. Poset Representation

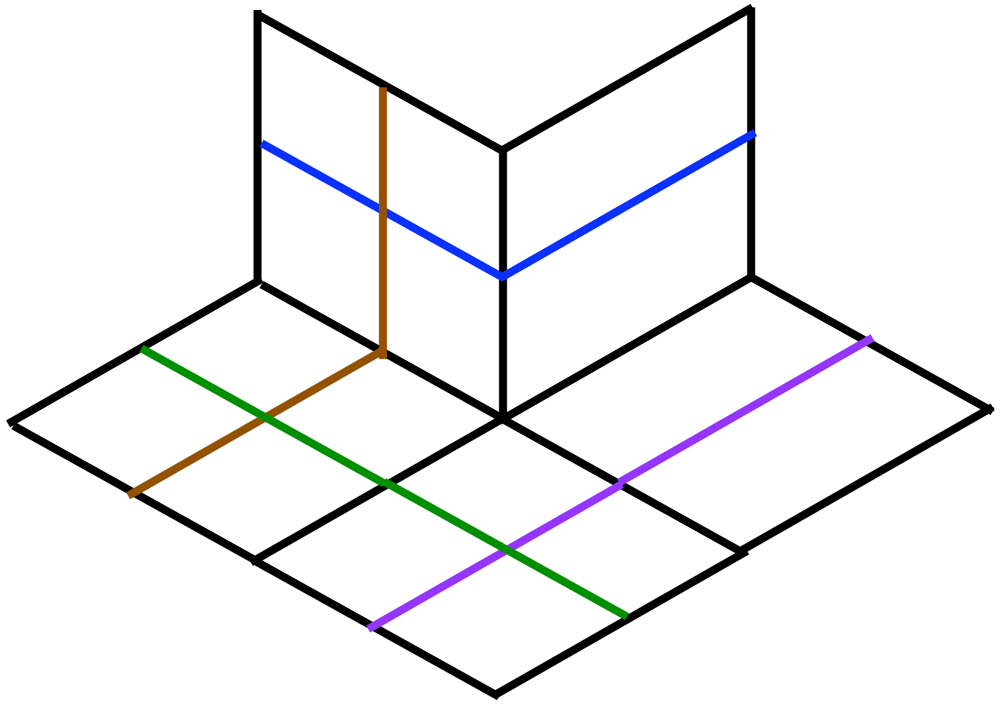
- goal: represent cube complex as a poset to induce coordinate system
- associate each cube edge with the perpendicular "hyperplane" that bisects it
- hyperplanes act as coordinates

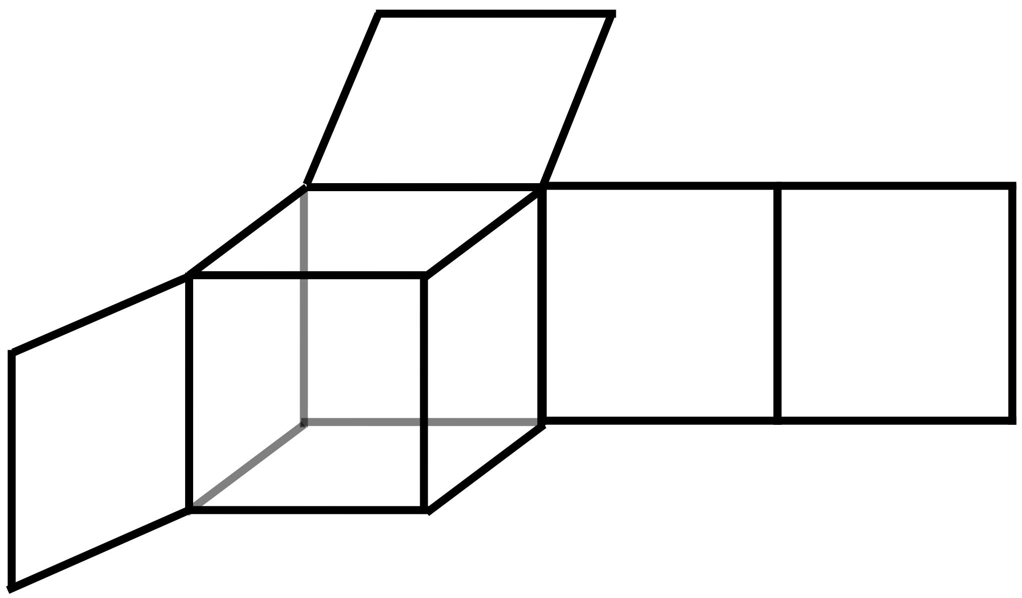




1. Poset Representation

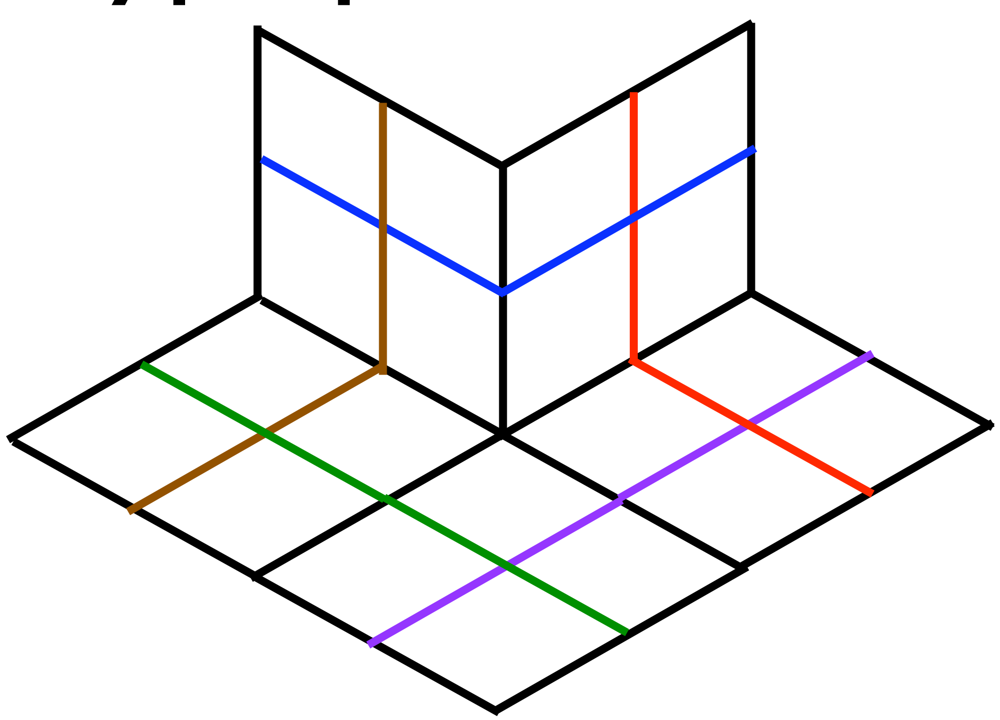
- goal: represent cube complex as a poset to induce coordinate system
- associate each cube edge with the perpendicular "hyperplane" that bisects it
- hyperplanes act as coordinates

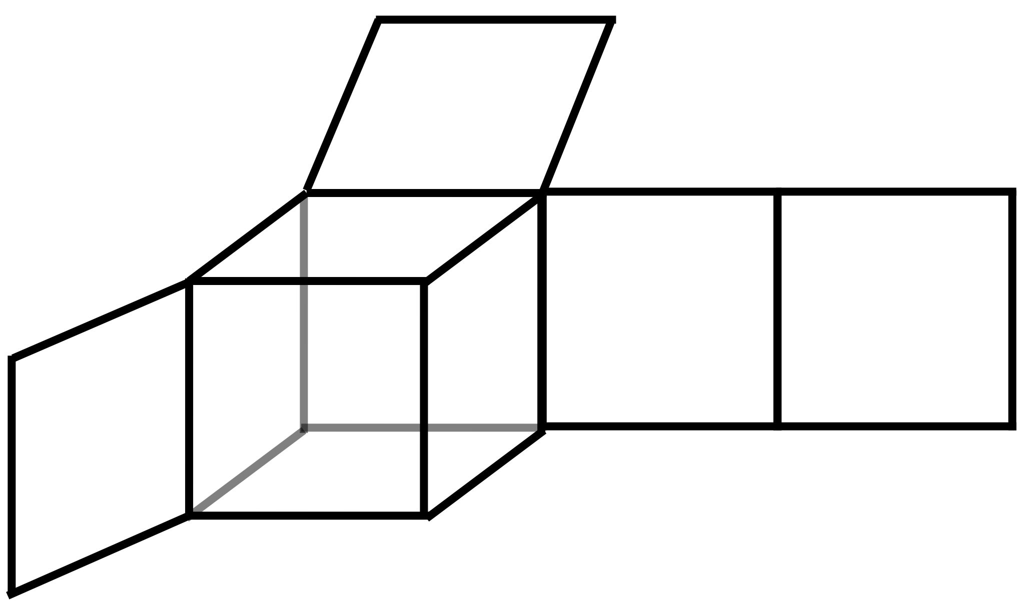




1. Poset Representation

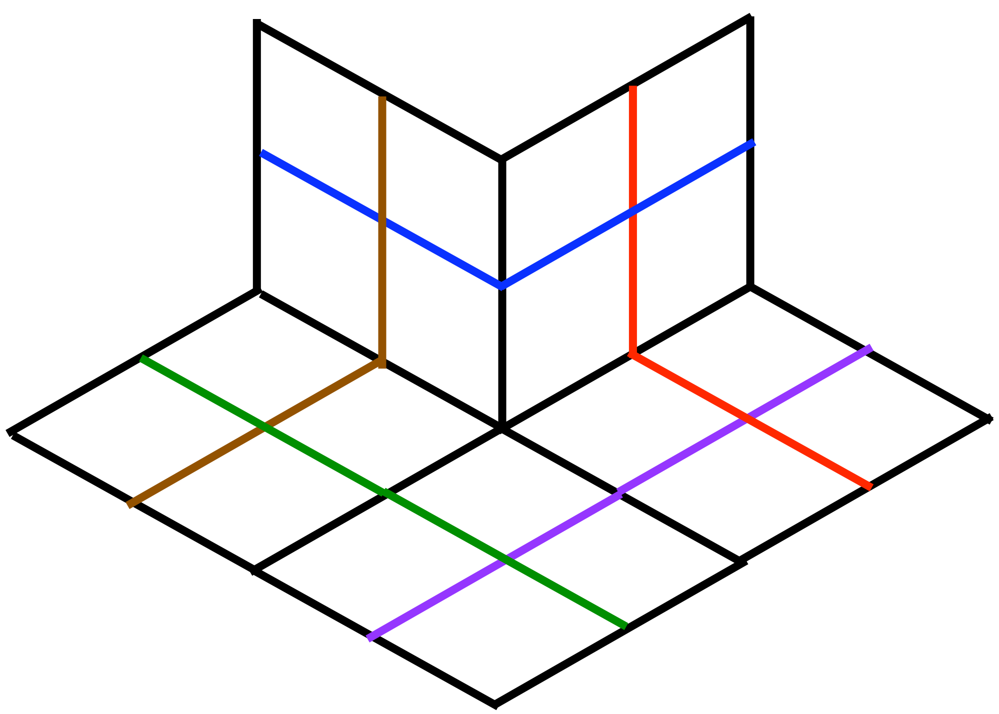
- goal: represent cube complex as a poset to induce coordinate system
- associate each cube edge with the perpendicular "hyperplane" that bisects it
- hyperplanes act as coordinates

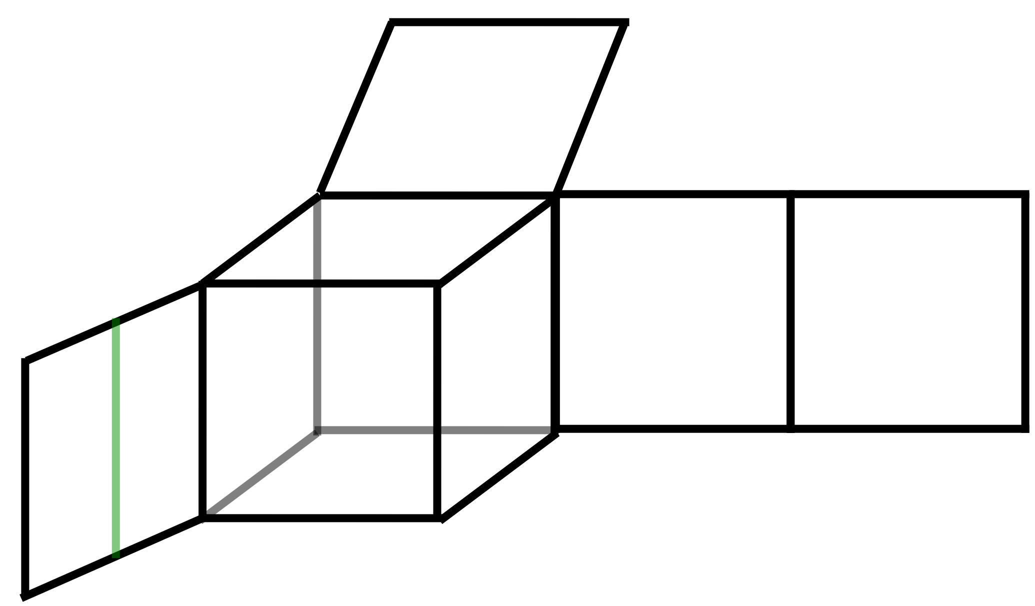




1. Poset Representation

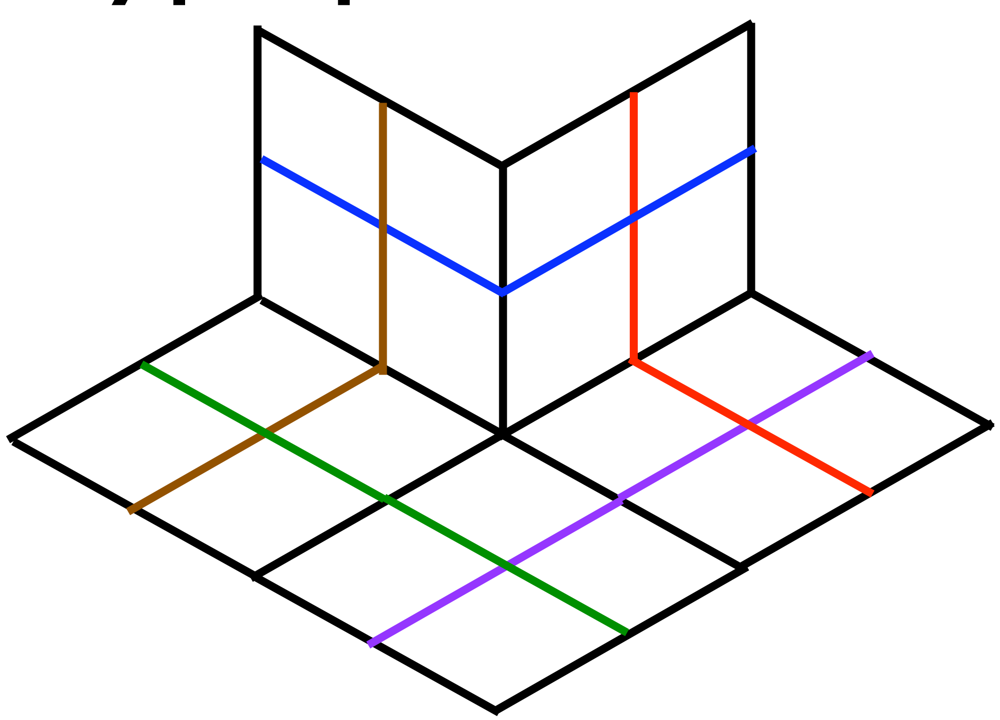
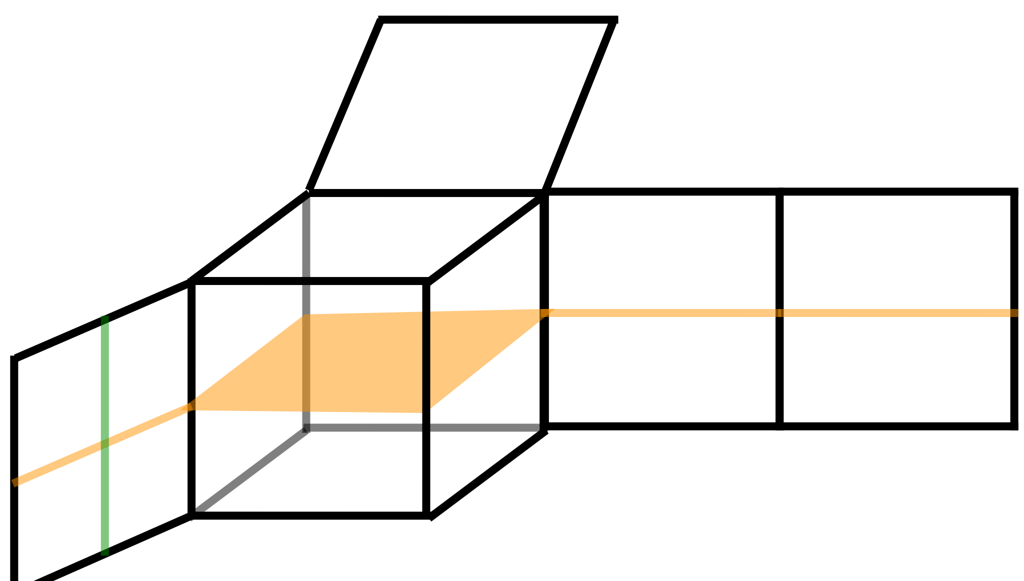
- goal: represent cube complex as a poset to induce coordinate system
- associate each cube edge with the perpendicular “hyperplane” that bisects it
- hyperplanes act as coordinates





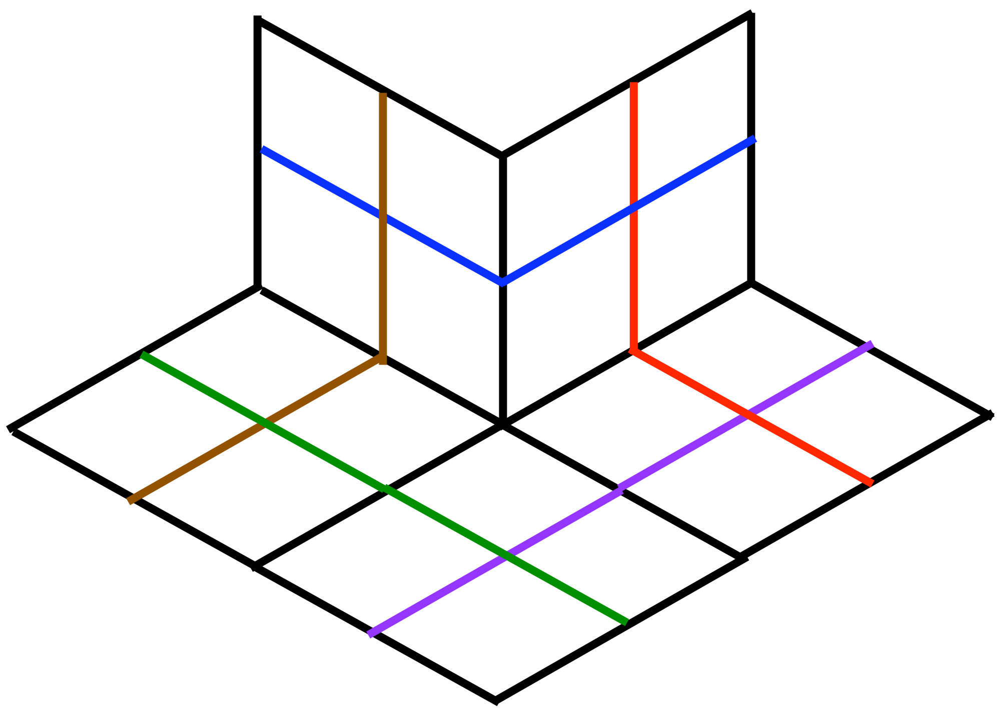
1. Poset Representation

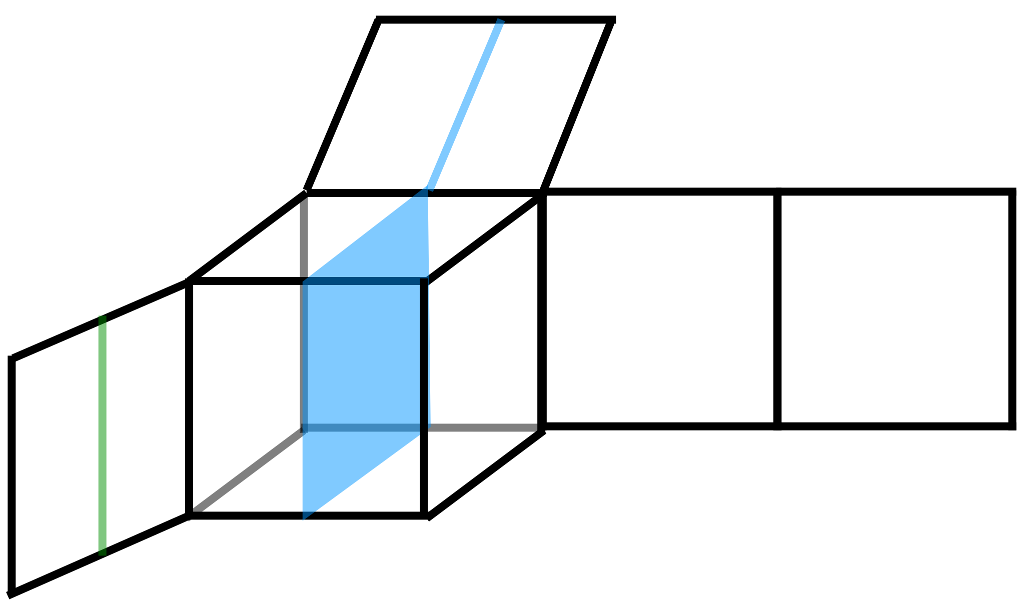
- goal: represent cube complex as a poset to induce coordinate system
- associate each cube edge with the perpendicular “hyperplane” that bisects it
- hyperplanes act as coordinates

1. Poset Representation

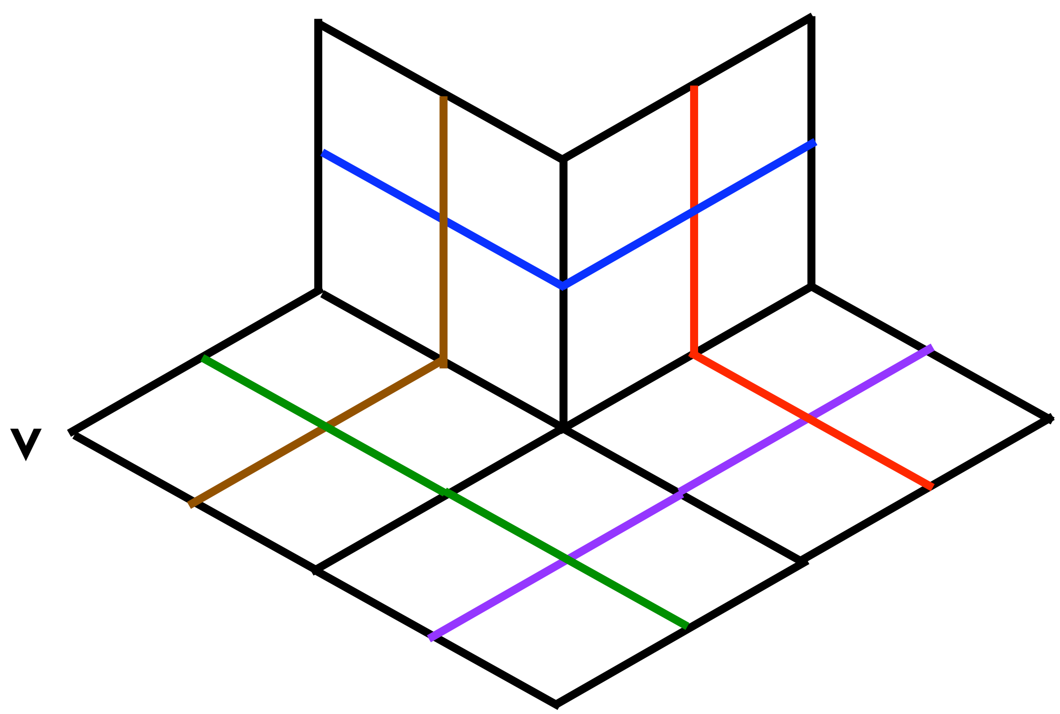
- goal: represent cube complex as a poset to induce coordinate system
- associate each cube edge with the perpendicular "hyperplane" that bisects it
- hyperplanes act as coordinates

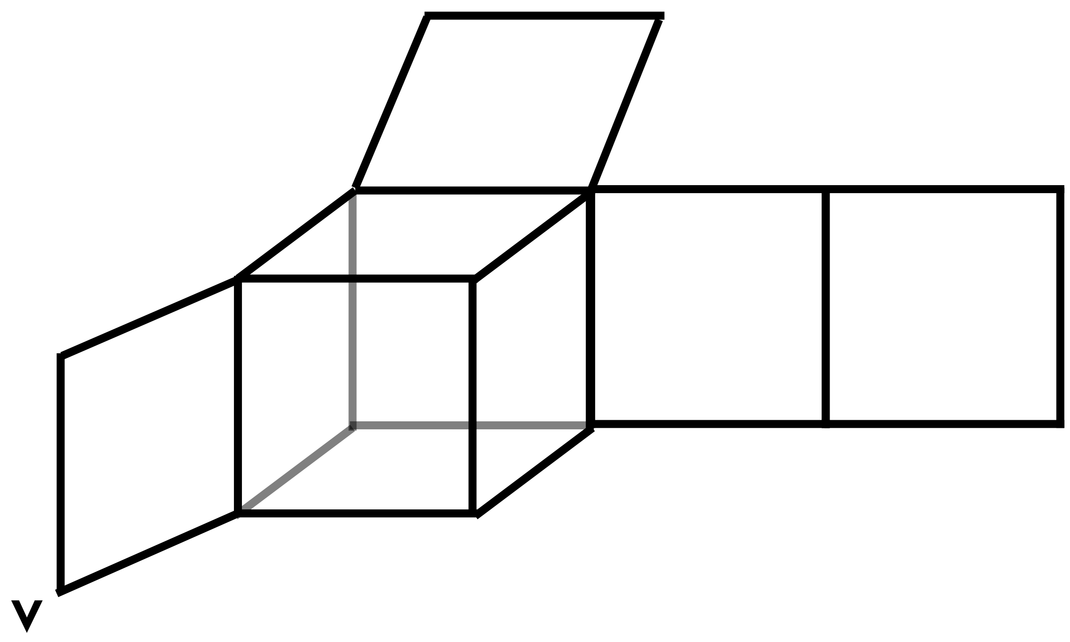




1. Poset Representation

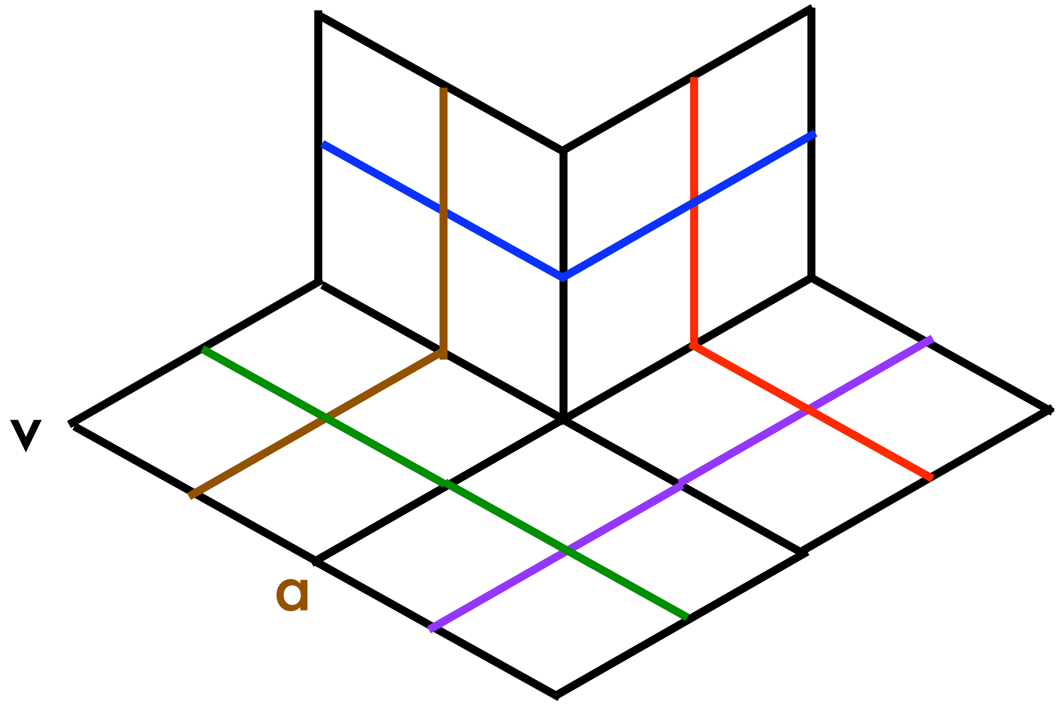
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

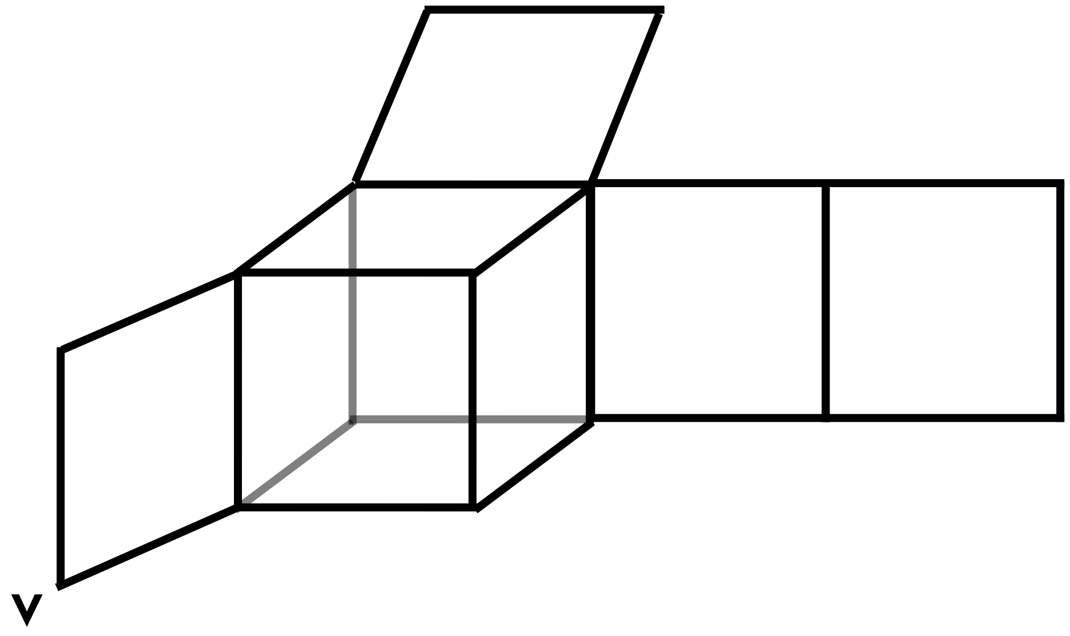




1. Poset Representation

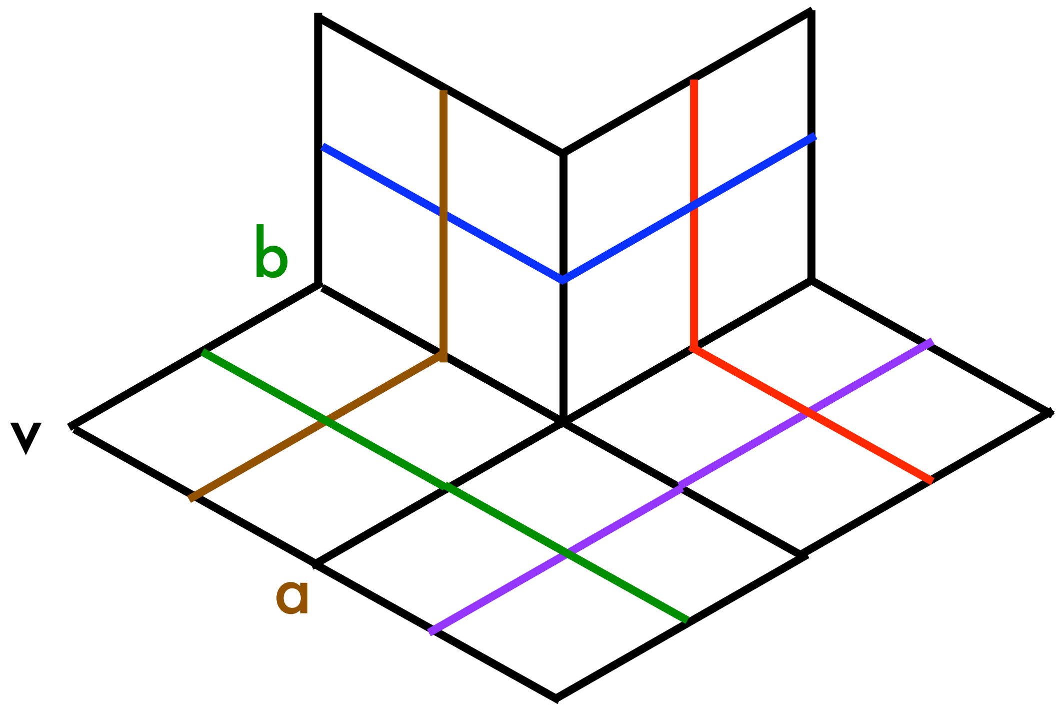
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

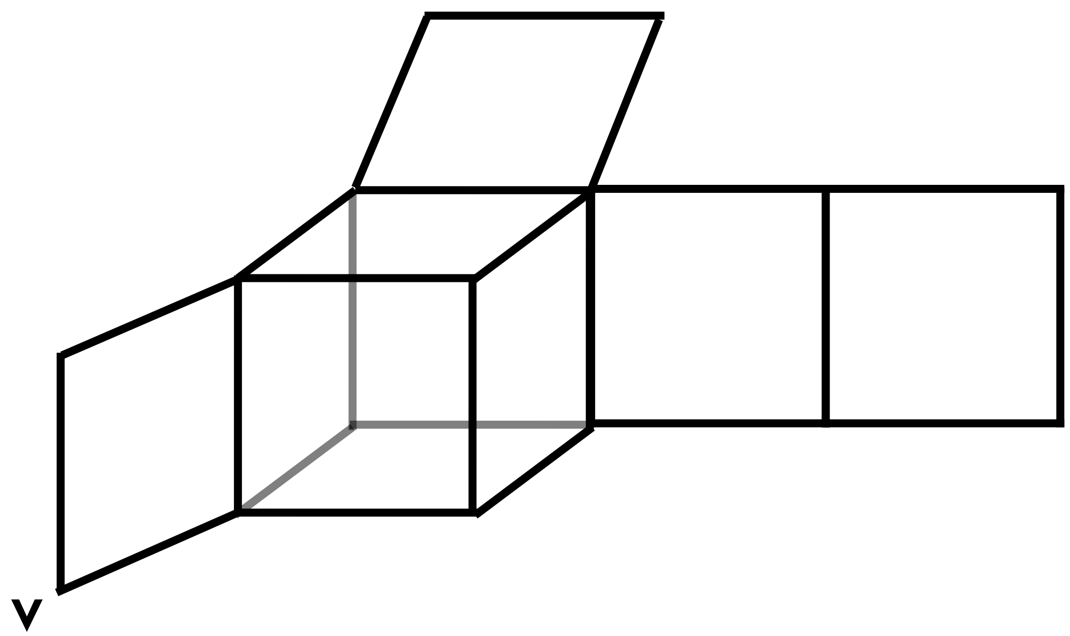




1. Poset Representation

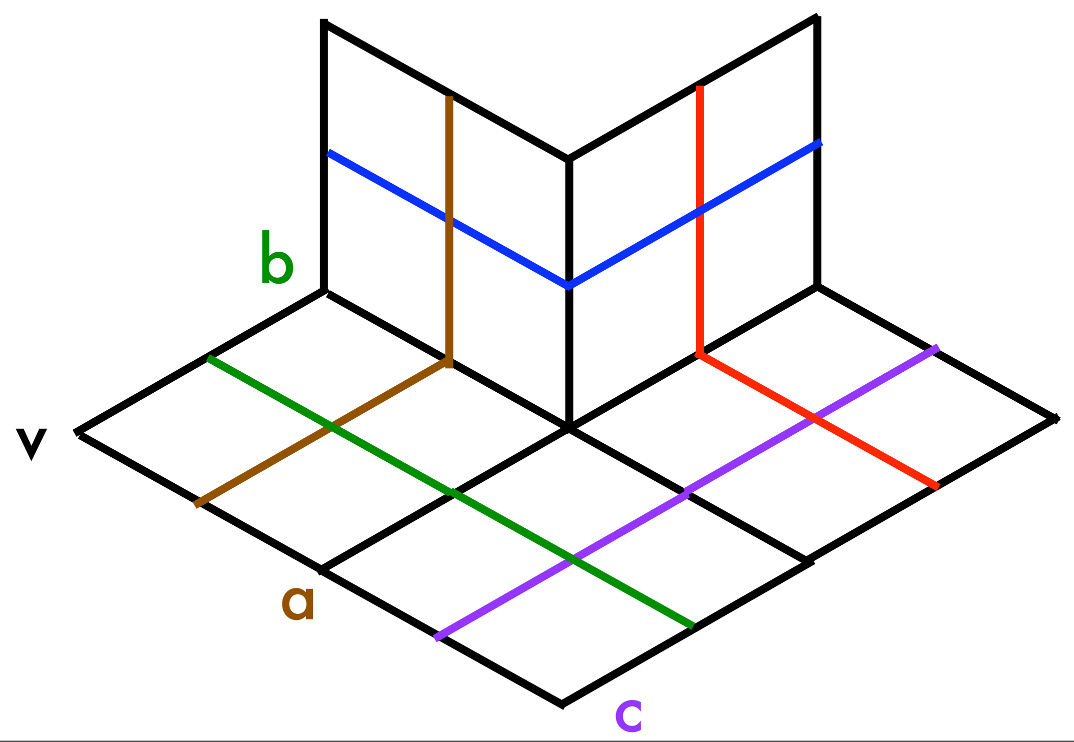
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

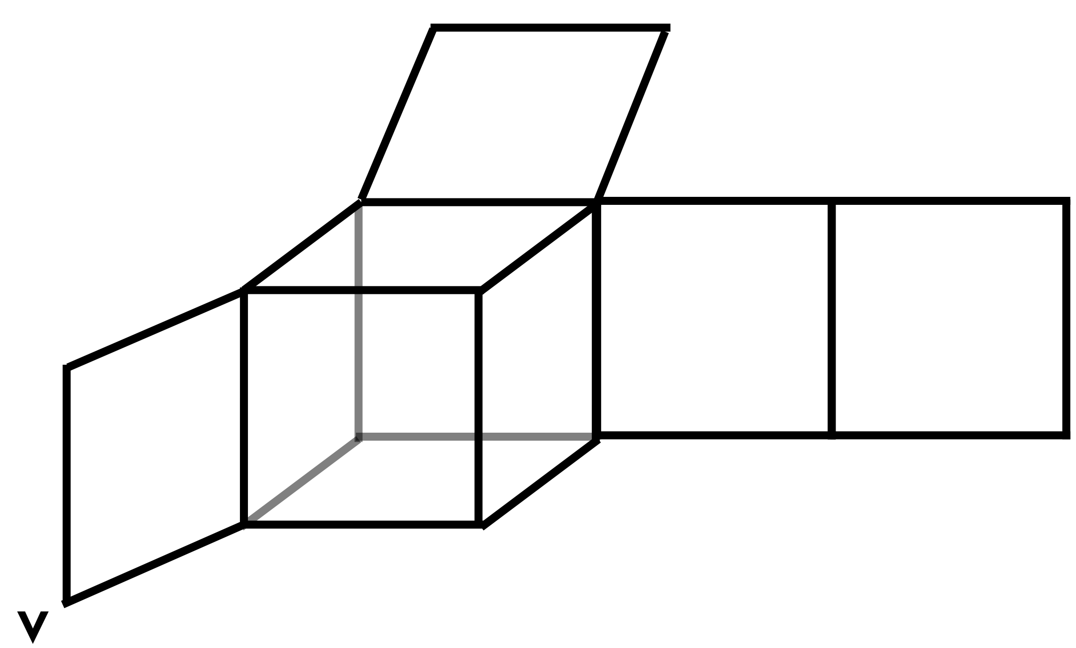




1. Poset Representation

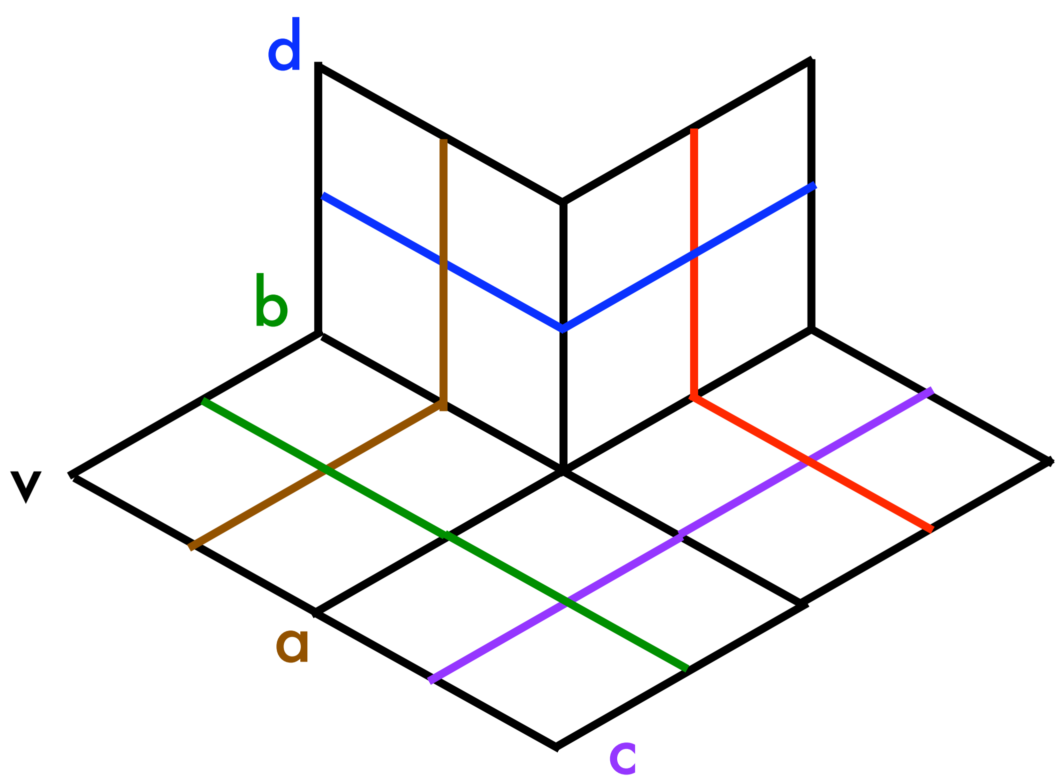
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

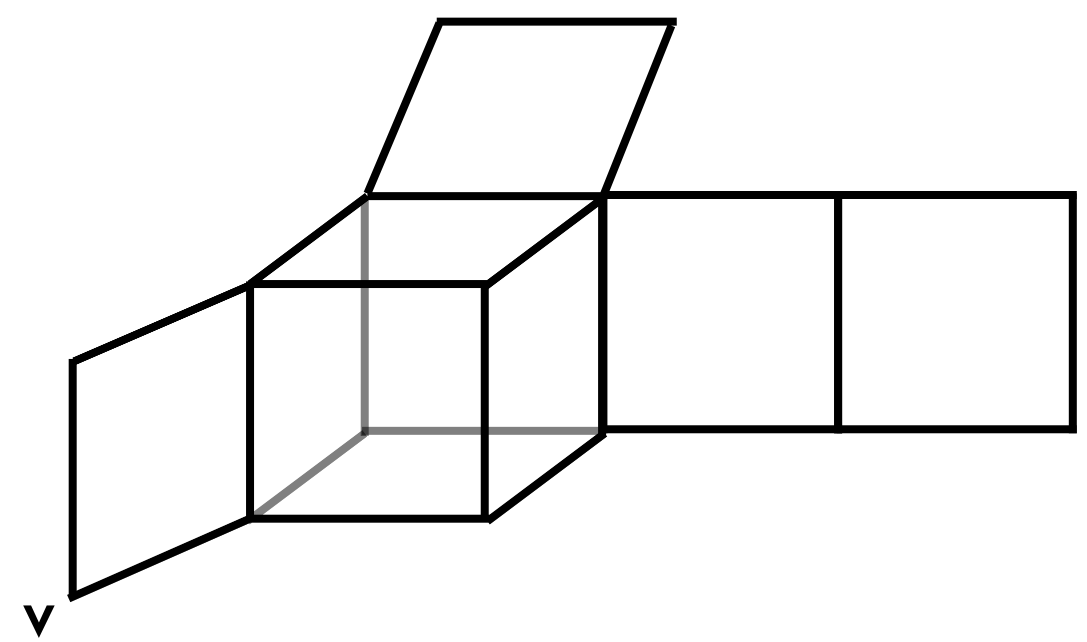




1. Poset Representation

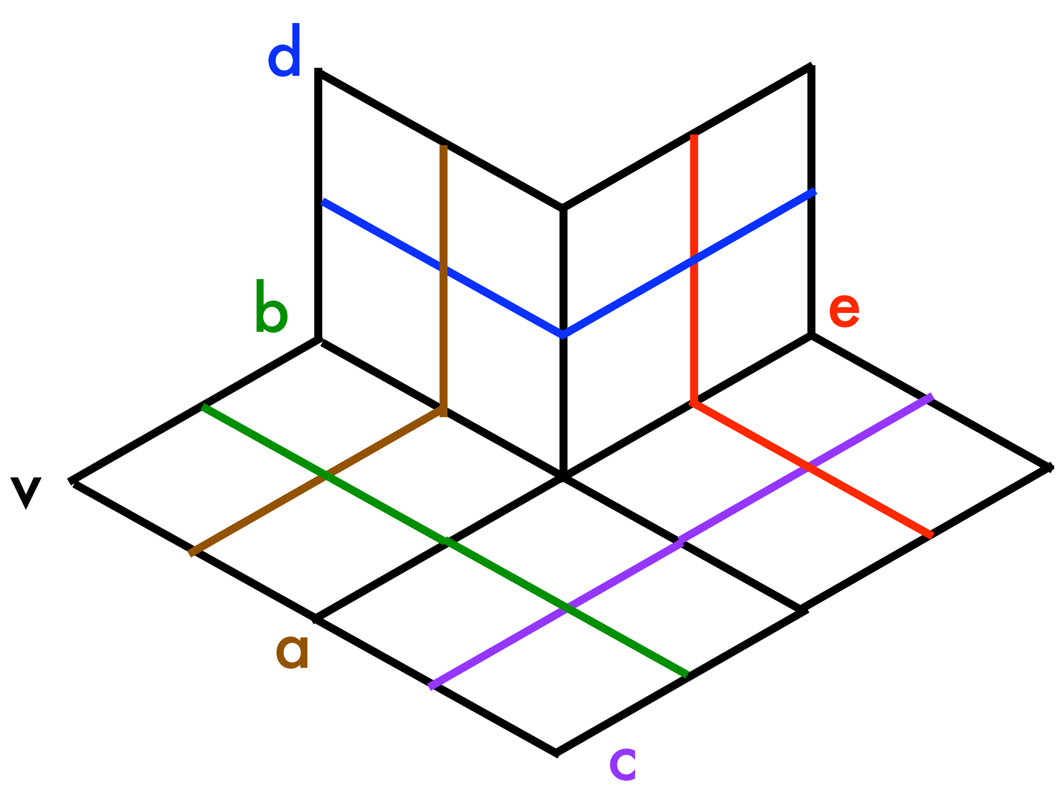
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

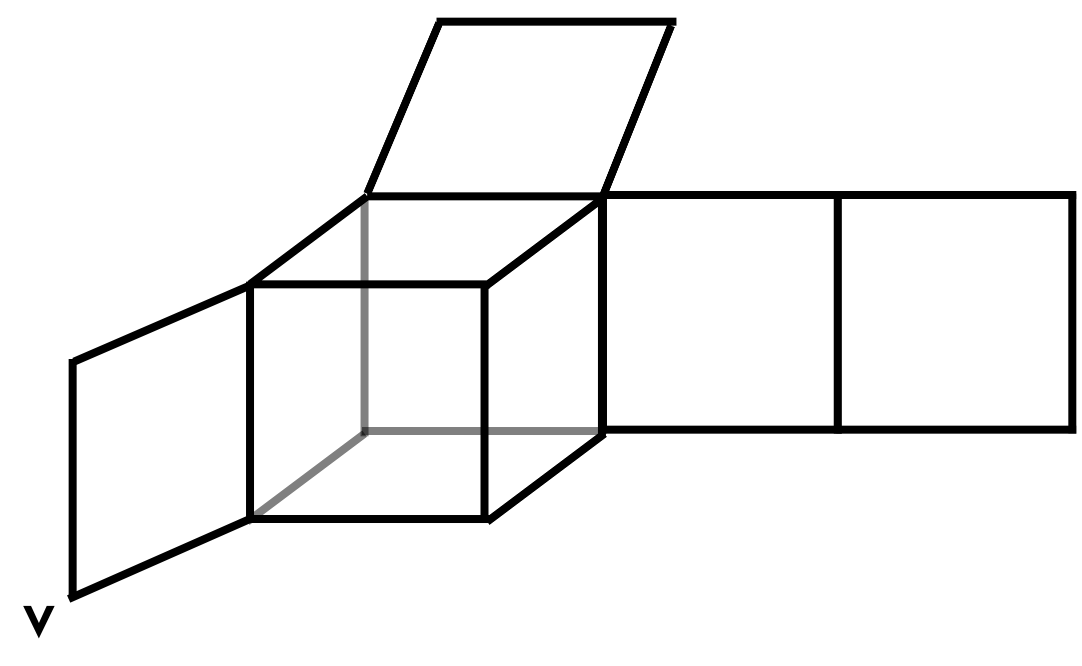




1. Poset Representation

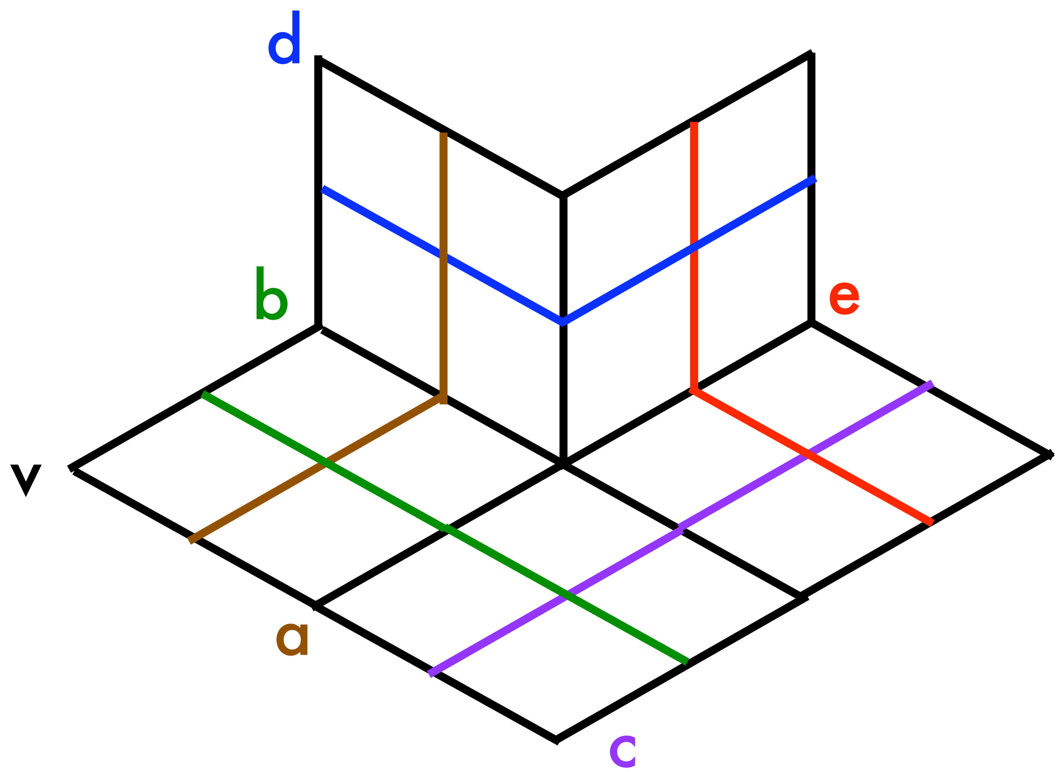
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

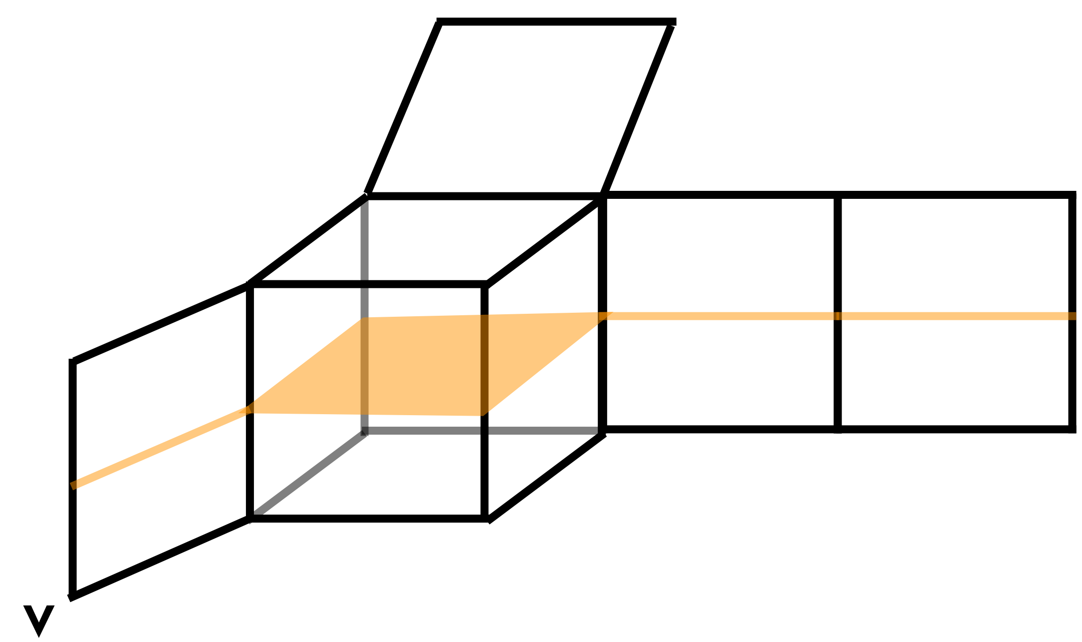




1. Poset Representation

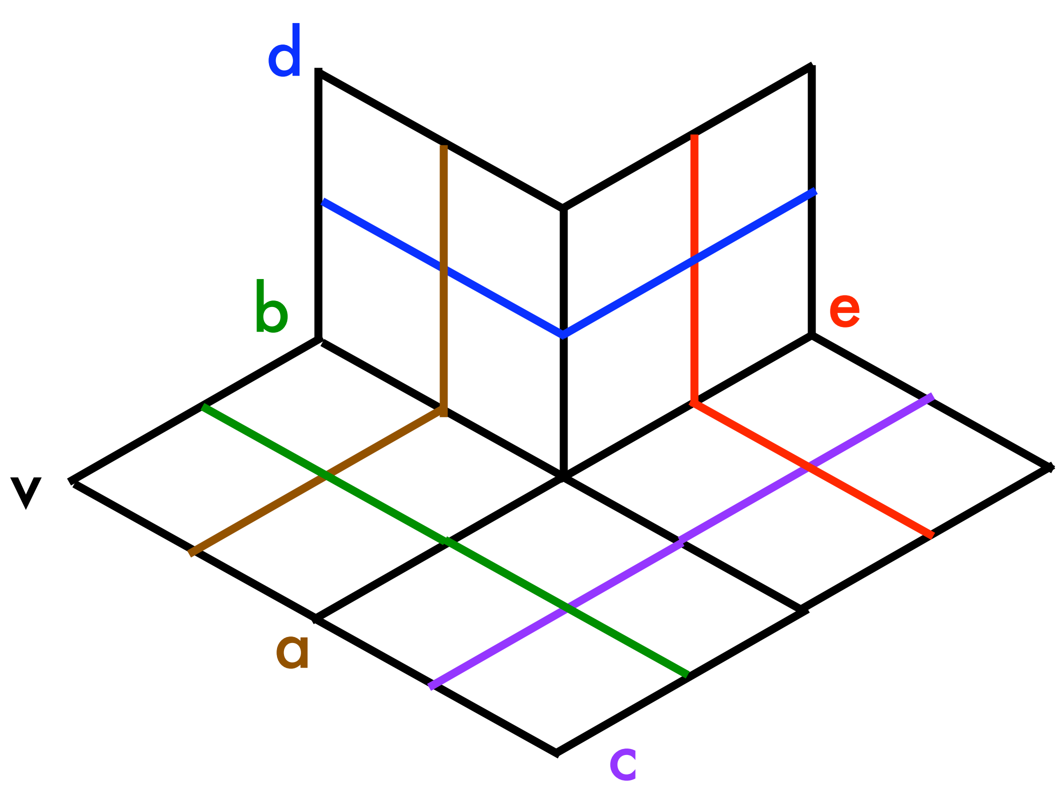
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

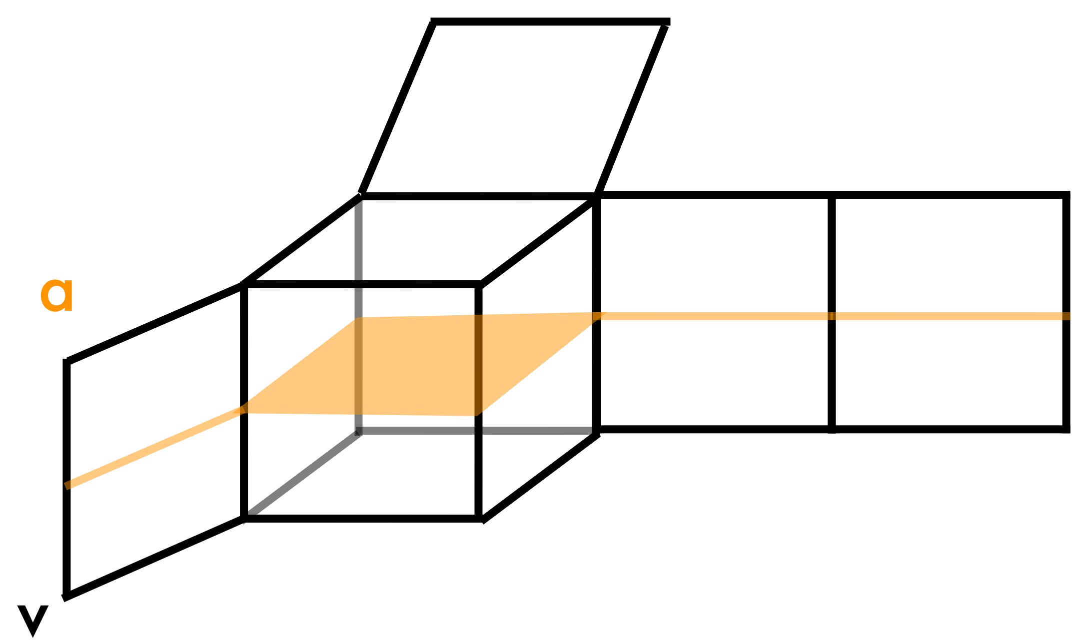




1. Poset Representation

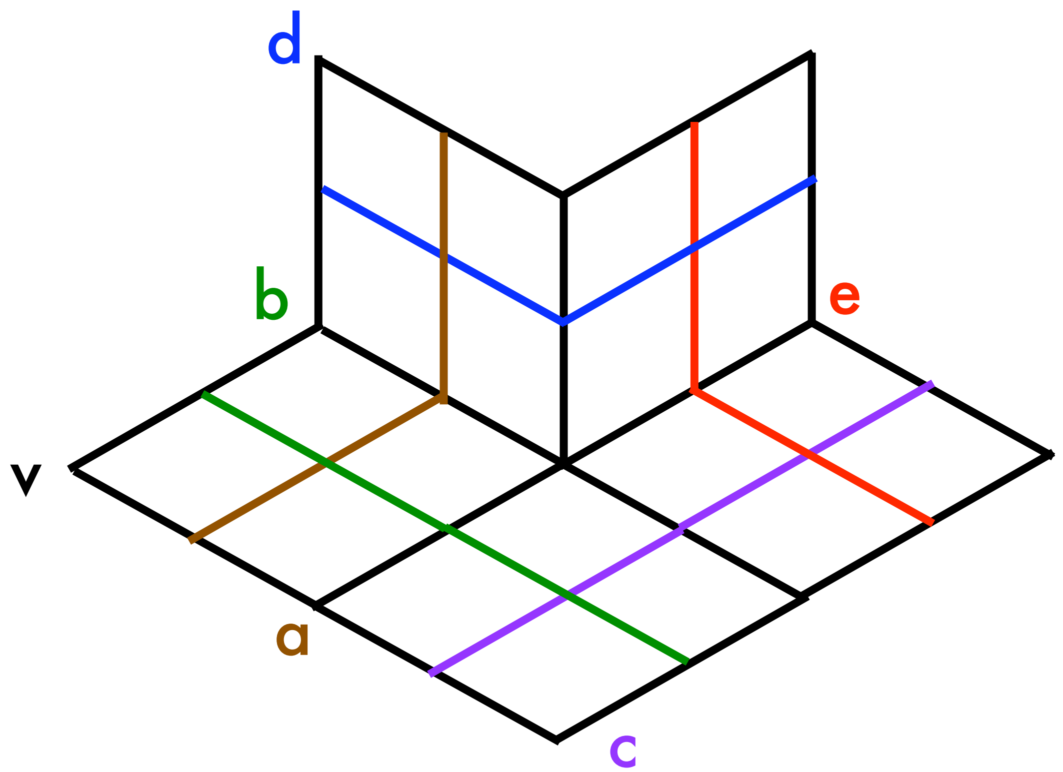
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

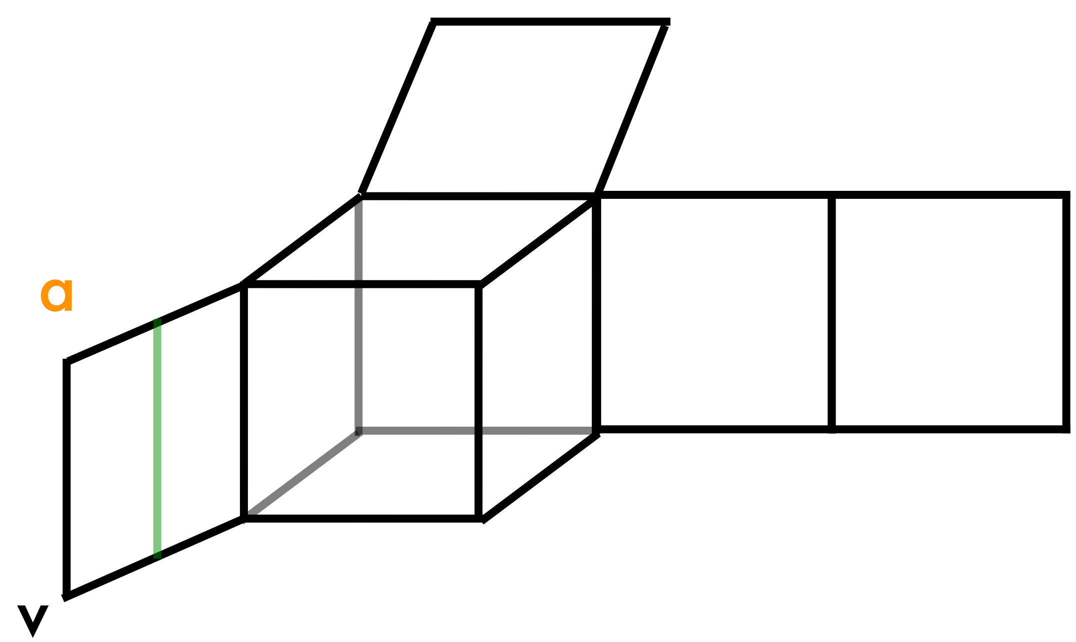




1. Poset Representation

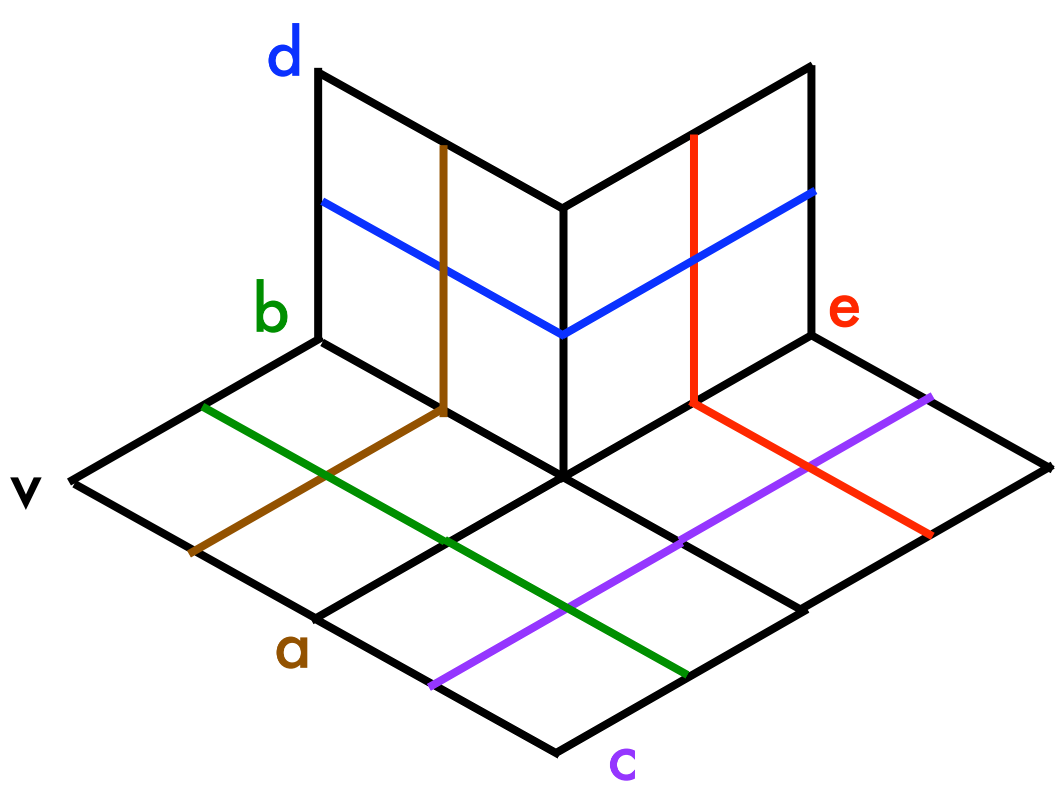
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

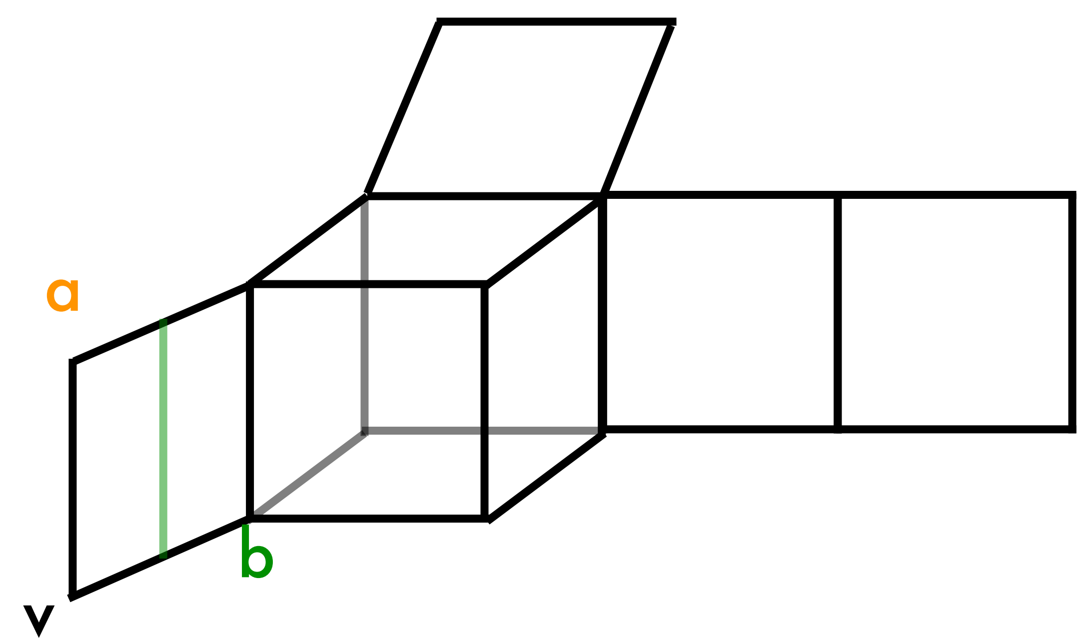




1. Poset Representation

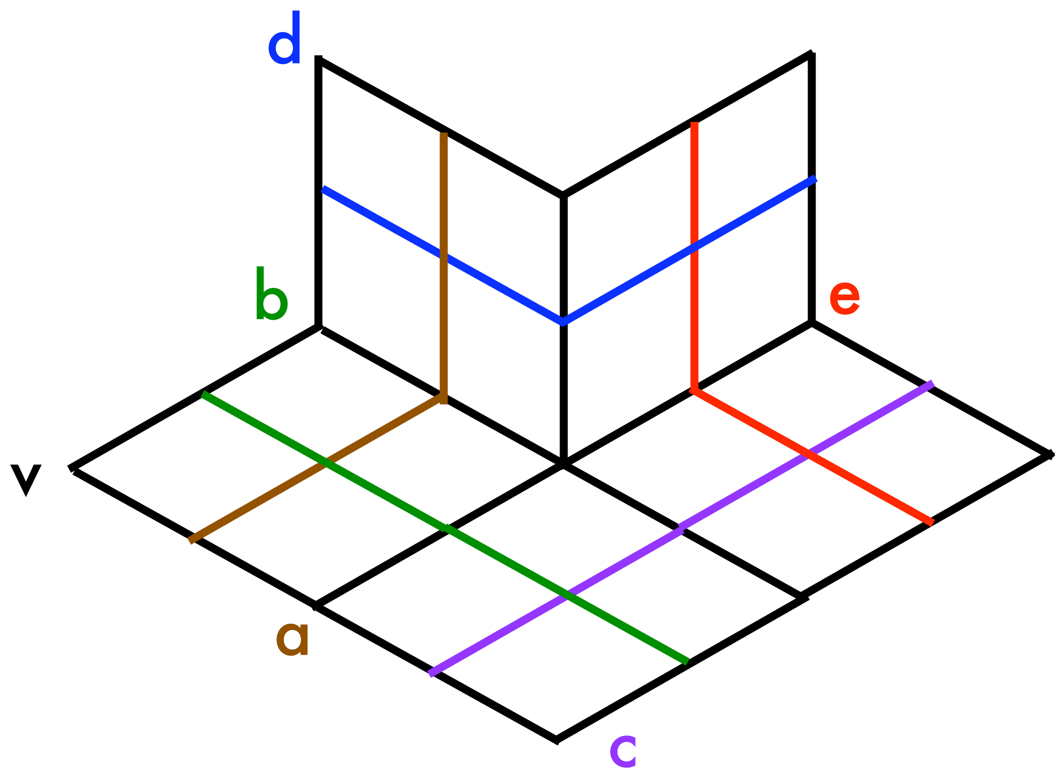
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

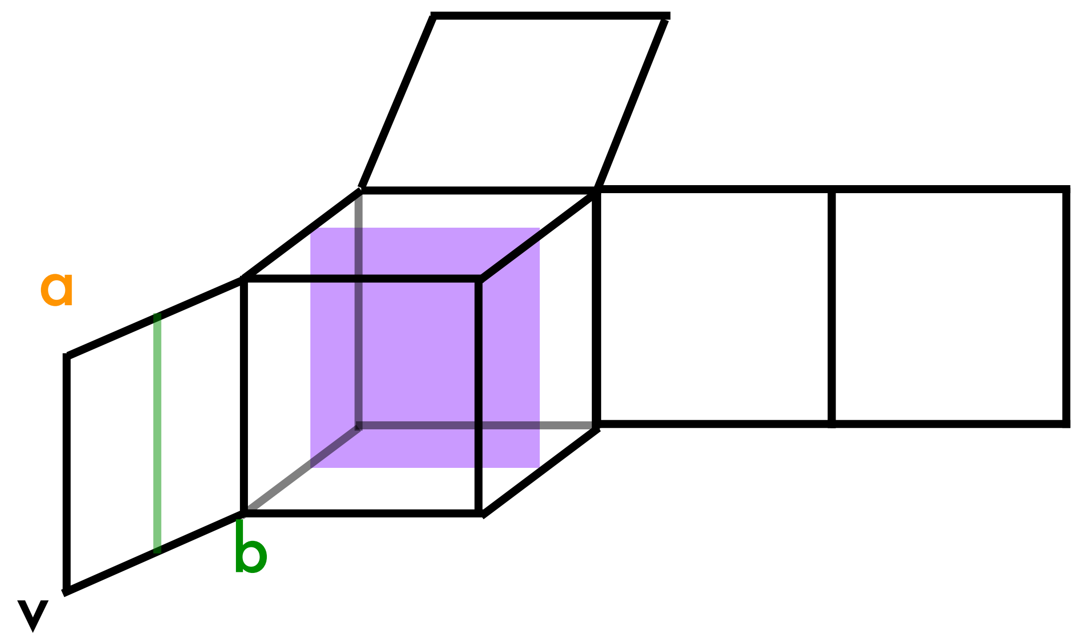




1. Poset Representation

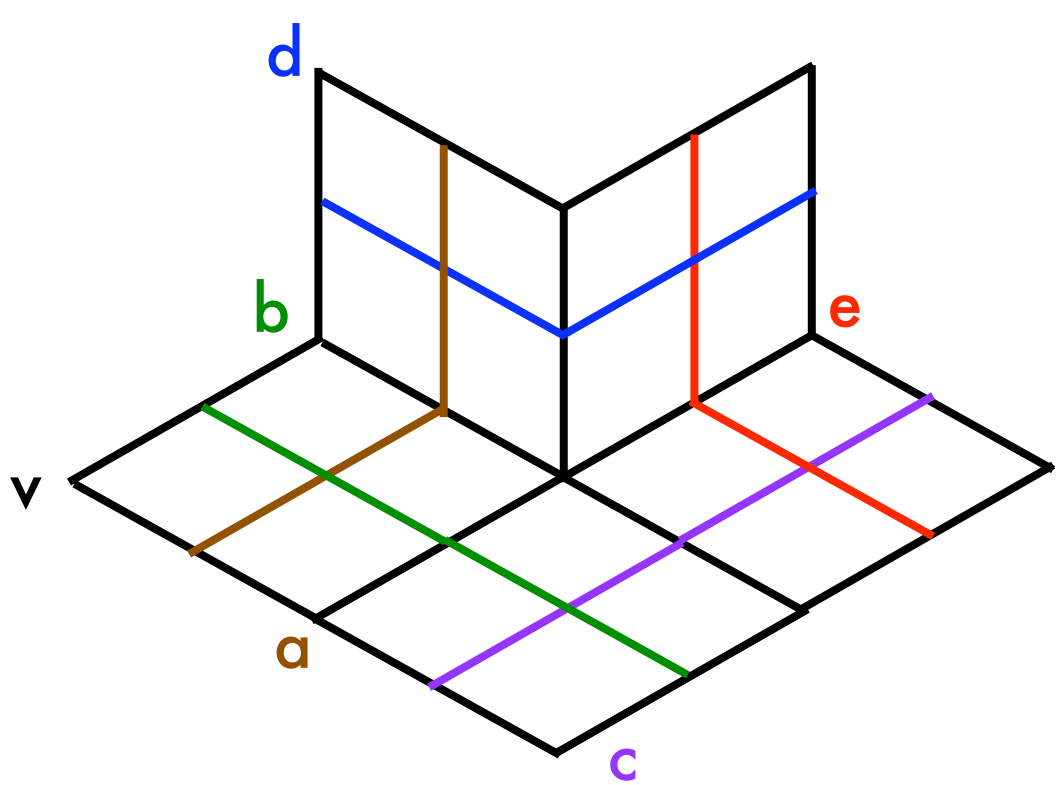
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

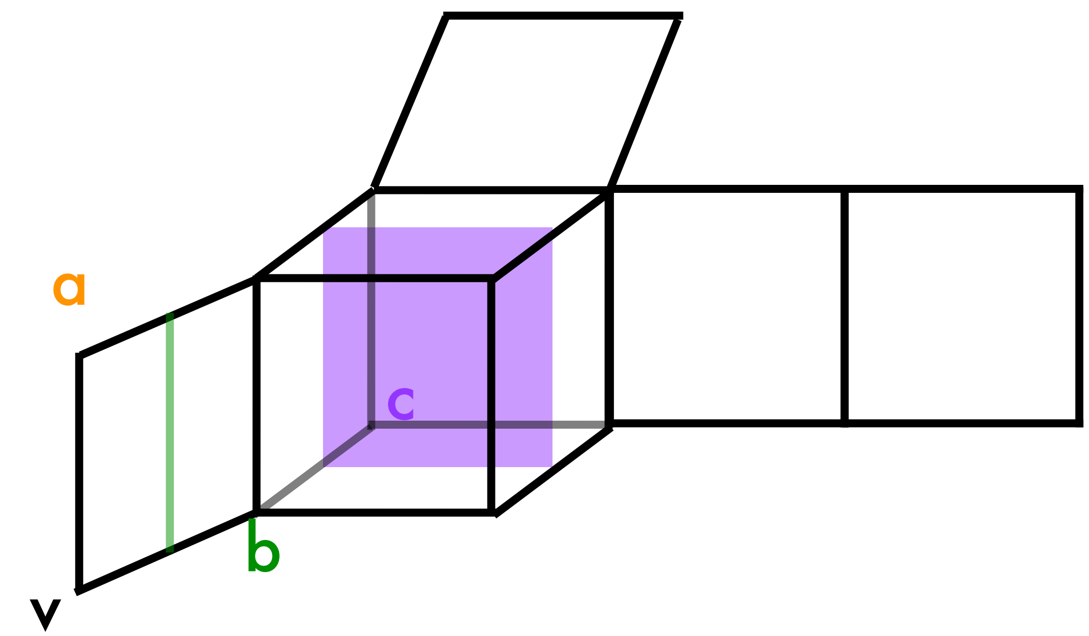




1. Poset Representation

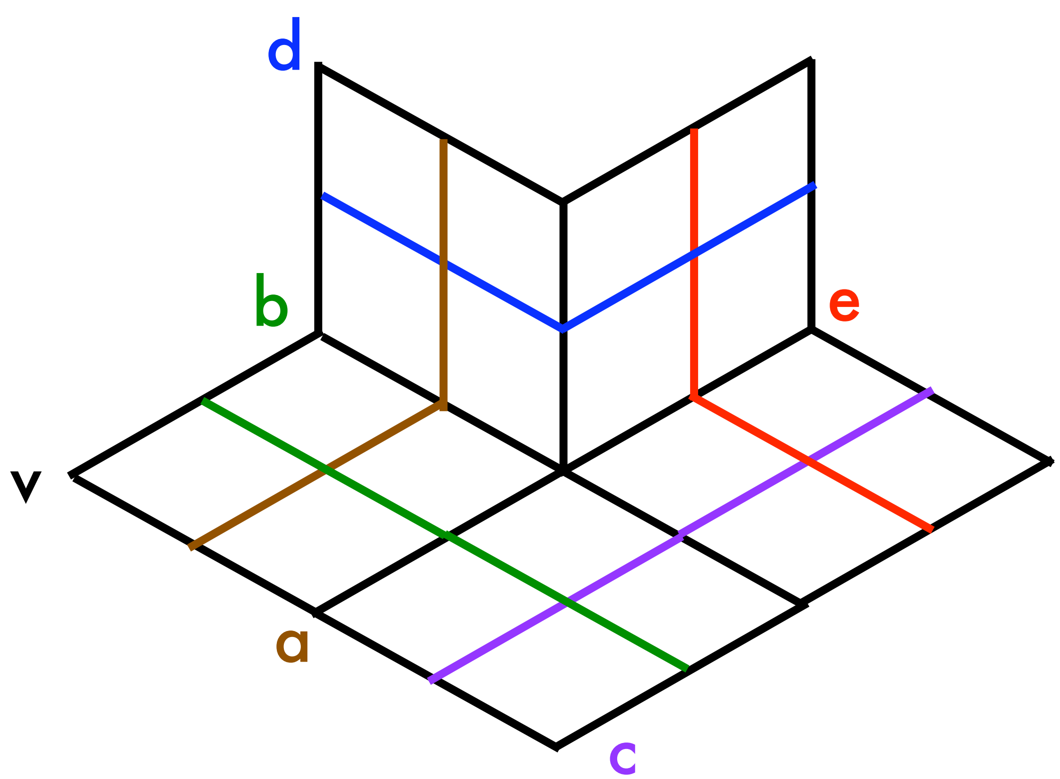
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

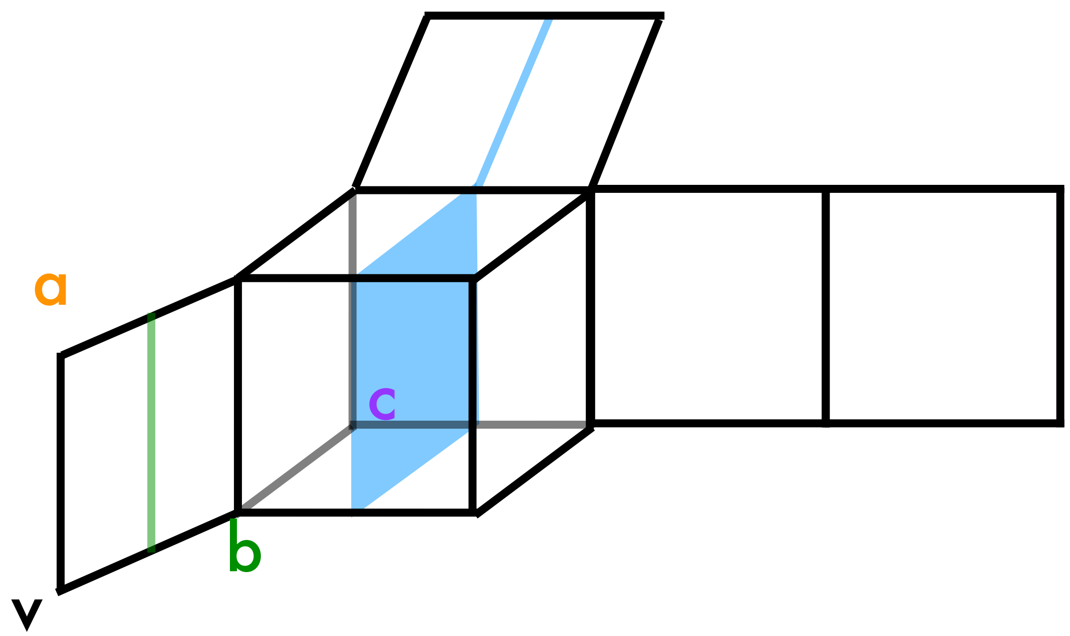




1. Poset Representation

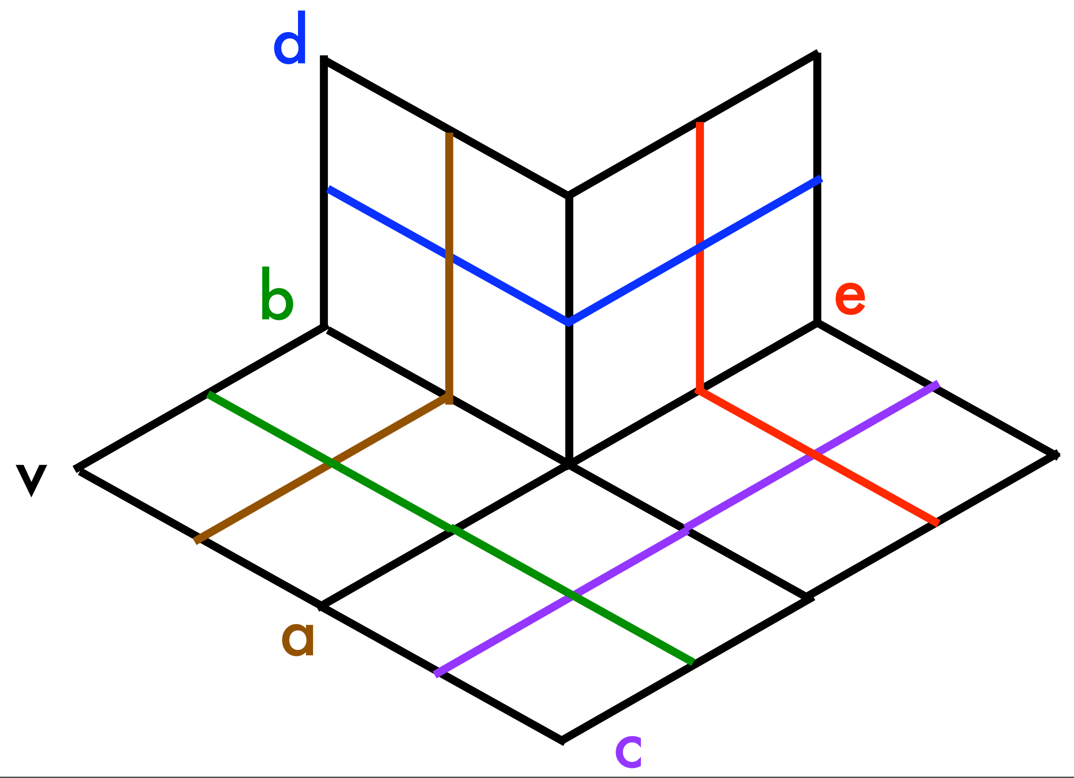
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

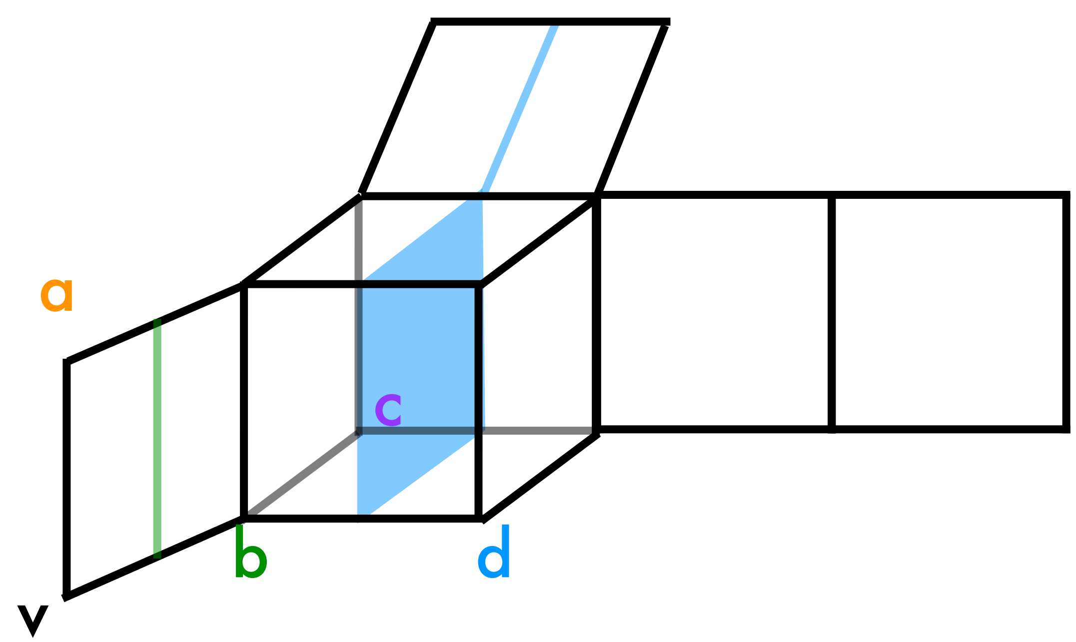




1. Poset Representation

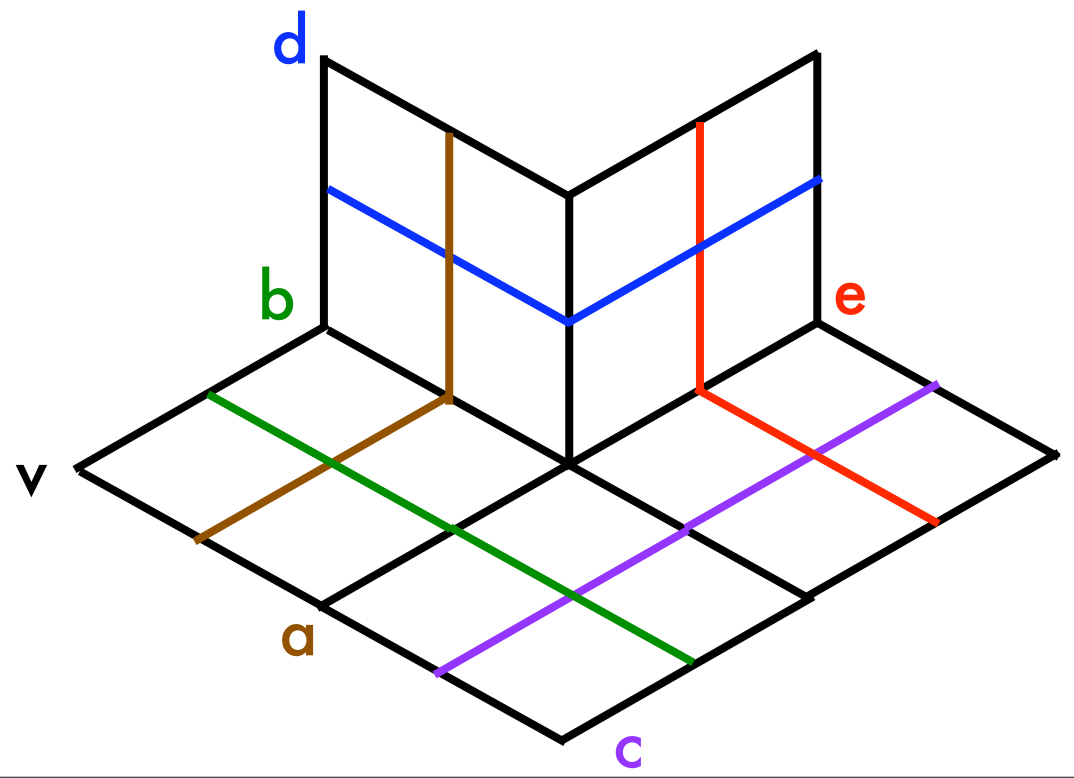
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

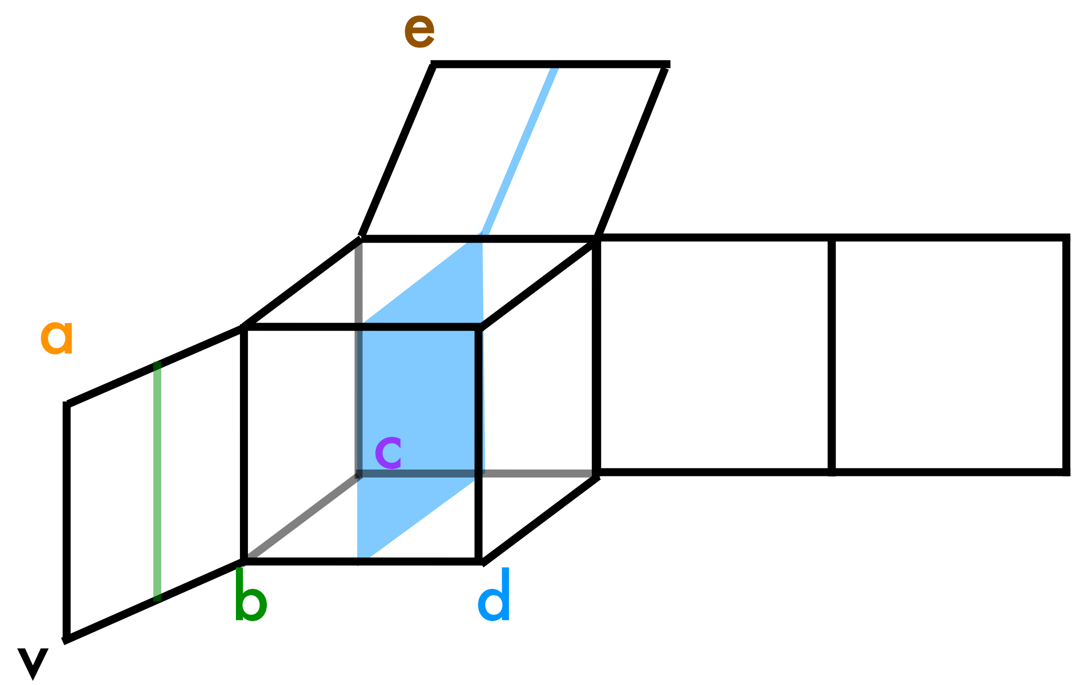




1. Poset Representation

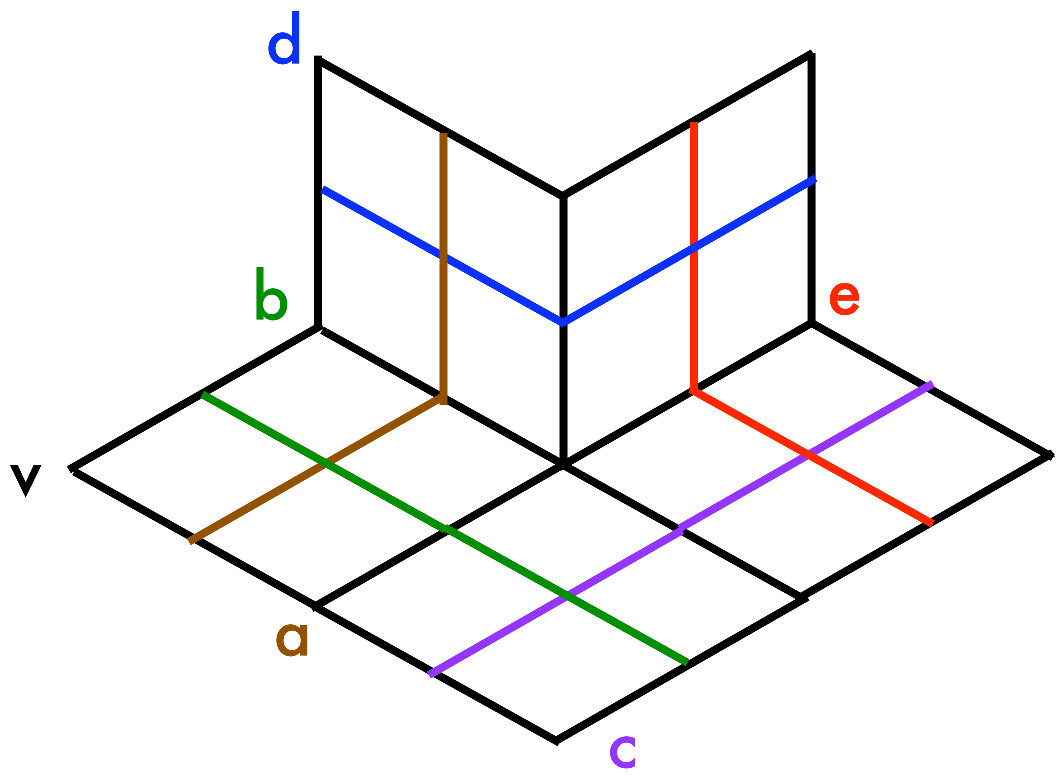
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

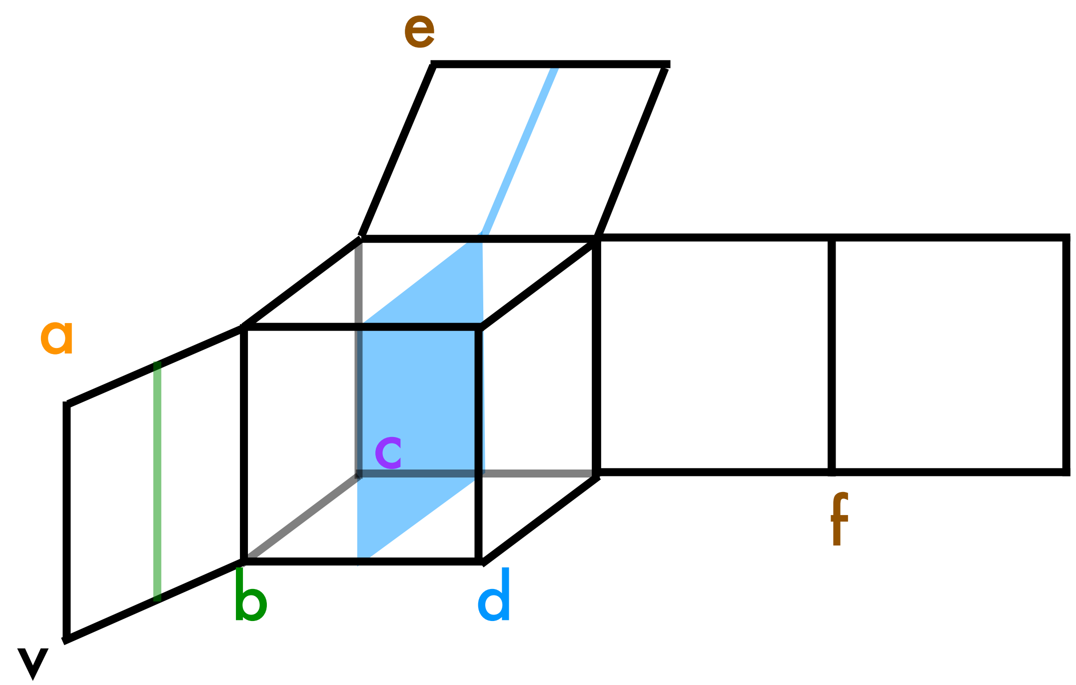




1. Poset Representation

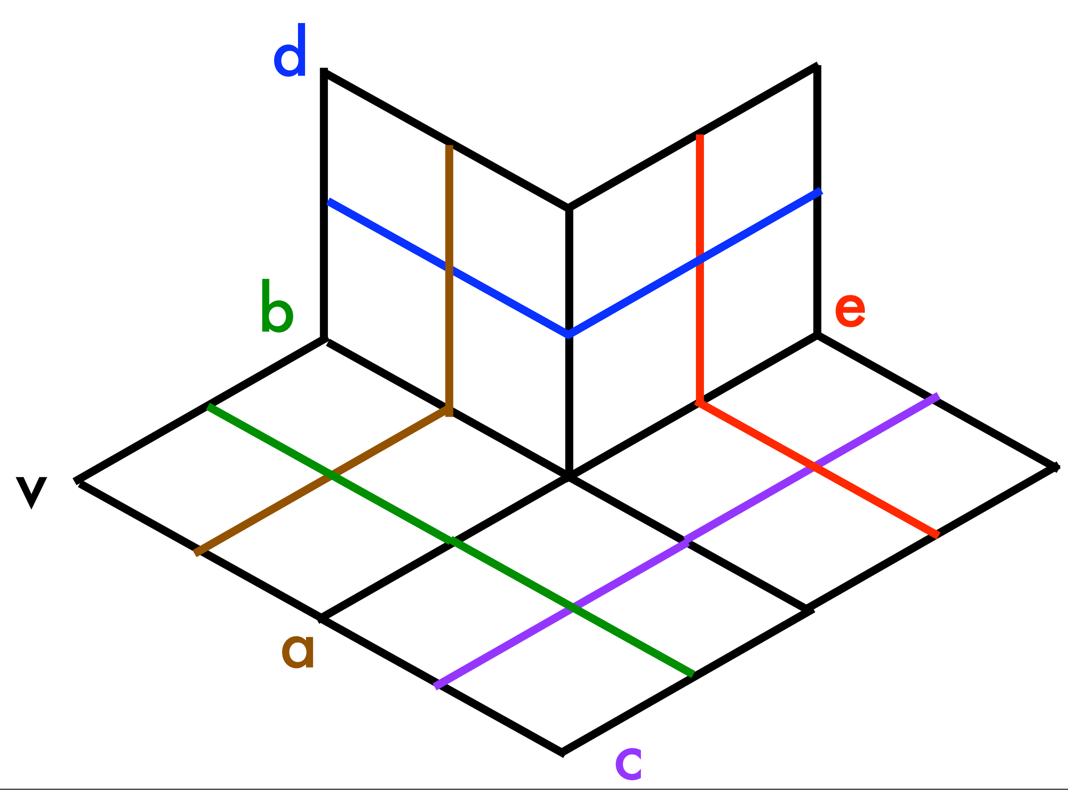
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v

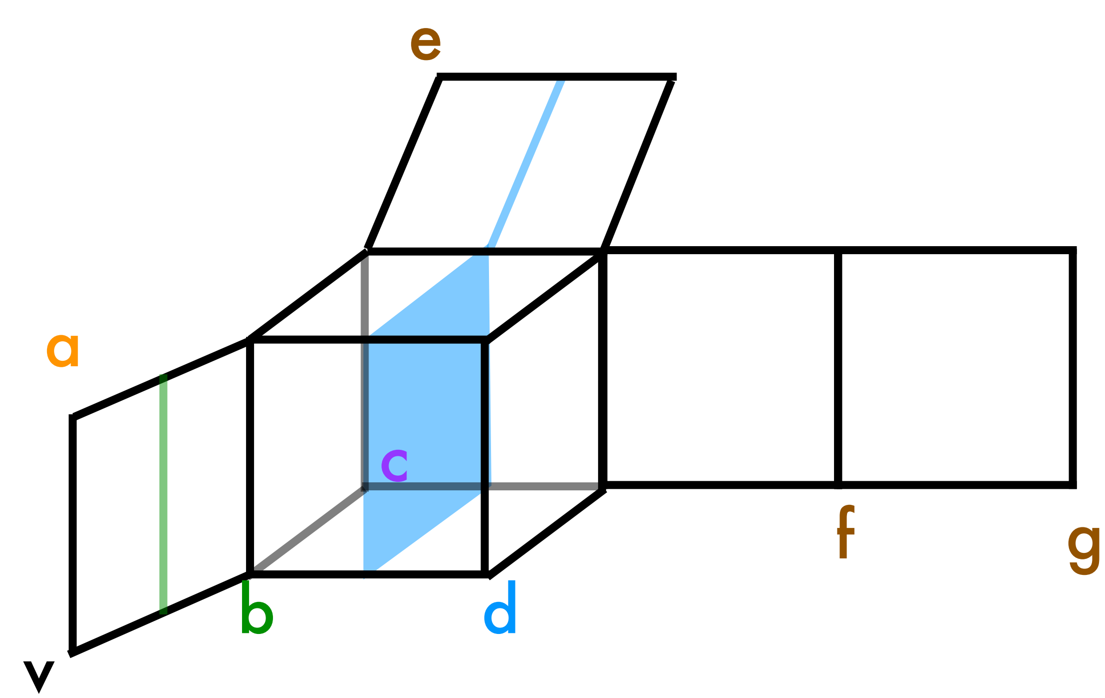




1. Poset Representation

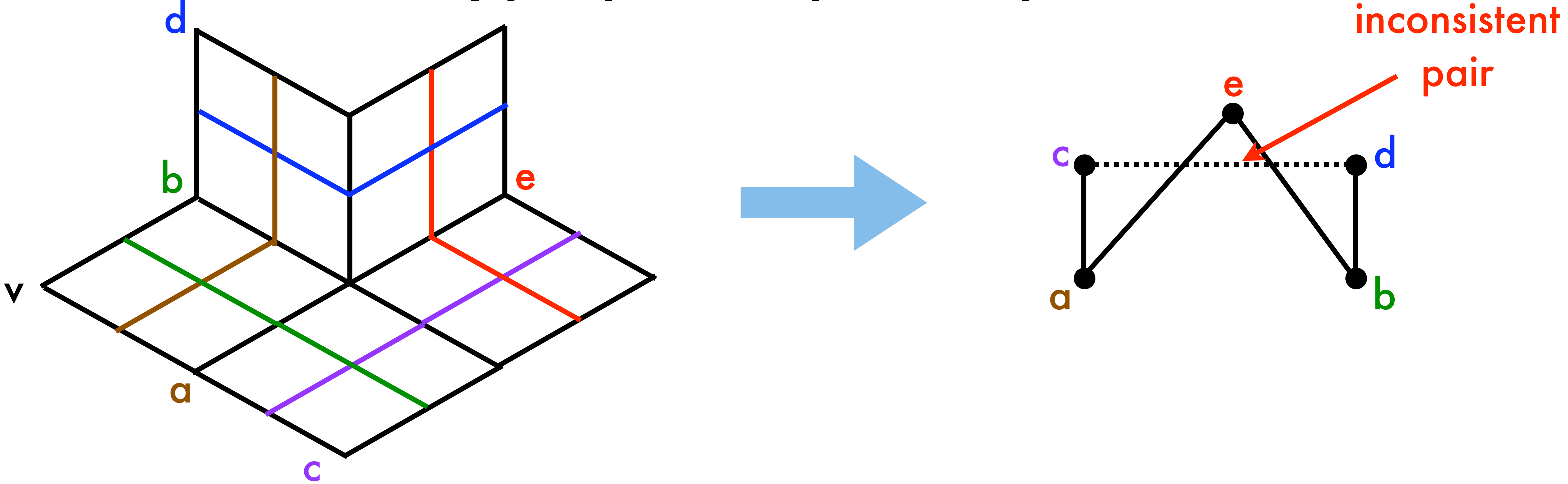
- fix a vertex v
- for each hyperplane, label the vertex closest to v on the opposite side of the hyperplane from v





1. Poset Representation

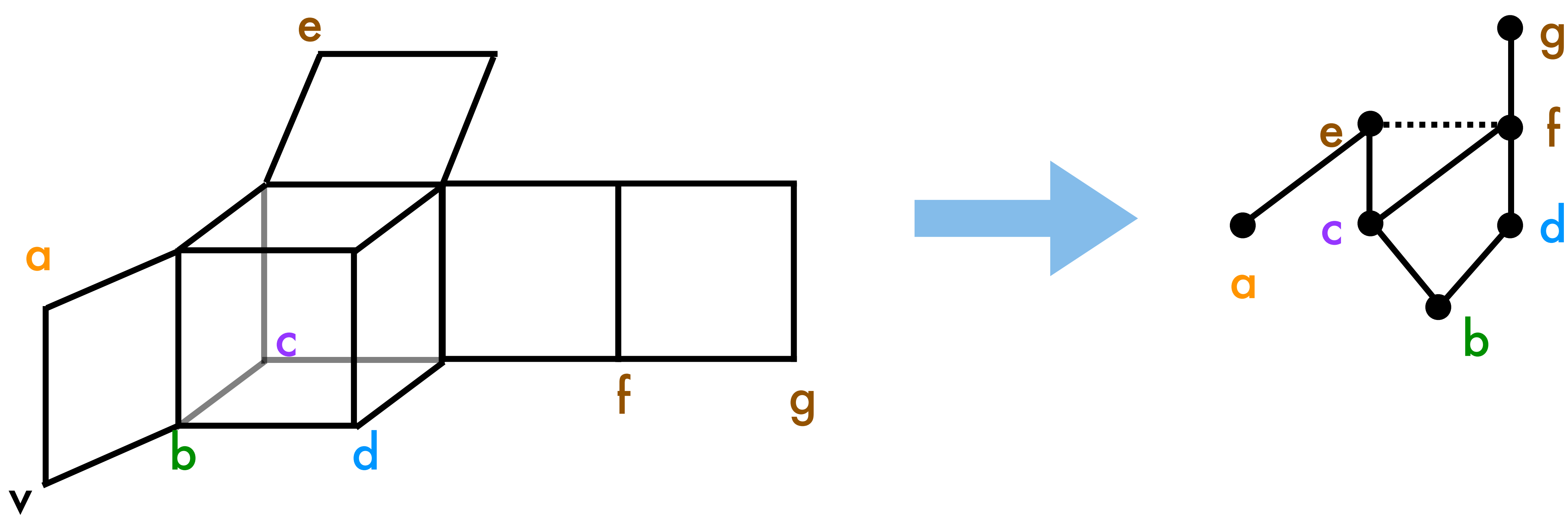
- labeled vertices form *poset with inconsistent pairs*:
  - u < w ⇔ any path from v to w crosses hyperplane associated with u
  - (p,q) is an *inconsistent pair* ⇔ no geodesic from v crosses both hyperplanes p and q



1. Poset Representation

- *poset with inconsistent pairs*
  = (~finite) poset P + set of inconsistent pairs {p,q} with:

  1. no r in P such that r ≥ p, r ≥ q
  2. p′ ≥ p, q′ ≥ q ⇒ {p′,q′} is inconsistent pair



1. Poset Representation

- *poset with inconsistent pairs*
  = (~finite) poset P + set of inconsistent pairs {p,q} with:

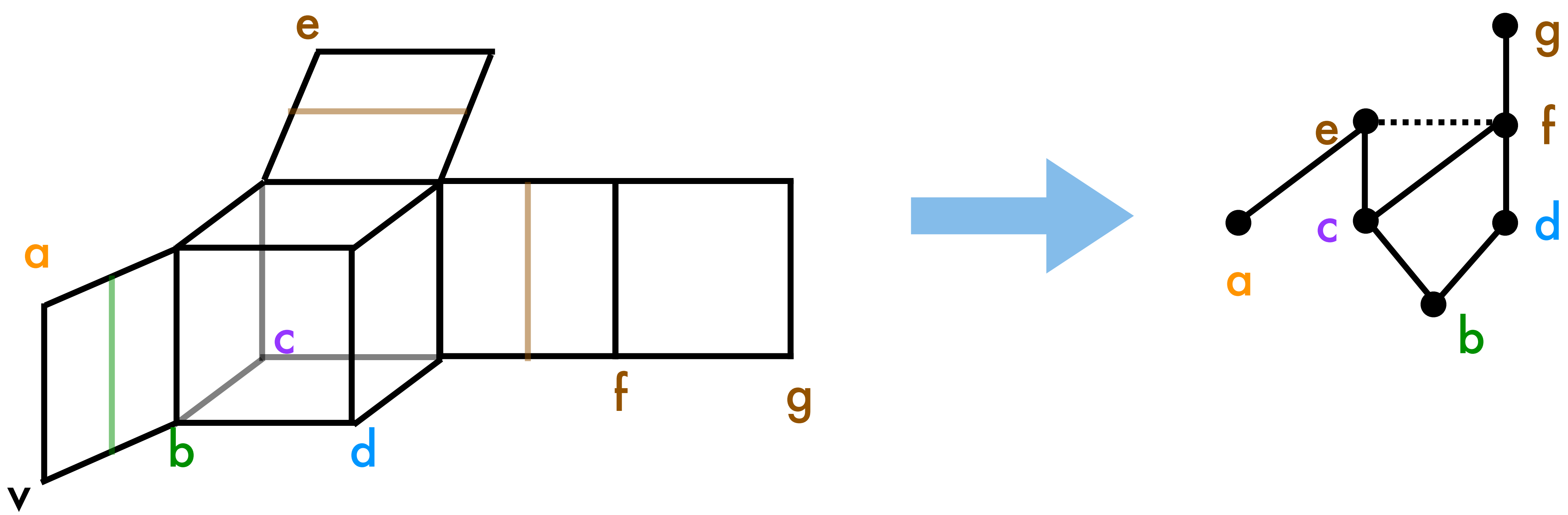
  1. no r in P such that $r \geq p$, $r \geq q$
  2. $p' \geq p$, $q' \geq q$ $\Rightarrow$ {p′,q′} is inconsistent pair

- standard embedding (P finite):

$$X_P = \{(x_1,\ldots,x_n) \in [0,1]^{|P|}:$$

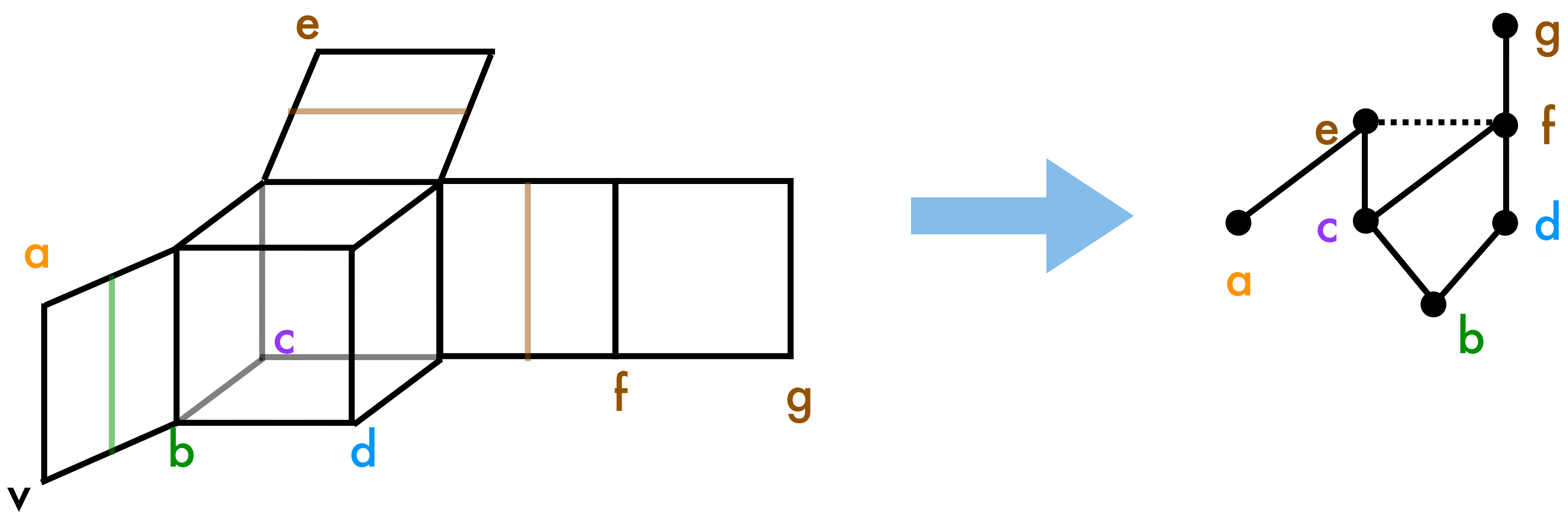
$$\text{if } u \leq_P w \text{ and } x_u < 1, \text{ then } x_w = 0$$
$$\text{and if } \{p,q\} \text{ inconsistent, then } x_p = 0 \text{ or } x_q = 0 \}$$

1. Poset Representation



Theorem (Ardila, O., Sullivant):
Fixing a vertex, there is a bijection between CAT(0) cube complexes and posets with inconsistent pairs.

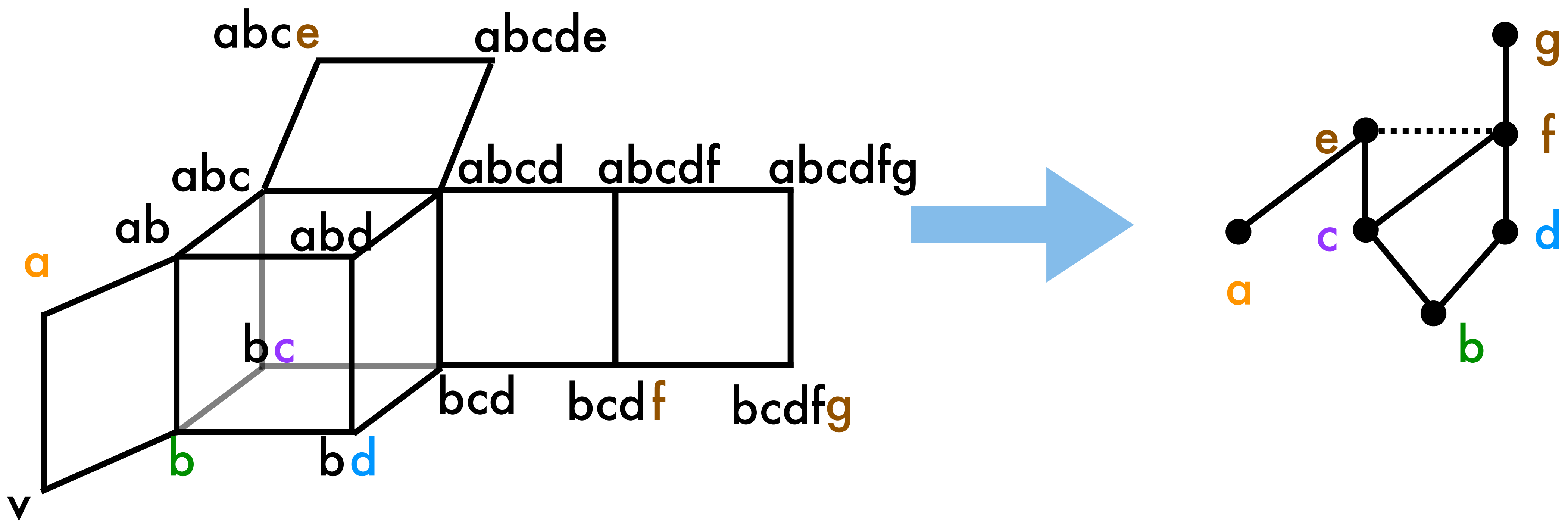
1. Poset Representation



Theorem (Ardila, O., Sullivant):
Fixing a vertex, there is a bijection between CAT(0) cube complexes and posets with inconsistent pairs.

vertices in cube complex $\longleftrightarrow$ order ideals with no inconsistent pairs

1. Poset Representation



Theorem (Ardila, O., Sullivant):
Fixing a vertex, there is a bijection between CAT(0) cube complexes and posets with inconsistent pairs.

vertices in cube complex $\longleftrightarrow$ order ideals with no inconsistent pairs

Outline

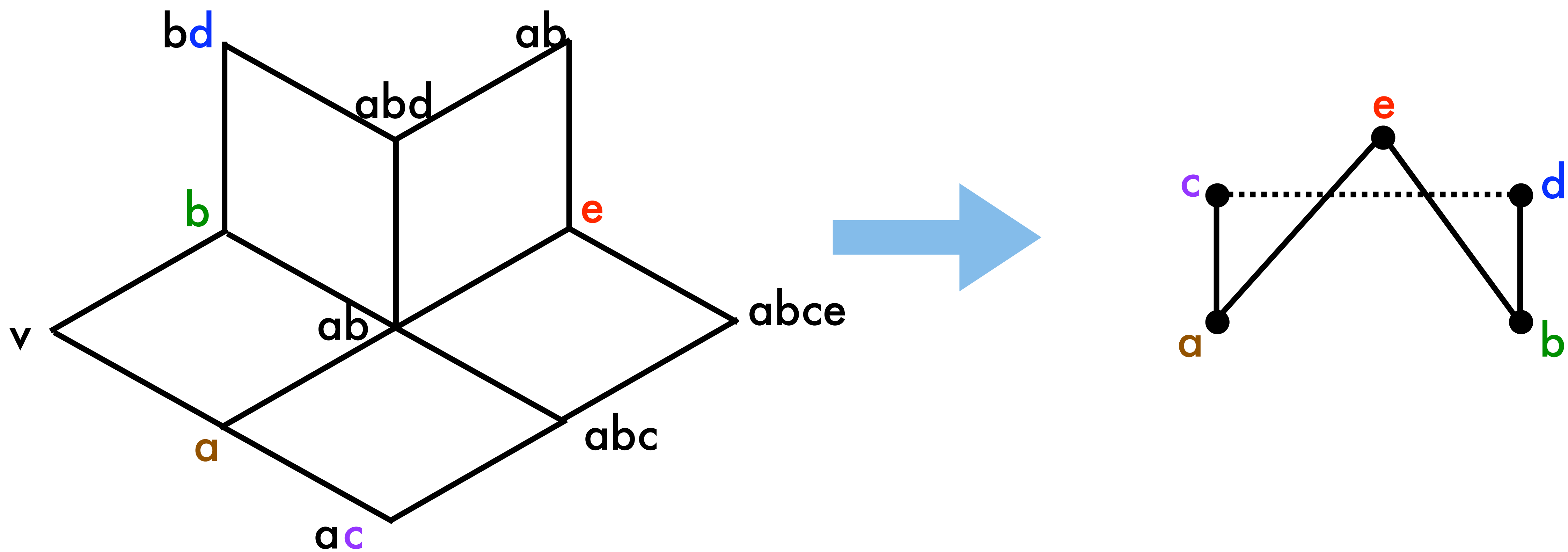
1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

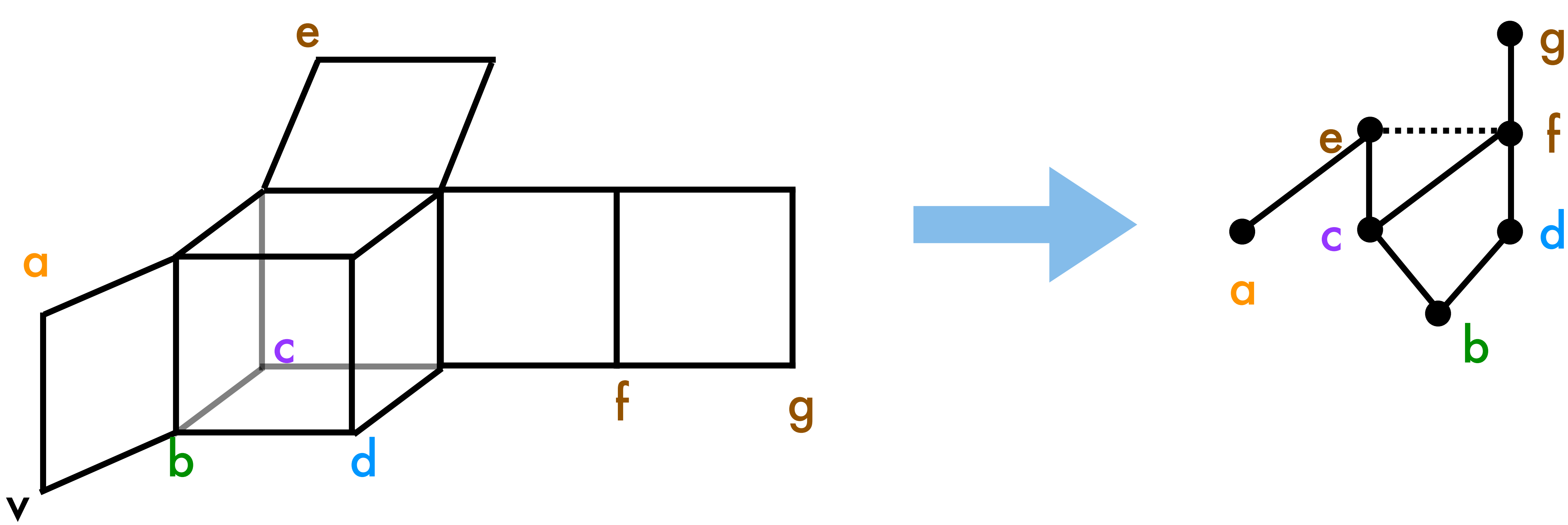
2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

3. Find geodesic through this cube sequence.

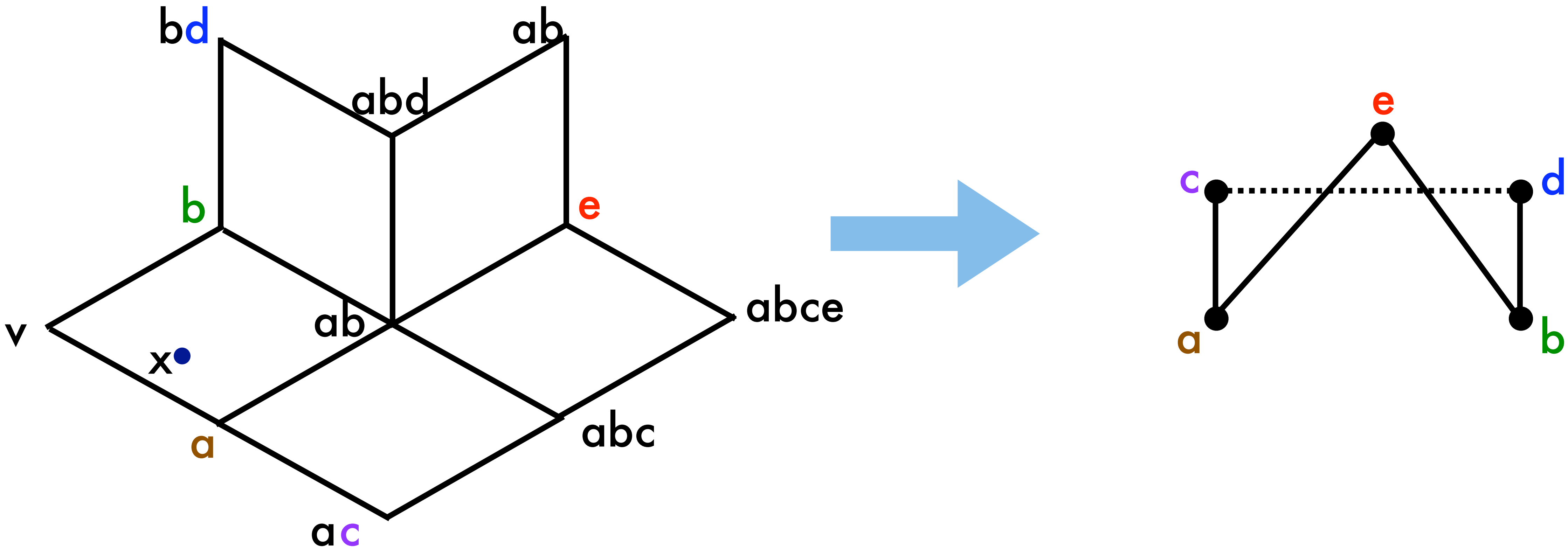
4. If possible, improve cube sequence and repeat from 3.

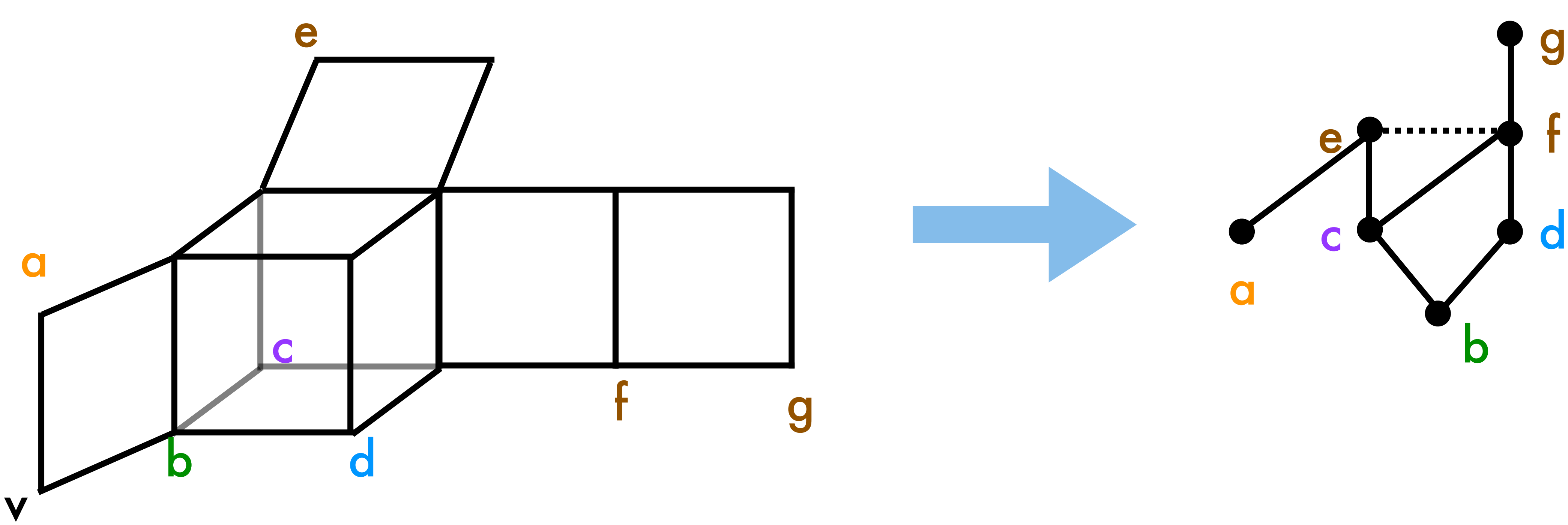
2. Reduce Poset



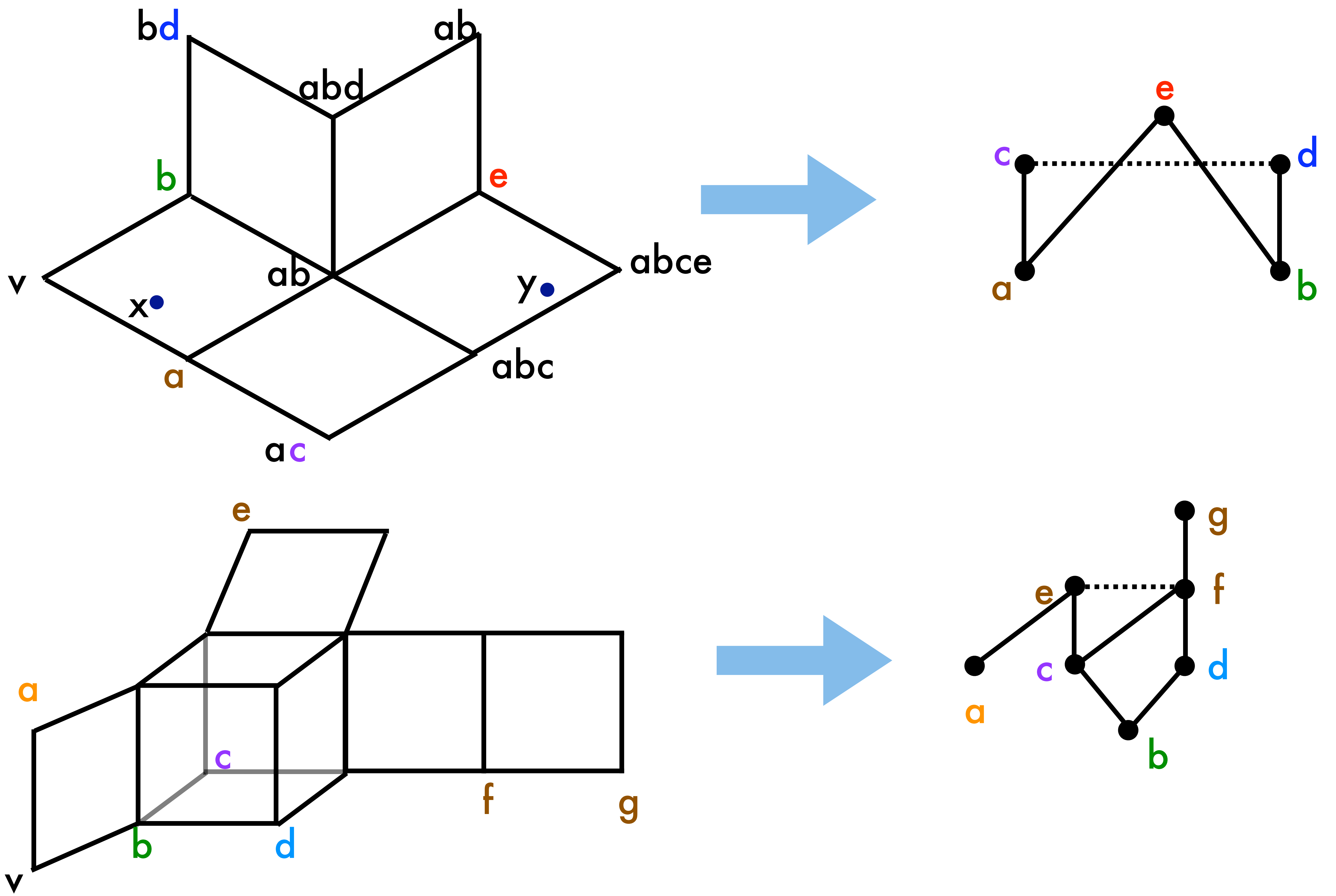


2. Reduce Poset

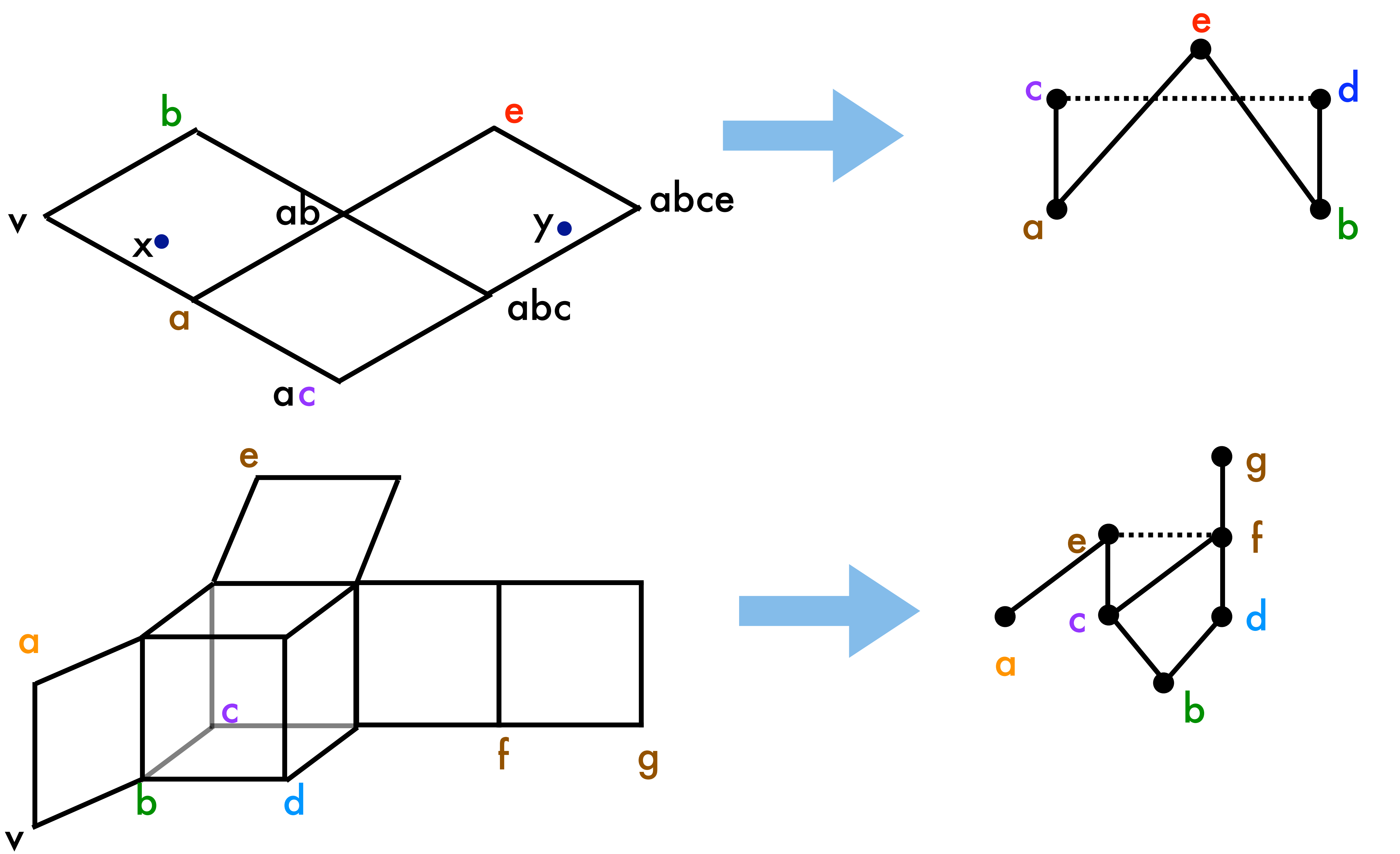




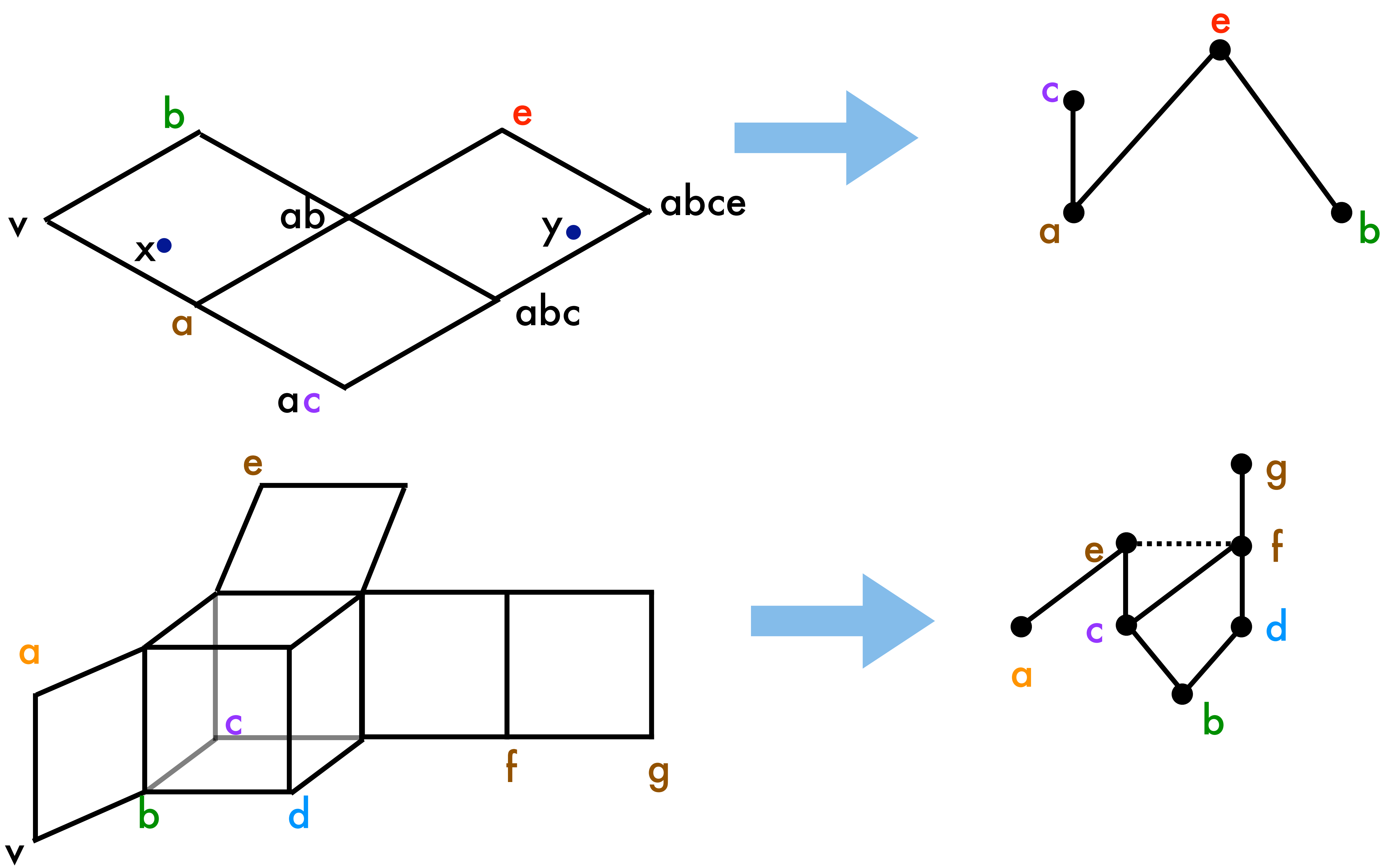
2. Reduce Poset



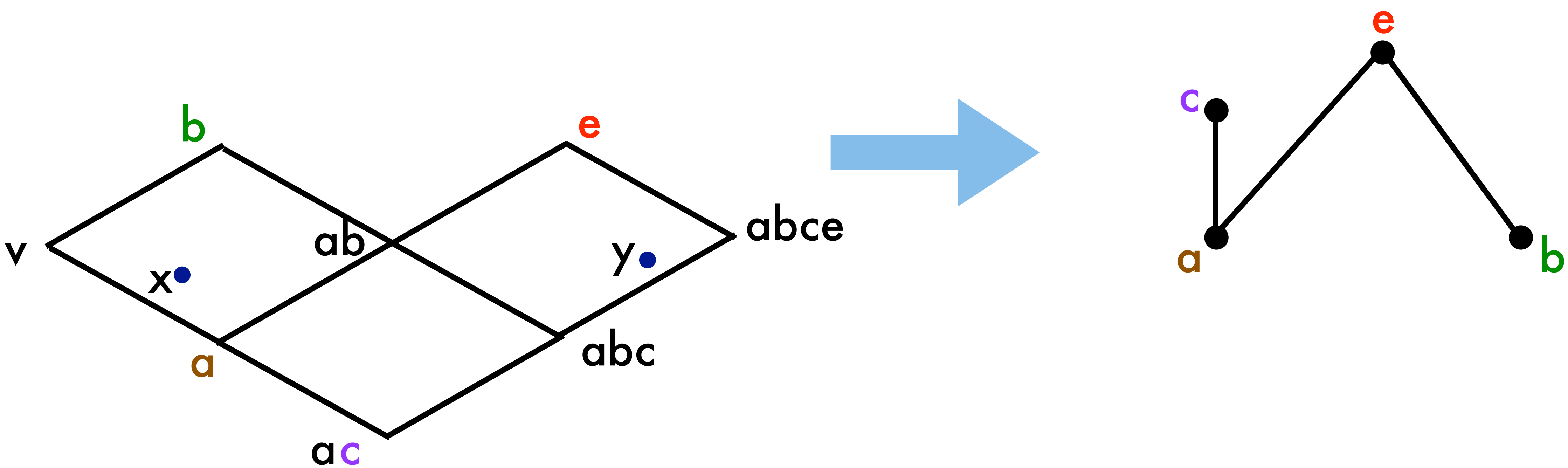
2. Reduce Poset

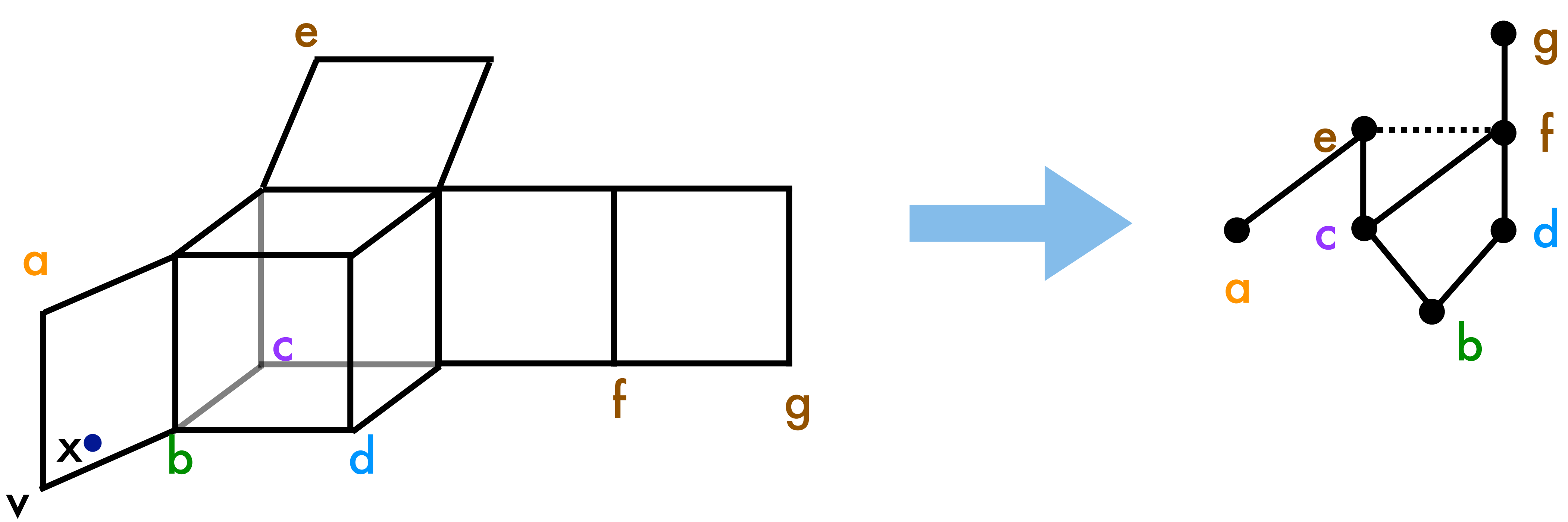


2. Reduce Poset

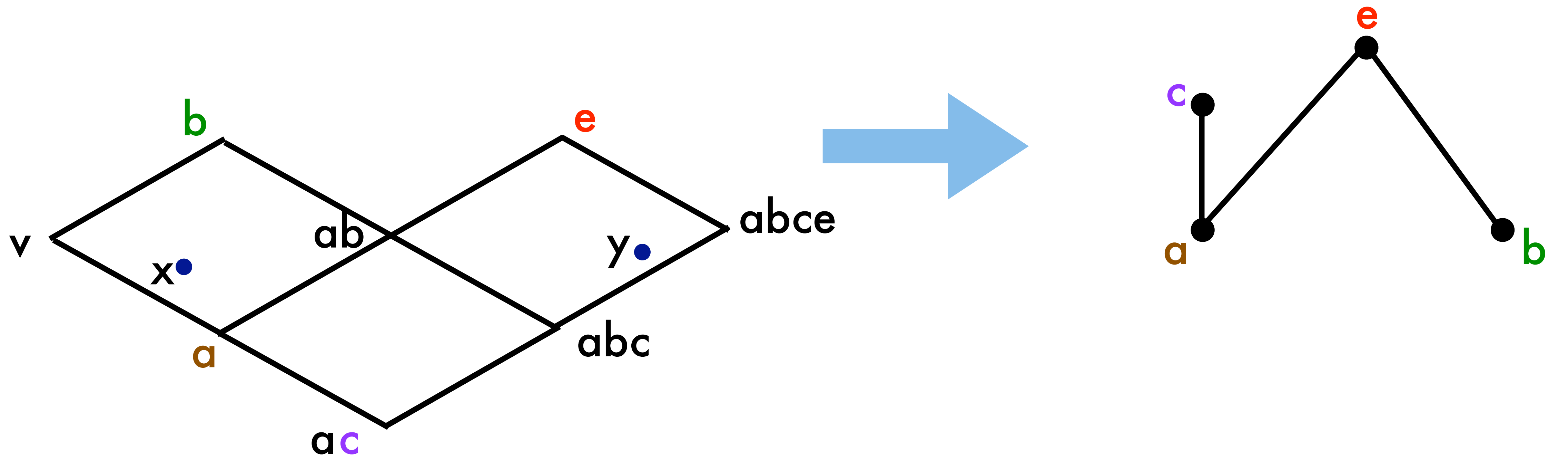


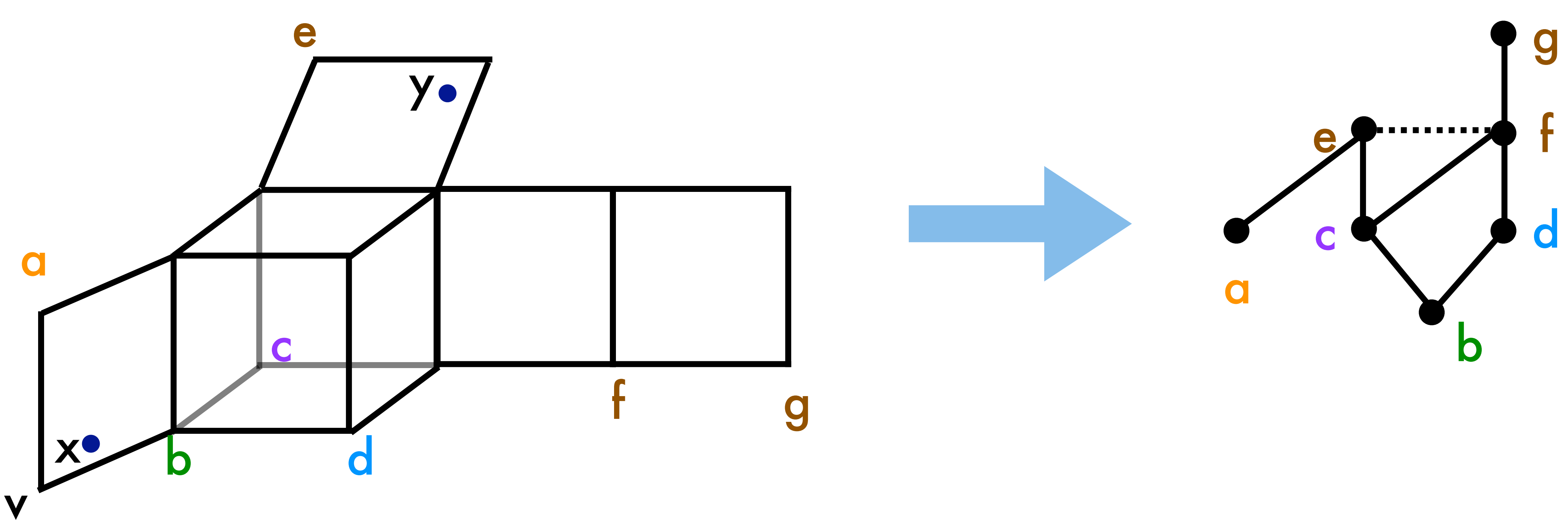
2. Reduce Poset



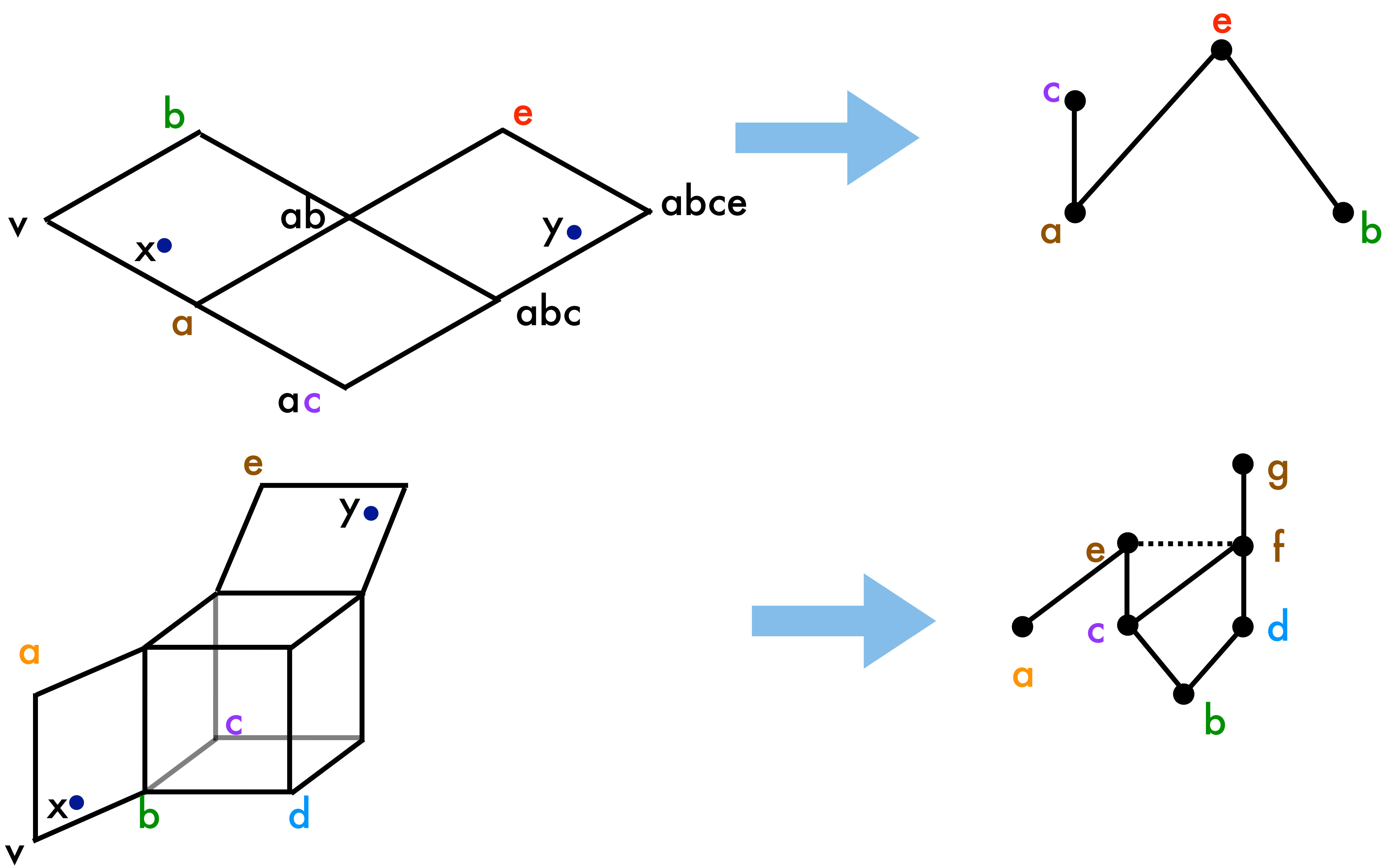


2. Reduce Poset

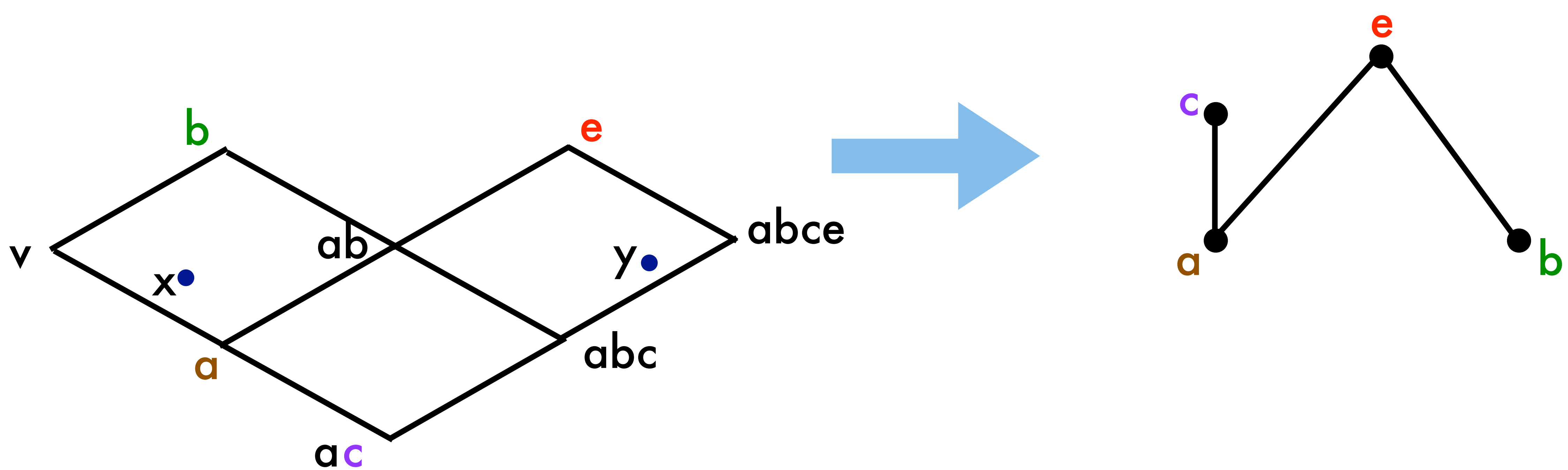




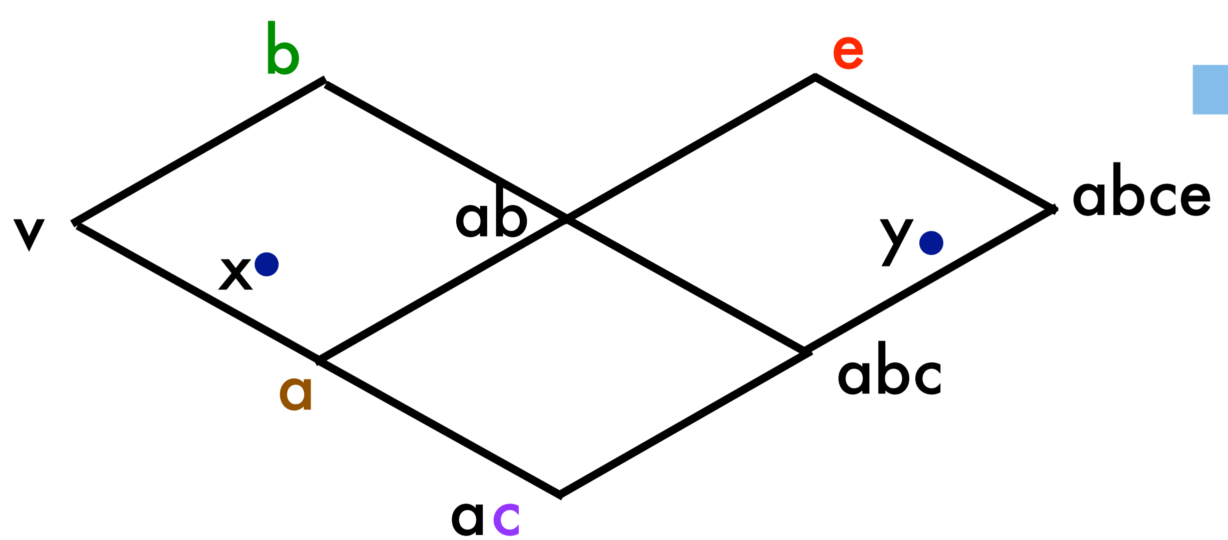
2. Reduce Poset

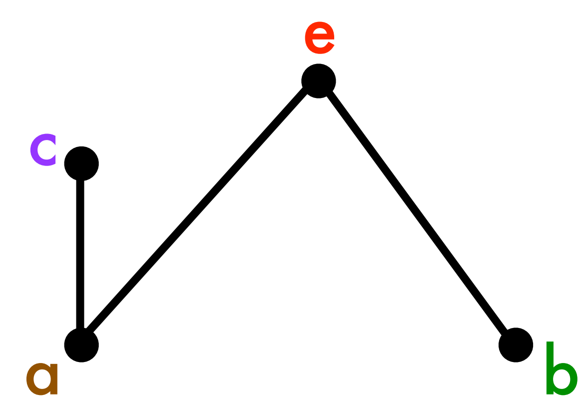


2. Reduce Poset

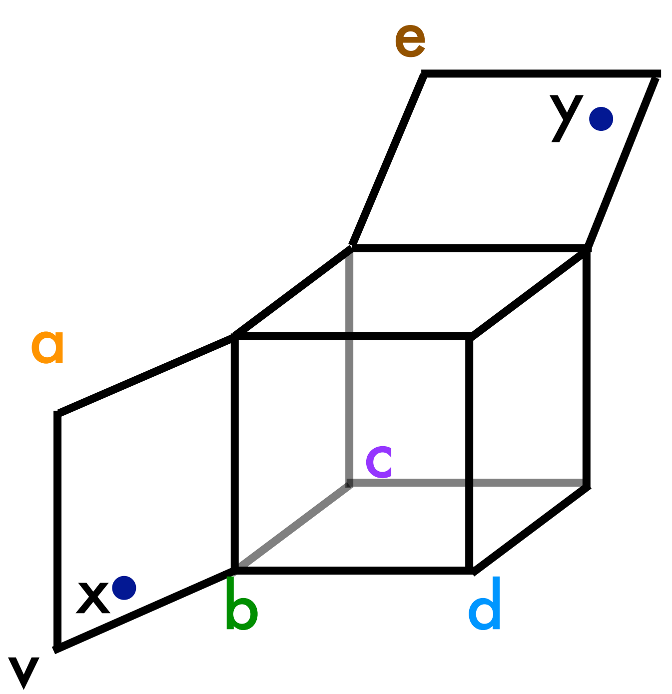


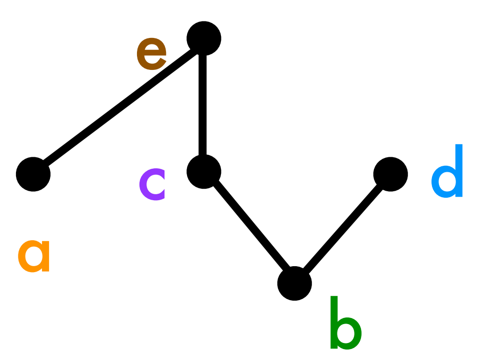
2. Reduce Poset



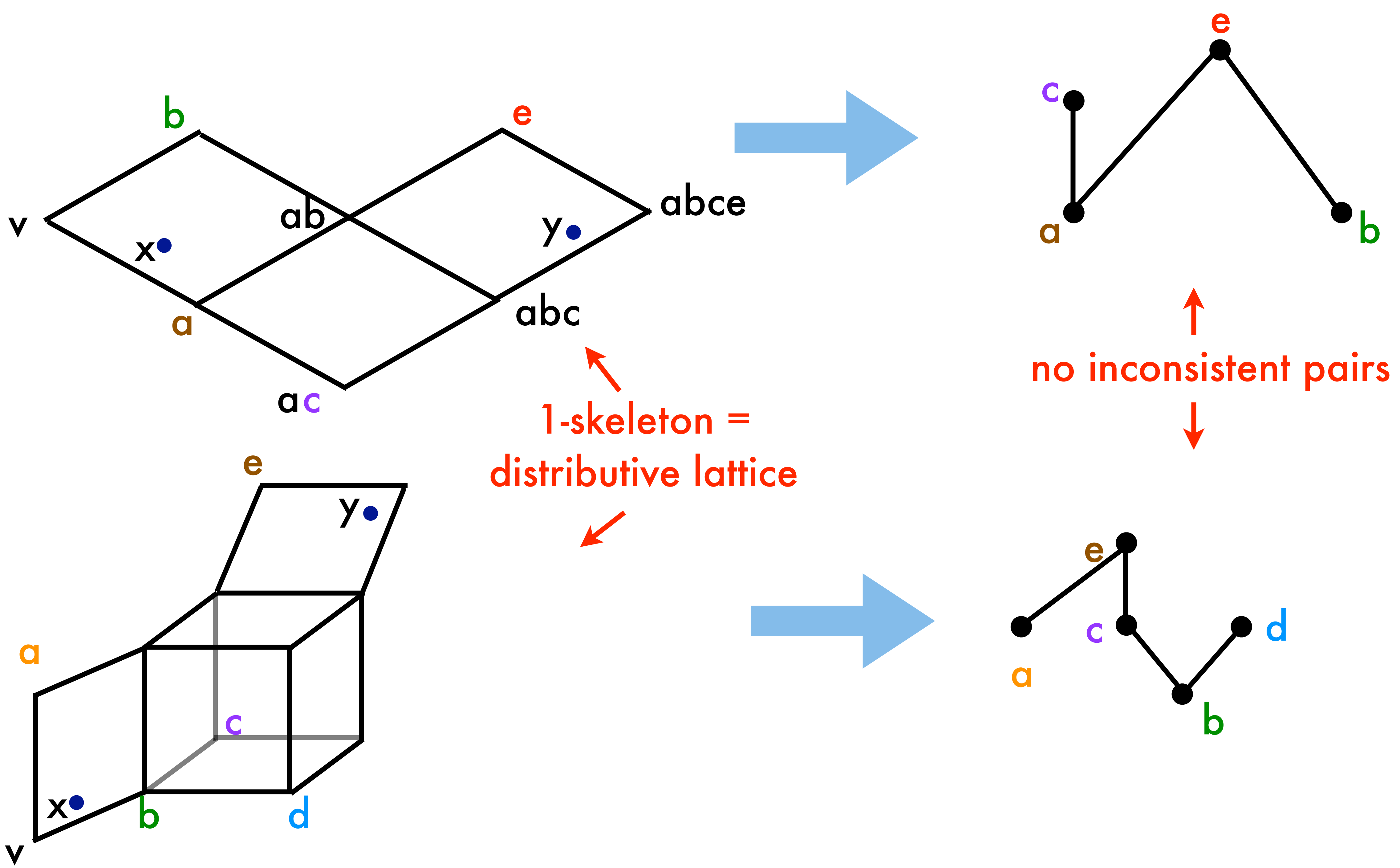


no inconsistent pairs



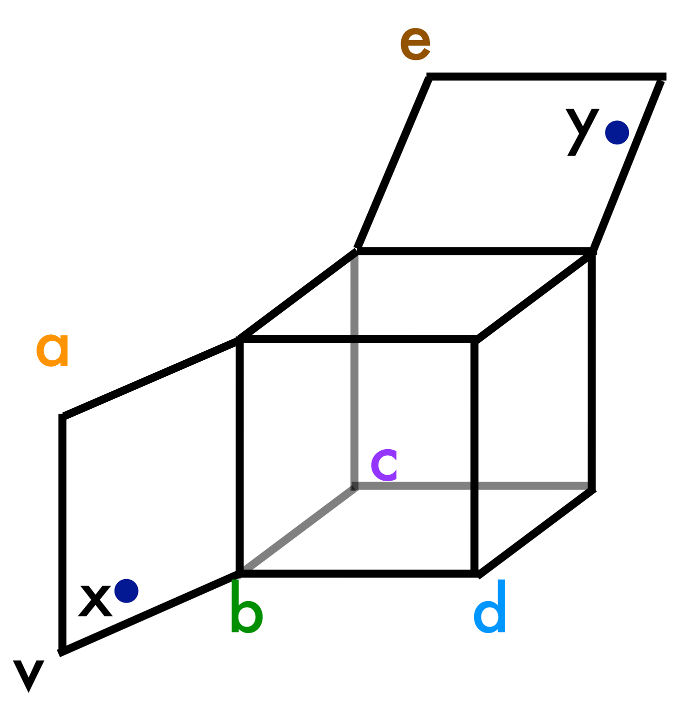


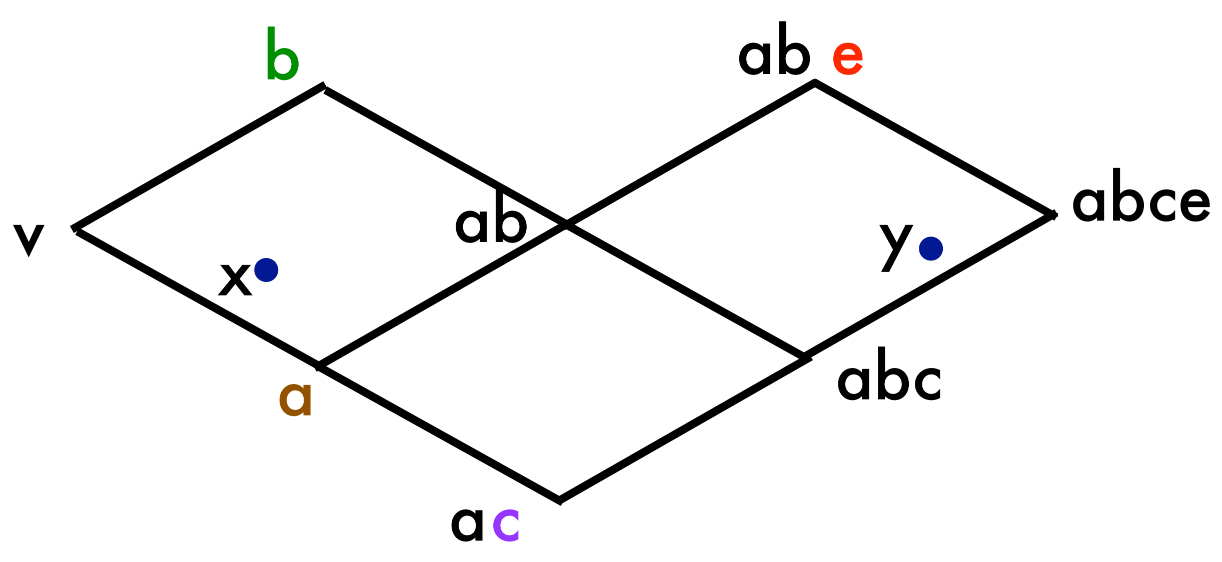
2. Reduce Poset



2. Starting Cube Sequence

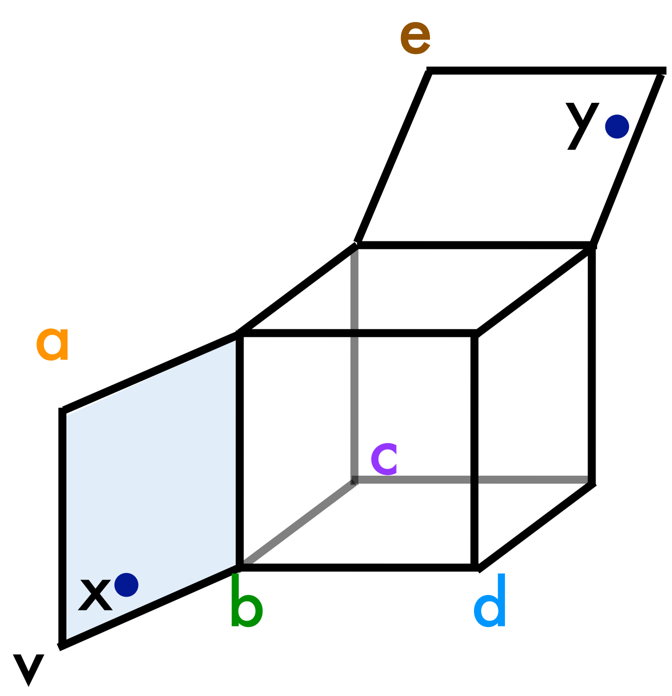
- choose a valid starting cube sequence based on x and y

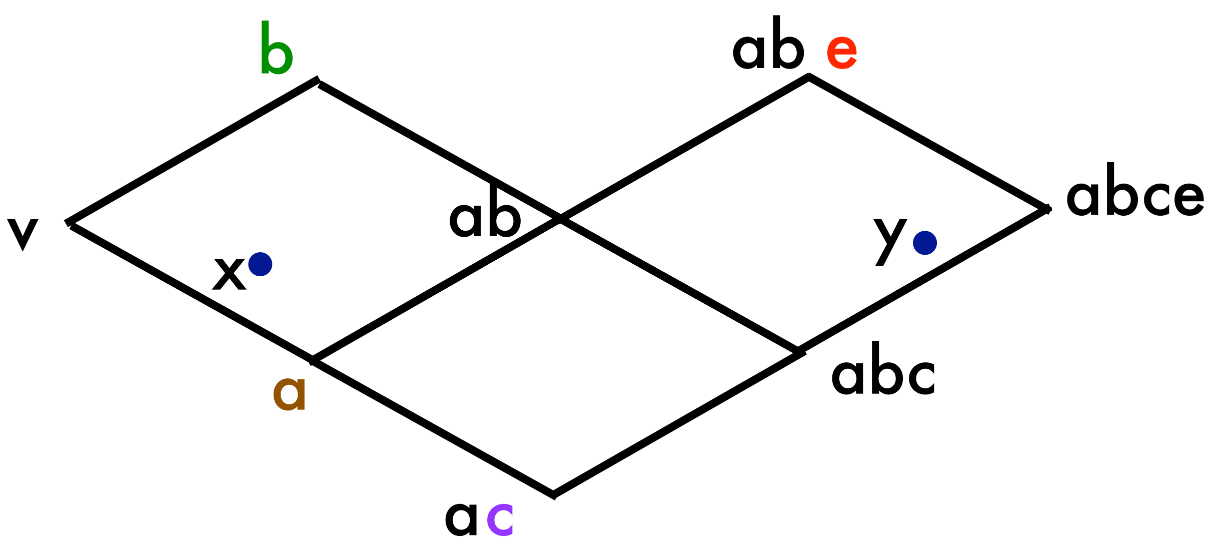




2. Starting Cube Sequence

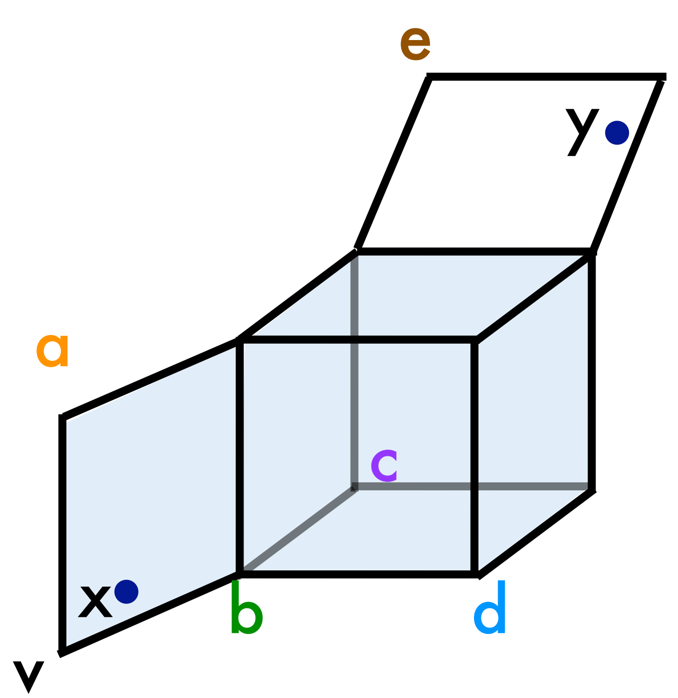
- choose a valid starting cube sequence based on x and y

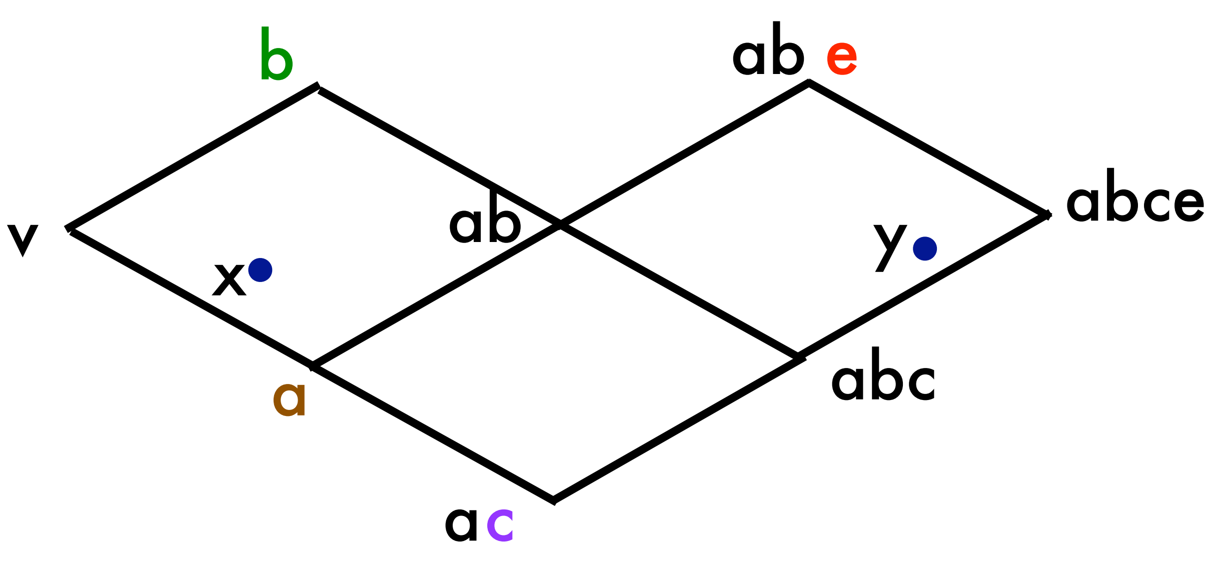




2. Starting Cube Sequence

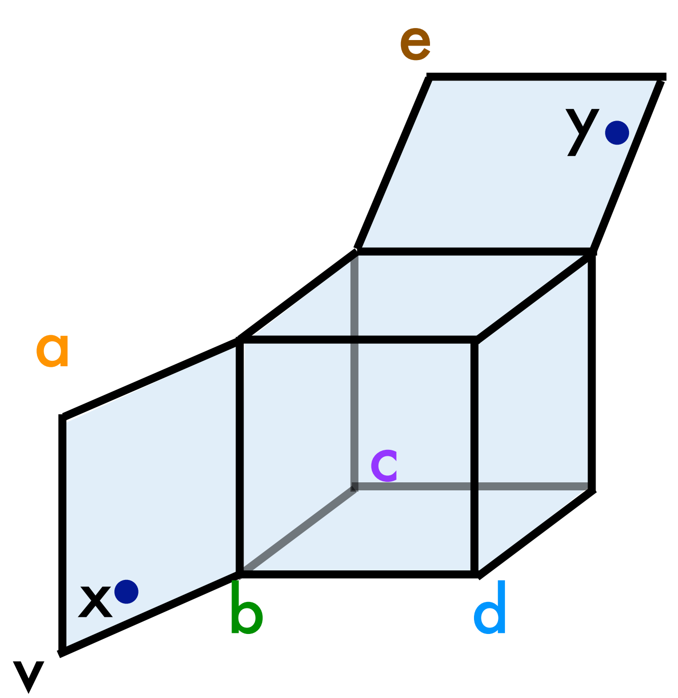
- choose a valid starting cube sequence based on x and y

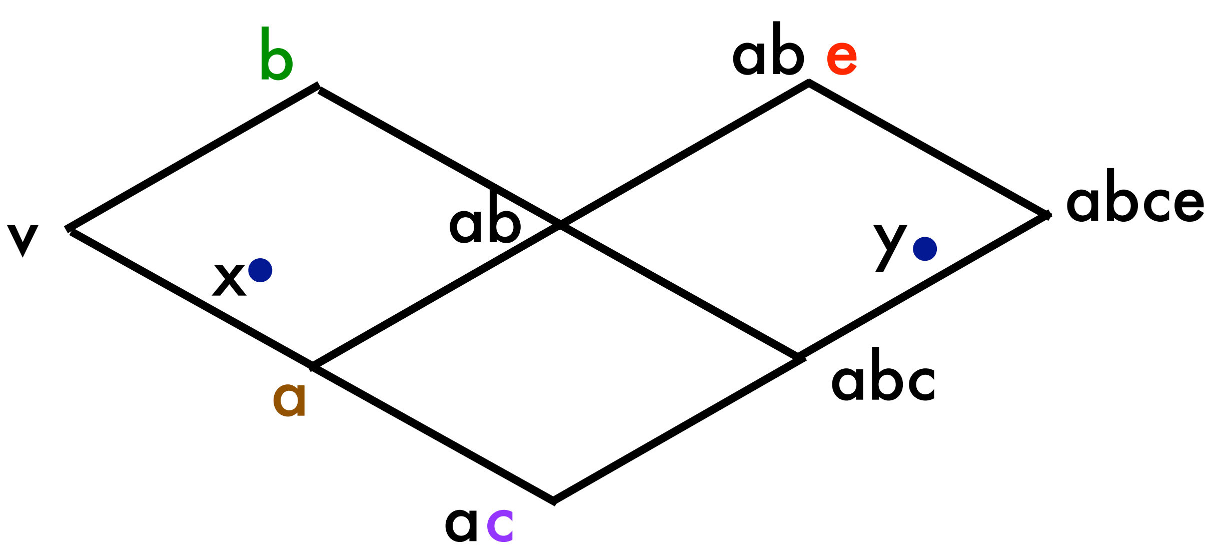




2. Starting Cube Sequence

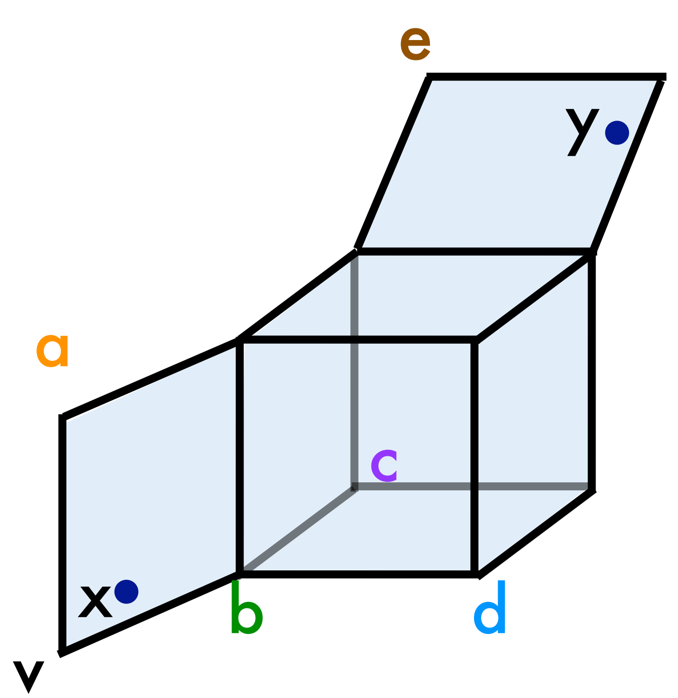
- choose a valid starting cube sequence based on x and y

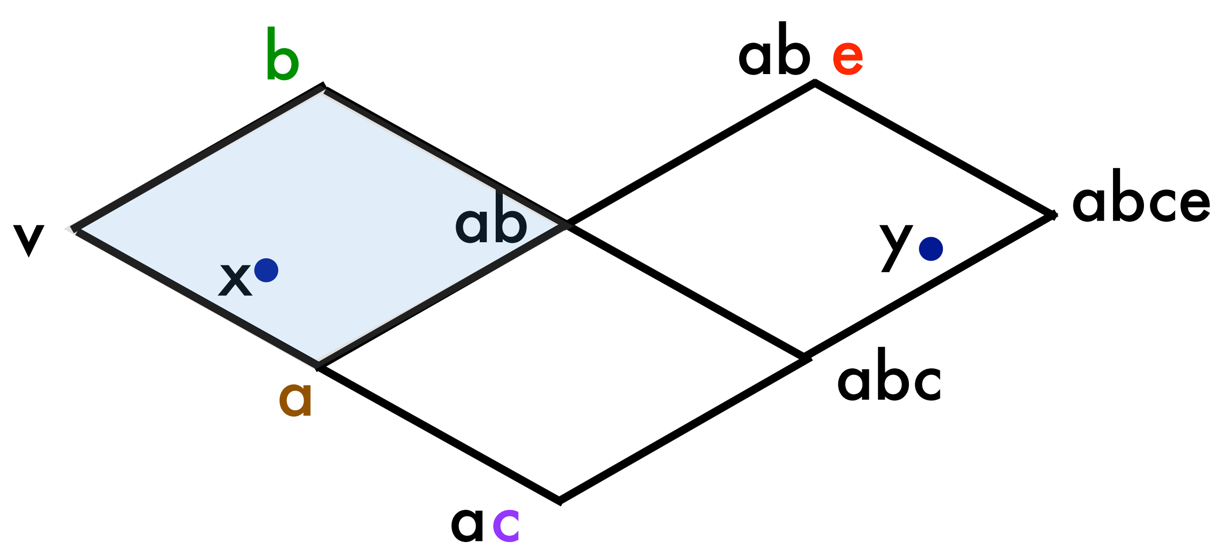




2. Starting Cube Sequence

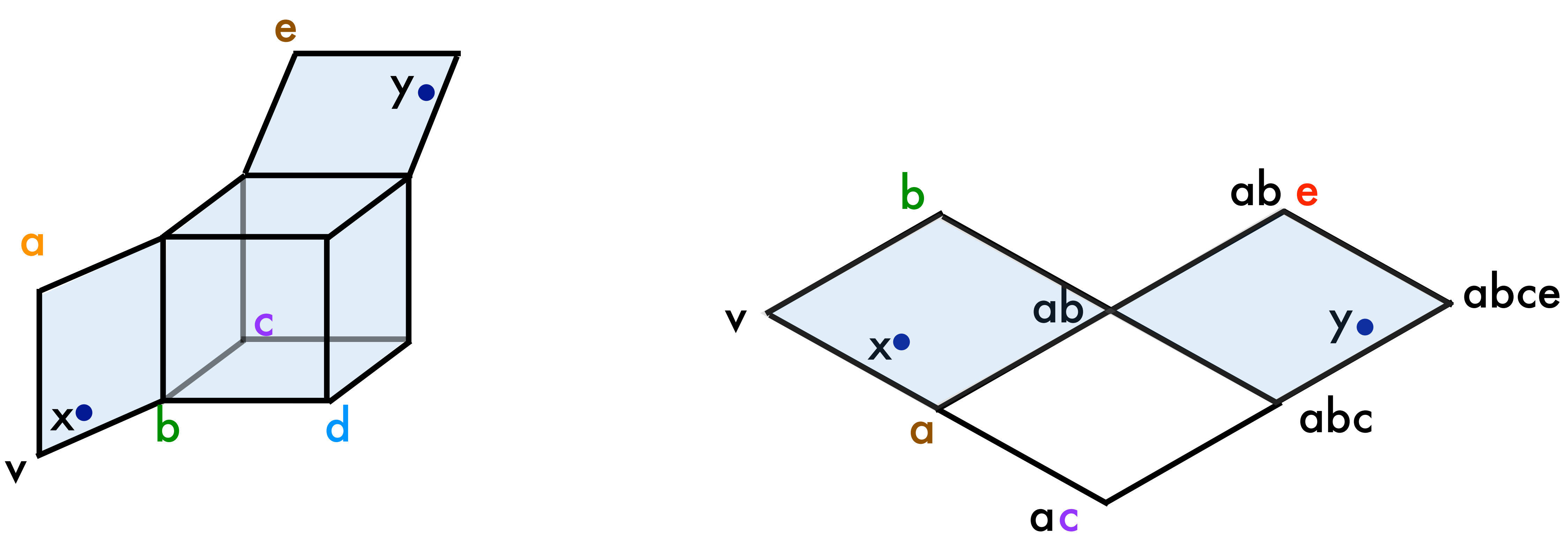
- choose a valid starting cube sequence based on x and y





2. Starting Cube Sequence

- choose a valid starting cube sequence based on x and y



Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

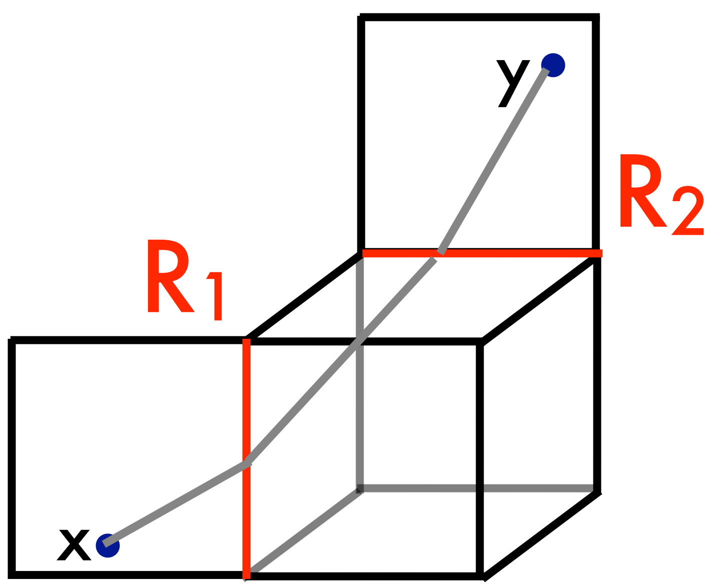
2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

3. Find geodesic through this cube sequence.

4. If possible, improve cube sequence and repeat from 3.

3. Touring Problem

- rephrase as convex optimization problem
- solvable as a second order cone problem in polynomial time (Polishchuk and Mitchell, 2005)



Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

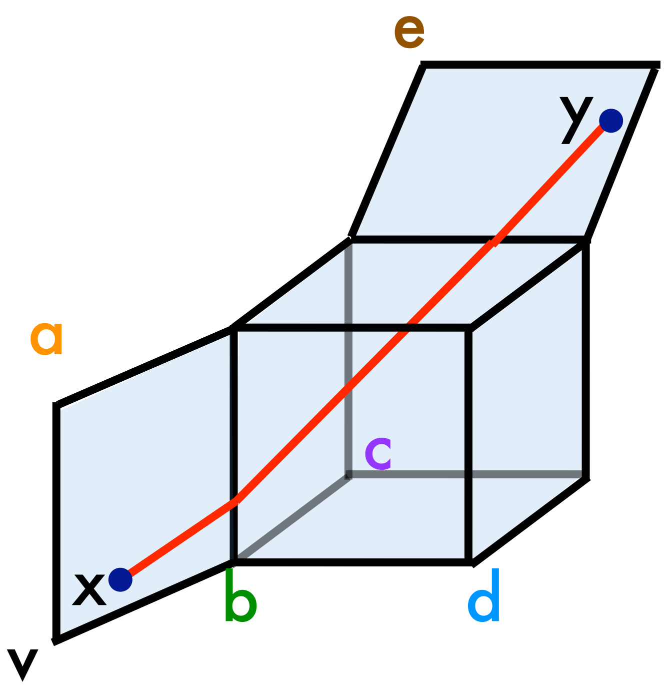
2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

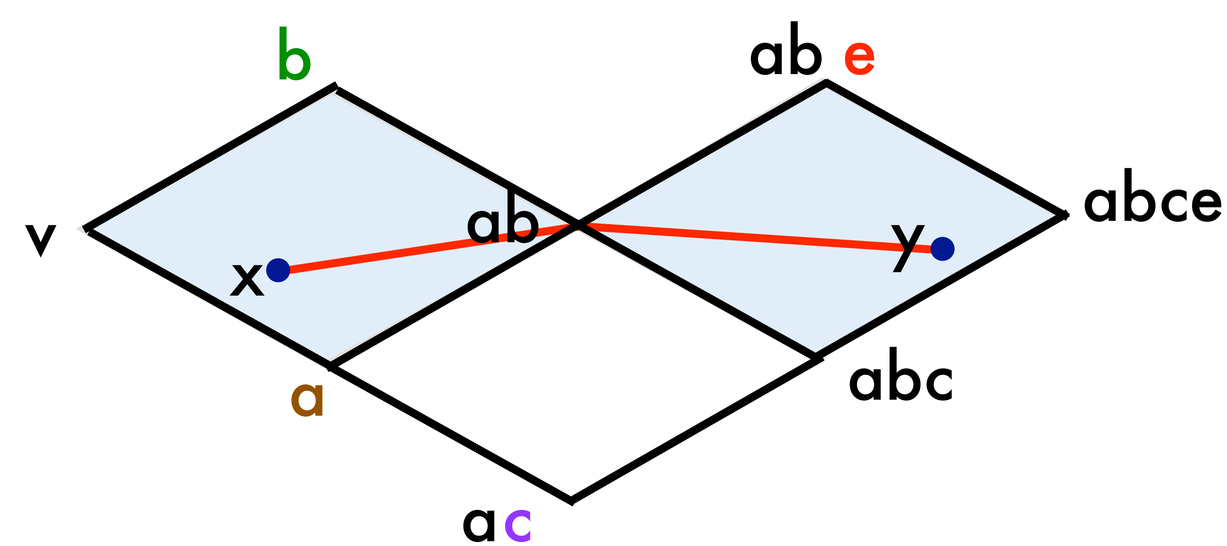
3. Find geodesic through this cube sequence.

4. If possible, improve cube sequence and repeat from 3.

4. Improve the Path

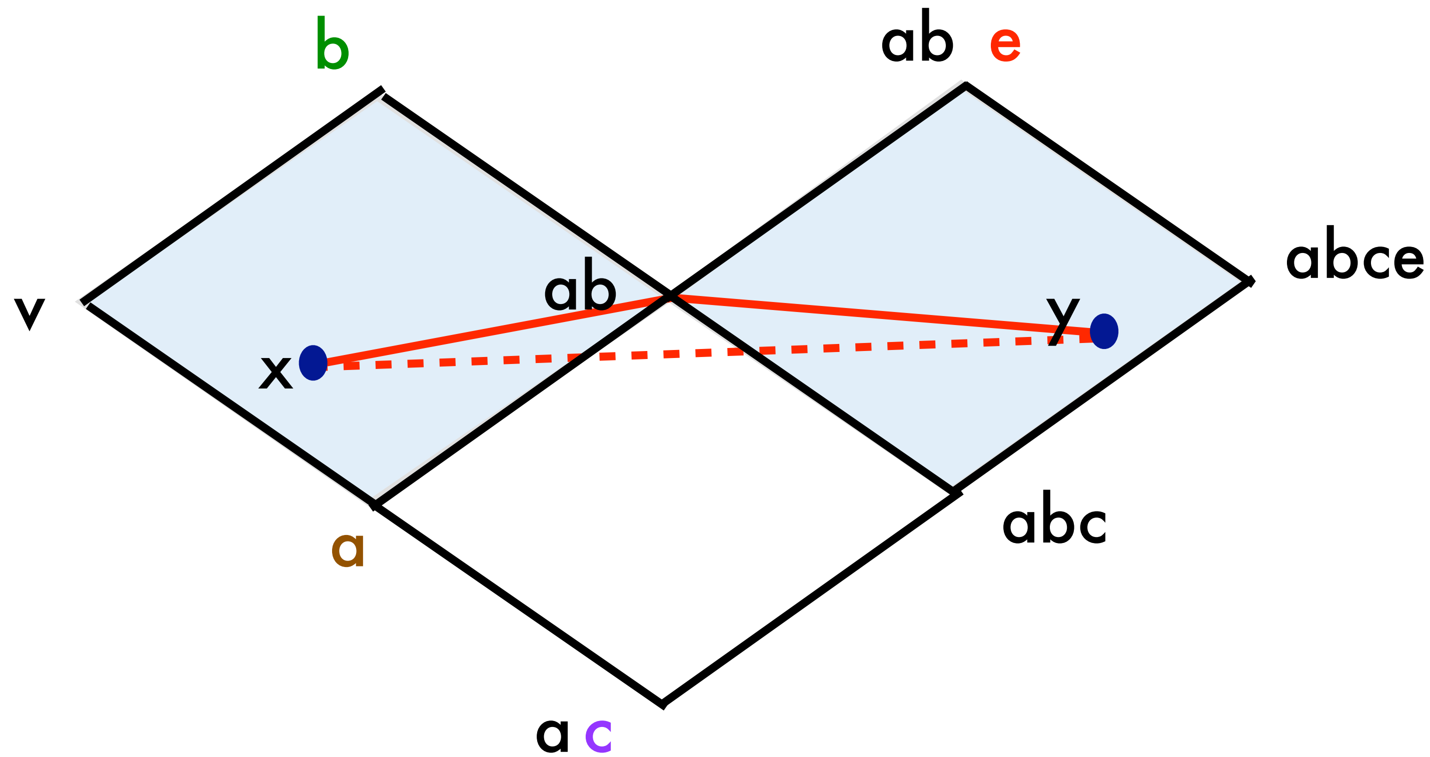
4. Can geodesic be improved?
   If yes, get a new cube sequence; go to step 3. If no, then done.





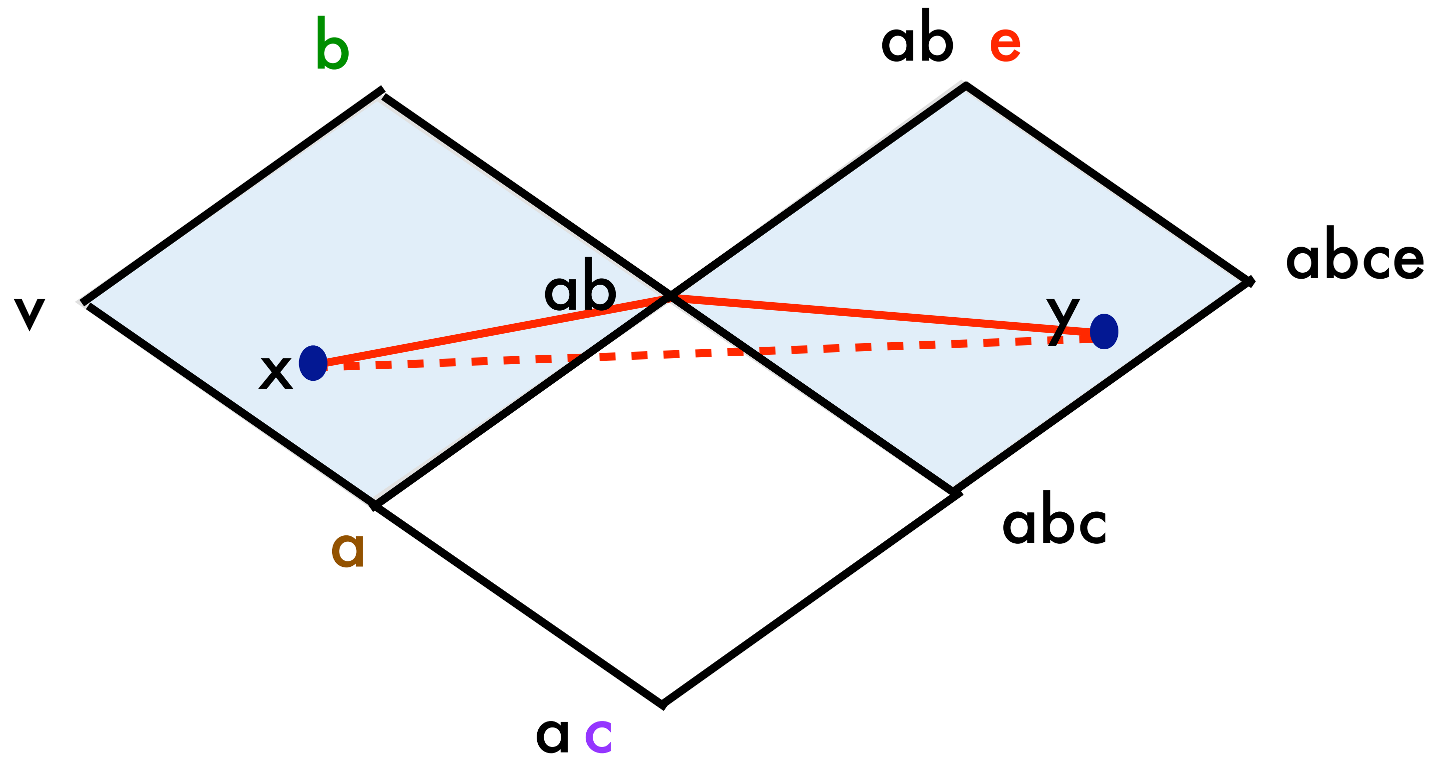
4. Improve the Path

- Only need to check geodesic bendpoints



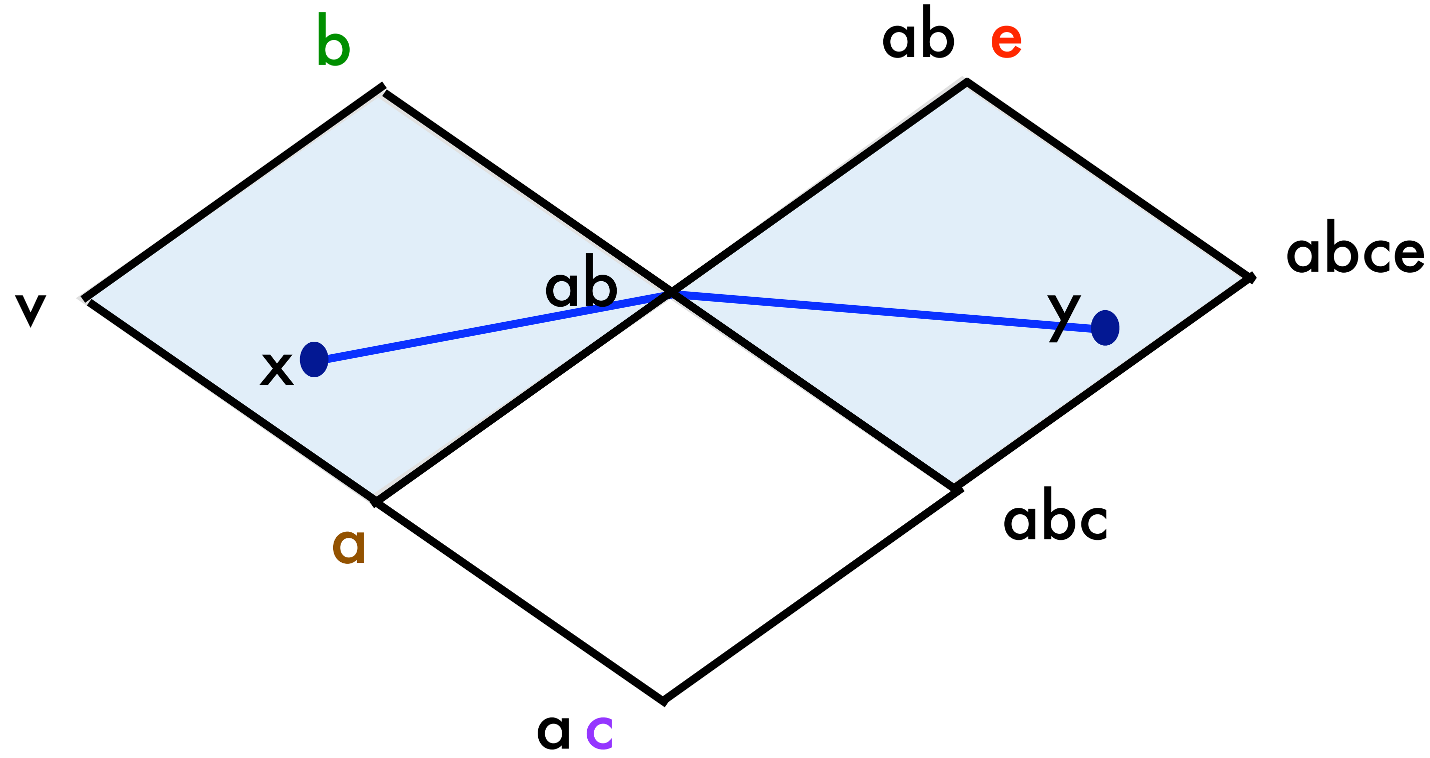
4. Improve the Path

- Only need to check geodesic bendpoints



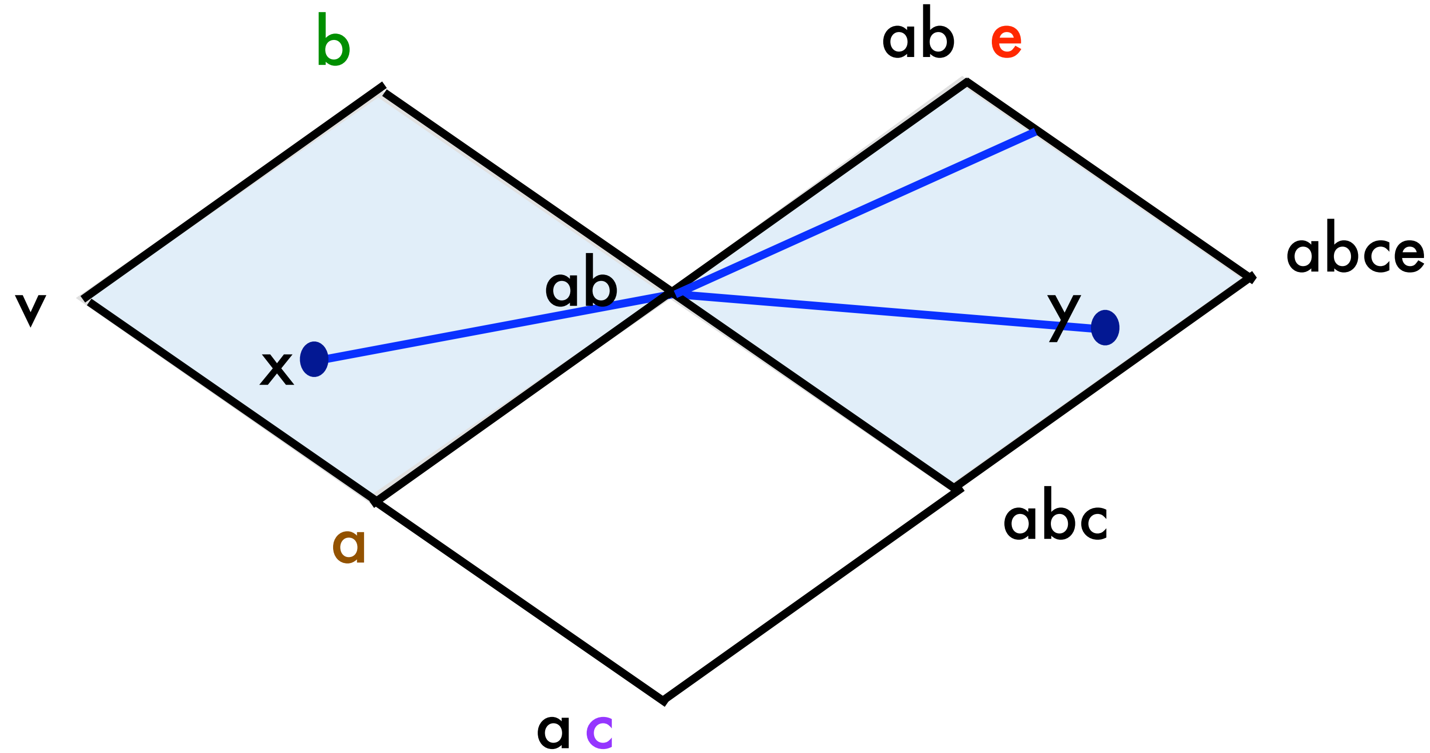
Cube exists?

4. Improve the Path



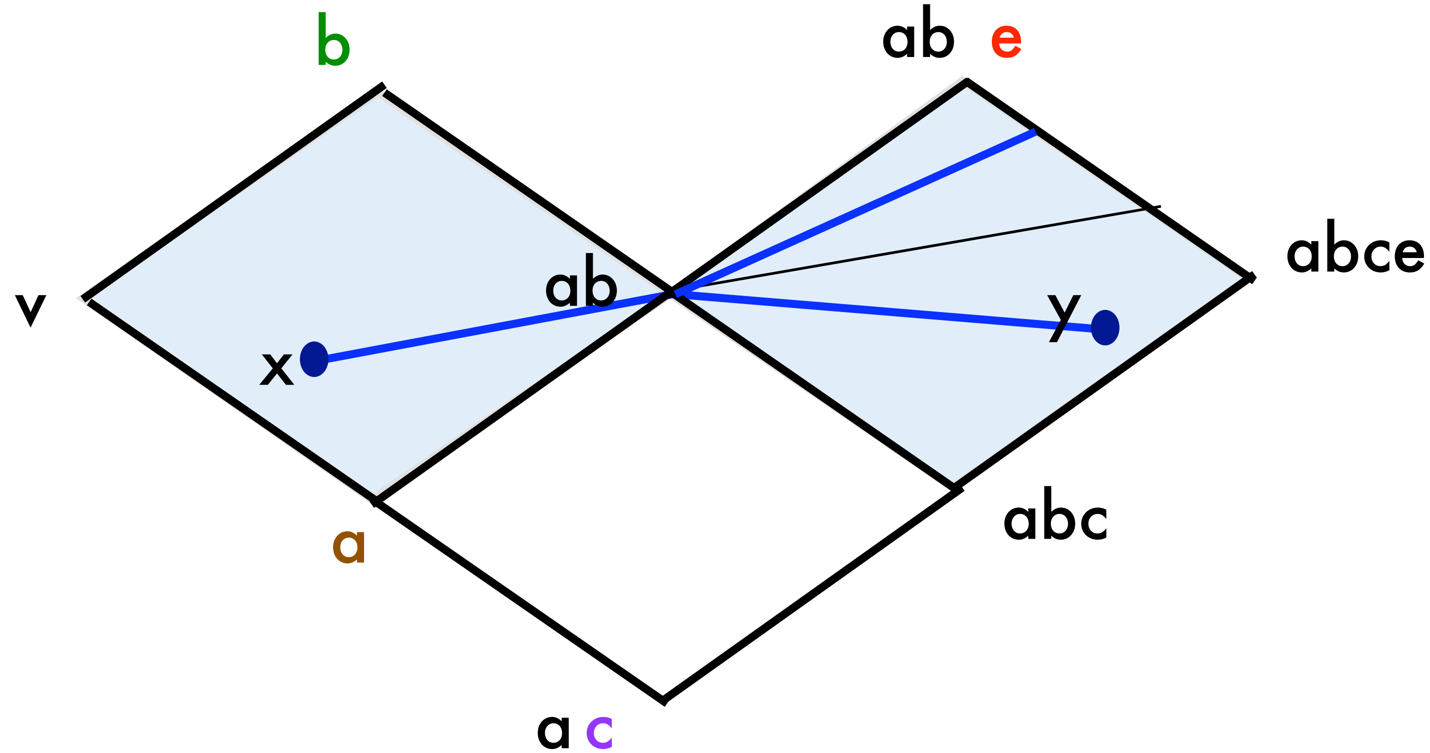
Shorter to go through this cube?

4. Improve the Path



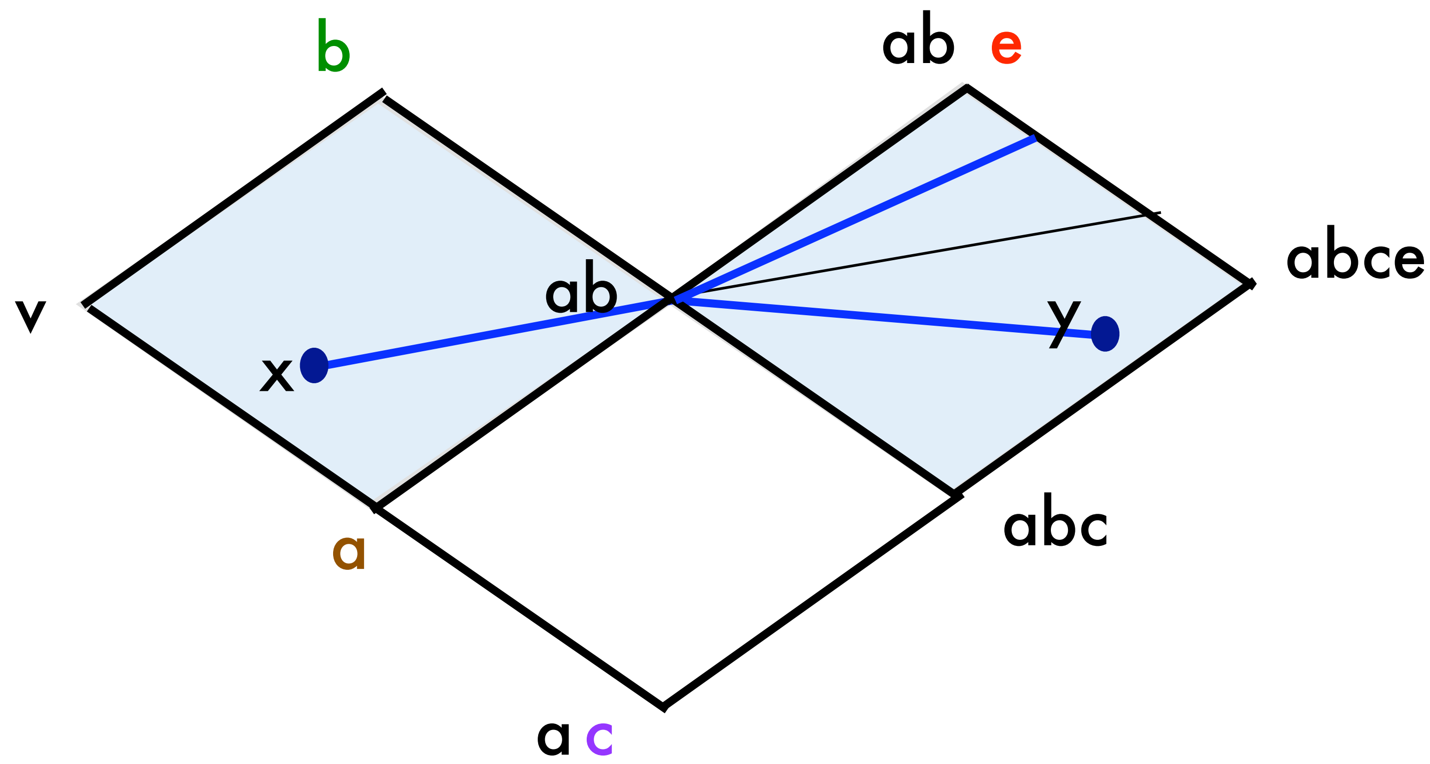
Shorter to go through this cube?

4. Improve the Path



Shorter to go through this cube?

4. Improve the Path



Shorter to go through this cube?
Check for both by finding a min weight vertex cover on a bipartite graph. (O. and Provan, 2011)

Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

3. Find geodesic through this cube sequence.

4. If possible, improve cube sequence and repeat from 3.

Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

polynomial

2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

3. Find geodesic through this cube sequence.

4. If possible, improve cube sequence and repeat from 3.

Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

   polynomial

2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

   polynomial

3. Find geodesic through this cube sequence.

4. If possible, improve cube sequence and repeat from 3.

Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

polynomial

2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

polynomial

3. Find geodesic through this cube sequence.

polynomial

4. If possible, improve cube sequence and repeat from 3.

Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

polynomial

2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

polynomial

3. Find geodesic through this cube sequence.

polynomial

4. If possible, improve cube sequence and repeat from 3.

polynomial

Outline

1. Coordinatize the CAT(0) complex: Establish a bijection with *posets with inconsistent pairs*. Coordinates = poset elements

polynomial

2. Reduce problem to subcomplex containing geodesic and find starting cube sequence.

polynomial

3. Find geodesic through this cube sequence.

polynomial

4. If possible, improve cube sequence and repeat from 3.

polynomial

unknown: # of iterations in general

Algorithm in Tree Space

Theorem (O. 2011, O. and Provan 2011):

In tree space:

- $\leq n - 2$ iterations
- geodesics can be computed using a linear algorithm instead of as touring problems
- complexity: $O(n^4)$

- can be iteratively used to compute mean and variance for a set of trees
- goal: Principal Component Analysis

Thank You

- references:

  - F. Ardila, M. Owen, S. Sullivant. Geodesics in CAT(0) cubical complexes, *Advances in Applied Mathematics, 48:142-163, 2012*. arXiv:1101.2428.
  - L. Billera, S. Holmes, K. Vogtmann. Geometry of the space of phylogenetic trees, *Advances in Applied Mathematics*, 27:733-767, 2001.
  - M. Owen and S. Provan. A fast algorithm for computing geodesic distances in tree space, *IEEE/ACM Trans. Comp. Biol. and Bioinform.*, 8:2-13, 2011.
  - M. Owen. Computing geodesic distances in tree space. Accepted *SIAM Journal on Discrete Mathematics*. arXiv:0903.0696.