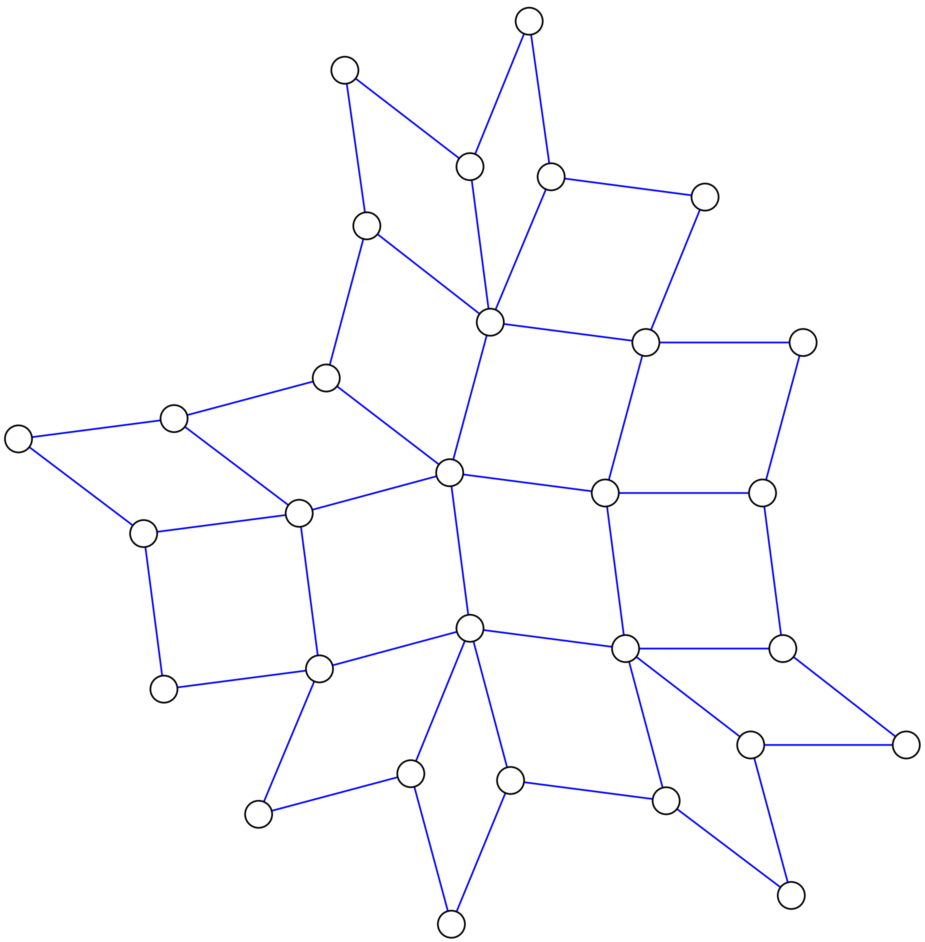
Principles of Graph Drawing

David Eppstein

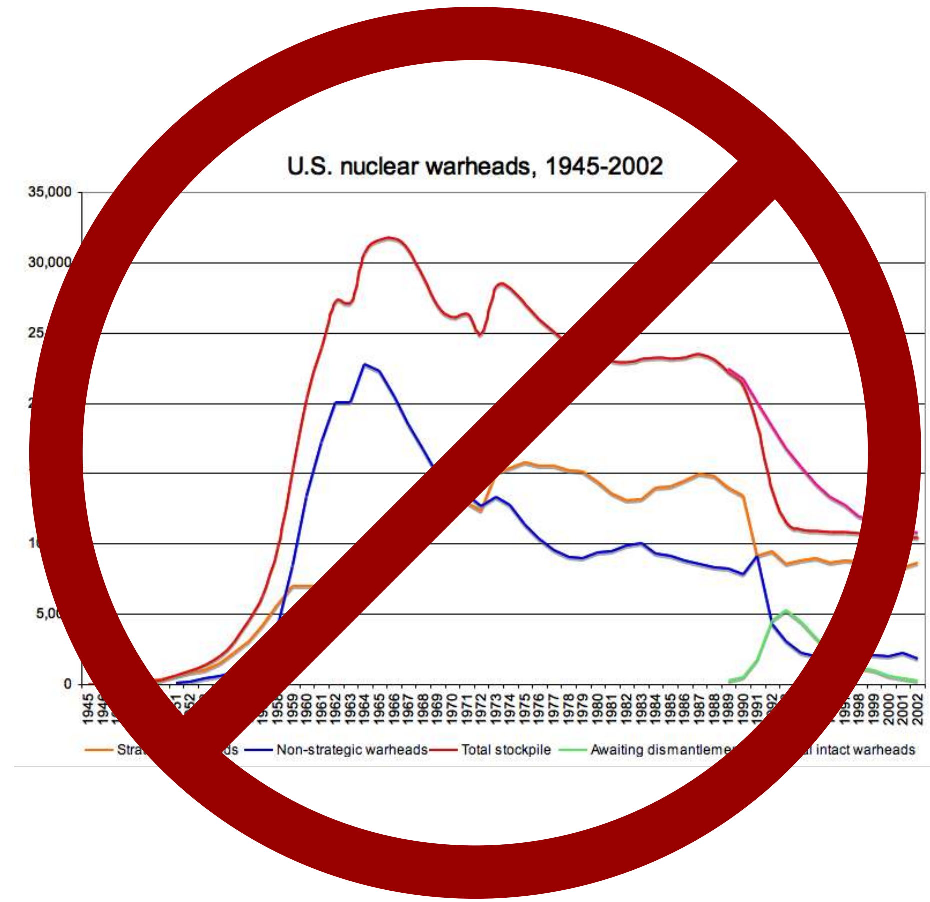
Univ. of California, Irvine
Computer Science Department

Graph drawing is...

... about combinatorial, not numerical data



A graph

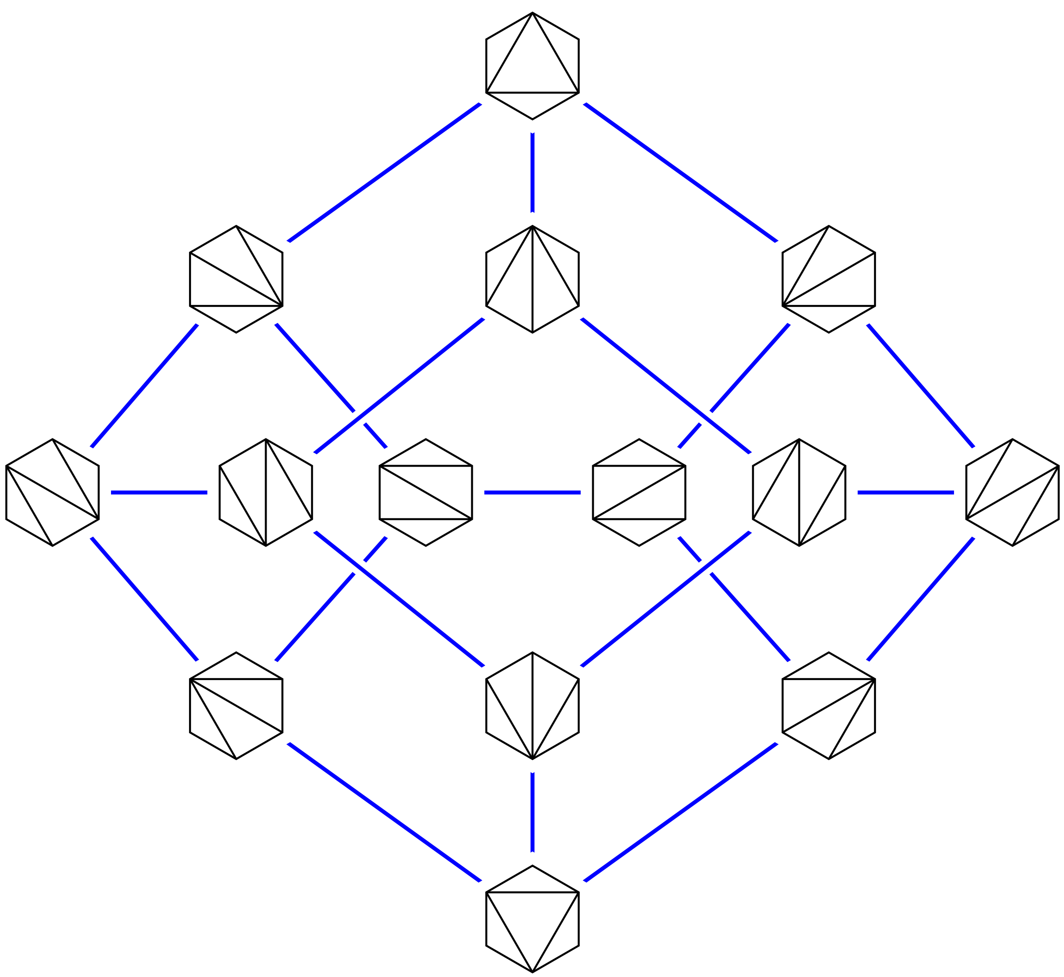


Not a graph

Graph drawing is...

... about combinatorial, not numerical data

... a way to communicate specific graphs to others



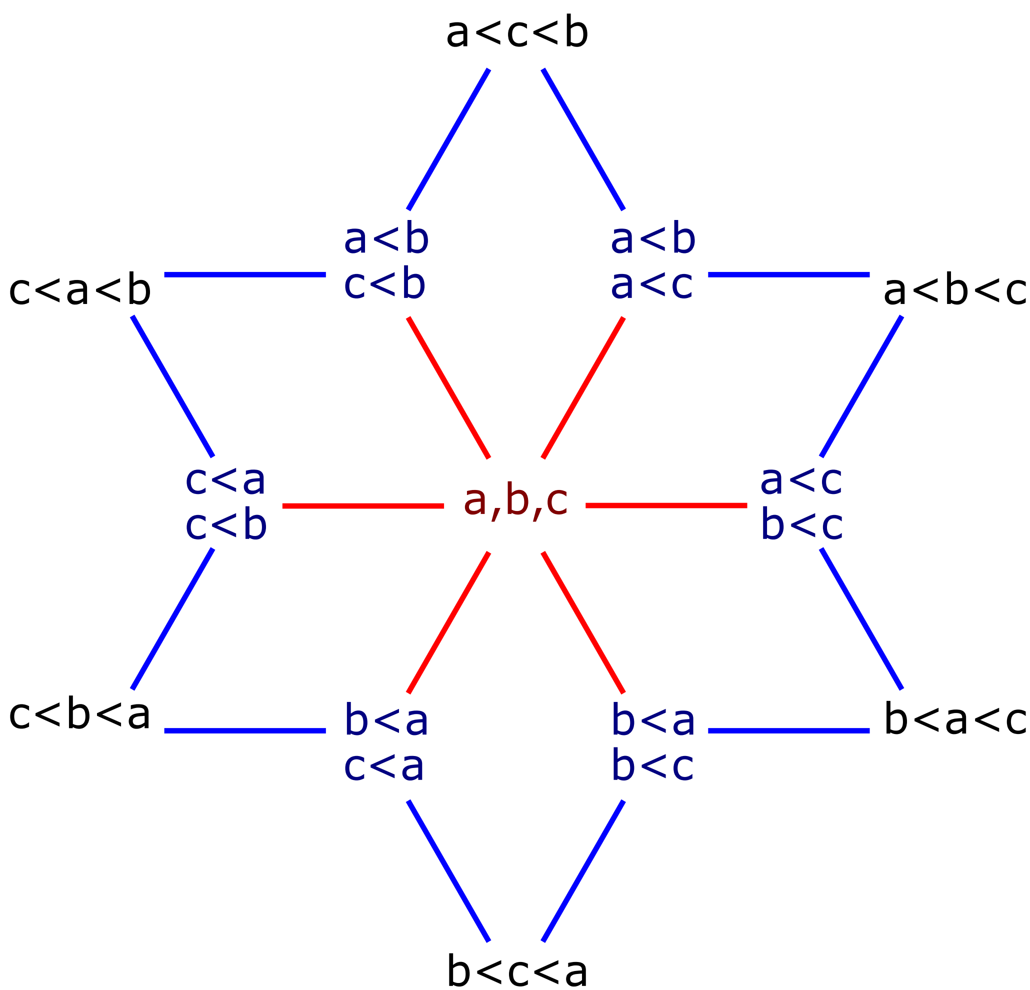
Vertices = hexagon triangulations

Edges = flips between triangulations

Graph drawing is...

... about combinatorial, not numerical data

... a way to communicate specific graphs to others

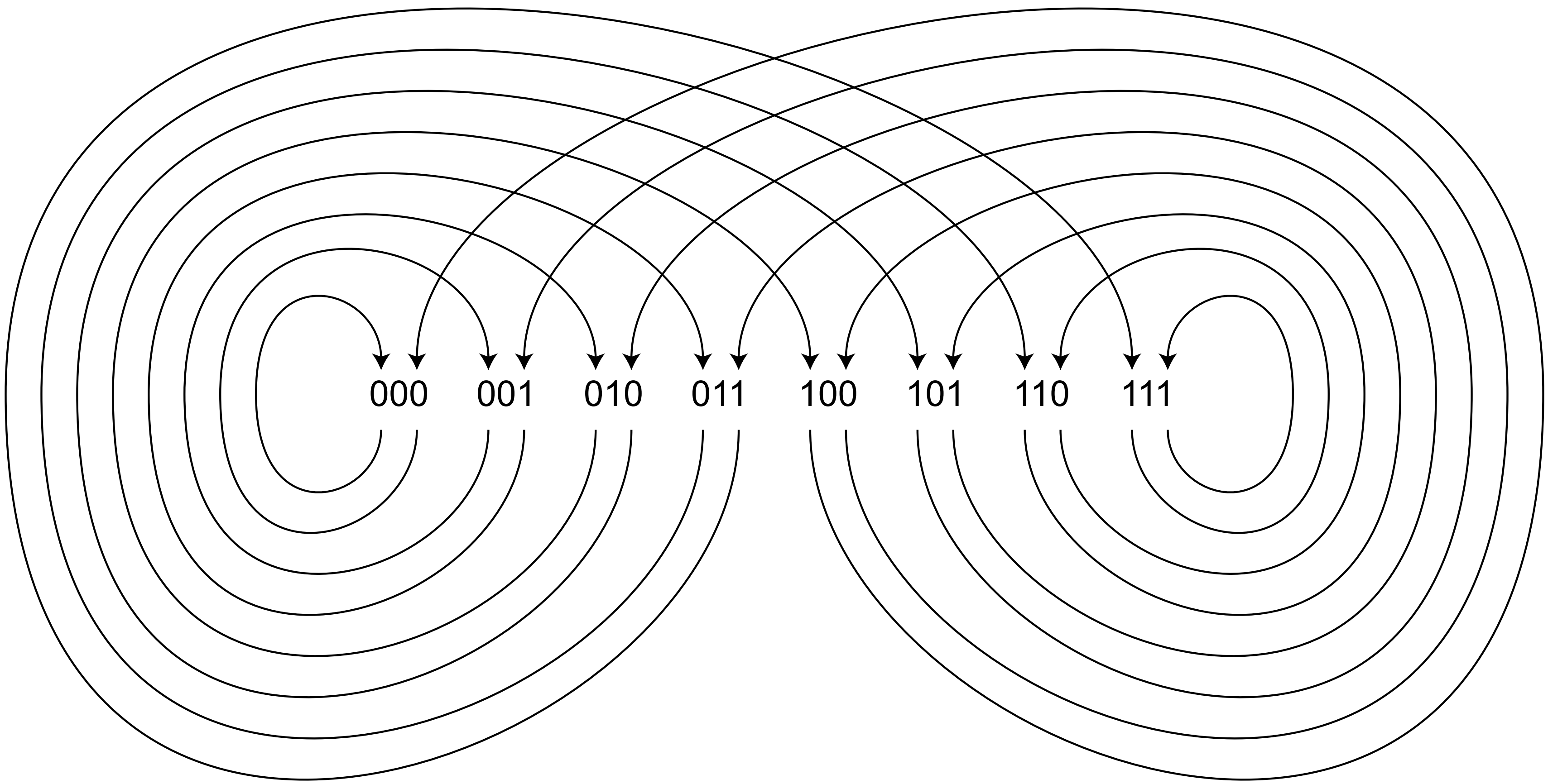


The 13 possible weak orderings on a set of three elements

Graph drawing is...

... about combinatorial, not numerical data

... a way to communicate specific graphs to others



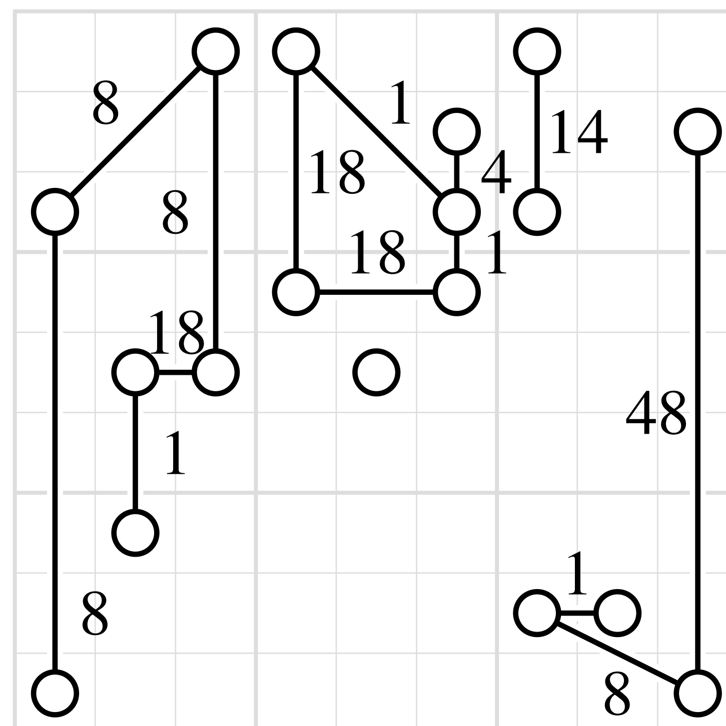
The three-dimensional binary De Bruijn graph

Graph drawing is...

... about combinatorial, not numerical data

... a way to communicate specific graphs to others

5	3			7	6		2	9
1	2	6	3	9		5	7	
	7	9	5	2			6	3
2	6	3		5		9	4	7
7			2		9	3	5	6
9	4	5	7	6	3	2	8	1
3		4	6	8	5	7	9	2
6	9	7	4	3	2			5
	5	2	9	1	7	6	3	



A graph in which each edge connects the only two cells that can hold a given digit in some row, column, or 3x3 block of a Sudoku puzzle

Graph drawing is...

... about combinatorial, not numerical data

... a way to communicate specific graphs to others

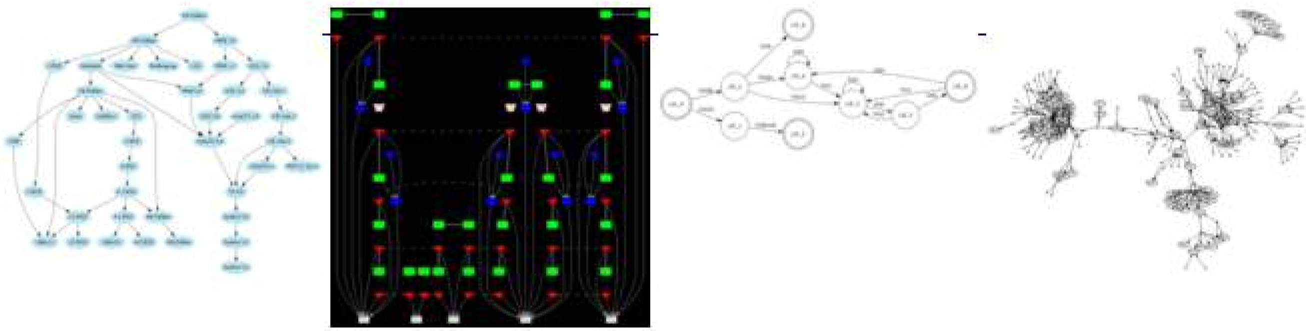
... a collection of knowledge, algorithms, and software
for efficiently and robustly finding legible layouts of graphs

Graphviz

http://www.graphviz.org/

About
Download
News
Gallery
Documentation
Theory
Bugs
MailingList
License
Resources
Credits

Graphviz - Graph Visualization Software



Graph Visualization

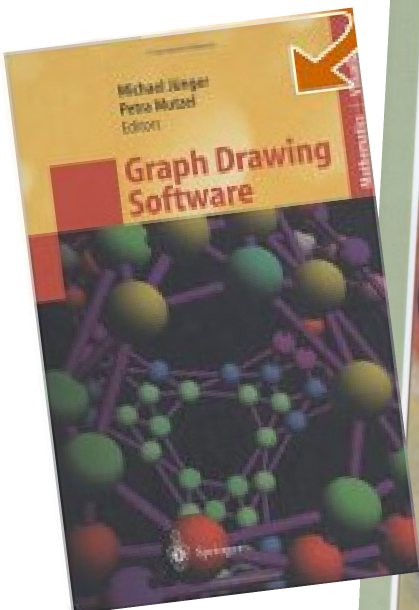
Graph visualization is a way of representing structural information as diagrams of abstract graphs and networks. Automatic graph drawing has many important applications in software engineering, database and web design, networking, and in visual interfaces for many other domains.

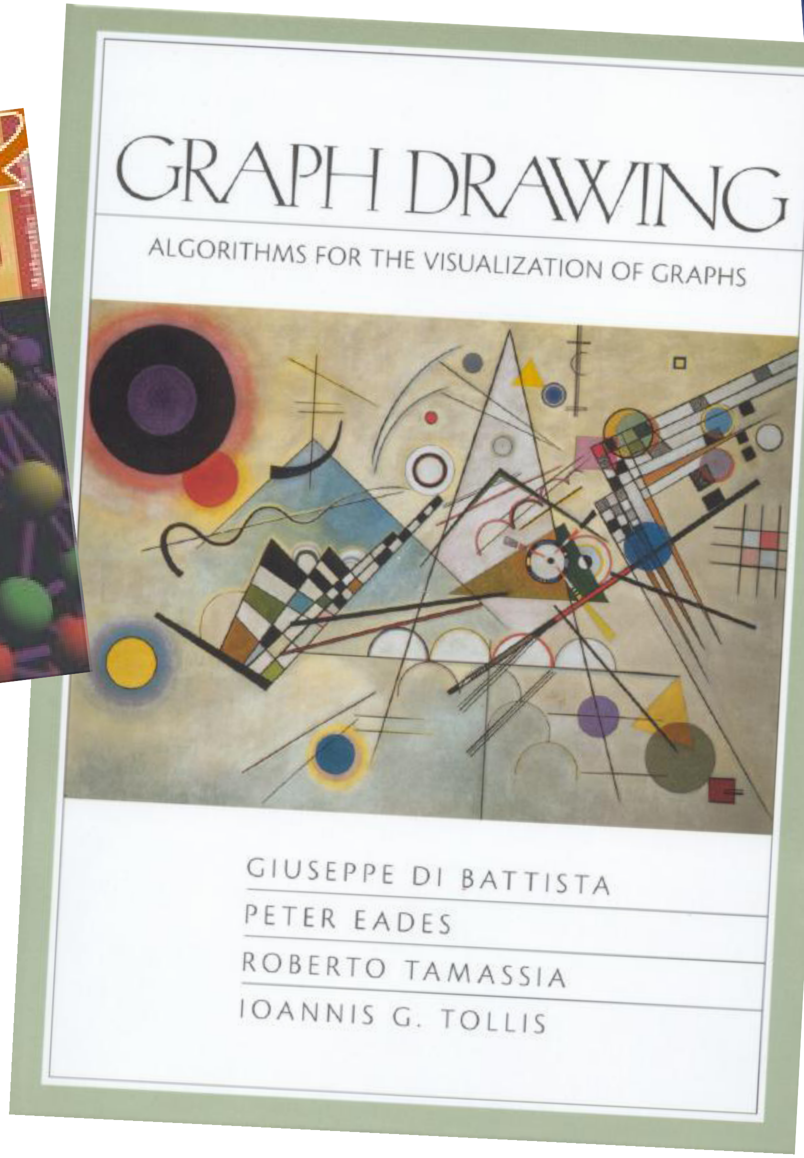
Graphviz is open source graph visualization software. It has several main graph layout programs. See the gallery for some sample layouts. It also has web and interactive graphical interfaces, and auxiliary tools, libraries, and language bindings.

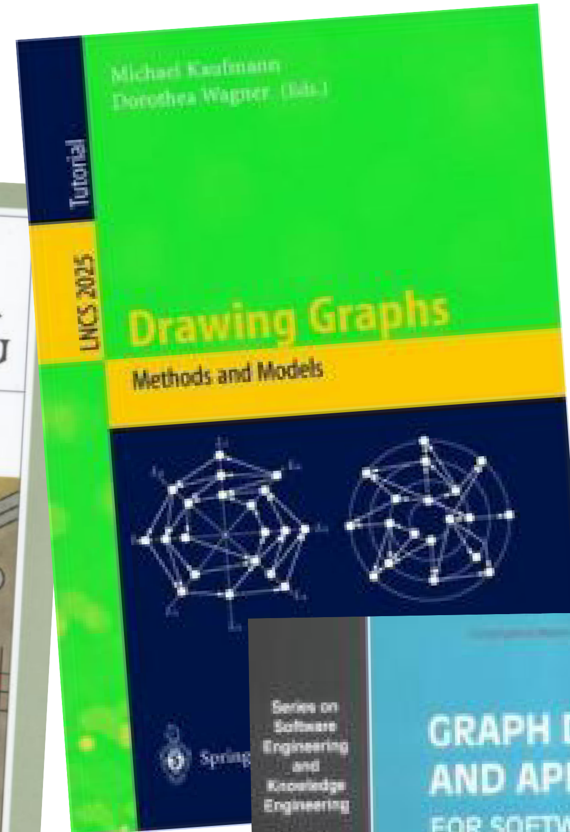
The Mac OS X edition of Graphviz, by Glen Low, won two 2004 Apple Design Awards.

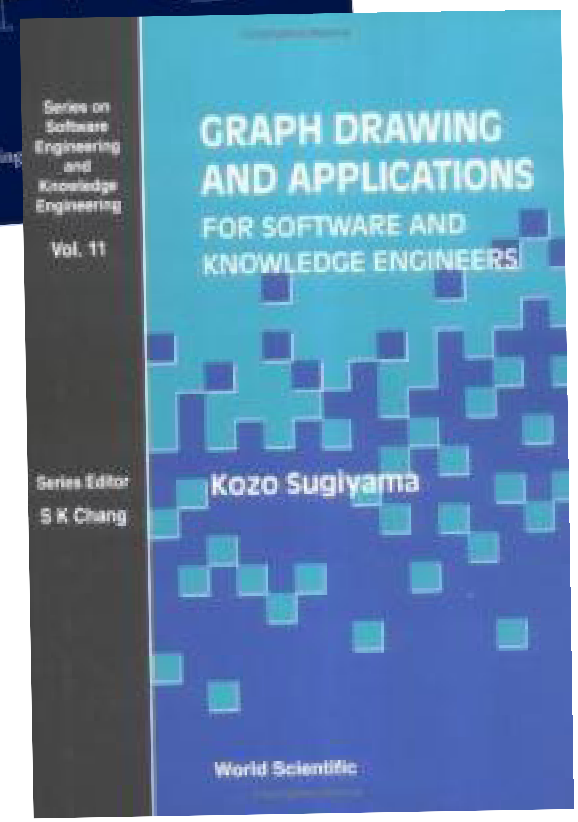
The Graphviz layout programs take descriptions of graphs in a simple text language, and make diagrams in several useful formats such as images and SVG for web pages, Postscript for inclusion in PDF or other documents; or display in an interactive graph browser. (Graphviz also supports GXL, an XML dialect.)

Graphviz has many useful features for concrete diagrams, such as options for colors, fonts, tabular node layouts, line styles, hyperlinks, and custom shapes.


Michael Jünger
Petra Mutzel
Editors
Graph Drawing Software


GRAPH DRAWING
ALGORITHMS FOR THE VISUALIZATION OF GRAPHS
GIUSEPPE DI BATTISTA
PETER EADES
ROBERTO TAMASSIA
IOANNIS G. TOLLIS


Michael Kaufmann
Dorothea Wagner (Eds.)
Tutorial
LNCS 2025
Drawing Graphs
Methods and Models


Series on Software Engineering and Knowledge Engineering
Vol. 11
GRAPH DRAWING AND APPLICATIONS
FOR SOFTWARE AND KNOWLEDGE ENGINEERS
Kozo Sugiyama
Series Editor
S K Chang
World Scientific

Graph drawing is...

... about combinatorial, not numerical data

... a way to communicate specific graphs to others

... a collection of knowledge, algorithms, and software
for efficiently and robustly finding legible layouts of graphs

... an active and welcoming international research community
with a 15-year-old annual peer-reviewed research conference

Graph Drawing 2008
Heraklion, Crete
September 21-24
http://www.gd2008.org/

Submission deadline May 31

Topics include
- Human legibility of drawings
- Algorithms for making drawings
- Graph drawing software systems
- Application areas
  (e.g. social nets, bioinformatics)
- Mathematics of graph embeddings

Graph Drawing 2009
will be in Chicago
(I am program co-chair)

I. Different drawings of the same graph will communicate different things about it

Example:

Graph with 24 vertices, describing the different ways of choosing a sequence of three out of four items

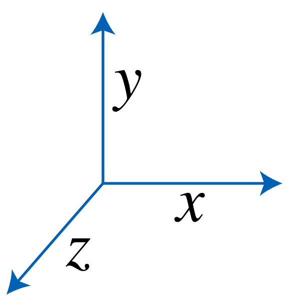
(1,2,3)	(1,2,4)	(1,3,4)	(1,3,2)	(1,4,2)	(1,4,3)
(2,1,3)	(2,1,4)	(2,3,1)	(2,3,4)	(2,4,1)	(2,4,3)
(3,1,2)	(3,1,4)	(3,2,1)	(3,2,4)	(3,4,1)	(3,4,2)
(4,1,2)	(4,1,3)	(4,2,1)	(4,2,3)	(4,3,1)	(4,3,2)

36 edges, connecting pairs that differ in a single position

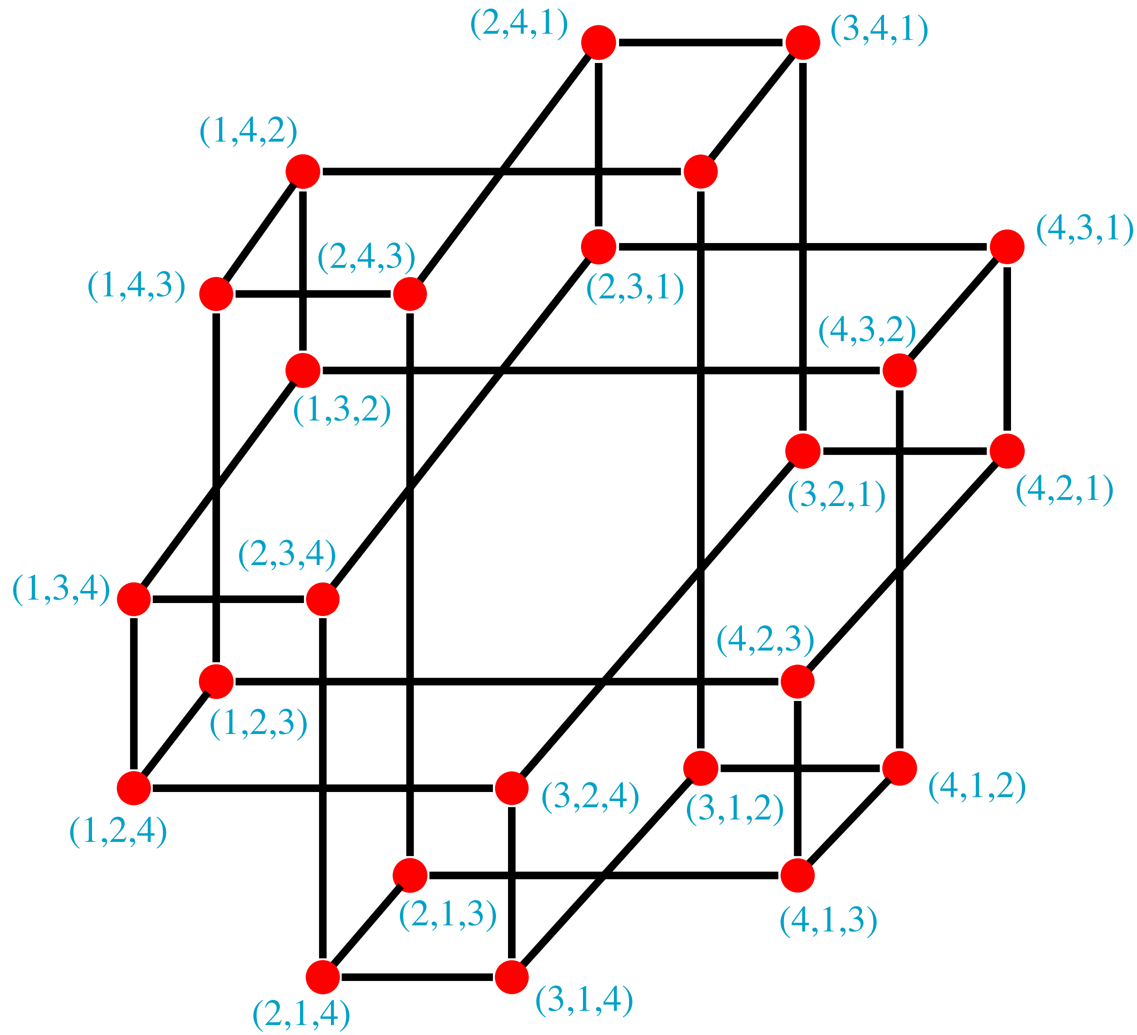
(1,2,3)—(1,2,4) (1,2,3)—(1,4,3) (1,2,3)—(4,2,3) etc.

Drawing 1: Use triples as spatial coordinates, project onto the plane

Direction of edge: which position of 3-tuple changes



Length of edge: which two values of 3-tuple change

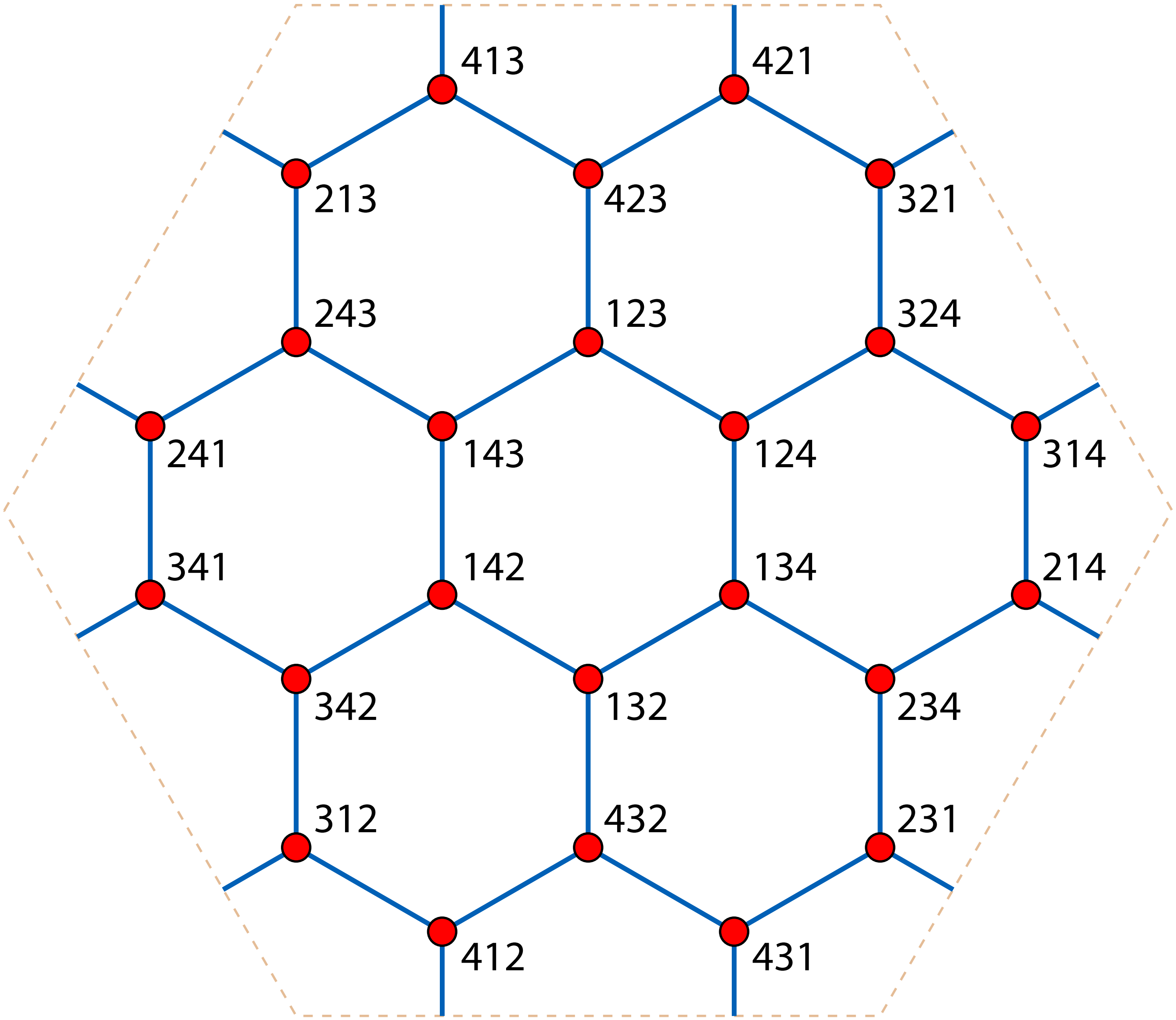


Shows plenty of symmetry

Has a natural embedding onto a 3d grid, edges are axis-parallel

("The Topology of Bendless Three-Dimensional Orthogonal Graph Drawing", E 2008, submitted)

Drawing 2: All edges unit length, 120 degree angles
Direction of edge set by the changed pair of values (not their positions!)



Forms an infinite repeating pattern in the plane

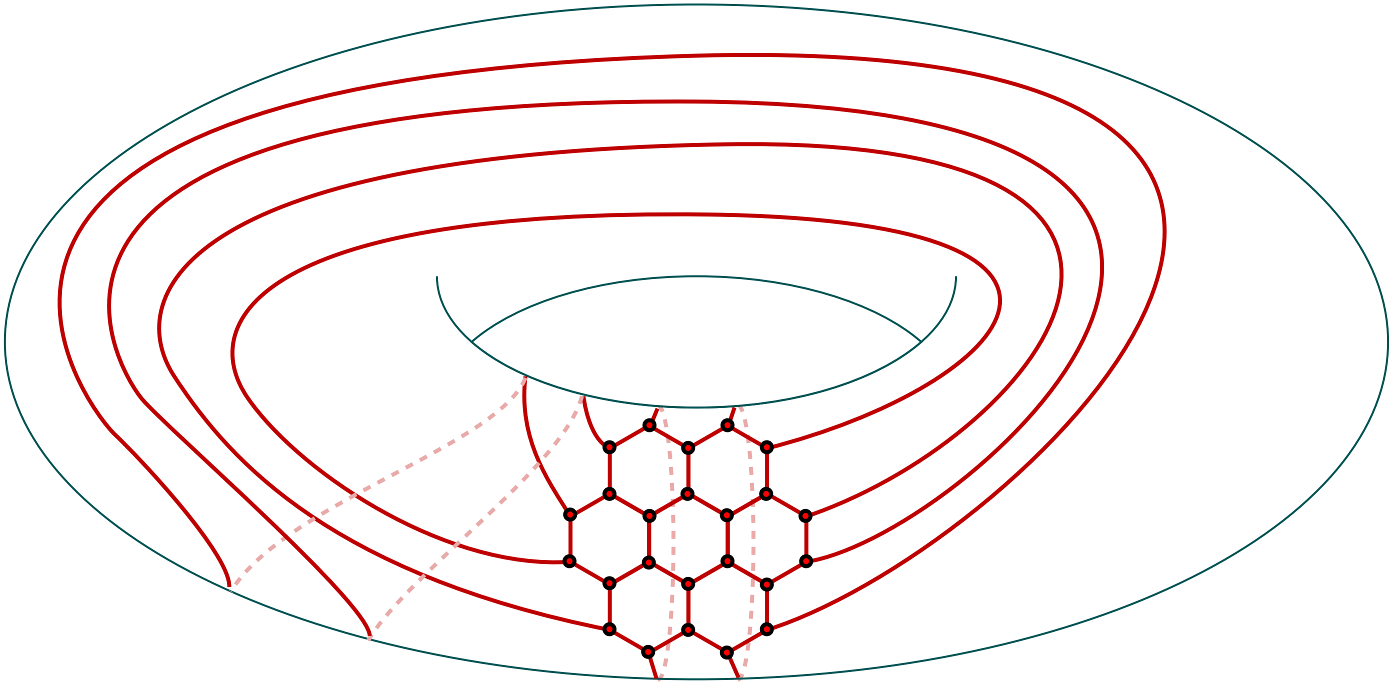
Small hexagonal cycles form a larger repeating hexagonal tile

Highly symmetric

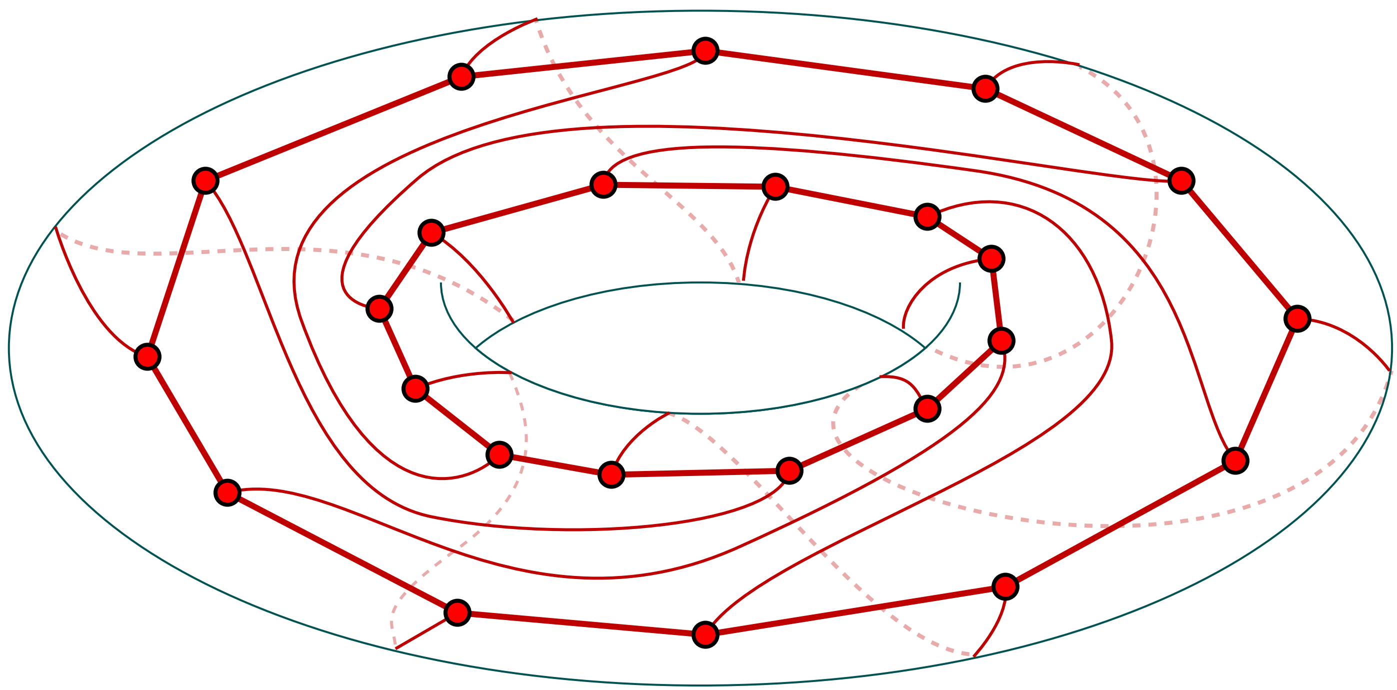
Topologically, gluing opposite edges of a hexagon forms a torus

Can we embed this torus into 3d space?

Drawings 3–4: Two different 3d torus embeddings

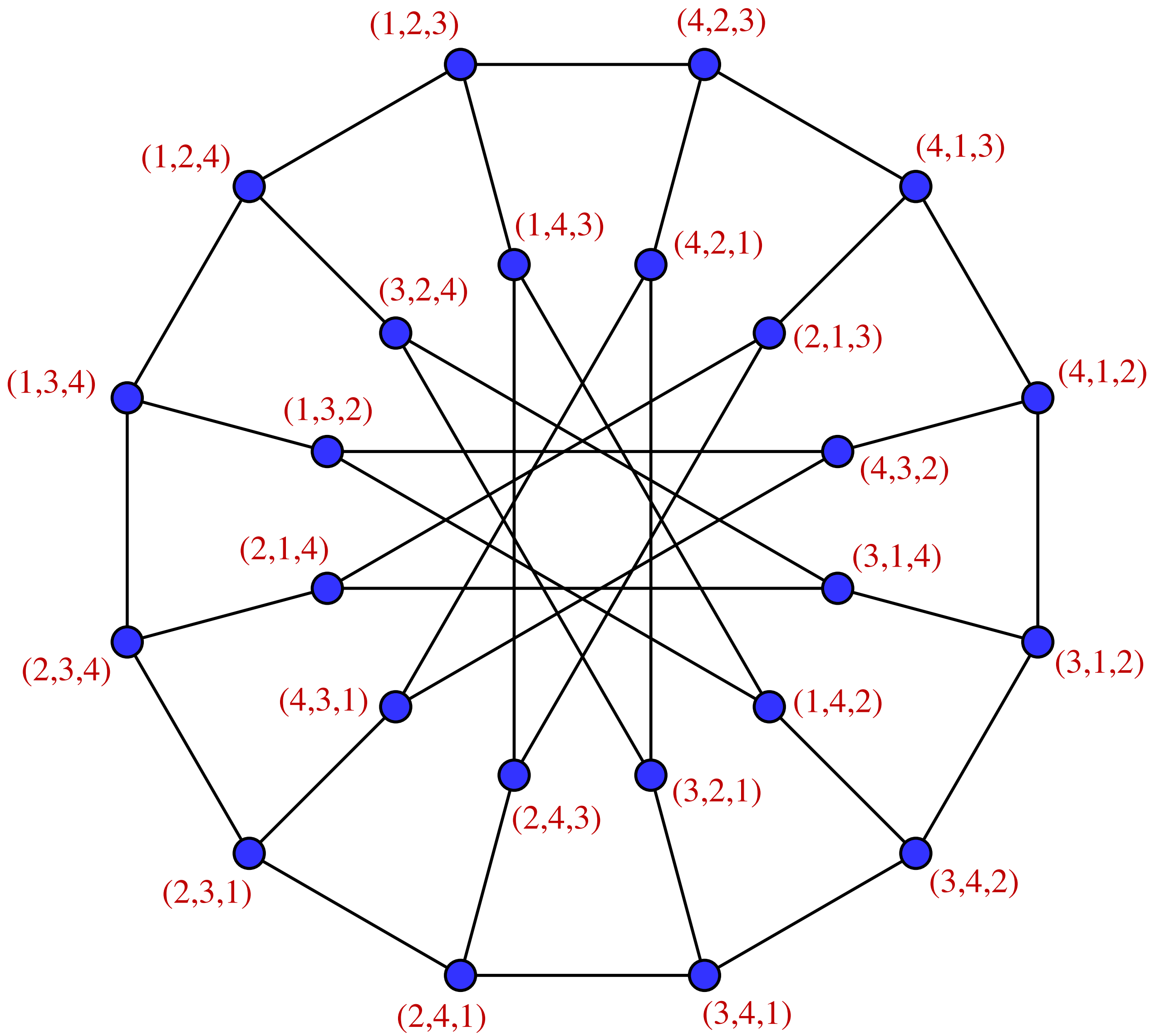


Closer connection to hexagonal tiling of the plane



More symmetric, shows decomposition into two 12-cycles

Drawing 5: Second torus, back into the plane



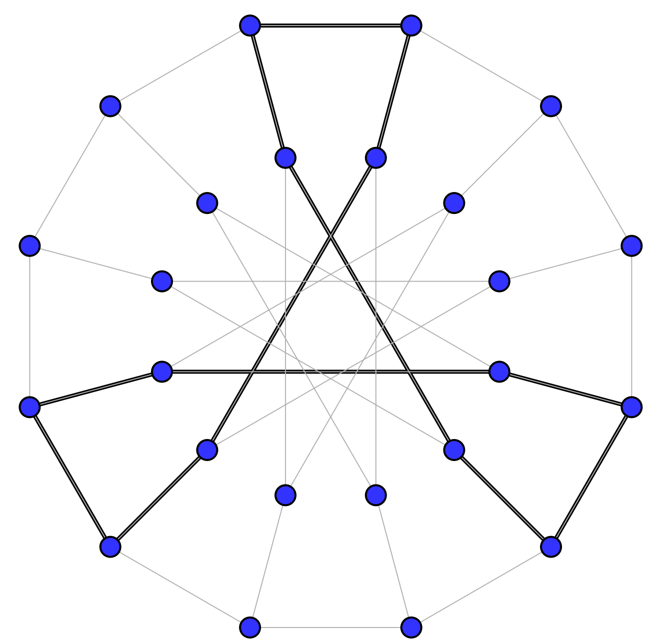
Generalized Petersen graph
(outer cycle connected to inner star, Watkins 1969)

12-fold rotational symmetry

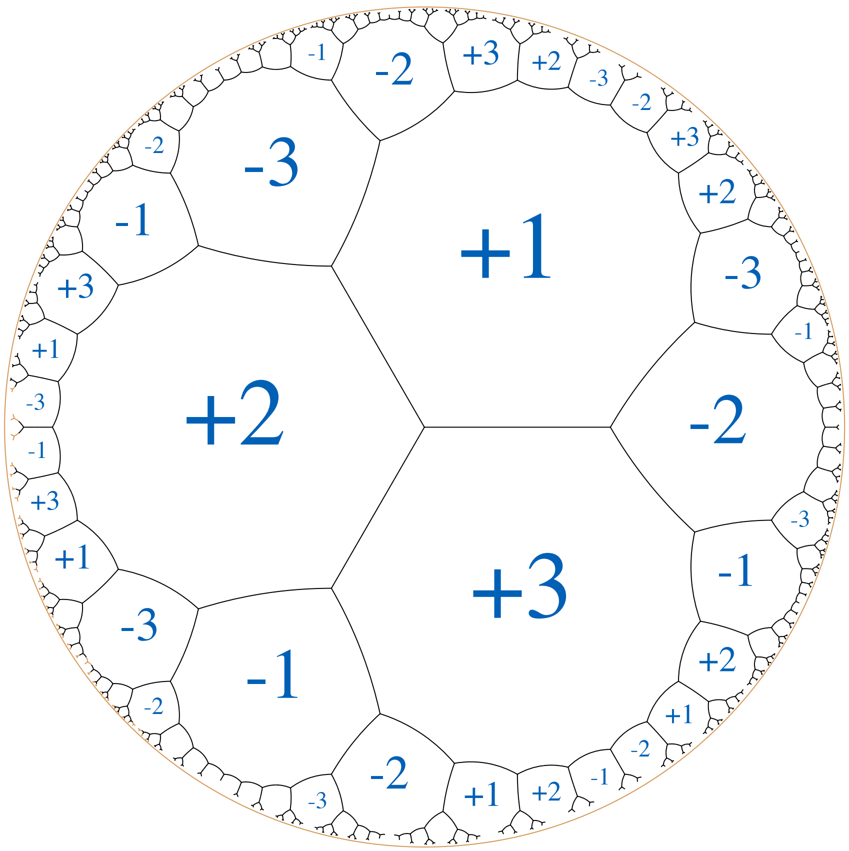
To walk around a 12-cycle: replace 3-tuple positions 1-2-3-1-2-3-1-2-3-1-2-3

Two obvious 12-cycles

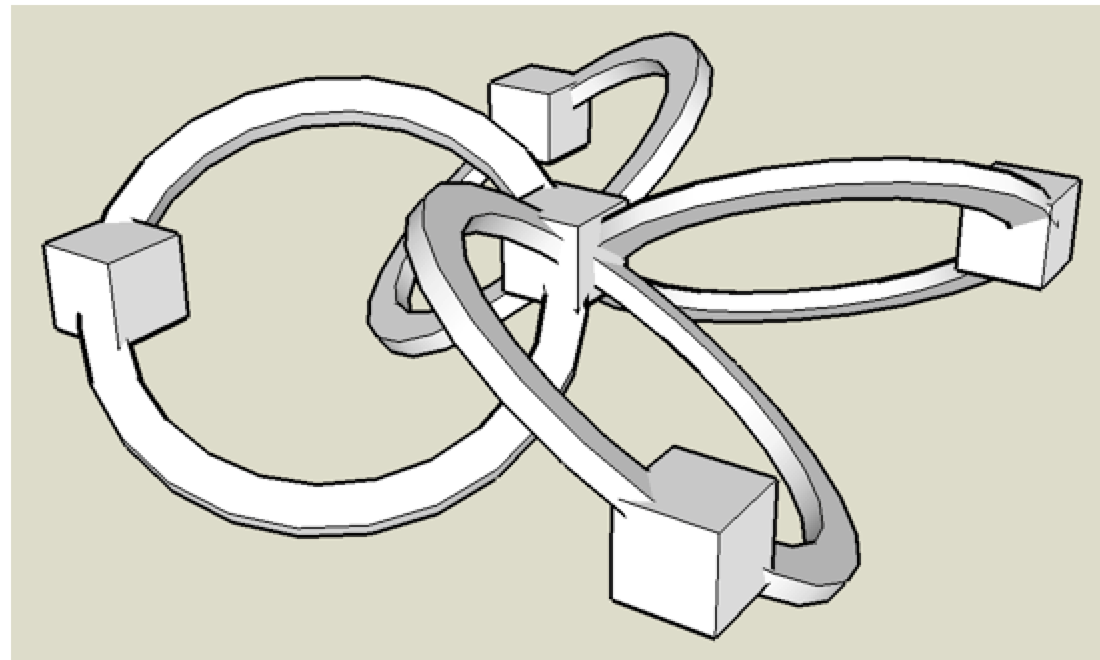
Four other 12-cycles cover the rest of the graph



Drawings 6-7: Six 12-cycles form an abstract topological surface with 4 handles

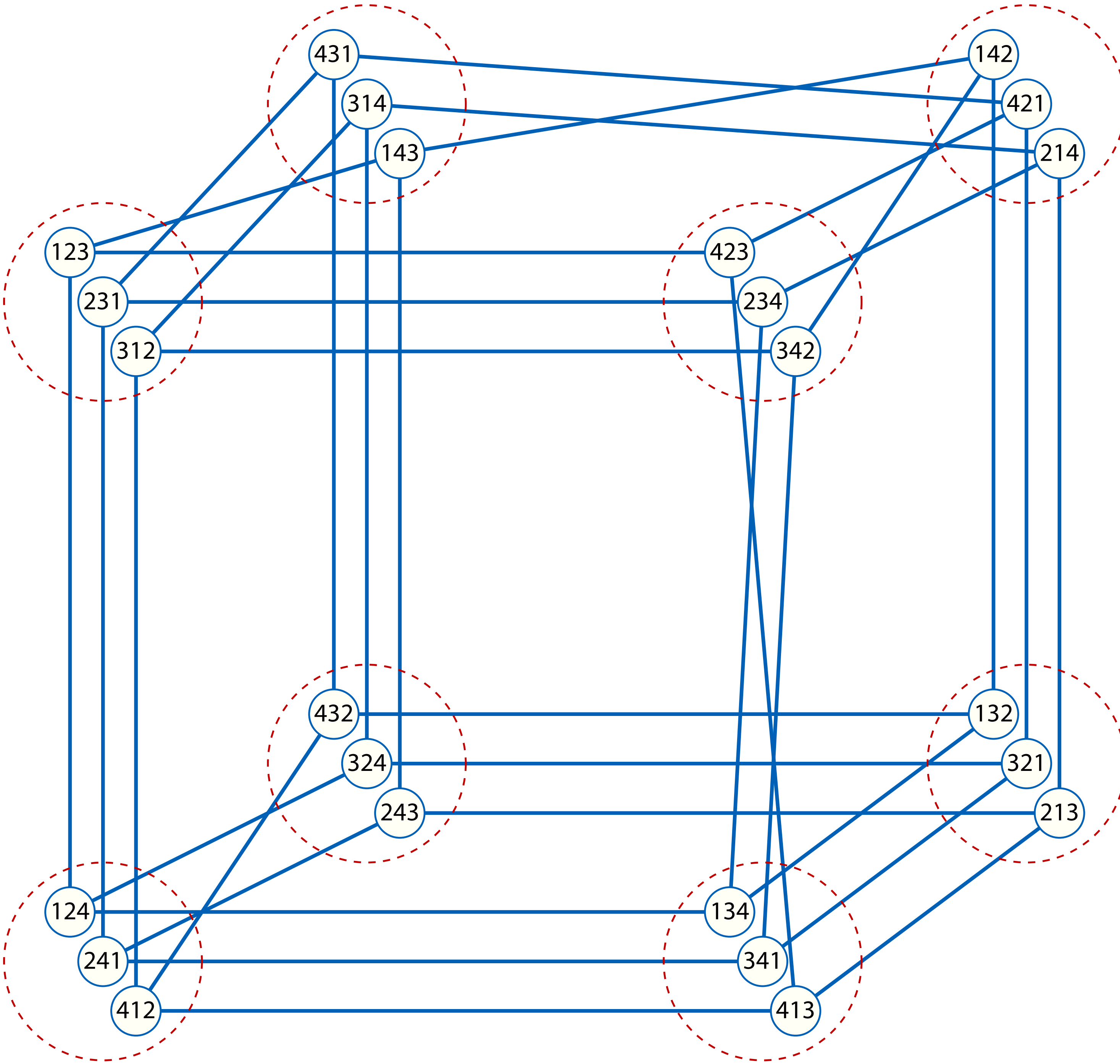


As an infinitely repeated tiling of
the hyperbolic plane



Rough mockup of a 3d embedding
(six vertices per outer cube,
three edges per connecting arm)

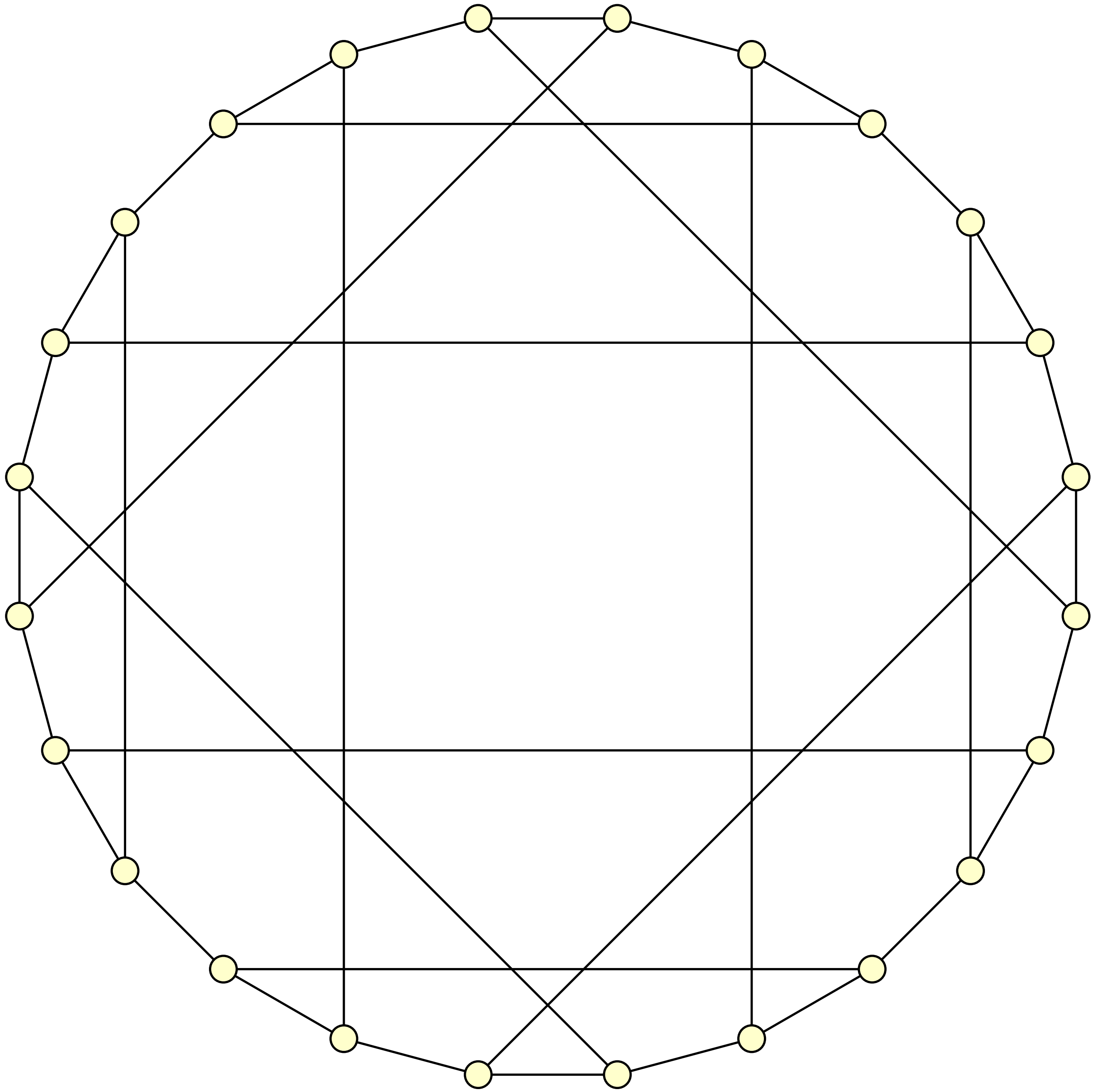
Drawing 8: Group together 3-tuples that are cyclic rotations of each other



Triple cover
of a cube

12-cycles collapse
to cube faces

Drawing 9: Hamiltonian cycle
(attributed by Coxeter to Tutte)

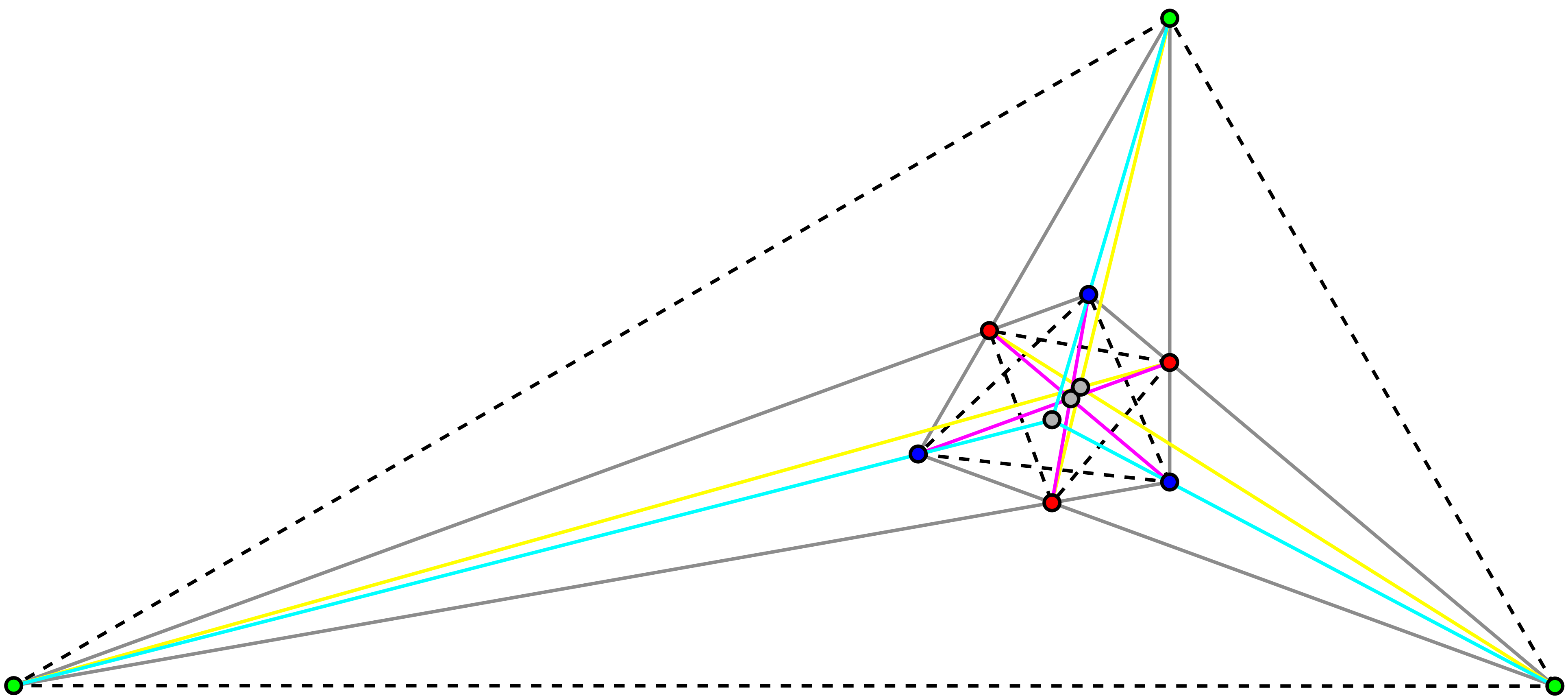


Drawing 10: Projective configuration (Zacharias 1941)

12 points (outer triangle, 2 inner triangles where outer trisectors meet,
3 centers of perspective of pairs of triangles)

12 lines (3 trisectors, 9 lines through corresponding points of two triangles)

Graph with 24 vertices (one per point or line), 36 vertices (point-line incidences)



II. Special families of graphs should be drawn using a drawing style specific to that graph family

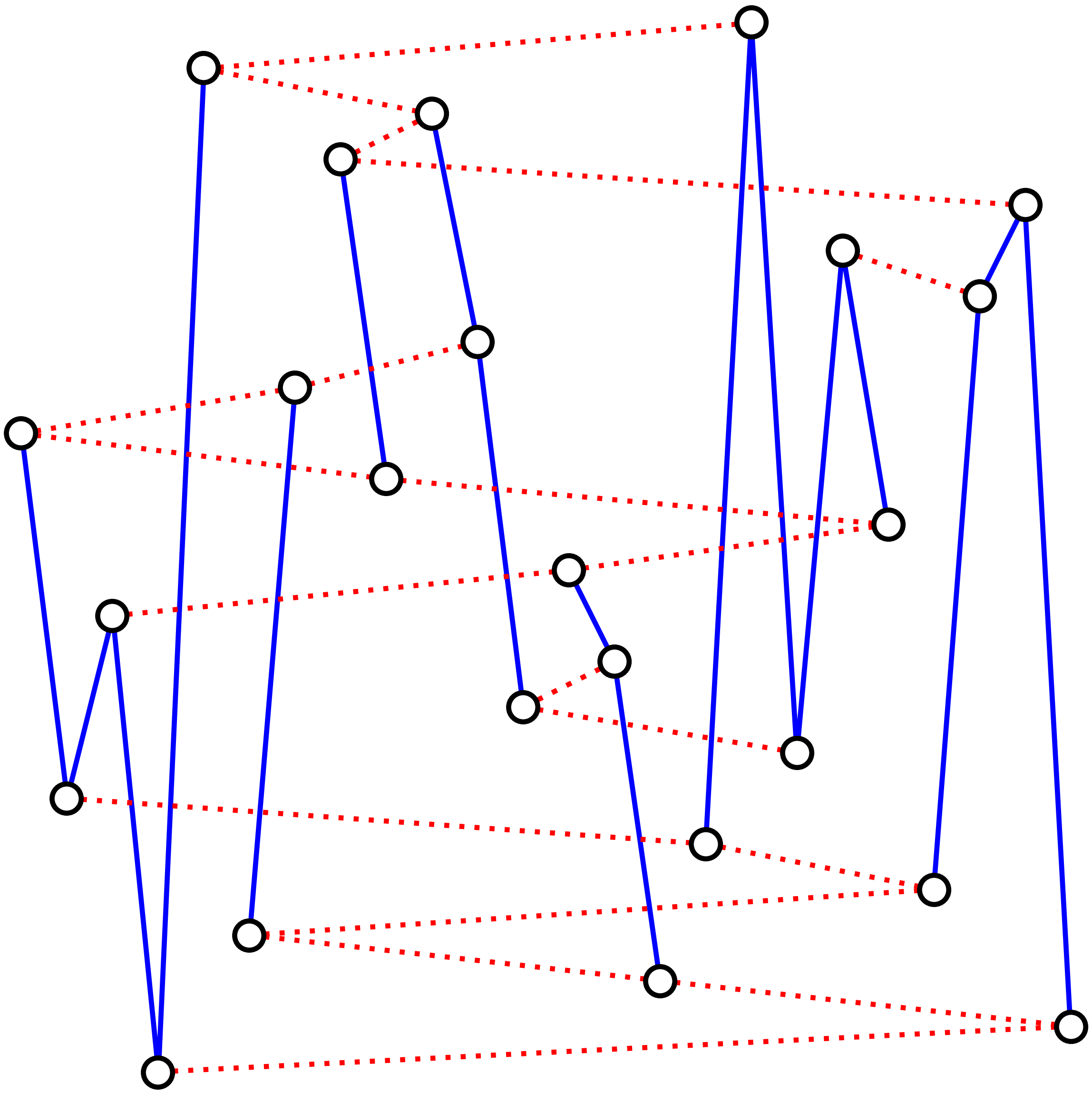
E.g., planar graphs should be drawn without crossings

Trees should be drawn so that they don't separate the plane

Symmetric graphs should be drawn symmetrically

If it's important that a graph has certain properties,
it's important to communicate those properties in a drawing,
and use a drawing style from which they are easily seen to hold

What happens when you use a general-purpose graph drawing technique on a graph with some special structure?



Same graph as before

Uses algorithm for drawing any graph with degree ≤ 4 in two non-crossing layers
[Duncan, E., Kobourov 2004]

Very legible, but looks like any other similar size 3-regular graph

No symmetry or other structure is visible

Example (related to joint work with Jean-Claude Falmagne):
two-dimensional learning spaces

Consider a set S of concepts that a human learner might learn
Some concepts may not be learnable until some prerequisites known

Initially given two different orderings by which all concepts
can eventually be learned (any prefix is a feasible state of knowledge)

(two textbooks, learner might read part of one then stop)

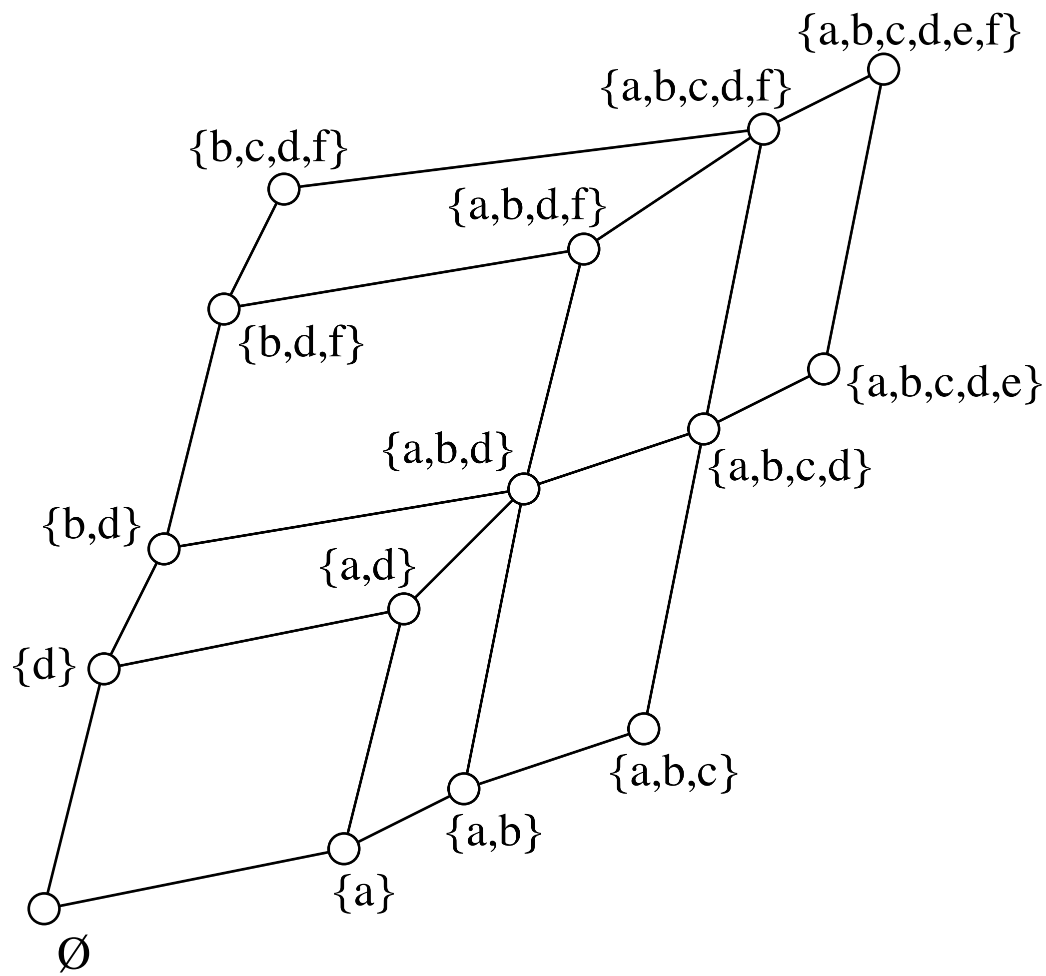
Form other states by unions of prefixes from both sequences

(read one text, stop, switch to other text)

Graph:

vertices = feasible states of knowledge
edges = two states that differ in a single concept

Example (related to joint work with Jean-Claude Falmagne):
two-dimensional learning spaces (continued)

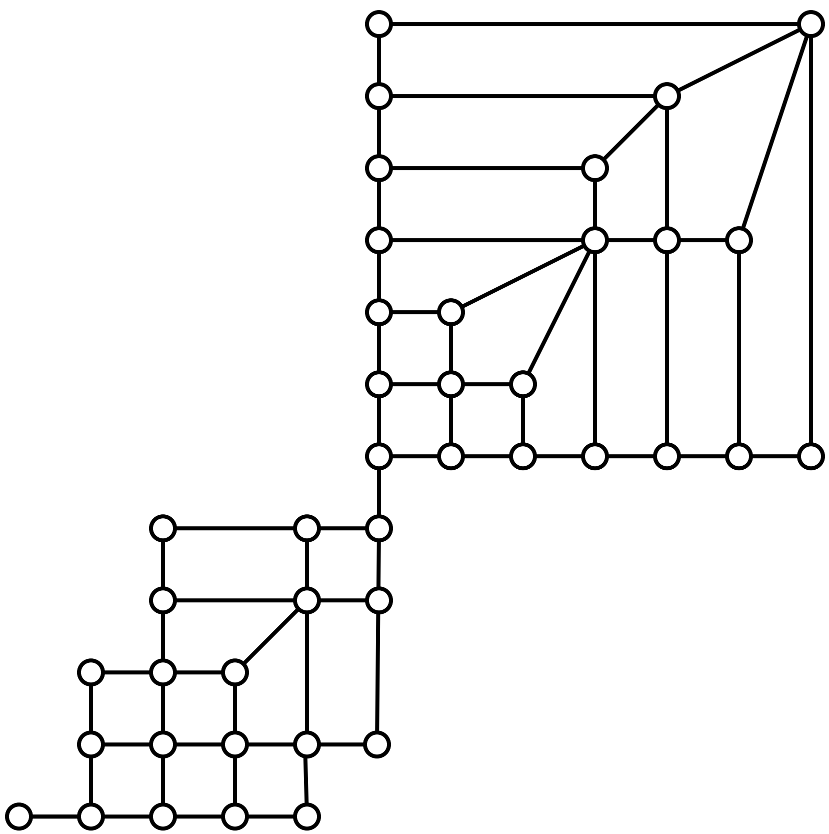


Two initial learning sequences:

a-b-c-d-e-f
d-b-f-c-b-e

All others formed by union of prefixes of the two e.g.
{a,b,d,f} = {a,b} u {d,b,f}

Example (related to joint work with Jean-Claude Falmagne):
two-dimensional learning spaces (continued)



Theorem:

G is the graph of a learning space formed from two learning sequences

if and only if

G has a planar drawing in which

- all faces are convex quadrilaterals
- each quad has axis-aligned bottom and left sides
- the drawing has a single top right vertex and a single bottom left vertex

[E, Graph Drawing 2006; *J. Graph Algorithms & Applications*, to appear]

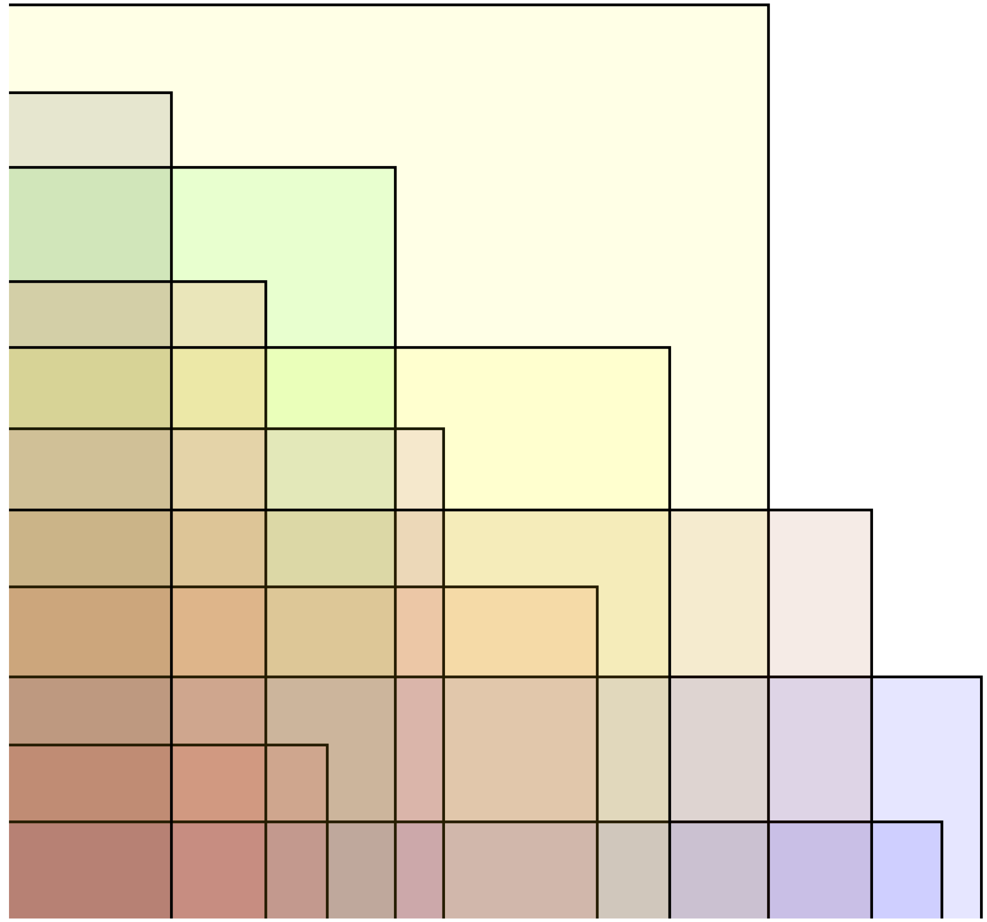
Proof sketch:

Show drawing of this type must correspond to a learning space via *arrangements of quadrants*:

Translate copies of the negative quadrant

No two translates share a boundary line

Adjacency of regions = graph of upright-quad drawing = two-sequence learning space



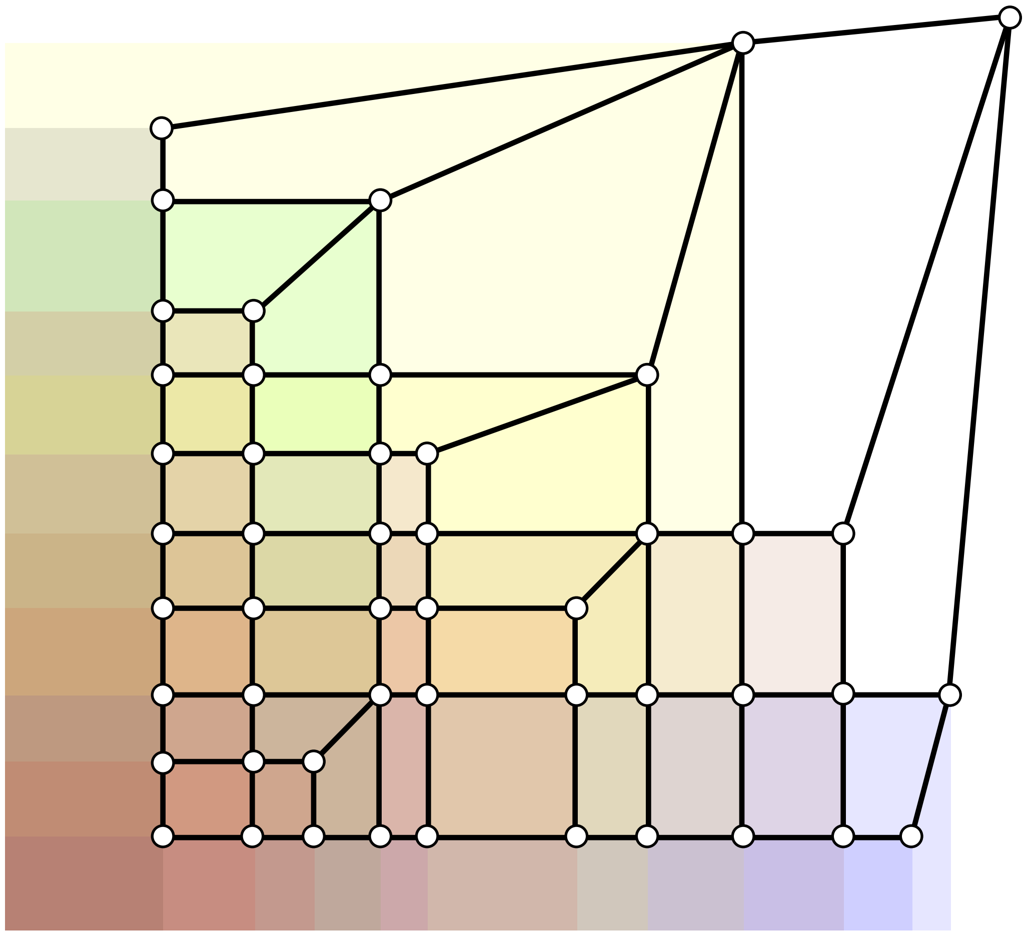
From arrangements to upright-quad drawings

Represent each region by its maximal point

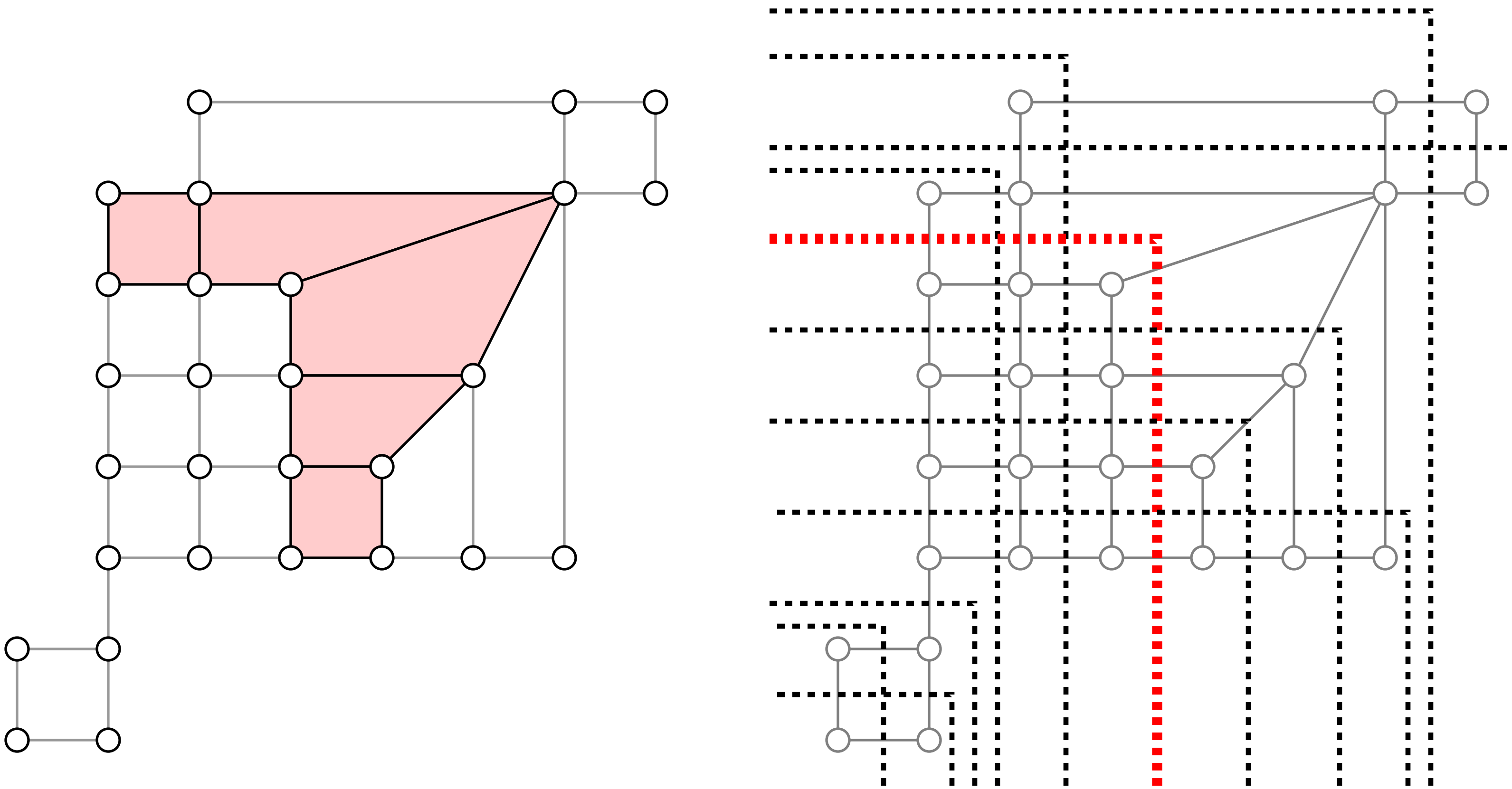
(for the unbounded region, choose a point dominating all other chosen points)

Connect points representing adjacent regions

Result has all faces convex quadrilaterals, bottom & left axis-aligned



From upright-quad drawings to arrangements



Define **zone** = set of cells sharing bottom-top, left-right, or top-right edges

Each face belongs to two zones

Cover by quadrant containing all lower left vertices (w/tie break rules)

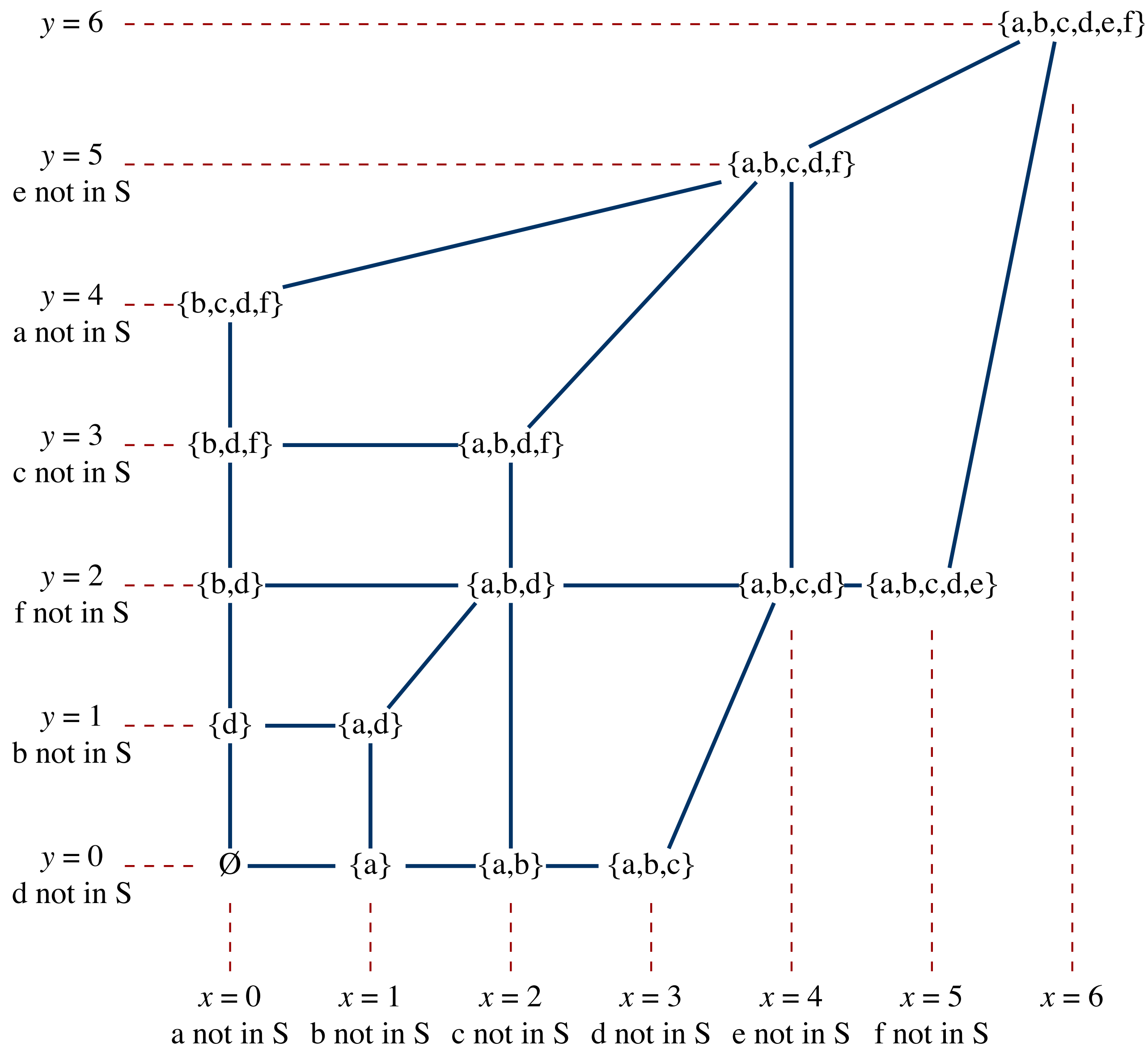
From two-sequence learning spaces to upright-quad drawings

x-coordinate of set S = position of first unknown concept in first sequence

y-coordinate of set S = position of first unknown concept in second sequence

The states from the two sequences form the lower left and upper right boundaries of the drawing

(Plus additional optimizations to allow states to share coordinate values creating a more compact drawing)



From arrangements to two-sequence learning spaces

Form one concept per quadrant

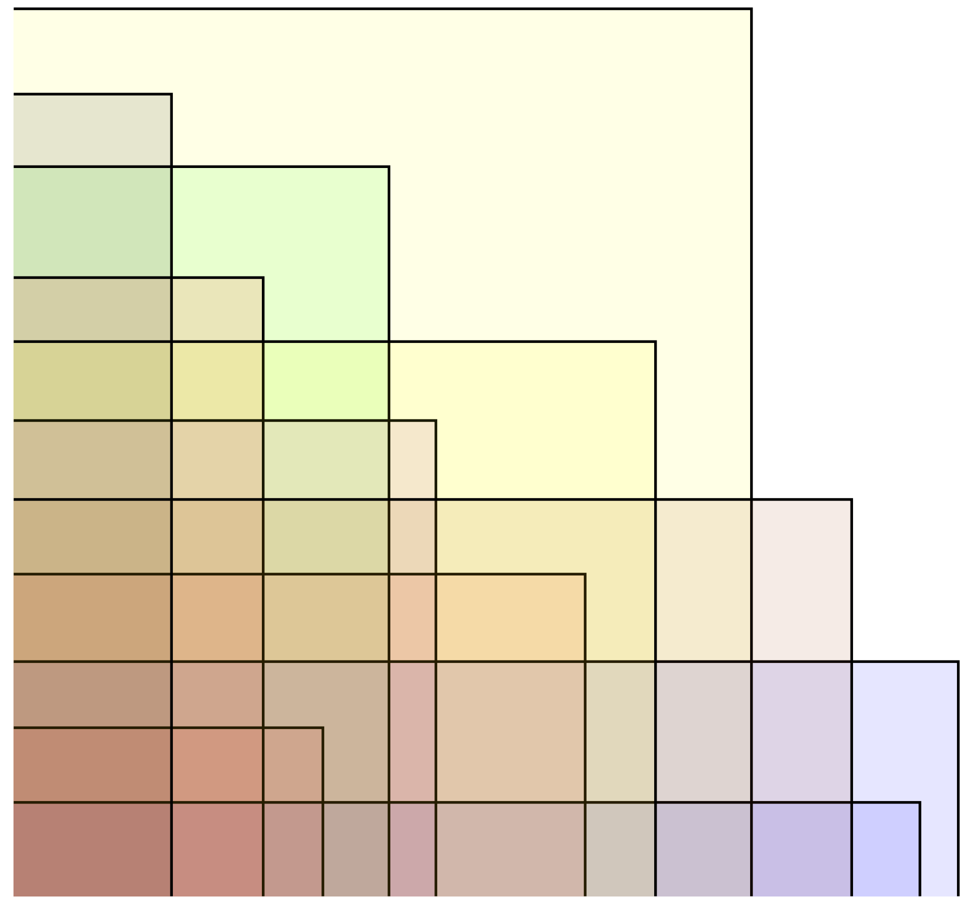
First sequence =
order of quadrants along bottom
of arrangement drawing

Second sequence =
order of quadrants along left
of arrangement drawing

Region corresponds to the set
of quadrants below or to the right of it

Quadrants to right = prefix of first seq
Quadrants below = prefix of second seq

So regions correspond with all possible
unions of prefixes of the sequences



III. Free parameters of a drawing style should be identified and optimized for legibility and aesthetics

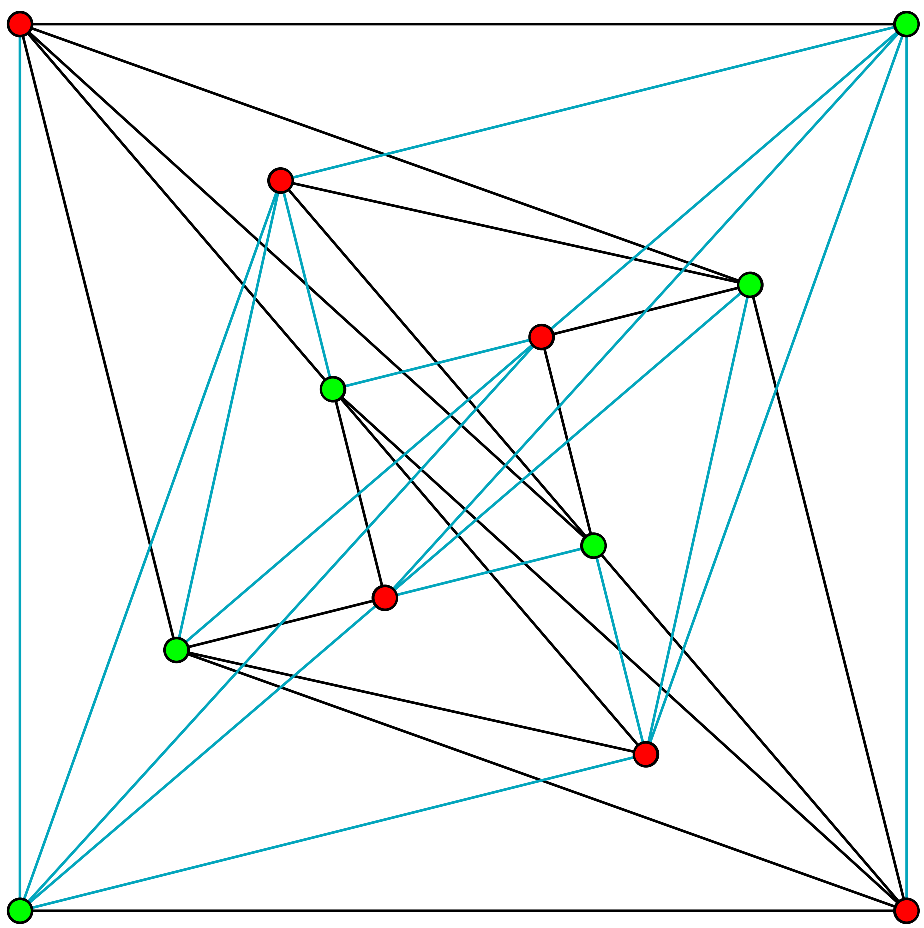
Typical parameters include:

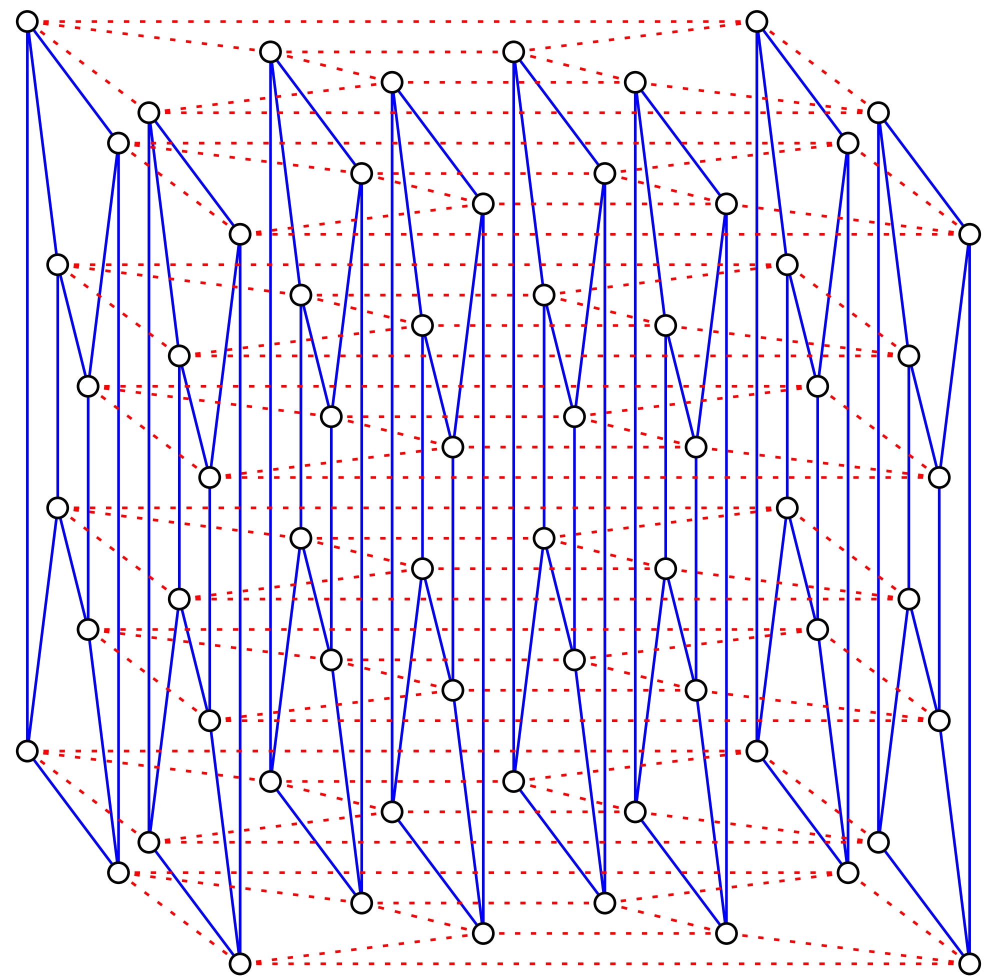
vertex position
vertex size and shape
bends (if allowed) or curves in the edges
vertex color
edge color
over/undercrossing choice
label placement, font, size

Using edge color to avoid crossings

"Geometric thickness": min number of colors needed
in a straight-line drawing such that each color forms a planar subgraph

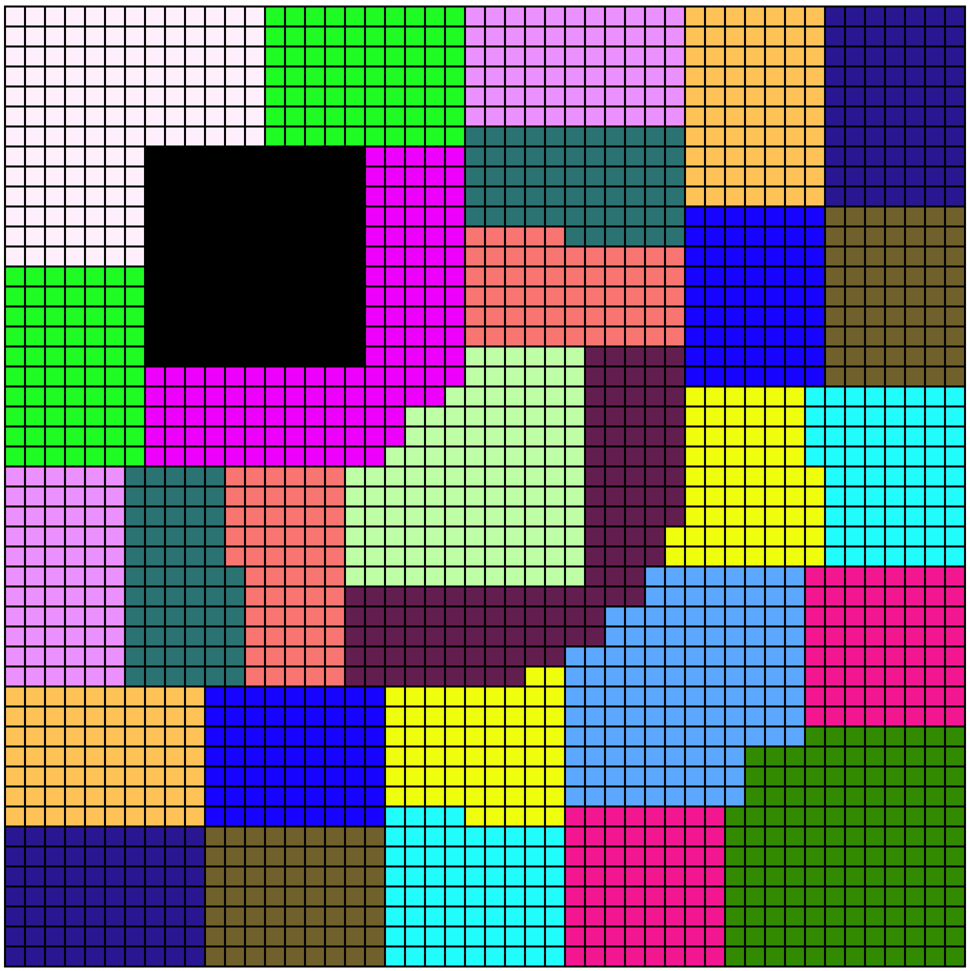
[Dillencourt, E., Hirschberg, Graph Drawing 1998 & JGAA 2000; various other papers]





Using vertex color to enhance contrast between adjacent regions

[Dillencourt, E., Goodrich, Graph Drawing 2006]



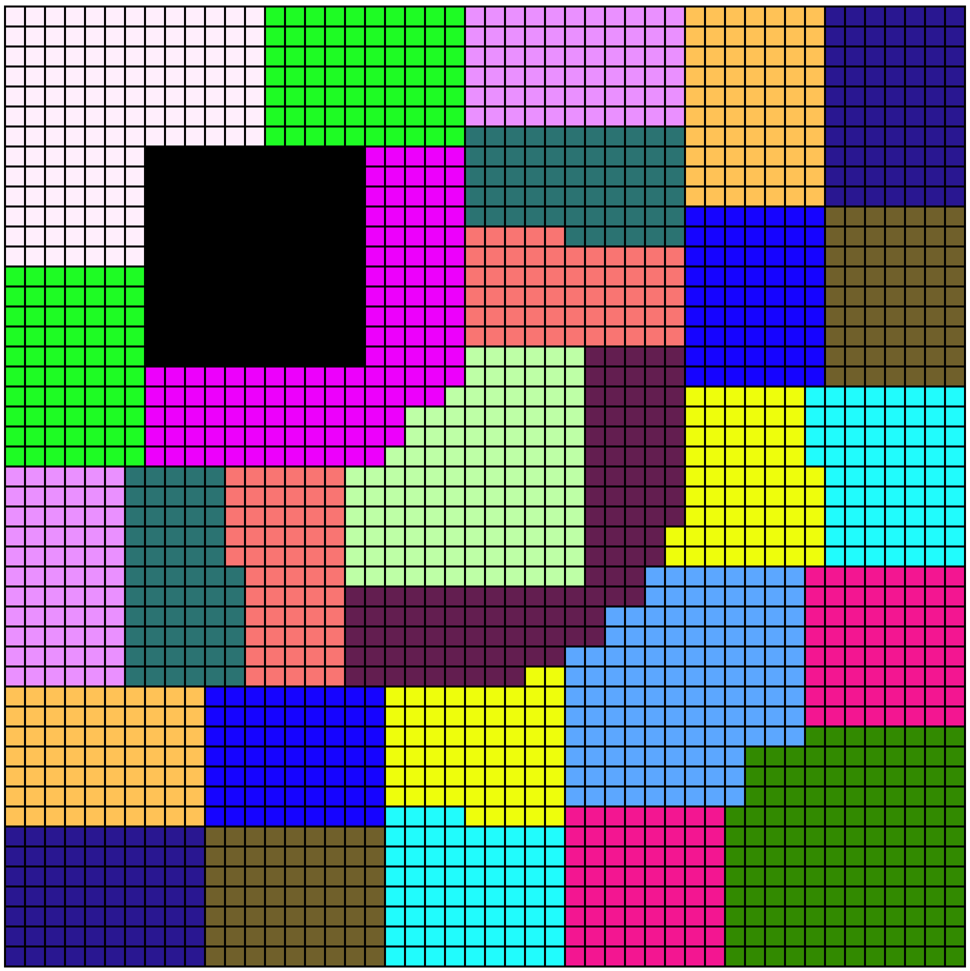
visualization of partitioned data matrix
for a distributed computing application

Goals:

- Show (possibly disconnected) regions of partition
- All regions should have distinctive colors
- Adjacent regions should be highly contrasting

Using vertex color to enhance contrast between adjacent regions

[Dillencourt, E., Goodrich, Graph Drawing 2006]



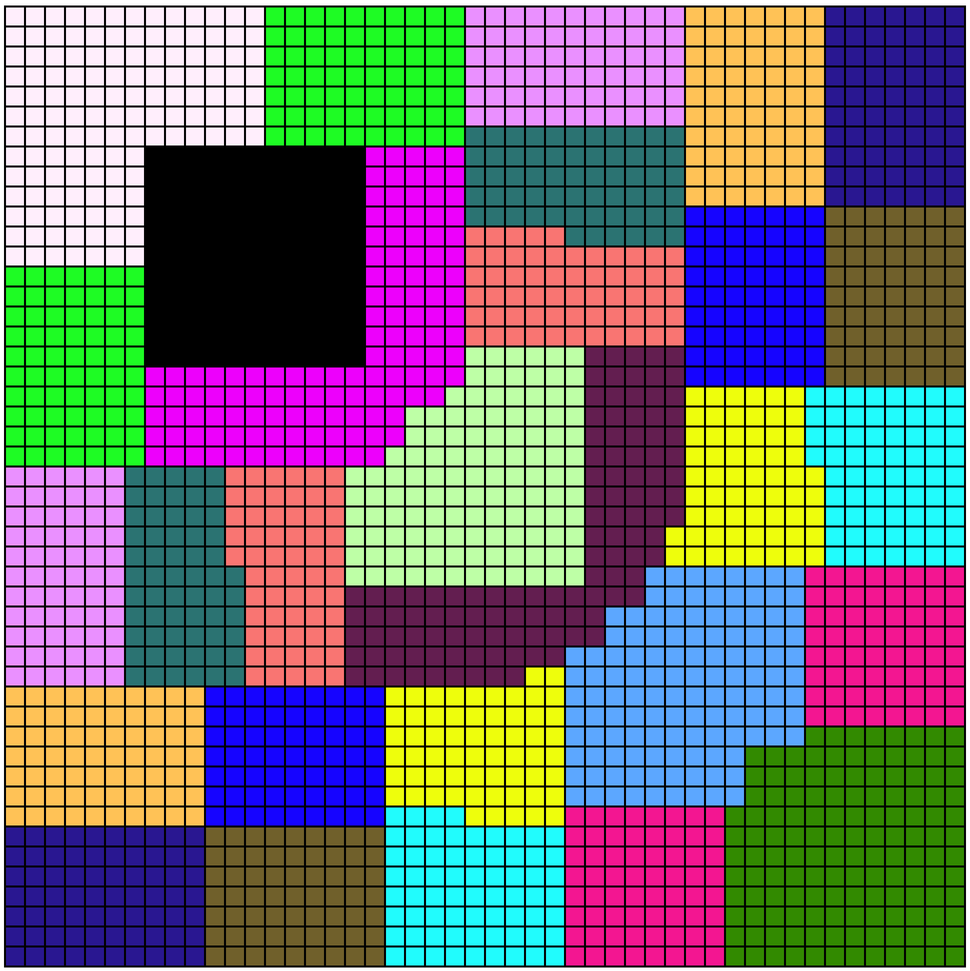
visualization of partitioned data matrix
for a distributed computing application

Method:

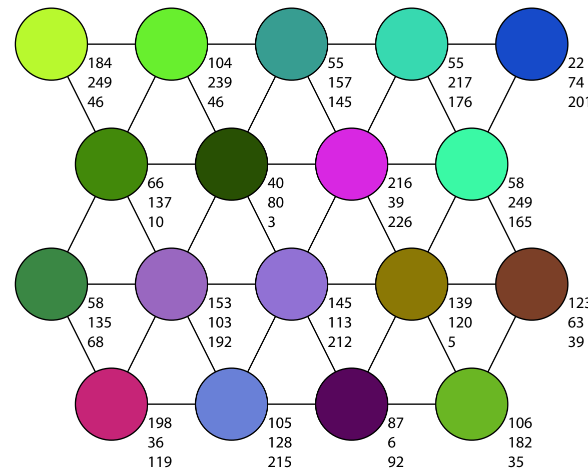
- Form a graph representing regions and adjacencies
- Place vertices in 3d color space (LAB used so that geometric distance correlates to human visual distinctiveness)
- Simulate spring system to spread vertices from each other
- Use stronger spring force for adjacent vertex pairs

Using vertex color to enhance contrast between adjacent regions

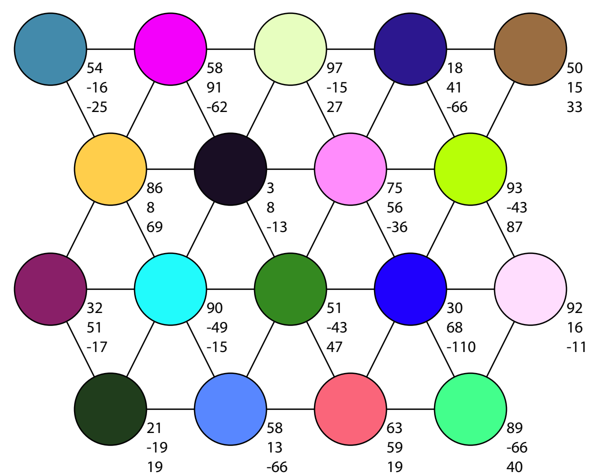
[Dillencourt, E., Goodrich, Graph Drawing 2006]



visualization of partitioned data matrix
for a distributed computing application



Random sRGB colors



Optimized LAB colors

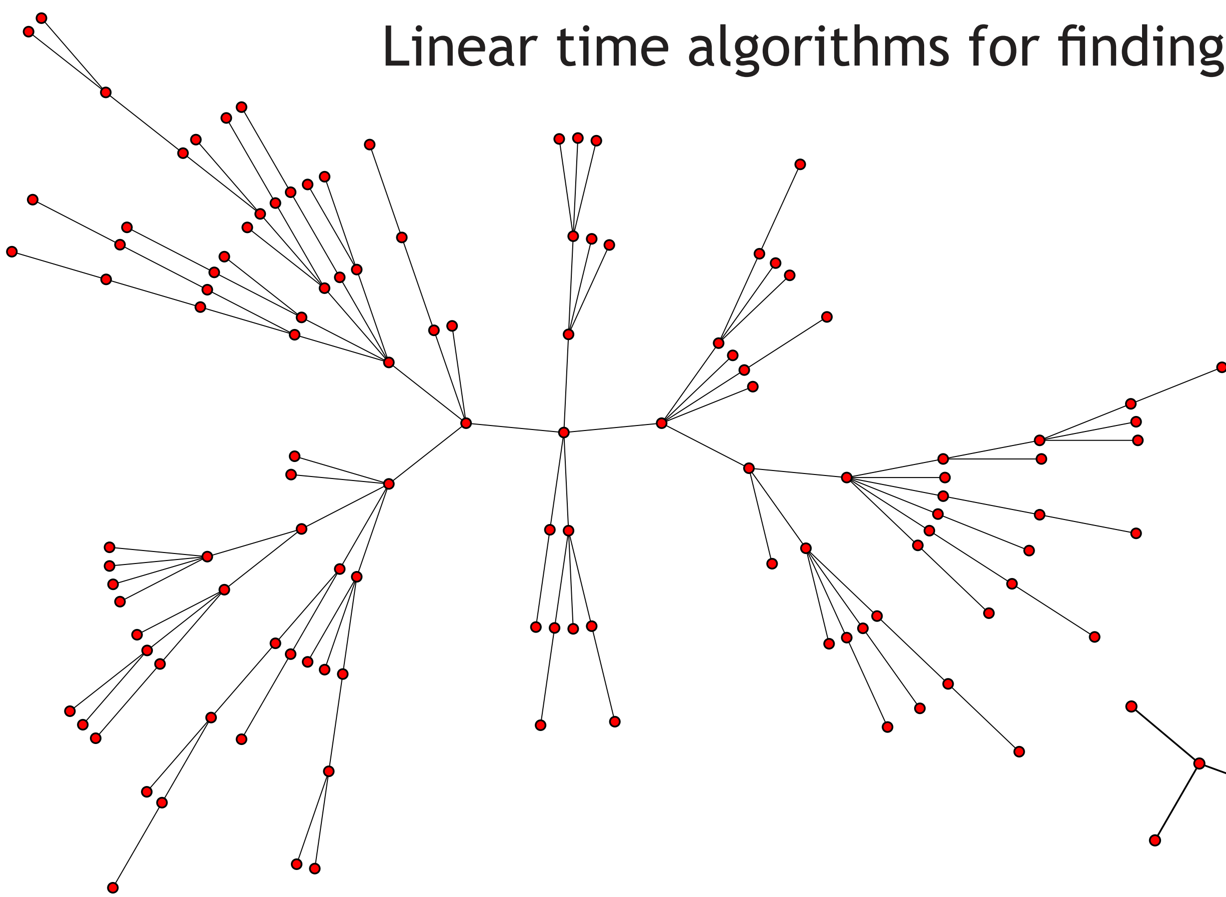
Optimal angular resolution for tree drawings

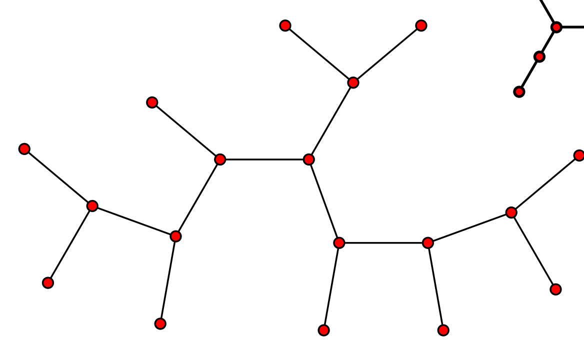
[Carlson, E., Graph Drawing 2006]

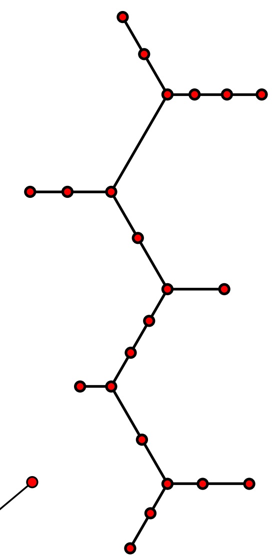
"Angular resolution": sharpest angle between adjacent edges in a drawing

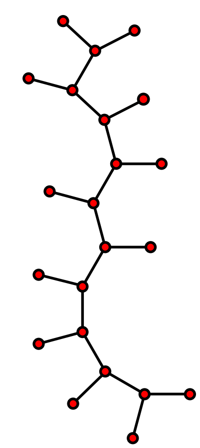
Tree drawings in which any leaf-leaf chain spans ≤ 180 degrees
(so extending the leaf edges to infinity partitions the plane into convex cells)

Linear time algorithms for finding optimal drawings





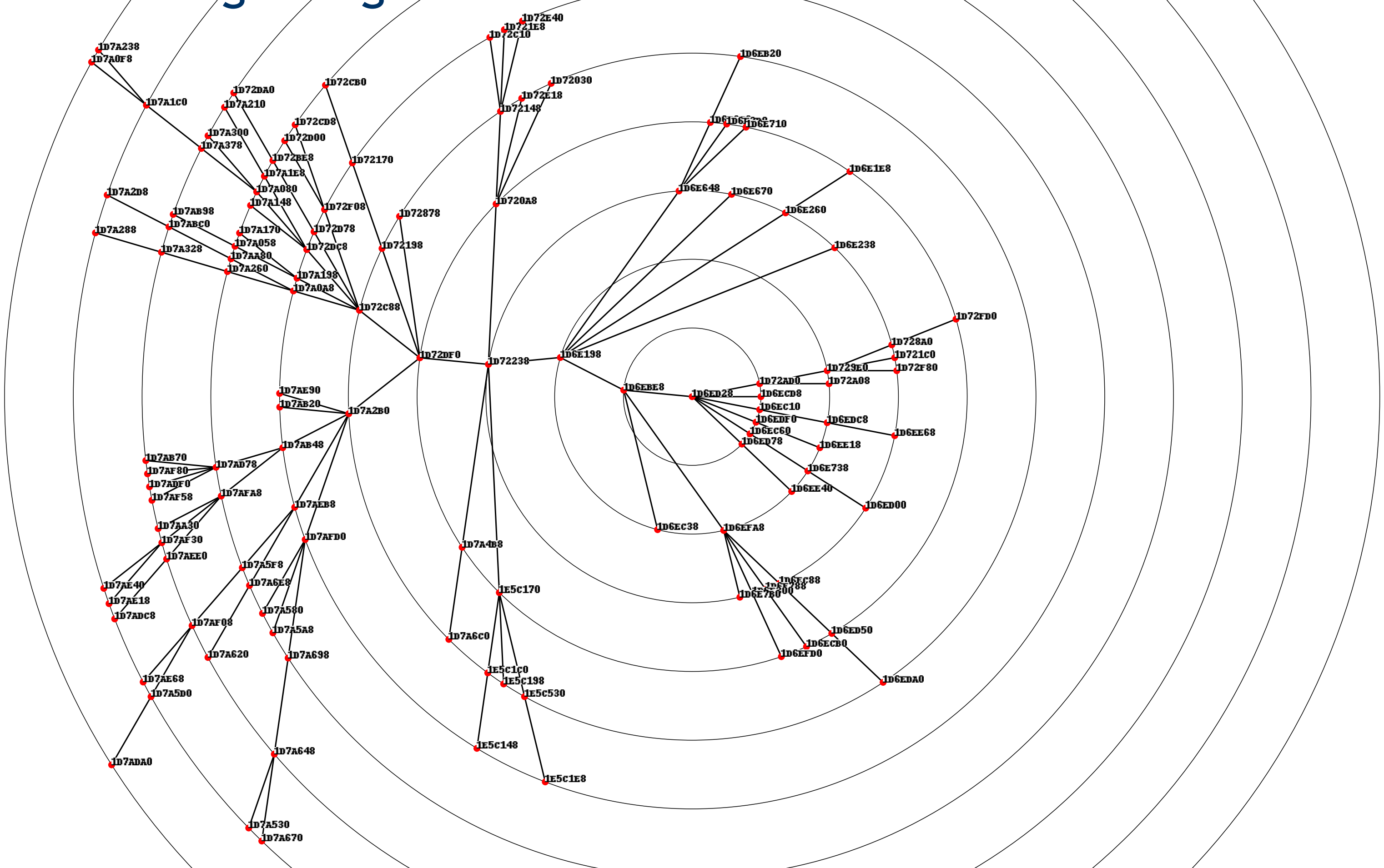




IV. When free parameters can be used to convey useful information, do so

For tree drawing with optimal angular resolution,
any choice of edge length leads to optimal drawing

E.g., choose edge length to show distance from center:



Conclusions

A picture is worth 1000 words:

Graph drawing is an important mode of communication
for the graphs arising in your work (social networks, learning spaces, etc)

Application specific knowledge leads to better graph drawings:

To make good drawings, you need to understand what structure in the graph
you want to convey and what kind of drawing will best convey it

To do my graph drawing research well, I need to understand
what types of graphs come up in your applications,
what you want to communicate about them,
and what kinds of drawings you have difficulty creating