Druid:

Representation of Interwoven Surfaces in 2½D Drawing

Dr. Keith Wiley

Dr. Lance R. Williams

This work completed while at:
University of New Mexico
Department of Computer Science
Albuquerque, NM 87131 USA

Current location:
Applied Physics Laboratory
University of Washington
Seattle, WA 98105 USA

Druid:

Toward a New Dimension In Vector Drawing

Dr. Keith Wiley

Dr. Lance R. Williams

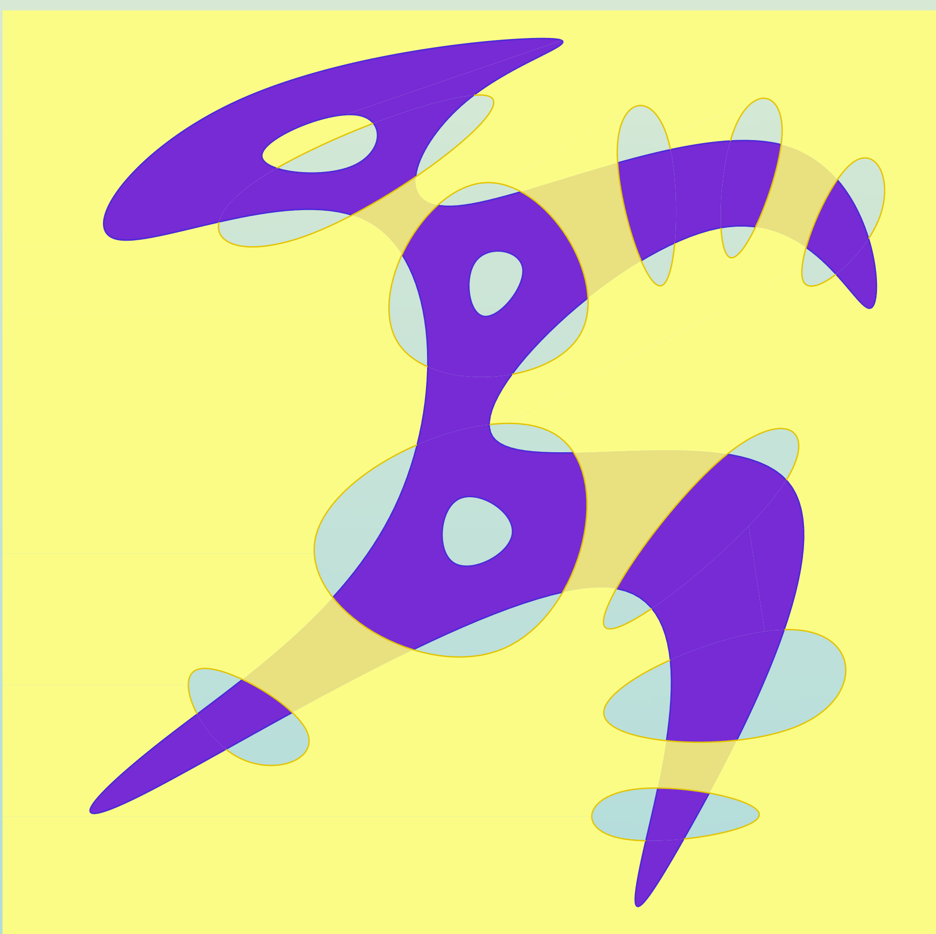
This work completed while at:
University of New Mexico
Department of Computer Science
Albuquerque, NM 87131 USA

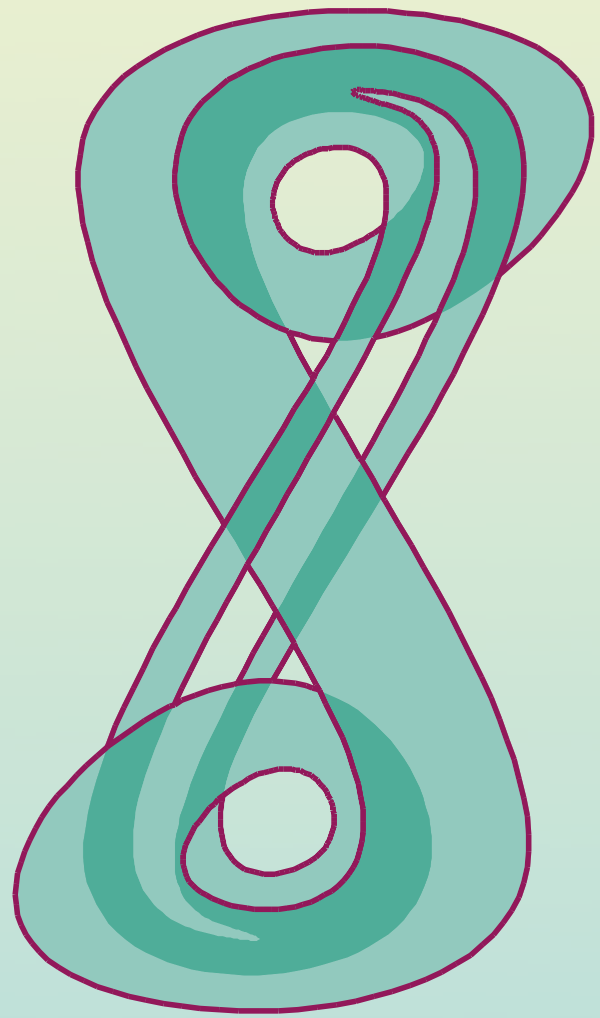
Current location:
Applied Physics Laboratory
University of Washington
Seattle, WA 98105 USA

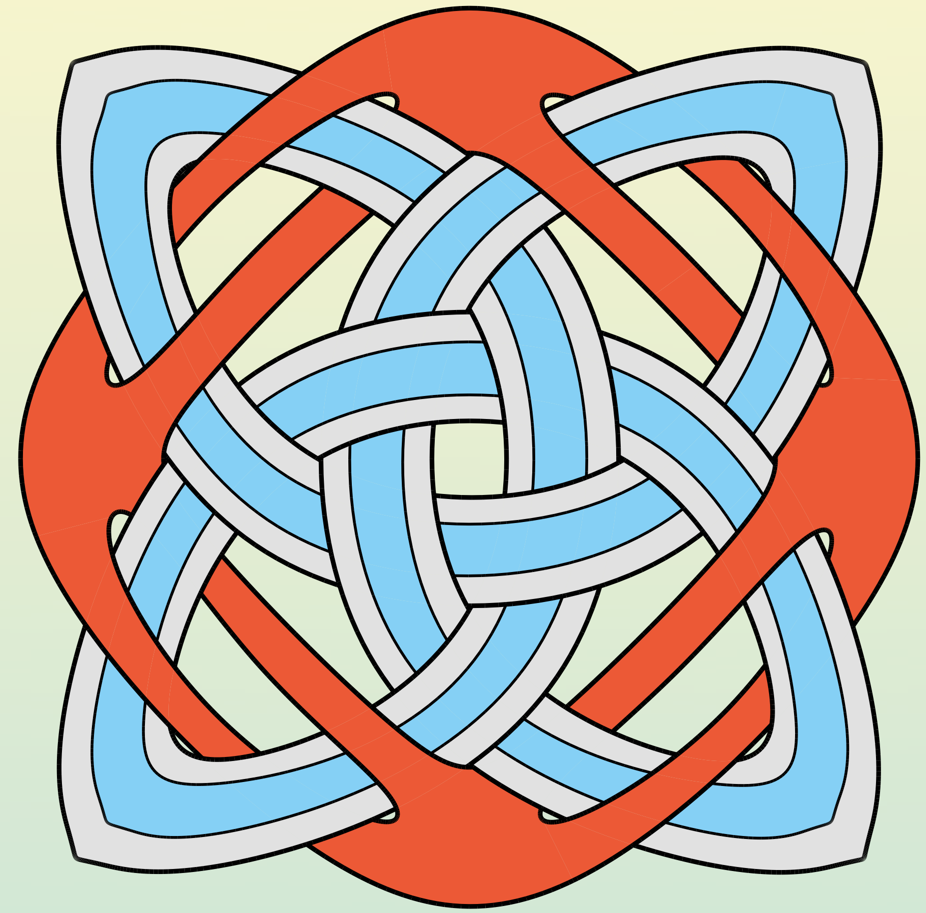
Talk Overview

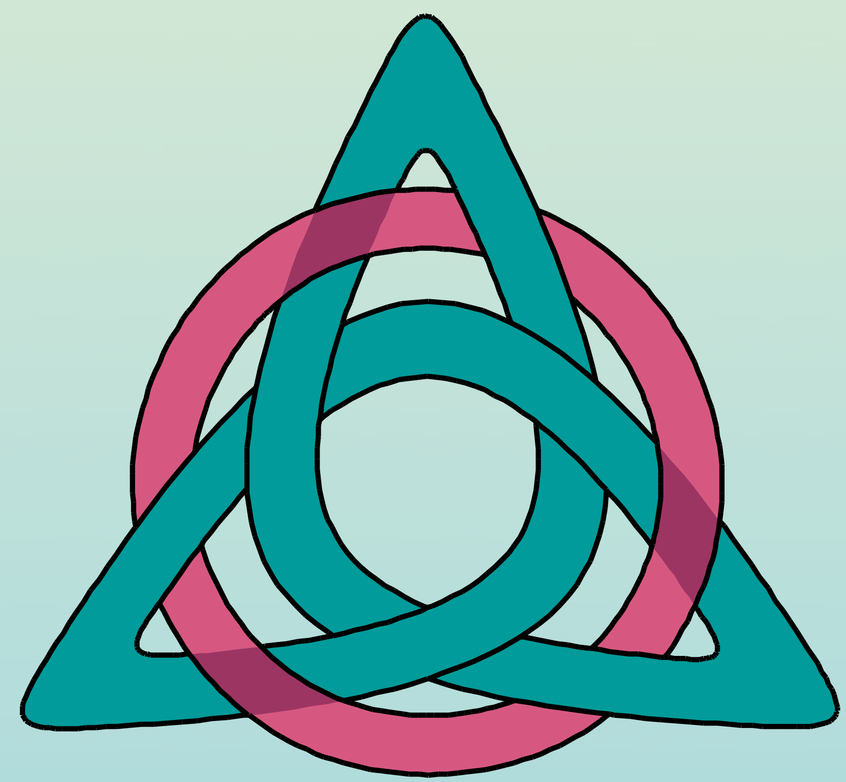
- **Introduction, Current State-of-the-Art**
- Druid Description, Usage
- Finding Legal Labelings
- Crossing-State Equivalence Classes
- Conclusions

Interwoven 2½D Scenes







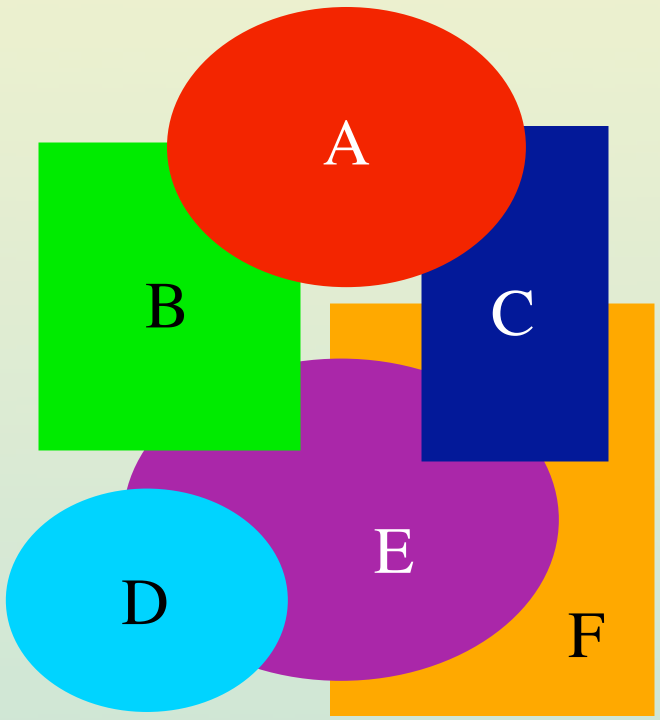


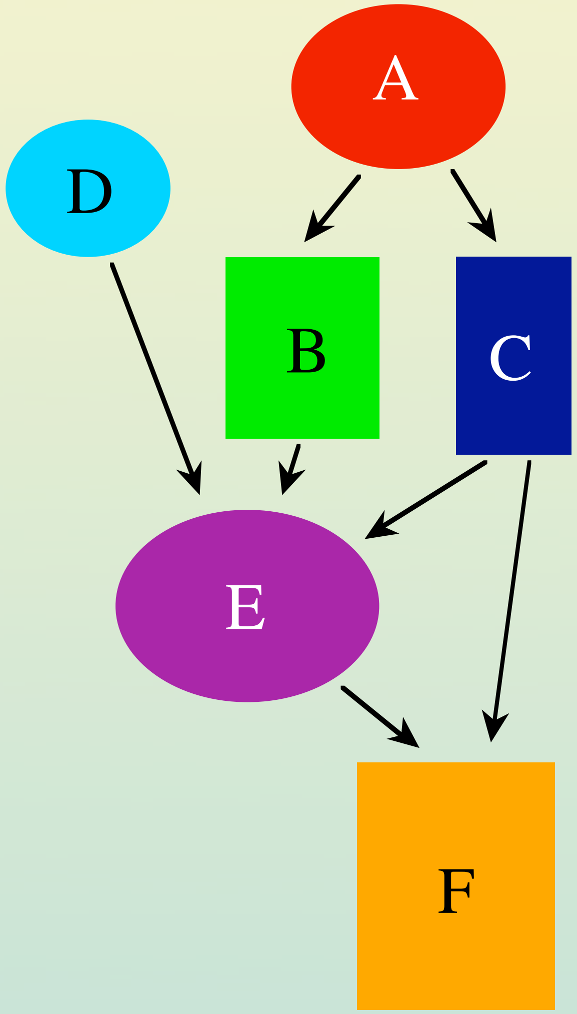
4

Introduction

Existing drawing programs:

- Use distinct layers
- Impose a DAG
- Do not permit interwoven surfaces





Our program, ***Druid***, does not suffer from these limitations.

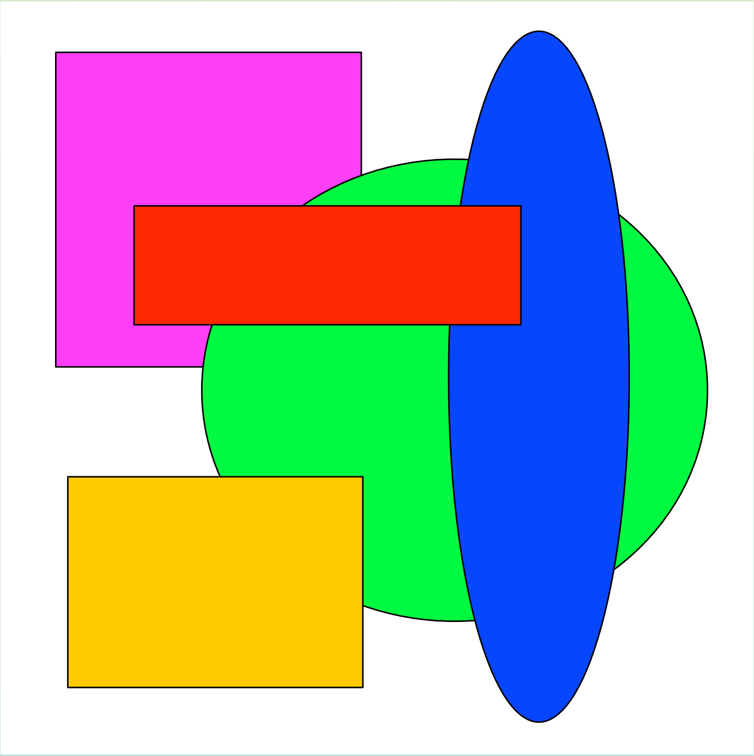
5

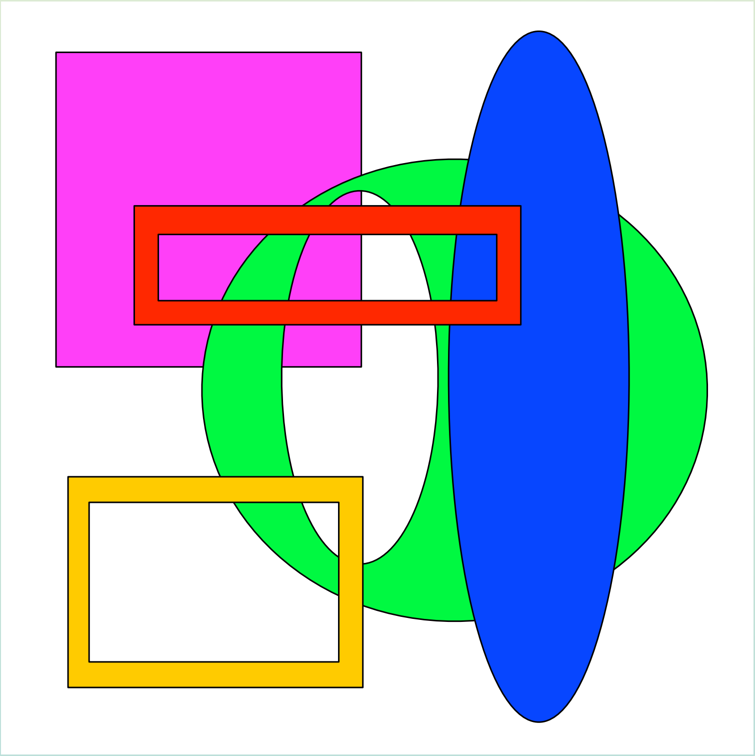
Existing Drawing Programs

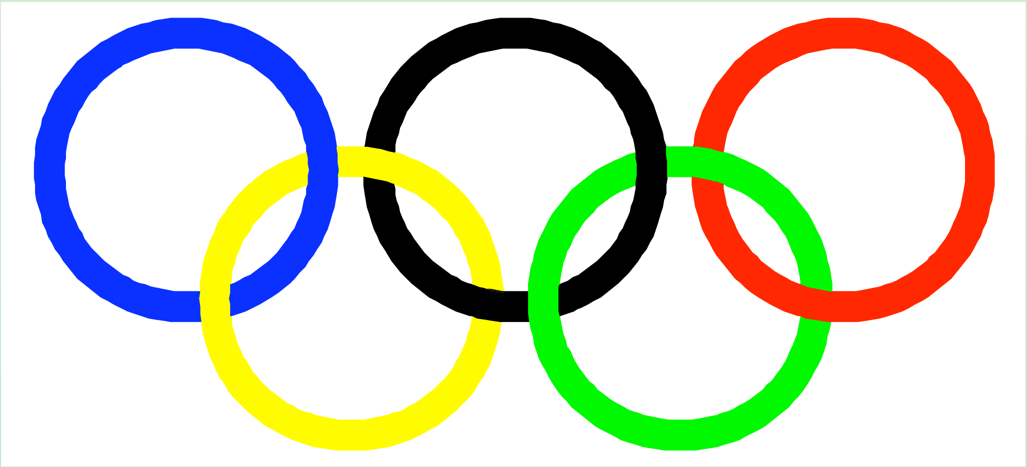
Noninterwoven layers

Boolean combinations of boundaries, *i.e.*, holes.

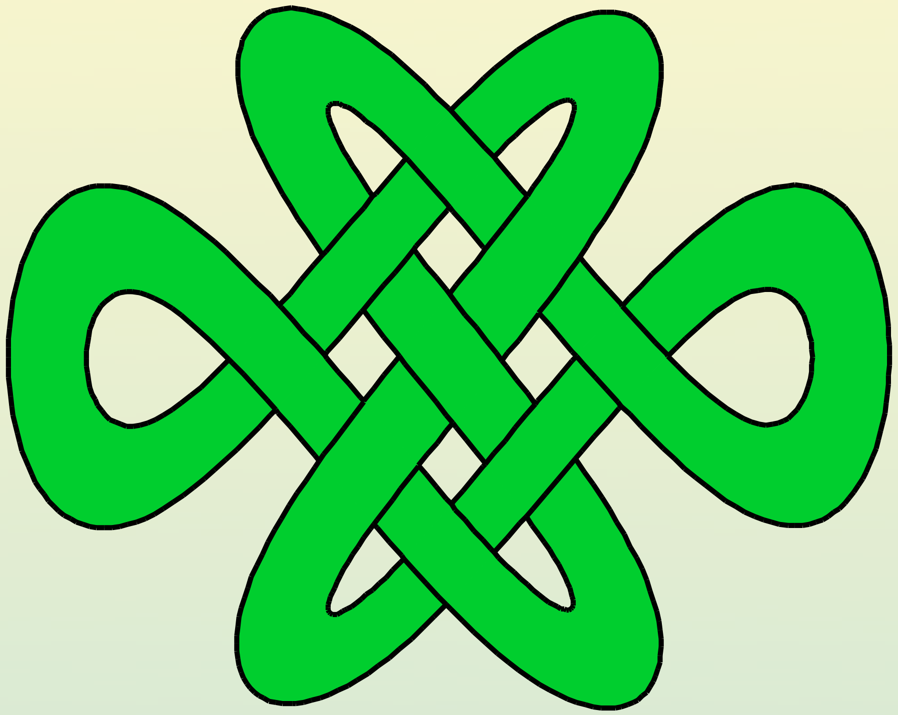
Do not span the full space of ***2½D scenes***.

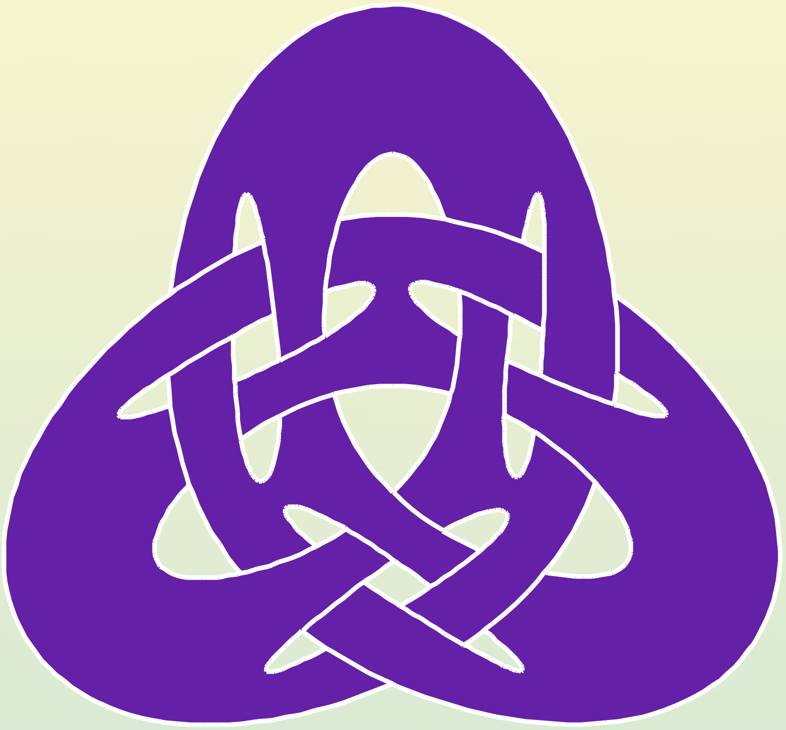


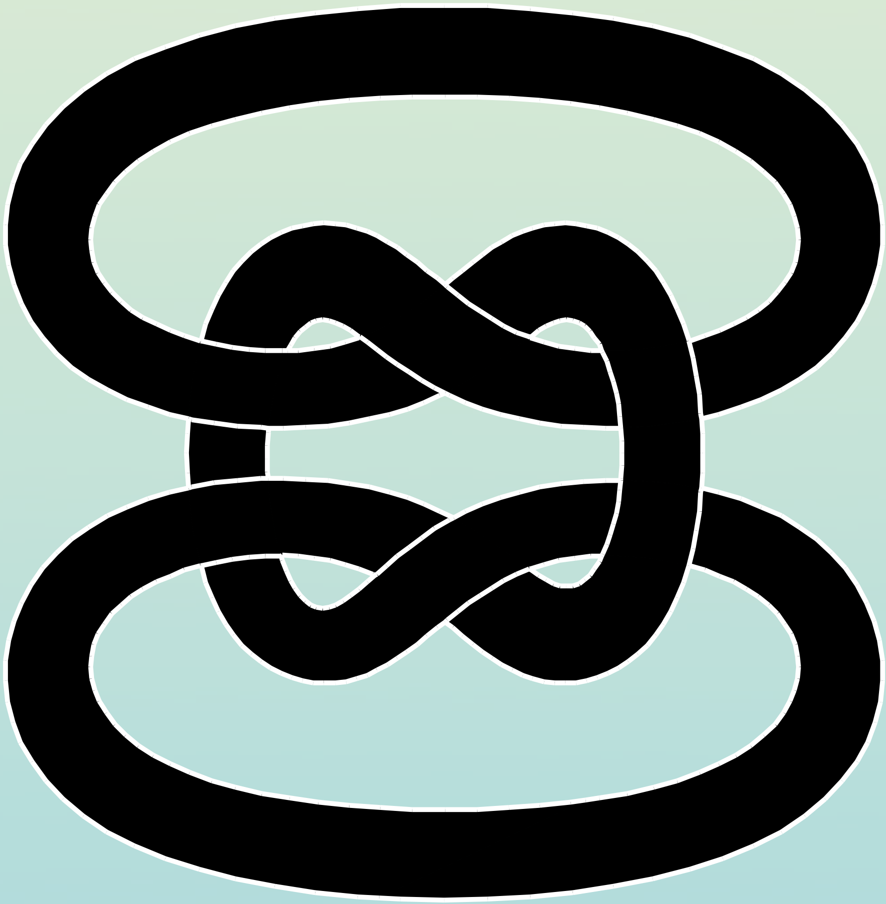




Knots vs. Interwoven Surfaces







7

Interwoven Surfaces in Conventional Drawing Programs

1. Spoofs

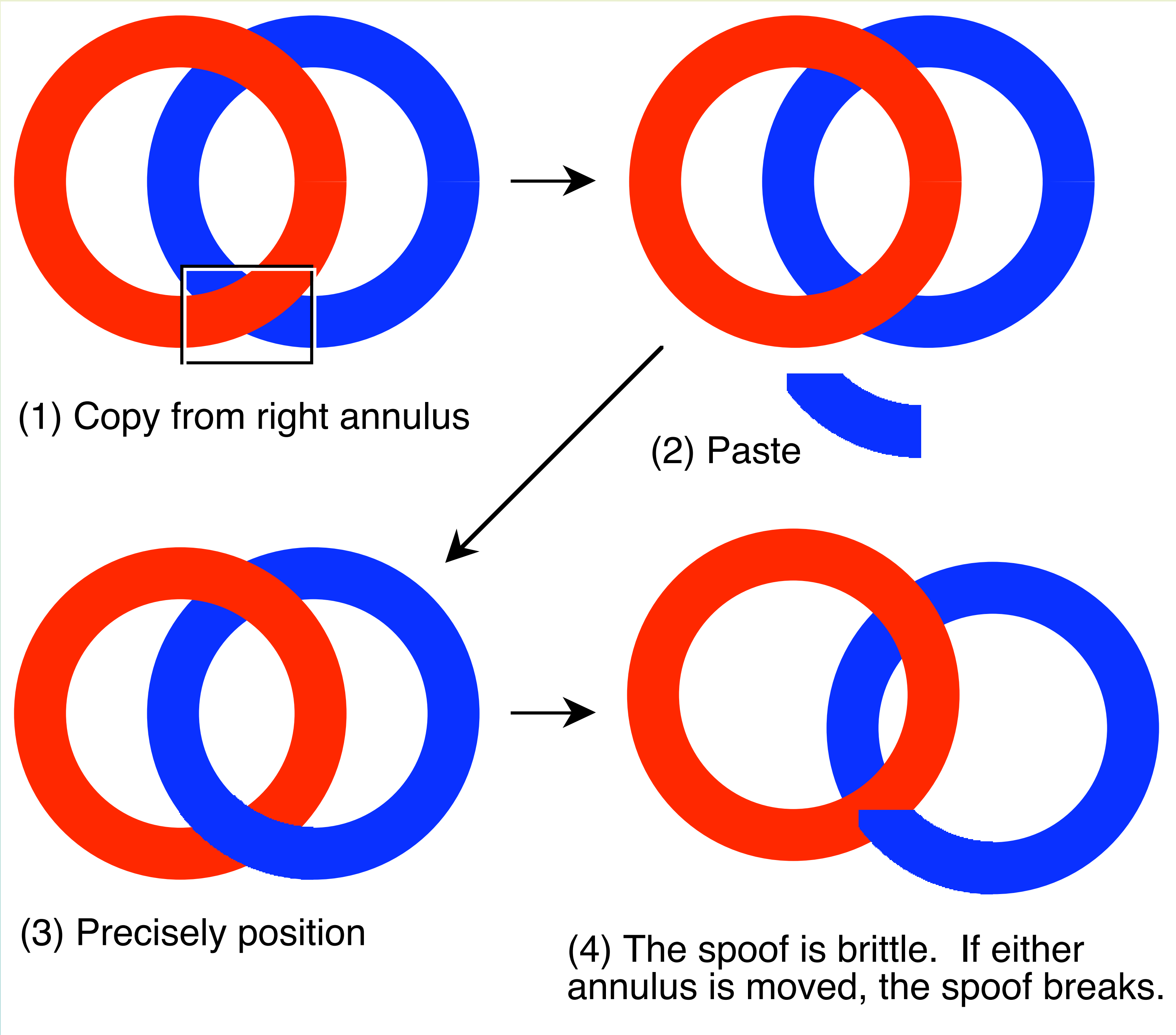
2. Painting planarized graphs,
   e.g., *Adobe Illustrator*

3. Local DAG manipulation,
   e.g., *MediaChance Real-Draw*

8

Spoofs

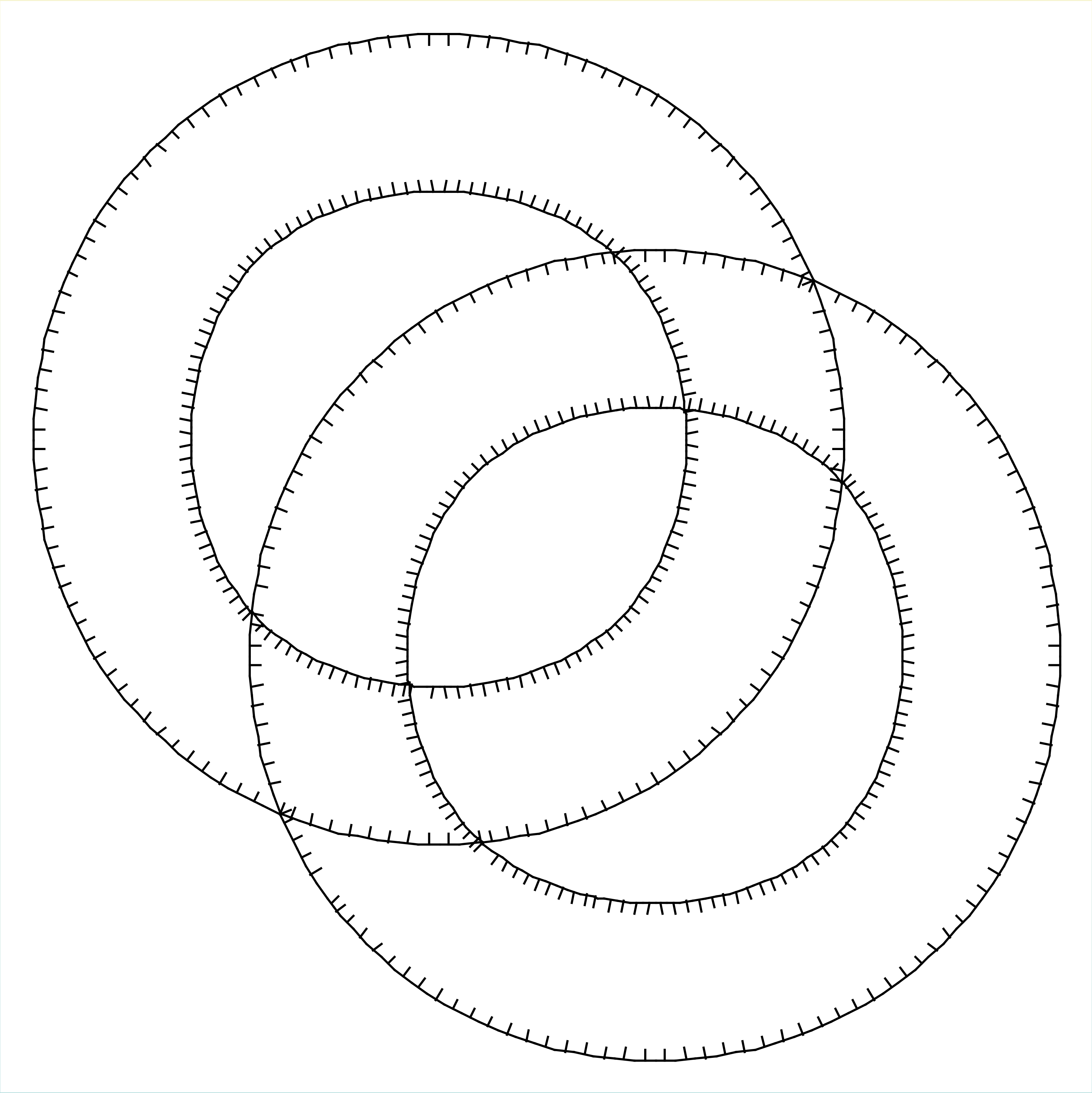
A layered arrangement that produces the illusion of interwoven surfaces



- Tedious to construct
- Tedious to maintain

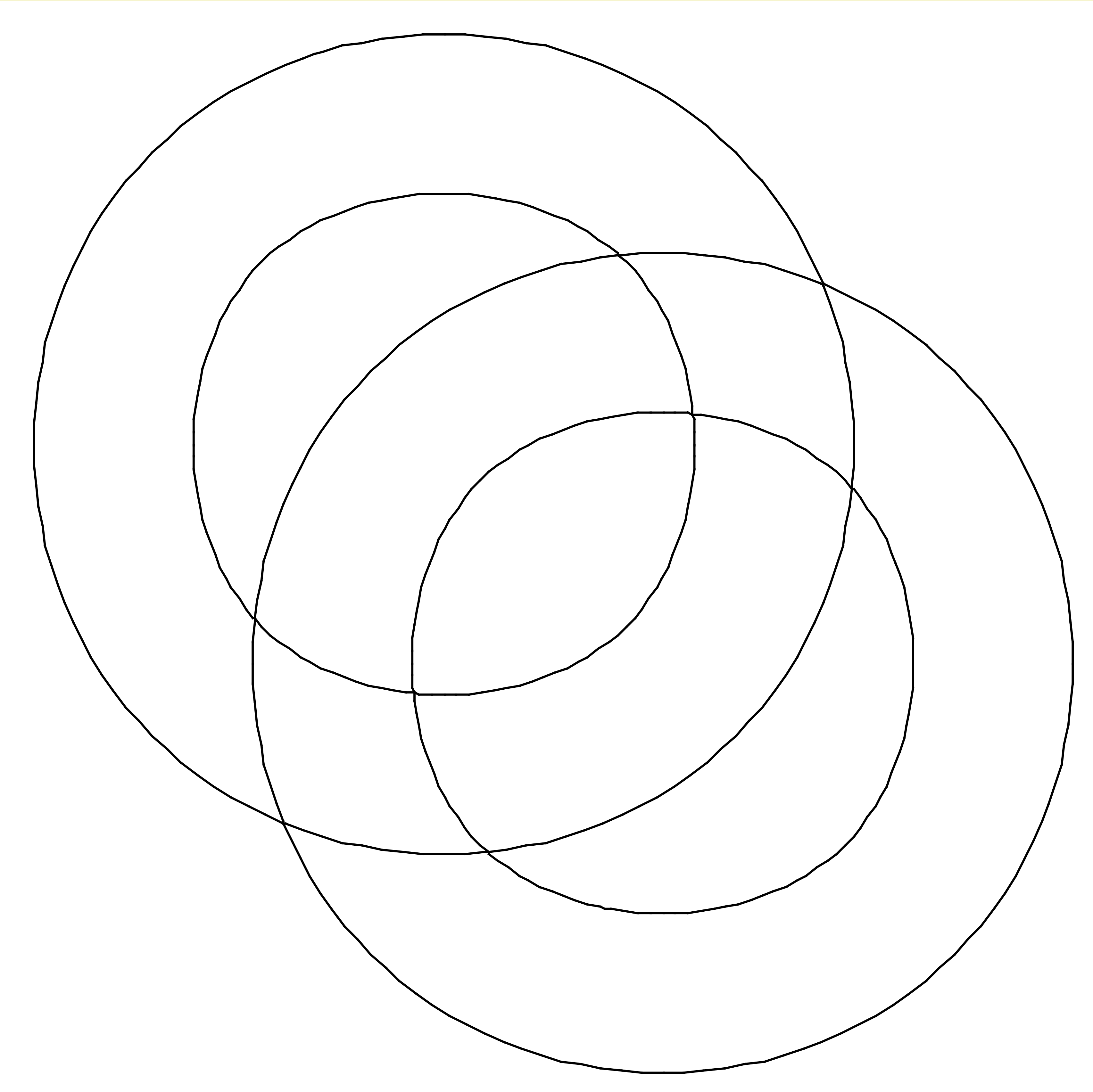
9

Adobe Illustrator Method



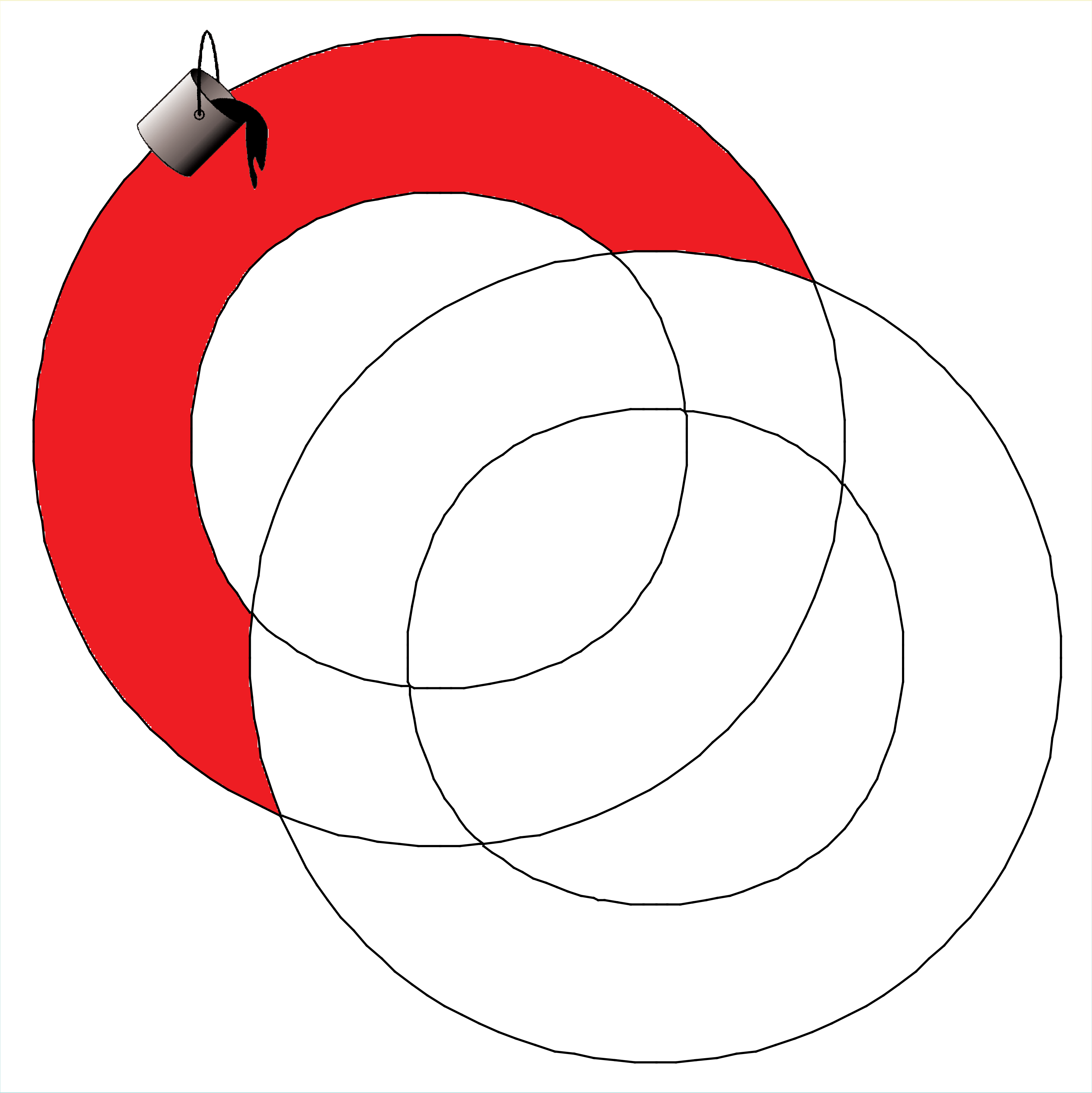
- Convert drawing to planar graph
- Paint faces of the graph independently

Adobe Illustrator Method



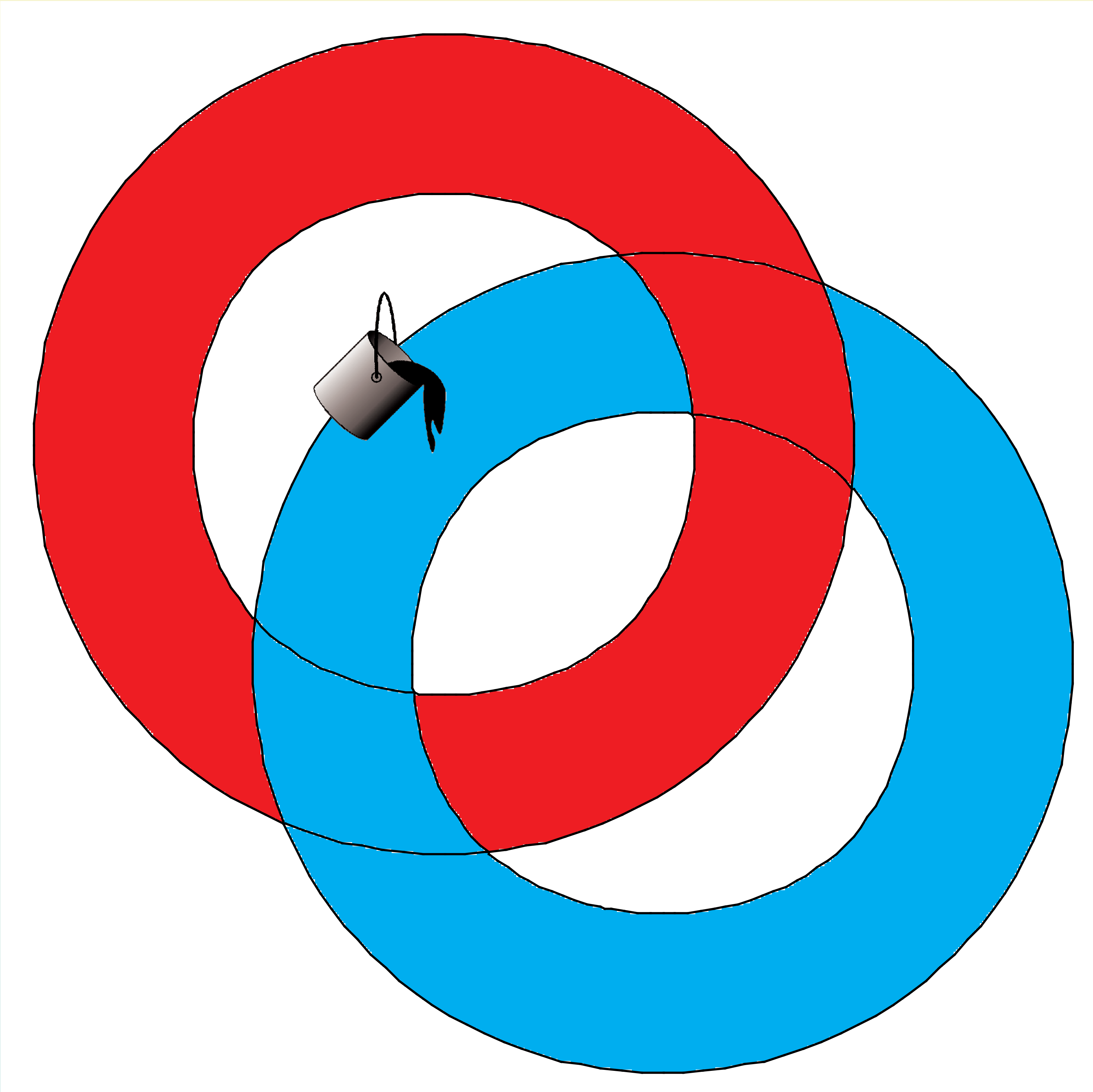
- Convert drawing to planar graph
- Paint faces of the graph independently

Adobe Illustrator Method



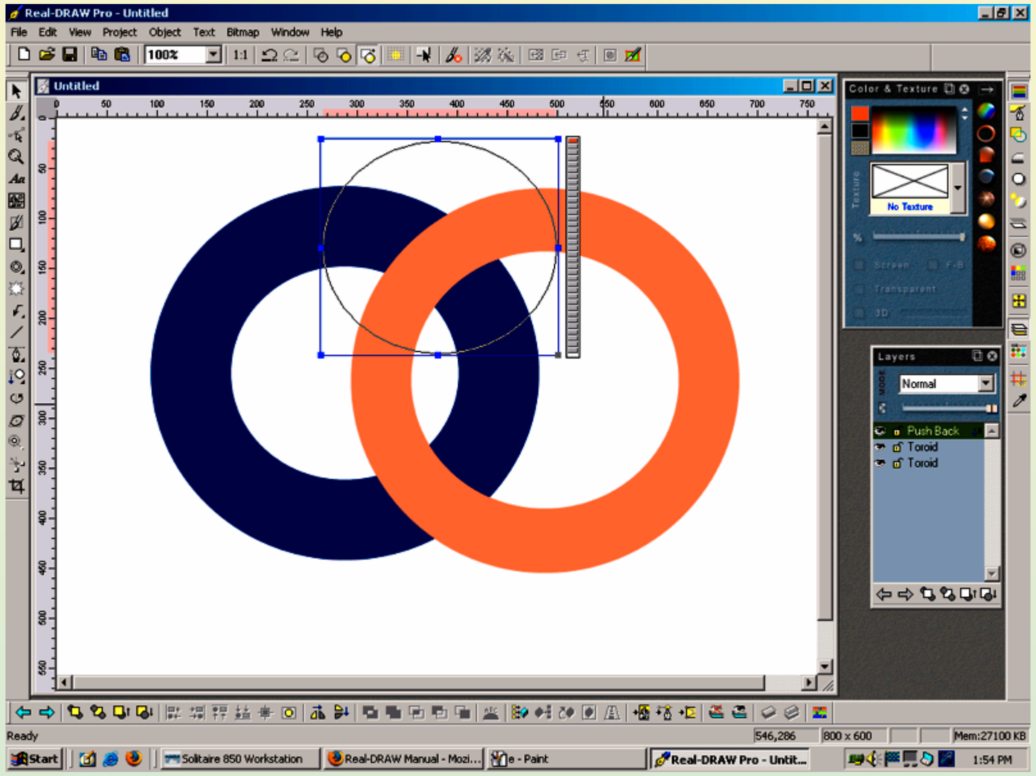
- Convert drawing to planar graph
- Paint faces of the graph independently

Adobe Illustrator Method



- Convert drawing to planar graph
- Paint faces of the graph independently

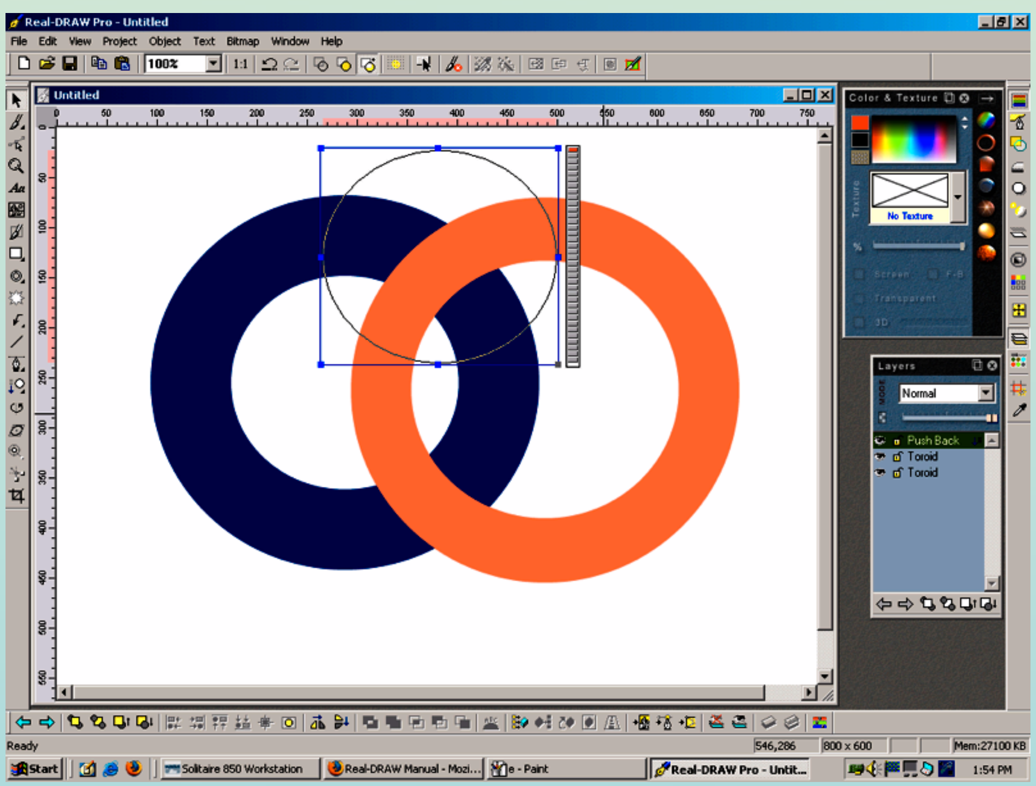
MediaChance Real-Draw Pro-3

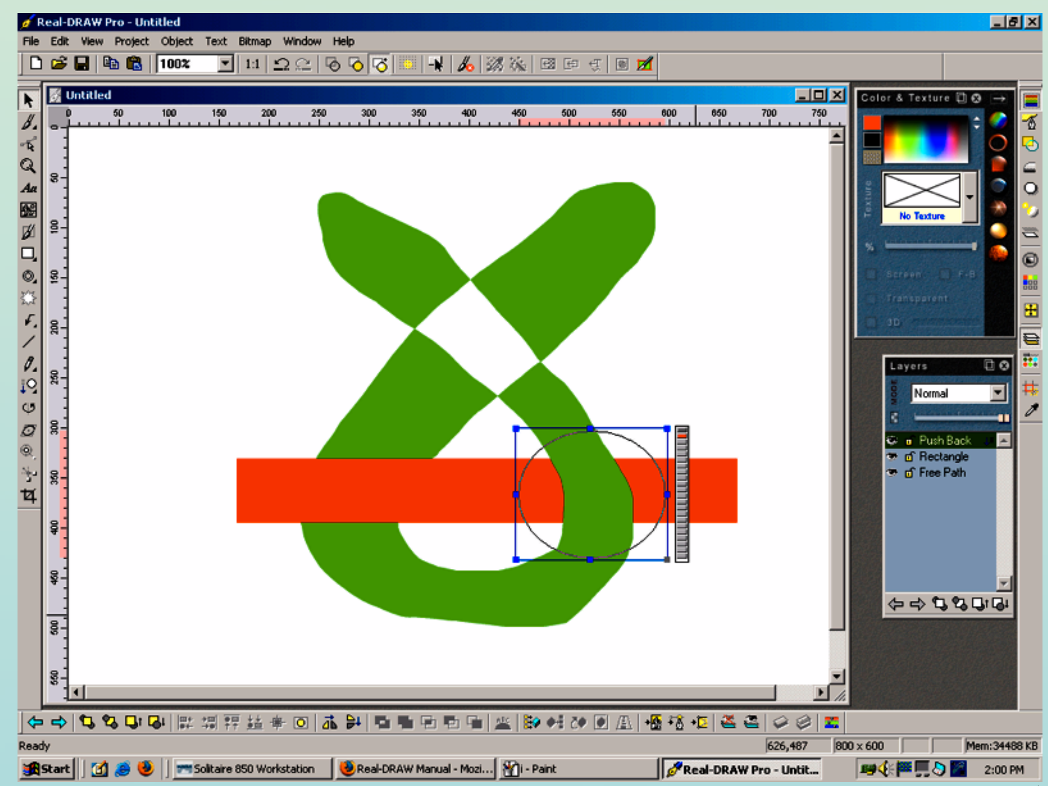


Push-back tool: The user can push the top layer down (figures left)

- Insufficient for transparent surfaces
- Cannot represent self-overlapping surfaces (figure below)

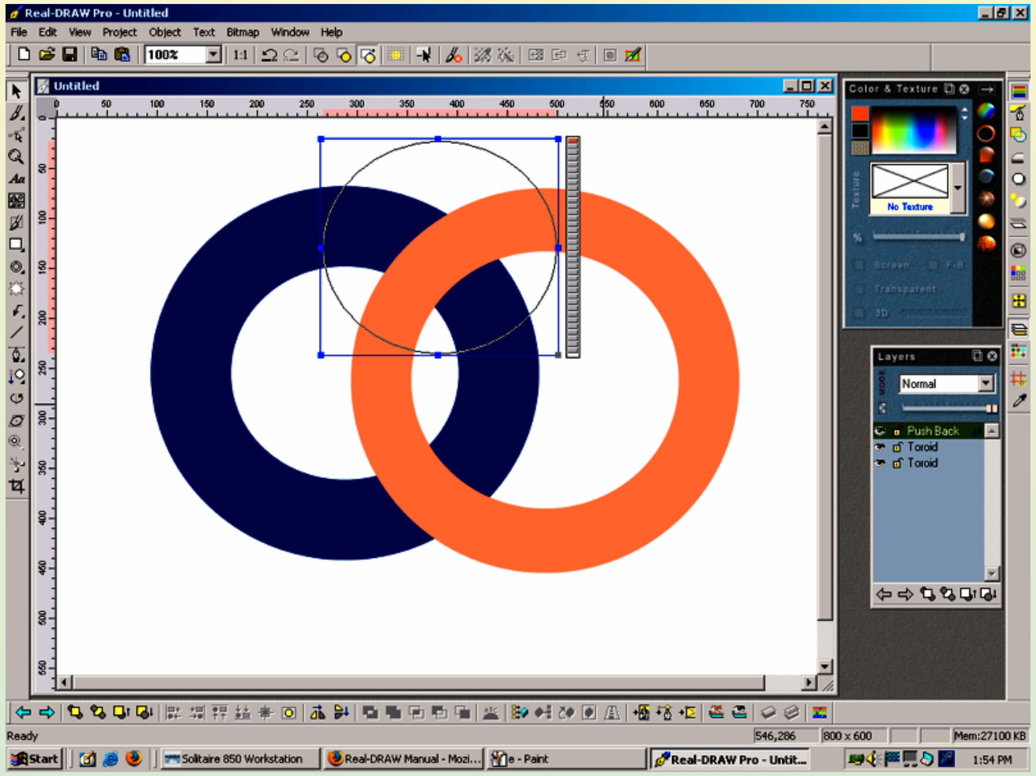
The right annulus is pushed down





11

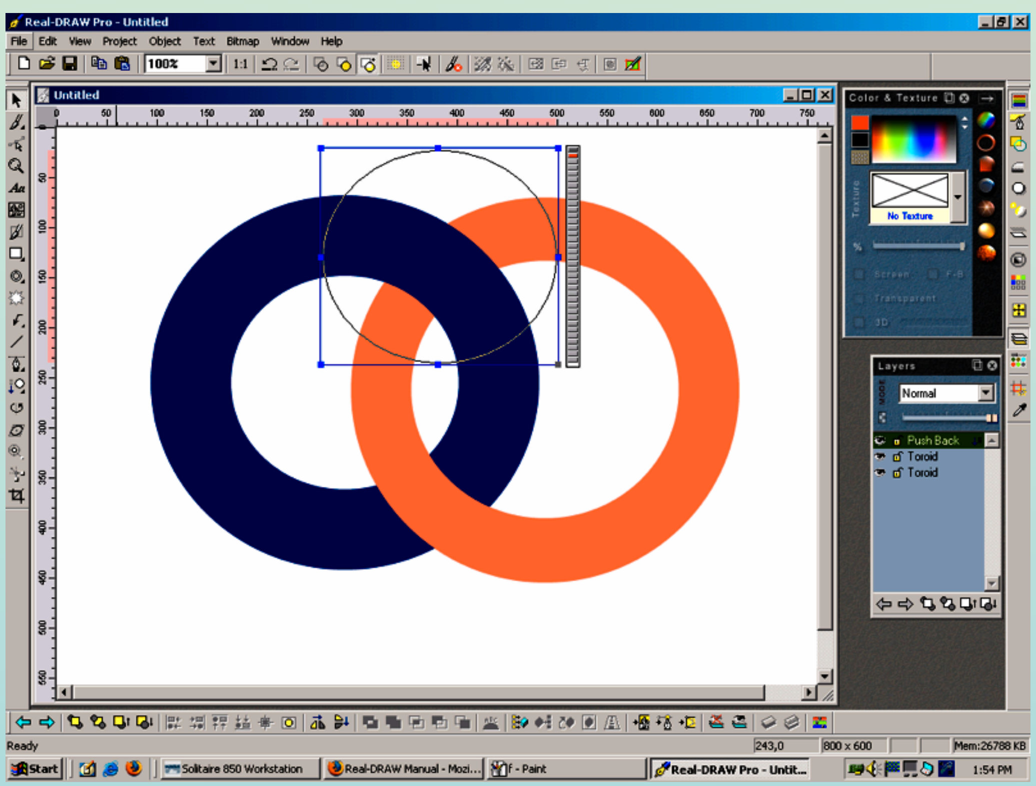
MediaChance Real-Draw Pro-3

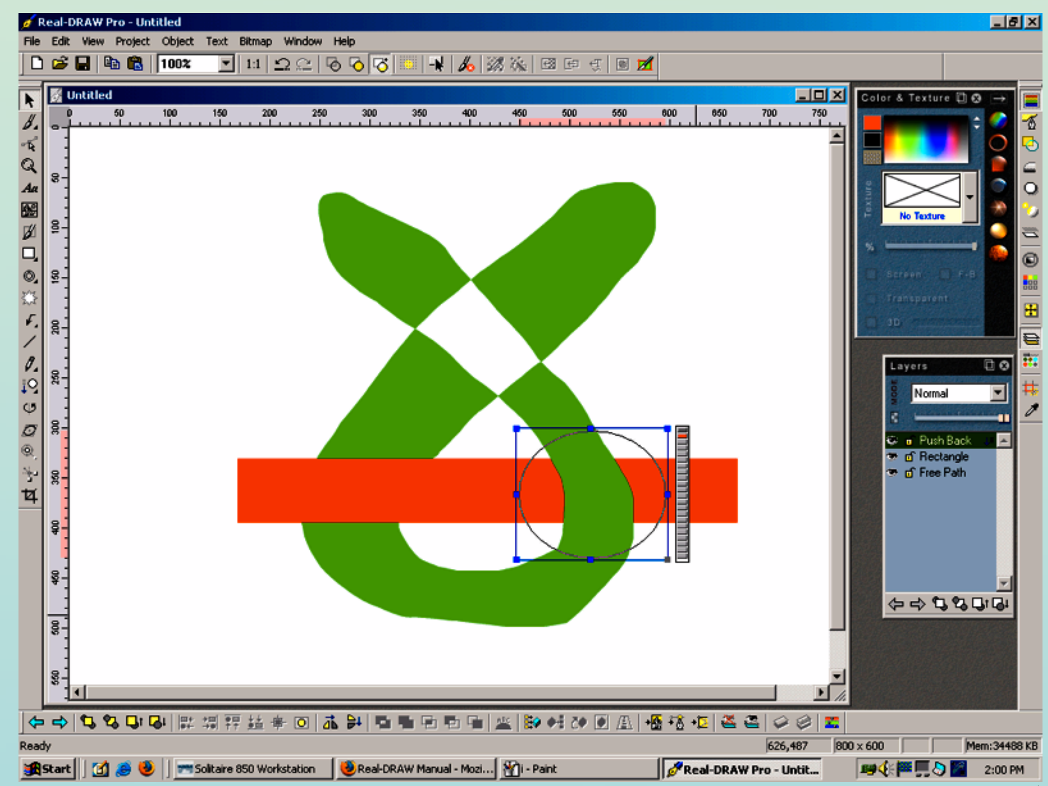


Push-back tool: The user can push the top layer down (figures left)

- Insufficient for transparent surfaces
- Cannot represent self-overlapping surfaces (figure below)

The right annulus is pushed down





11

Affordances

- ***Feasability*** is not the sole issue. ***Convenience*** and ***naturalness*** are also issues.

- ***Affordances***: The set of interactions that a physical object suggests for itself (Norman '02).

- Unlike conventional drawing programs, *Druid's* affordances are isomorphic to those of idealized physical surfaces.

- Thus, the user's experience is of interacting with surfaces, not with pictures of surfaces.

Norman, D. A., *The Design of Everyday Things*, Basic Books, 2002.

12

Talk Overview

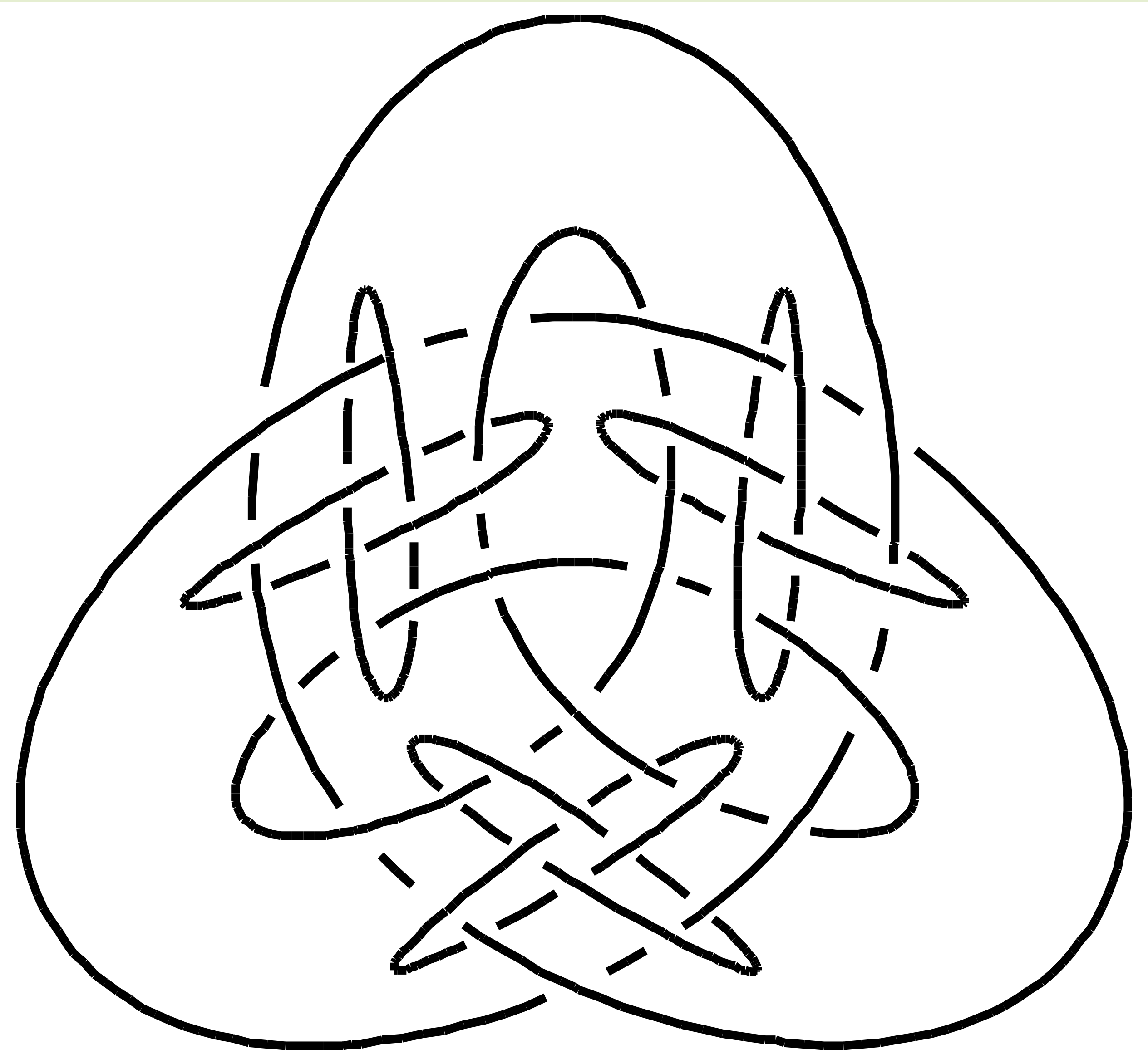
- Introduction, Current State-of-the-Art
- **Druid Description, Usage**
- Finding Legal Labelings
- Crossing-State Equivalence Classes
- Conclusions

13

Druid's Representation

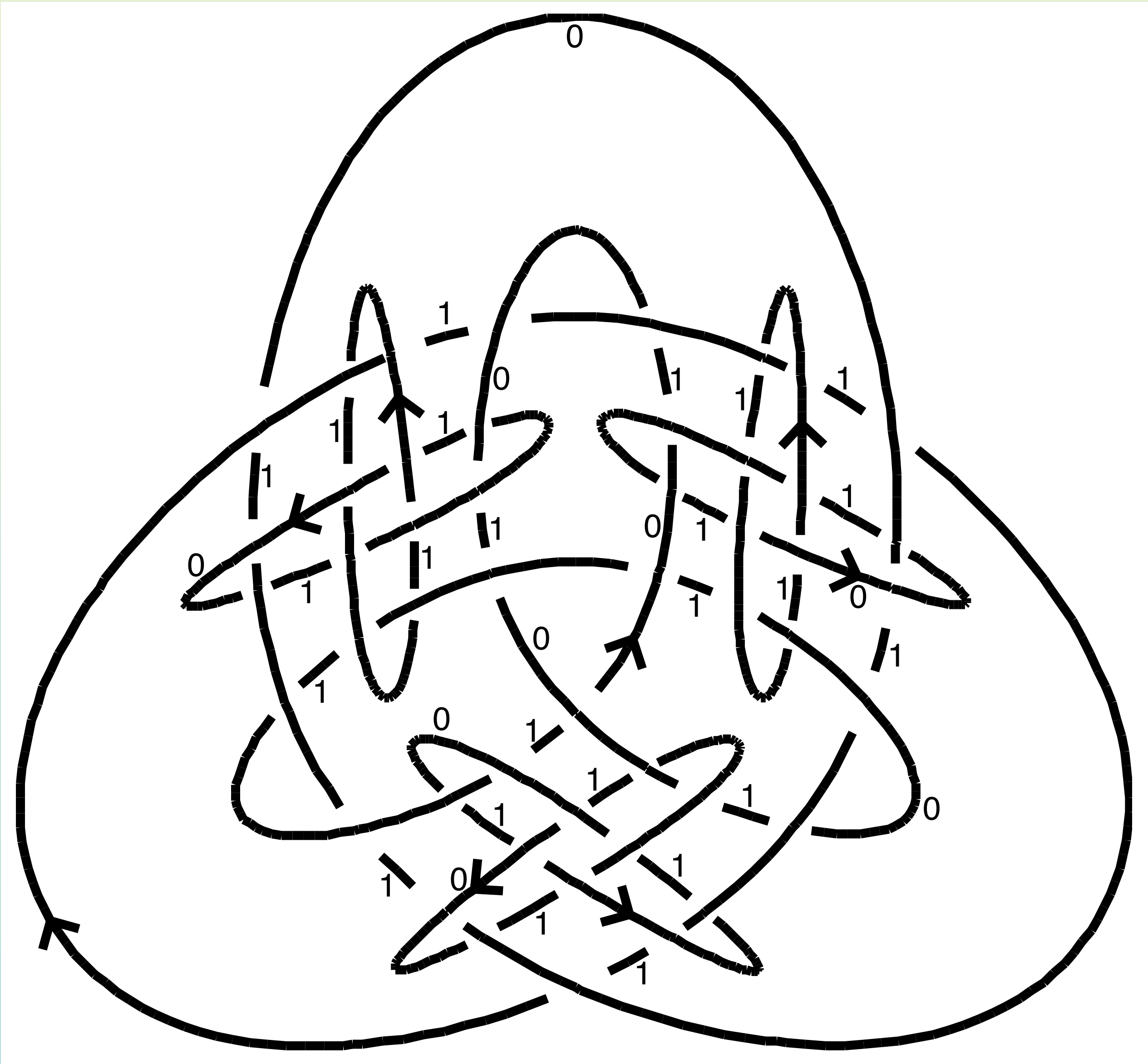
Knot-diagram:

A projection of closed curves indicating which curve is above where two cross



Labeled knot-diagram (Williams '94):

Sign of occlusion for every boundary (arrows)
Depth index for every boundary segment



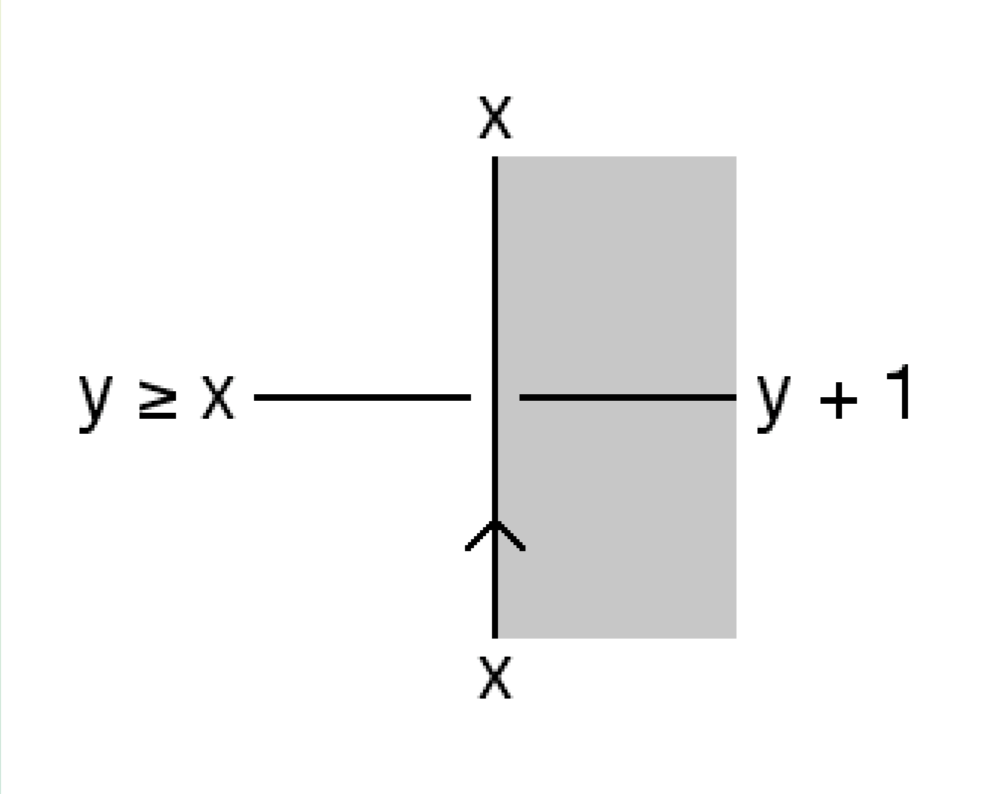
Williams, L. R., *Perceptual Completion of Occluded Surfaces*, PhD dissertation, Univ. of Massachusetts at Amherst, Amherst, MA, 1994.

14

Labeling Scheme

Imposes local constraints on the four boundary segment depths at a crossing

x, *y*: boundary segment depths

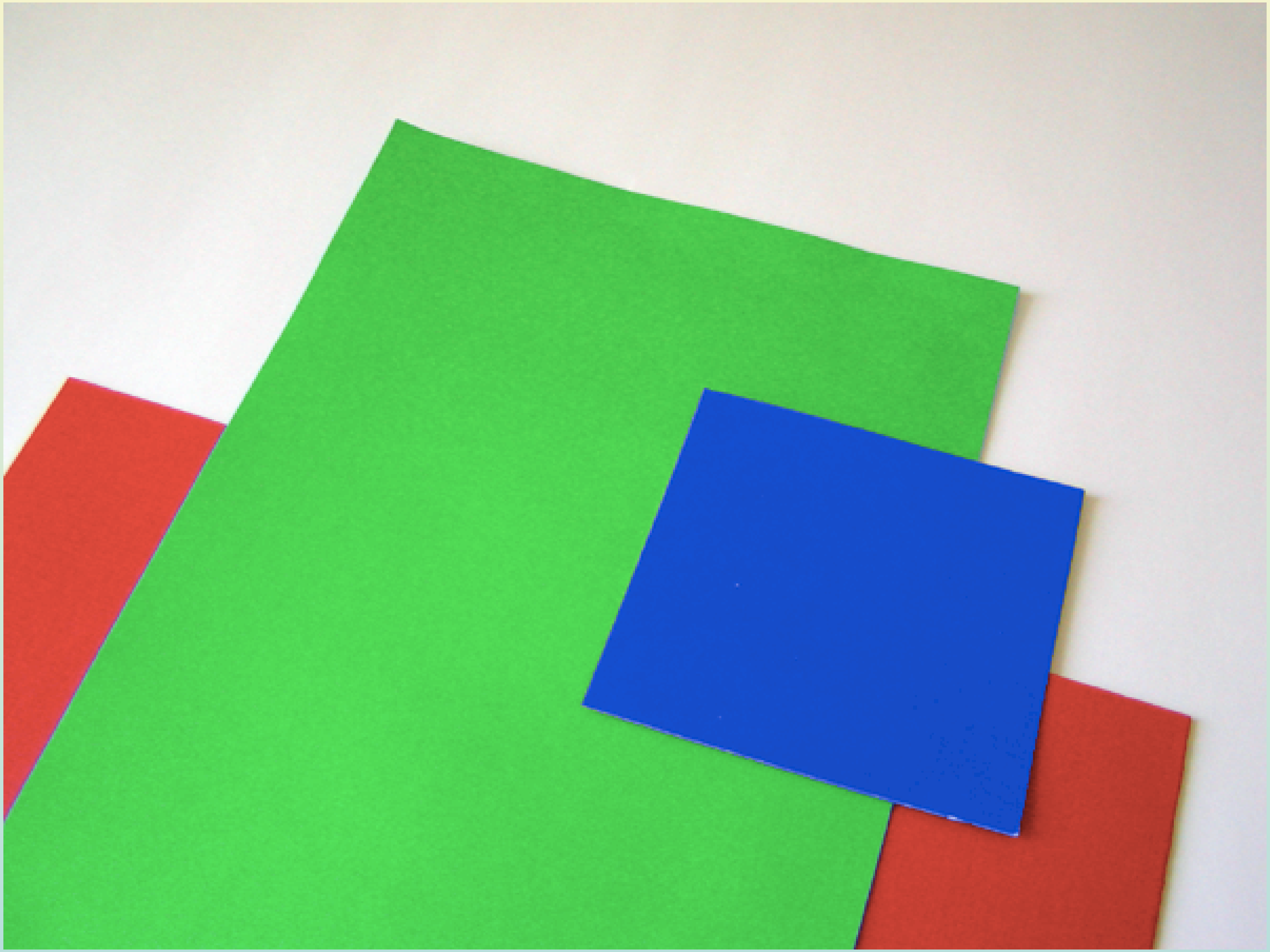


Legal labeling: A labeling in which every crossing satisfies the ***labeling scheme*** (Williams '94)

Williams, L. R., *Perceptual Completion of Occluded Surfaces*, PhD dissertation, Univ. of Massachusetts at Amherst, Amherst, MA, 1994.

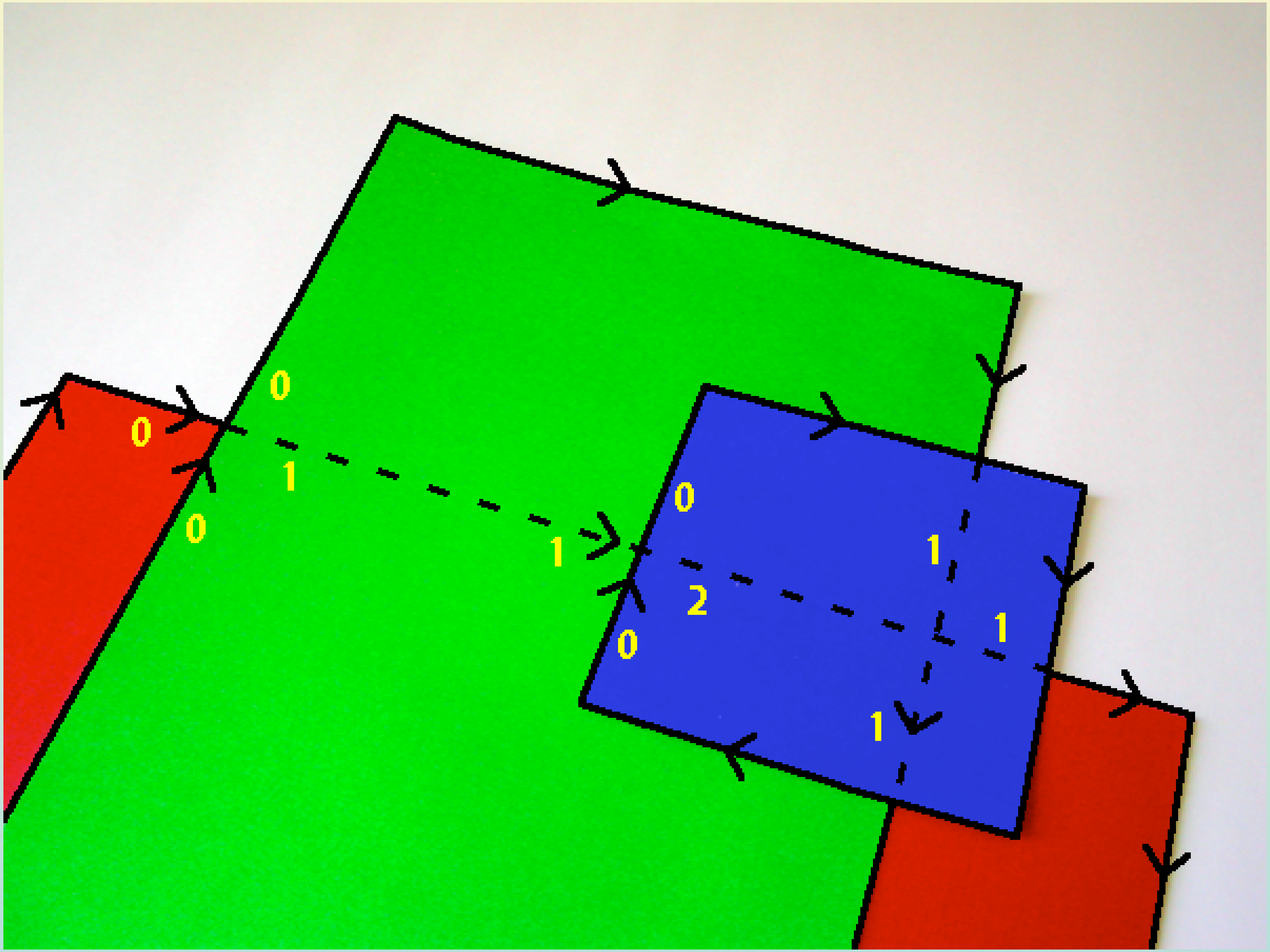
15

Labeling Scheme Justification

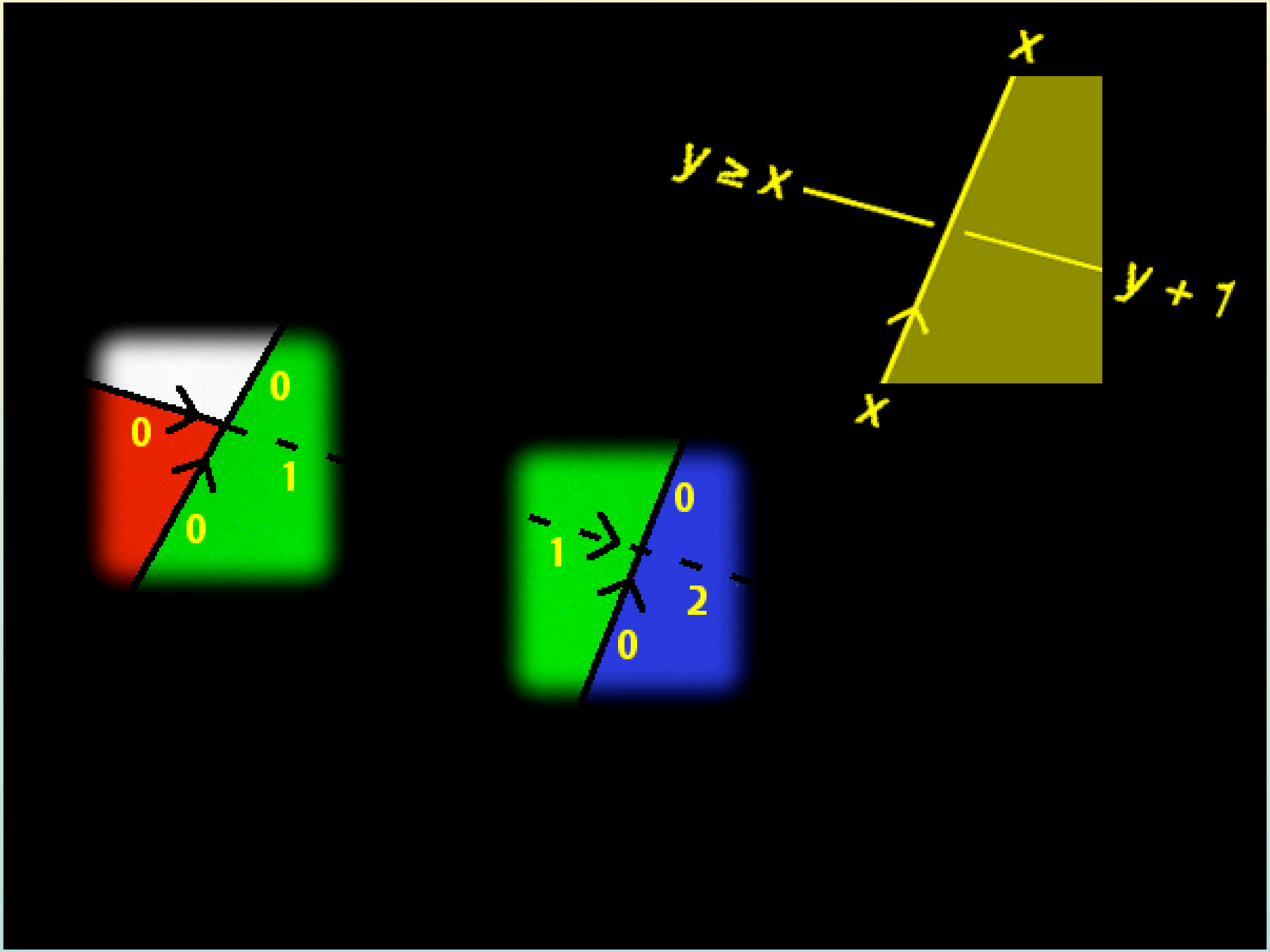


16

Labeling Scheme Justification



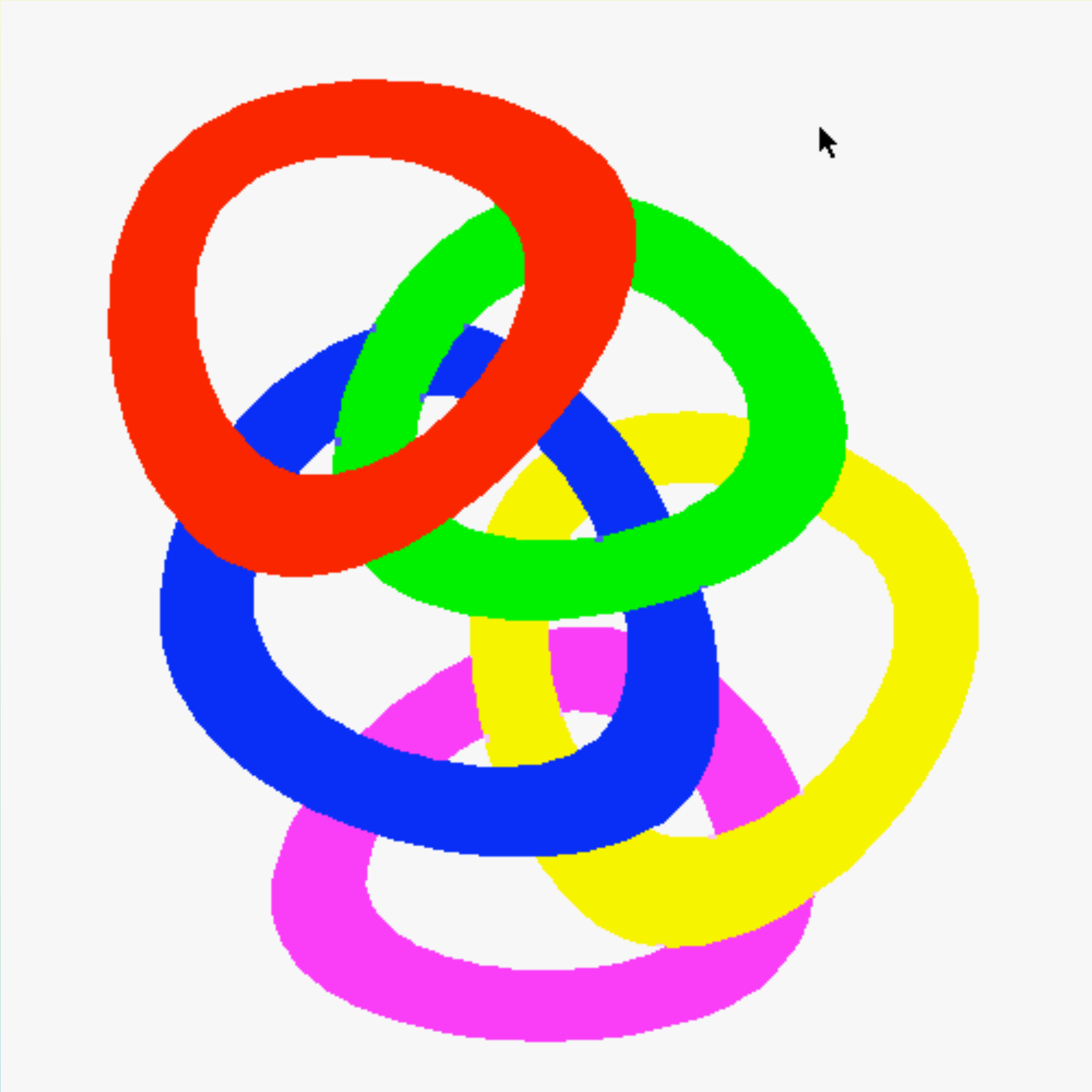
Labeling Scheme Justification



16

Using *Druid*

The *Crossing-Flip* Interaction



Drawing Program Interactions

- Create & delete boundaries
- Reshape & drag boundaries
- Crossing flip (Invert two surfaces' relative depths in an area of overlap)
- Sign-of-occlusion flip

19

Effects of Interactions on the Labeling

- Creation & deletion of crossings
- Reordering of crossings around boundaries
- Sign-of-occlusion flips
- Crossing-state flips
- Reshaping or dragging boundaries without causing topological changes

Effects of Interactions on the Labeling

Requiring ***labeling*** (topological change)

- Creation & deletion of crossings
- Reordering of crossings around boundaries
- Sign-of-occlusion flips

Requiring ***relabeling*** (topological change)

- Crossing-state flips

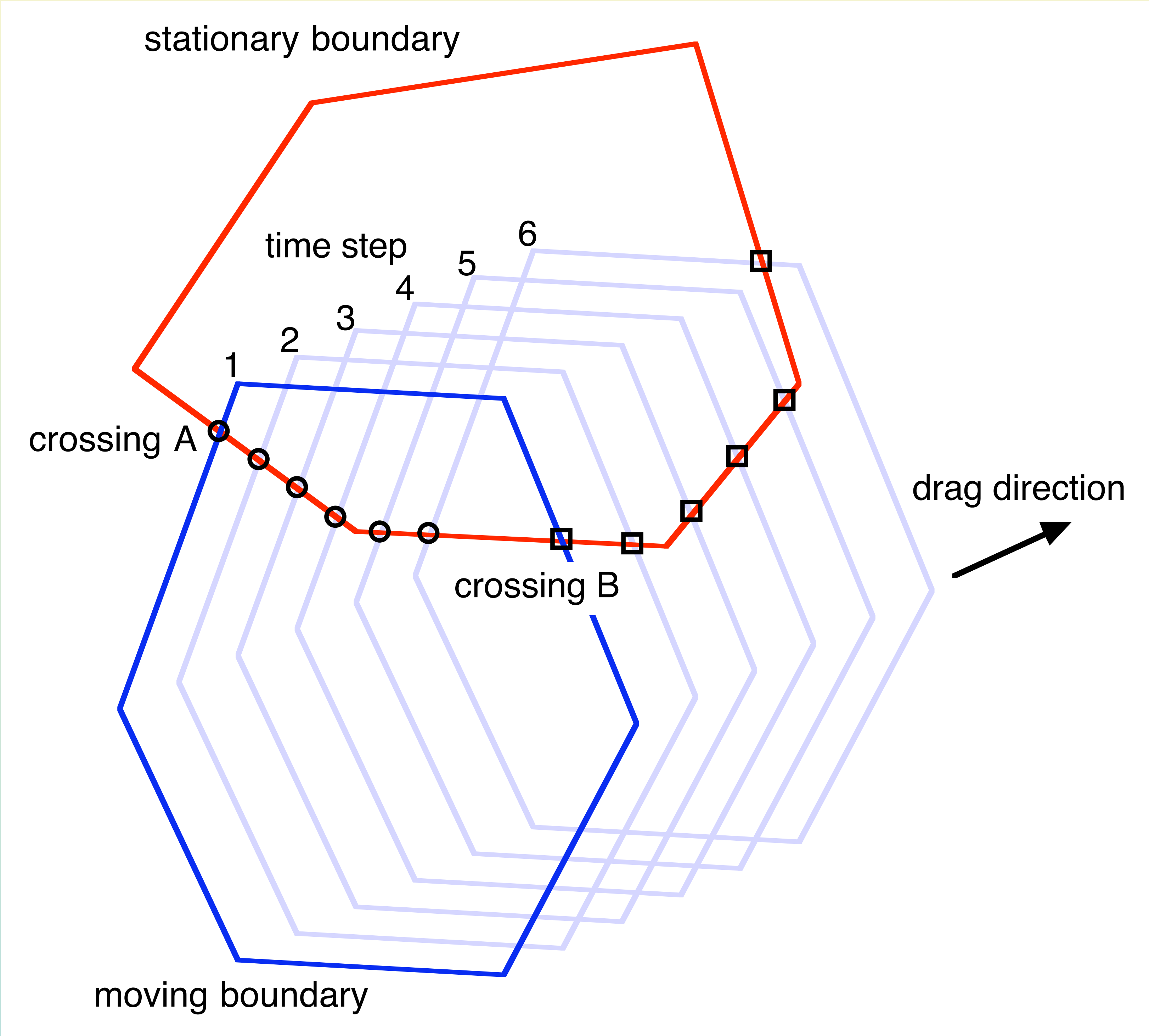
Maintaining labeling (no topological change)

- Reshaping or dragging boundaries without causing topological changes

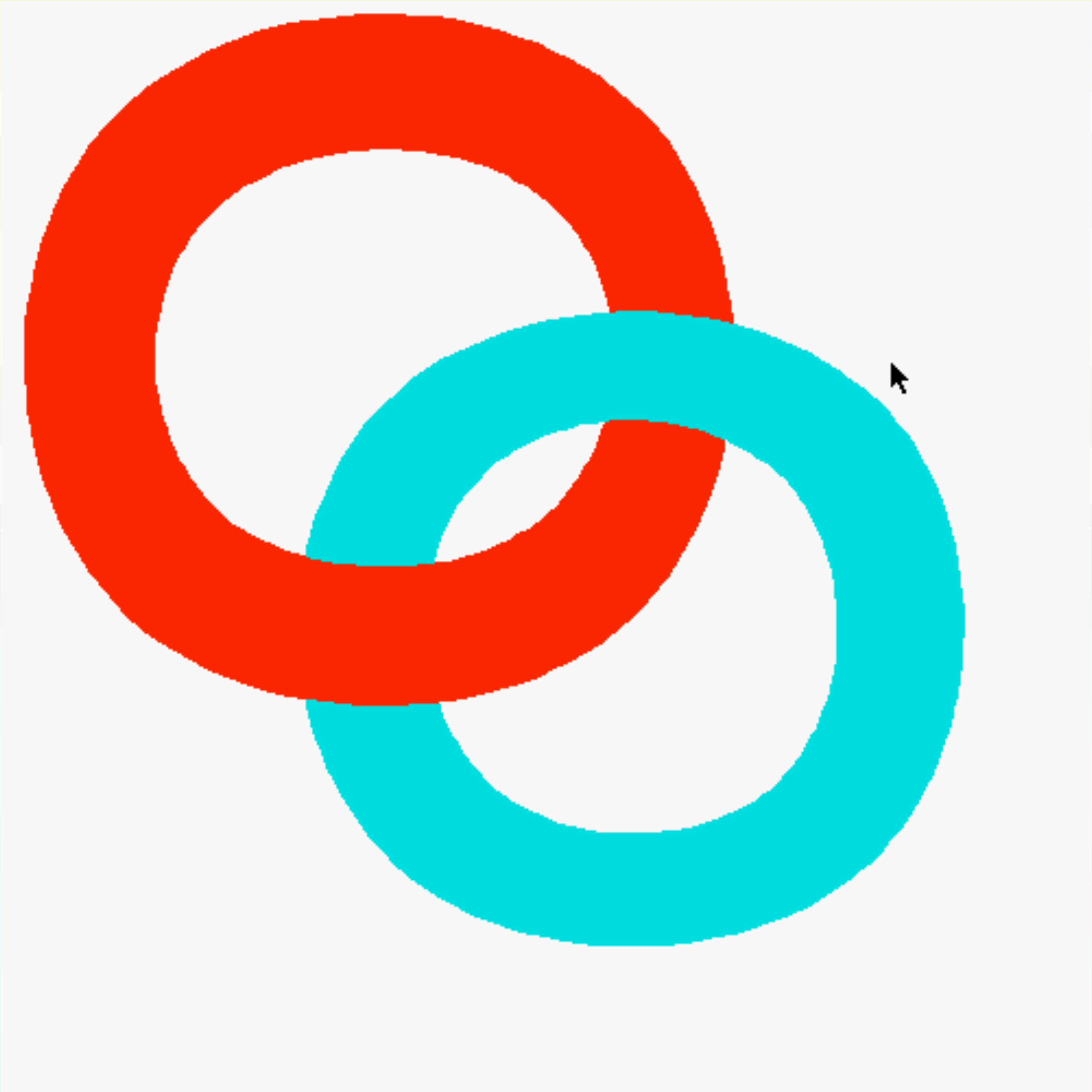
20

Crossing Projection

- Important to preserve crossing-states
- Naive destruction/ rediscovery of crossings would lose crossing-states
- *Druid* **projects crossings as they move around boundaries**



Demonstration of *Druid*



- *Druid* knows to move both boundaries at once.
- *Druid* relabels when the interlock breaks.

22

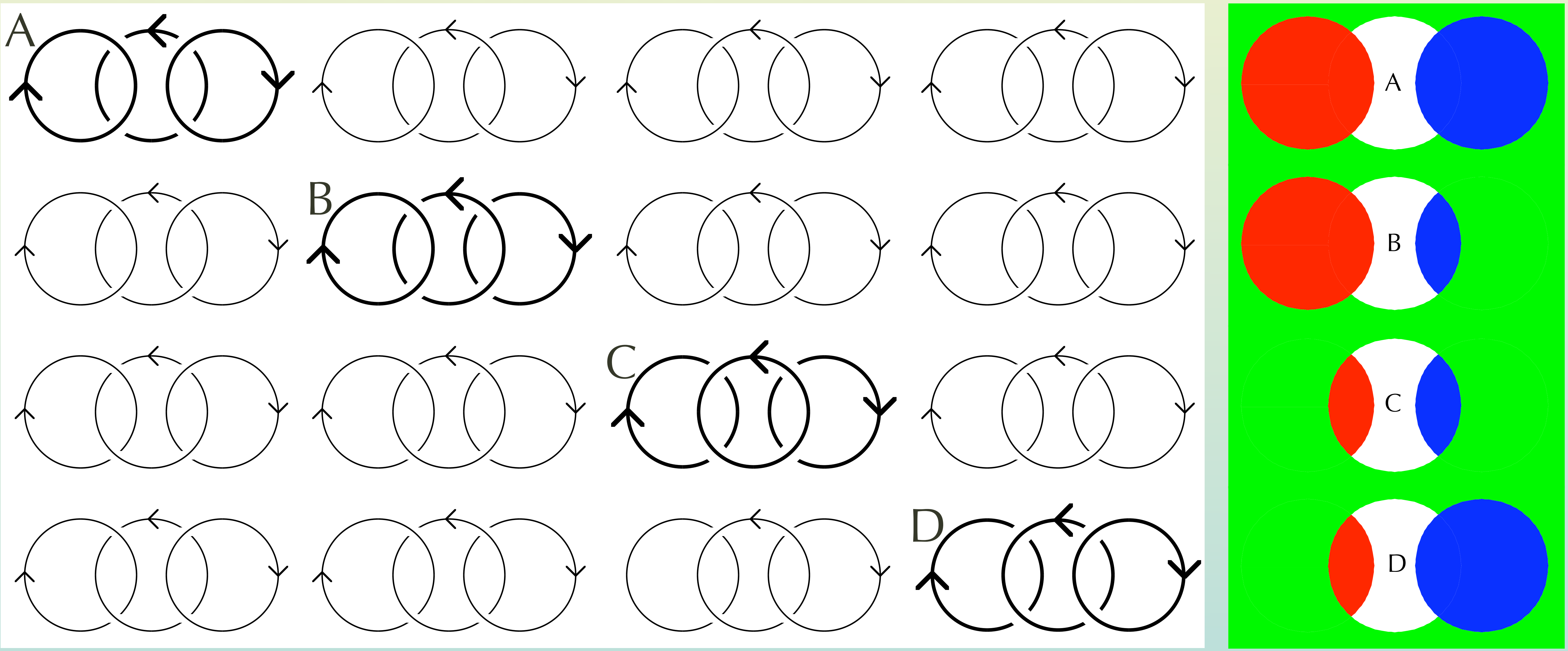
Talk Overview

- Introduction, Current State-of-the-Art
- Druid Description, Usage
- **Finding Legal Labelings**
- Crossing-State Equivalence Classes
- Conclusions

23

Finding a Legal Labeling

Labeling space: All possible labelings for a labeled knot-diagram. Labeling space size: 2^C

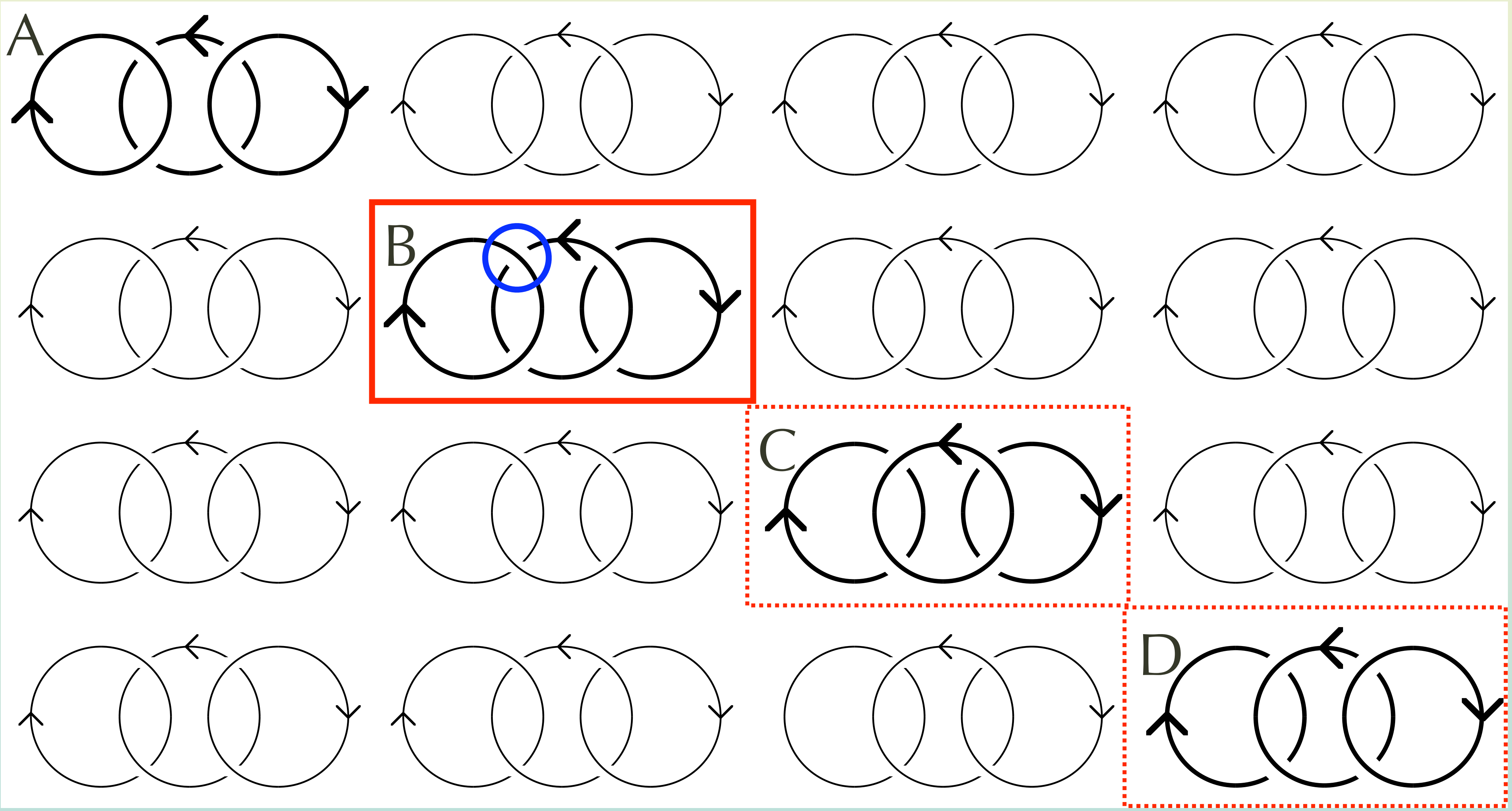


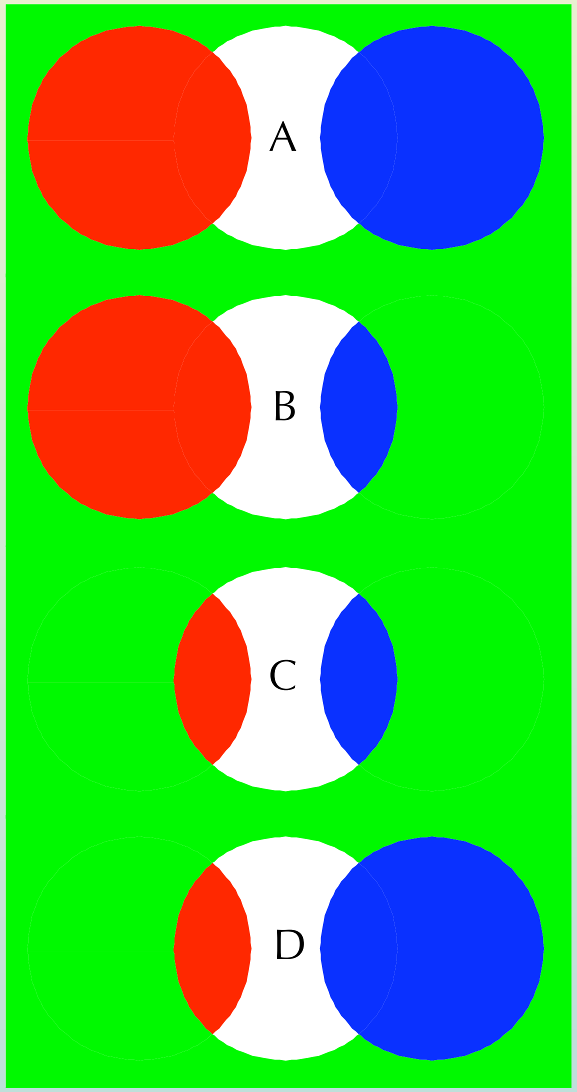
Druid maintains a legal labeling automatically.

24

Minimum-Difference Search

Druid searches the *labeling space* for the ***minimum-difference labeling***.





- Labeling is currently in state *B*.
- User clicks the blue-circle marked crossing.
- *C* and *D* are possible solutions, *C* is minimum difference from *B*.

The Labeling Search

- Branch-and-bound
- Constraint propagation
- Iterative deepening
- Timeouts

26

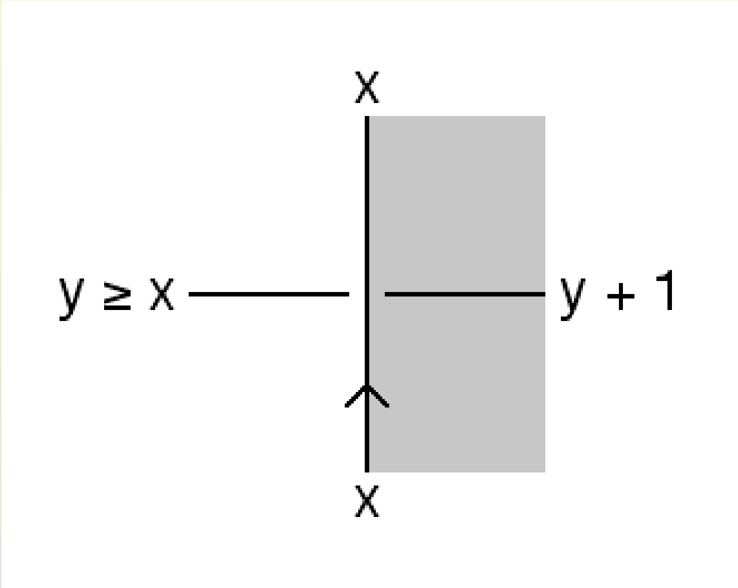
Branch-and-bound

- Search goal: ***minimum difference labeling***
- Node expansion can never decrease the accumulated labeling difference.
- Minimum difference legal solution gives the bound.
- Search is truncated when the accumulated current difference exceeds the bound.

27

Constraint Propagation (Waltz '75)

- Orders the search so that legal solutions are found earlier.



- Legal solutions define bounds.
- Constraint propagation works in concert with branch-and-bound to increase search efficiency.

Waltz, D. L., Understanding line drawings of scenes with shadows, McGraw-Hill, New York, pp. 19-92, 1975.

28

Iterative Deepening

- Branch-and-bound works best if good solutions are found earlier.
- In good solutions, changes are localized to the ***area of interest***.
- Search is restarted with increasing ***search horizons***.

29

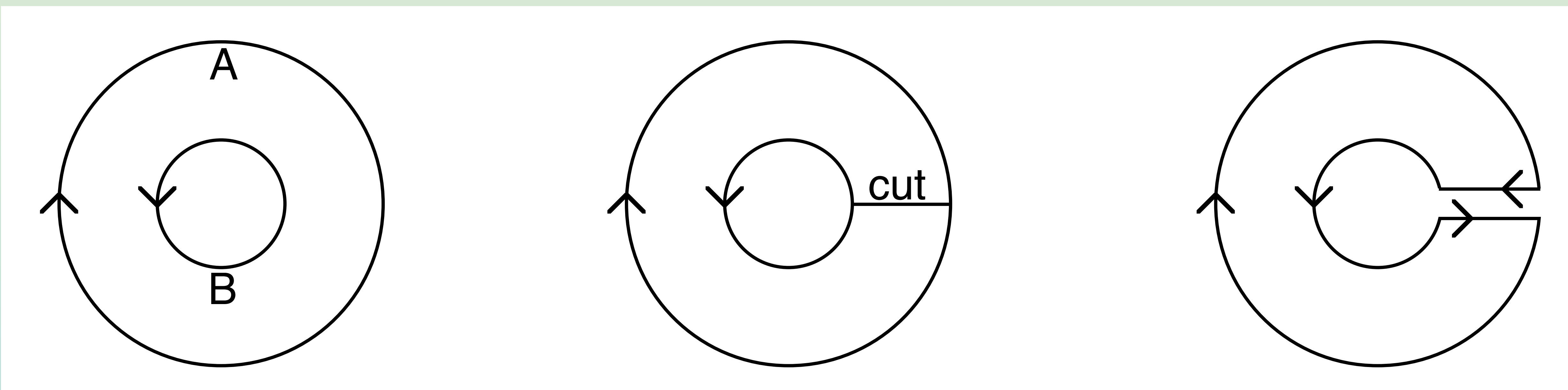
Timeouts

- The search can take too long.
- Two timeouts:
  - ***Very short timeout (0.1 sec)***: If a solution has been found during the search
  - ***Longer timeout (5.0 sec)***: If no solution has been found yet

30

Boundary Grouping with Cuts

- Some surfaces have multiple boundaries.
- This can cause problems.
- A ***cut*** between two different boundaries reduces the number of boundaries by one.

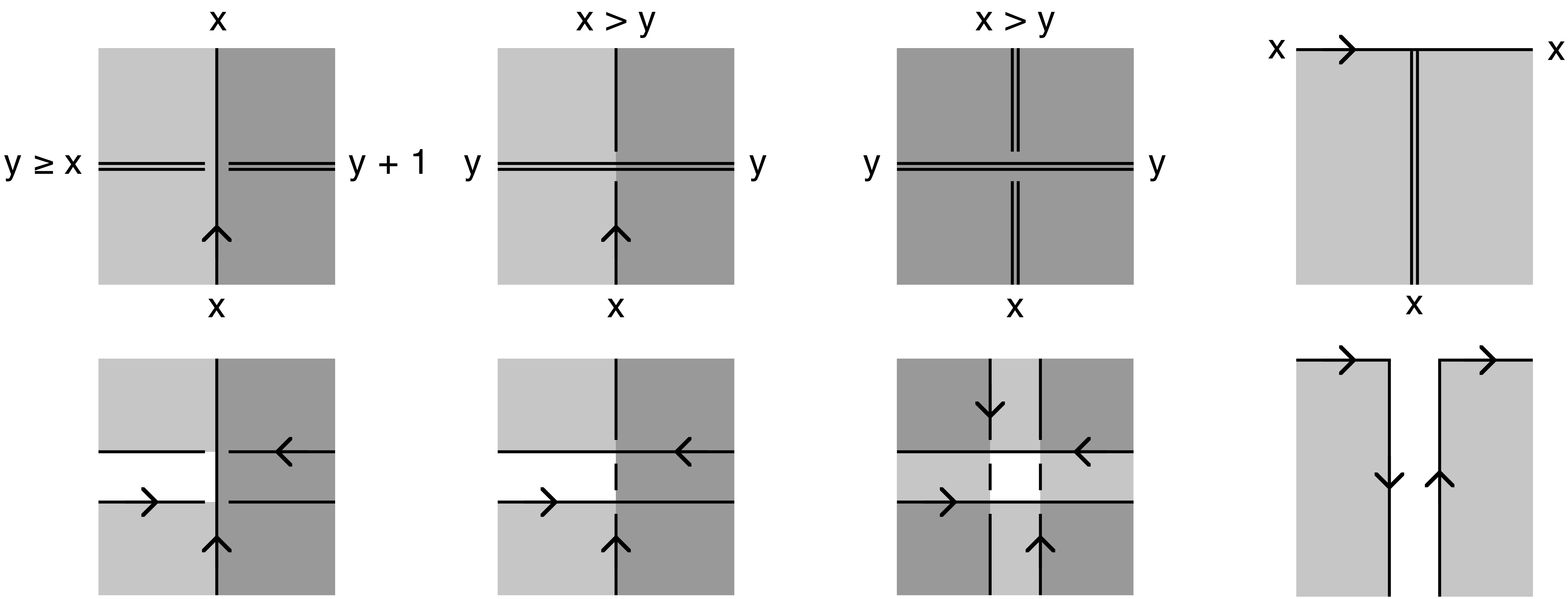


Cuts are a geometric device. Needn't be horizontal or straight.

31

Cut Labeling Schemes

Using cuts requires four new labeling schemes.



1 A boundary crosses a cut with the boundary above

2 A boundary crosses a cut with the boundary below

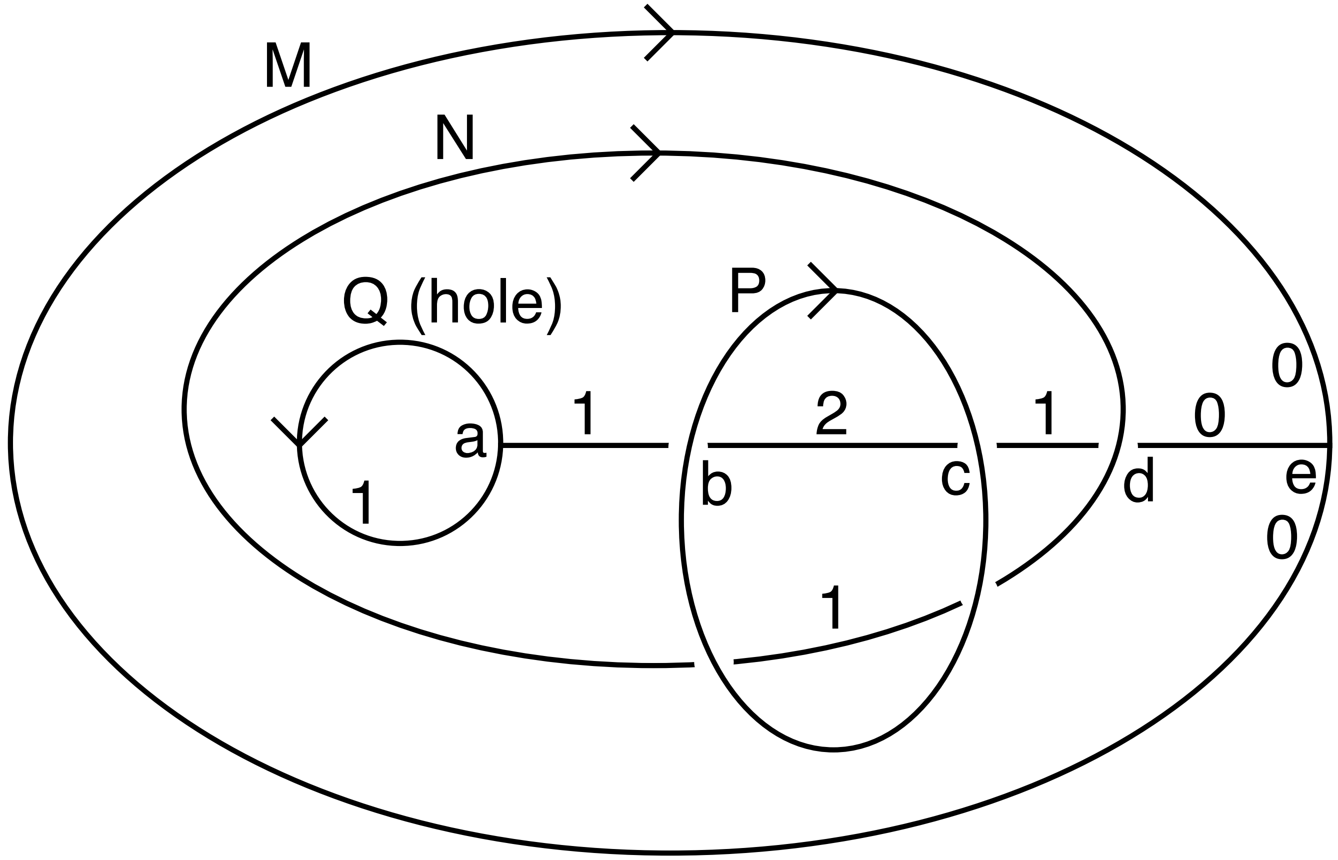
3 A cut crosses a cut

4 A cut *ends* and attaches to a boundary, *i.e.*, a *T-junction*

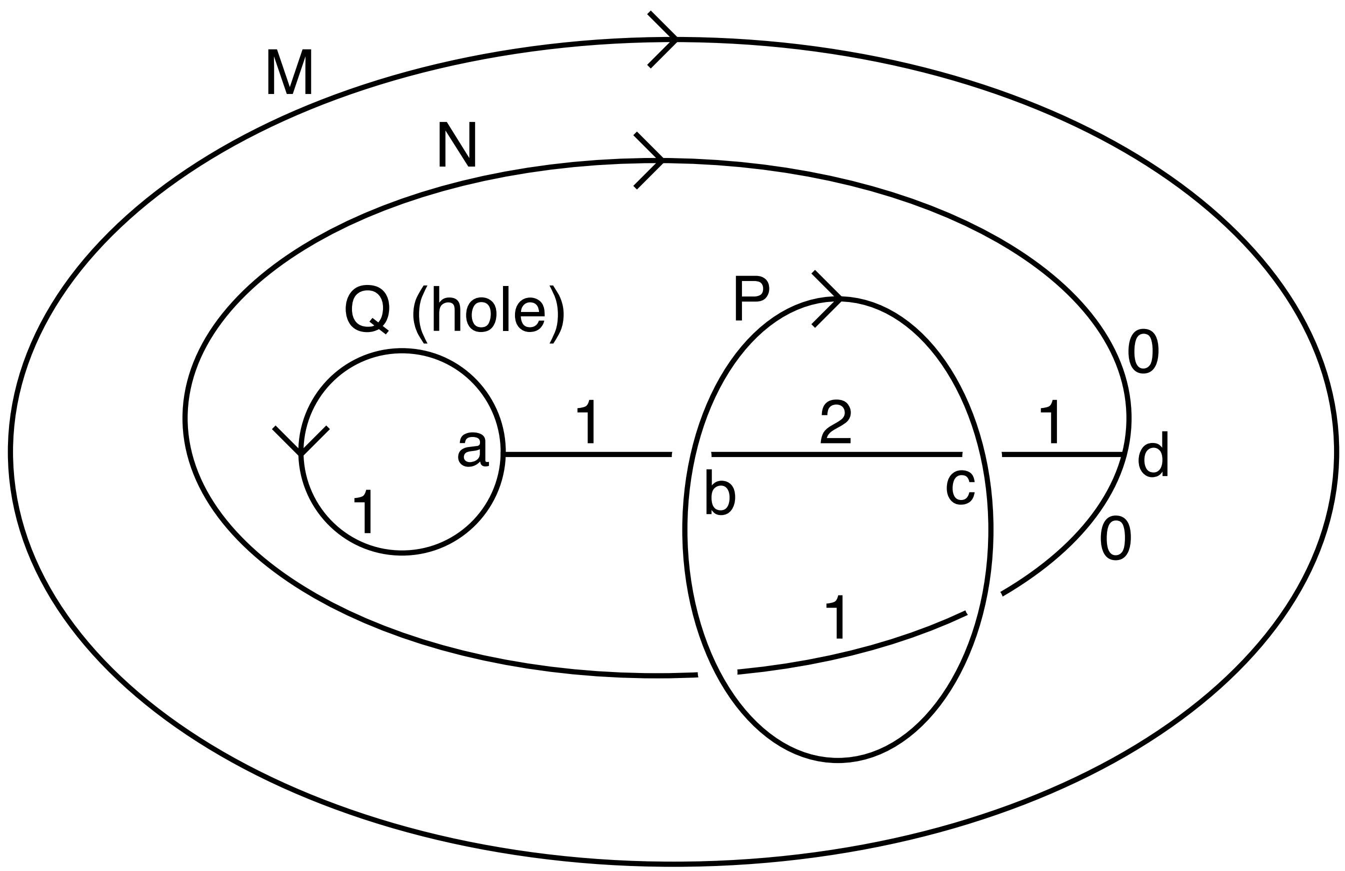
Cuts denoted with a double line (top row) and a gap (bottom row)

32

Finding Legal Cuts



A successful cut:
Last crossing (*e*) is legal.

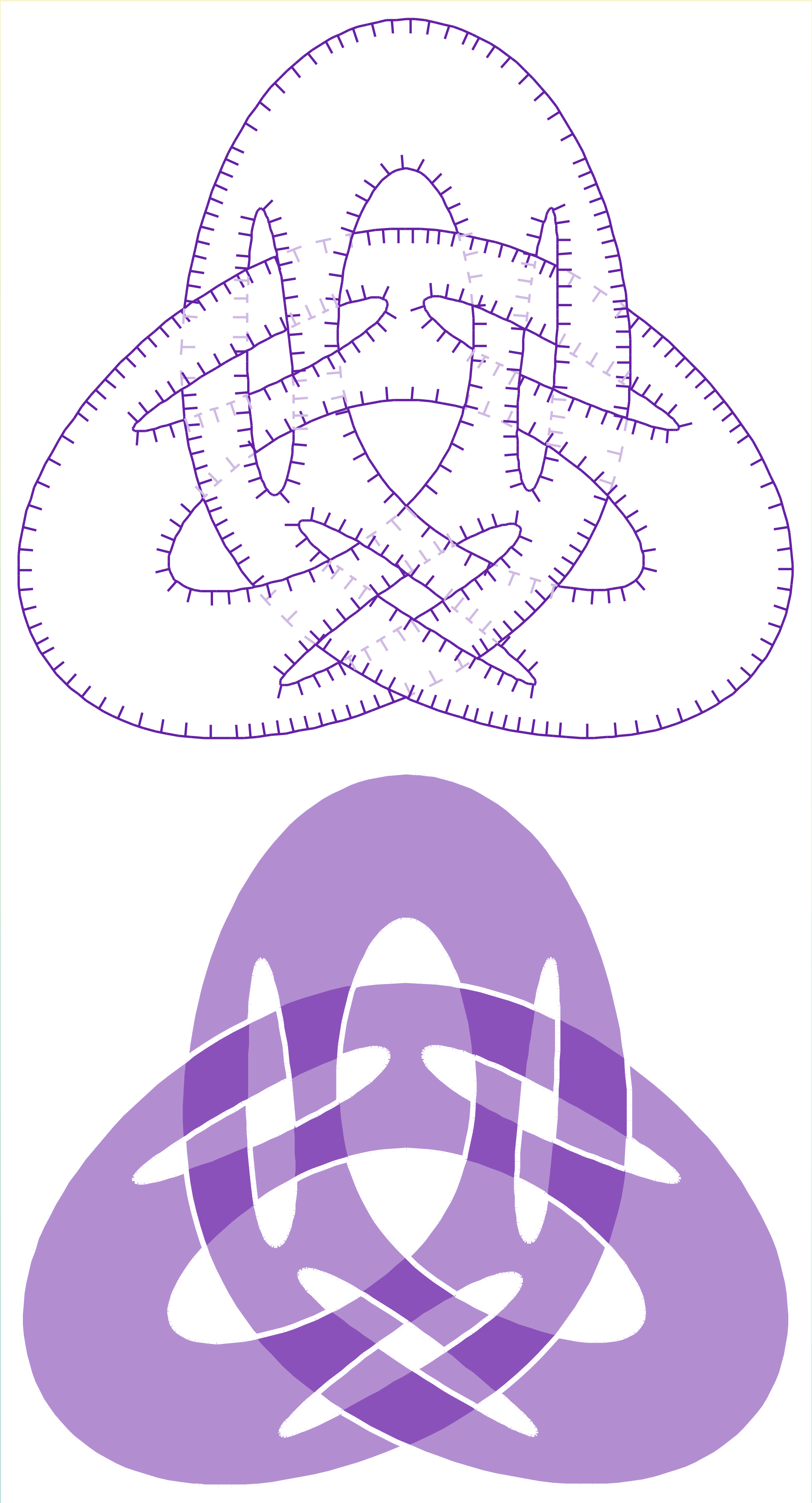


An unsuccessful cut:
Last crossing (*d*) is illegal.

33

Rendering

- Conversion of a labeled knot-diagram to an image with solid fills
- Requires full depth ordering of all surfaces covering each region
- *Druid* uses the ***episcotister model*** (Metelli '74)

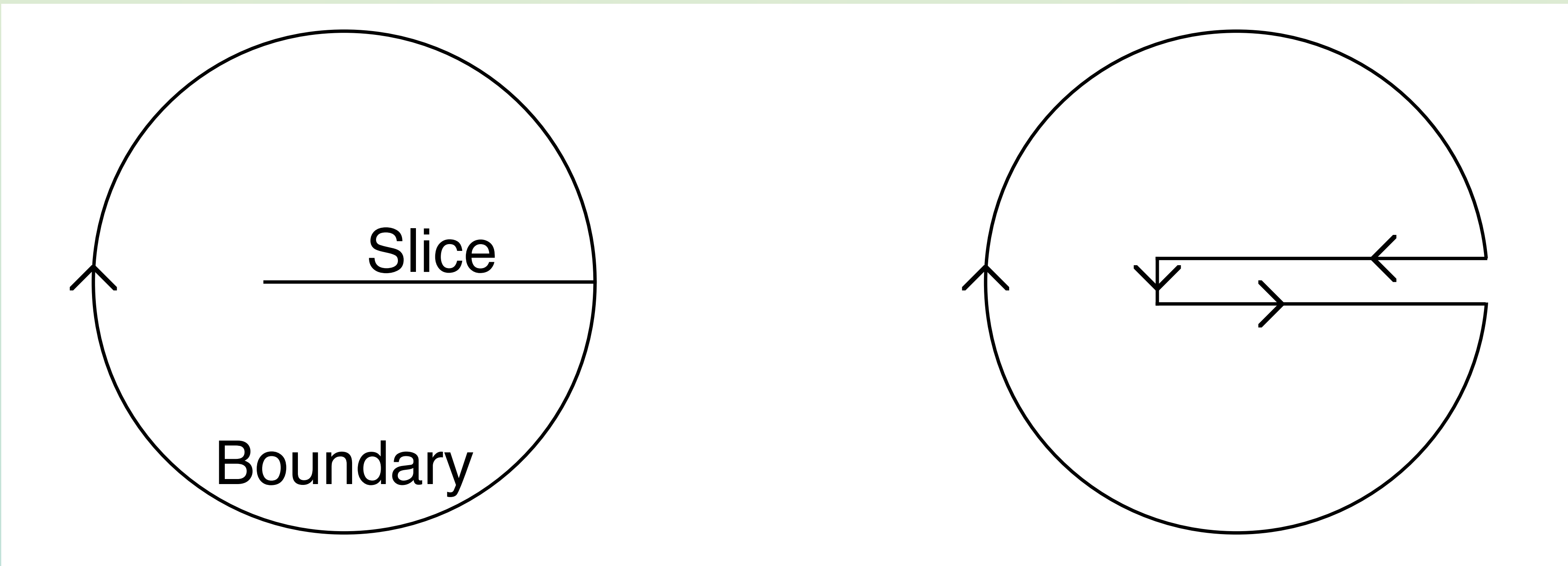


Metelli, F., The perception of transparency, Scientific American, 230(4), pp. 90-98, 1974.

34

Slice

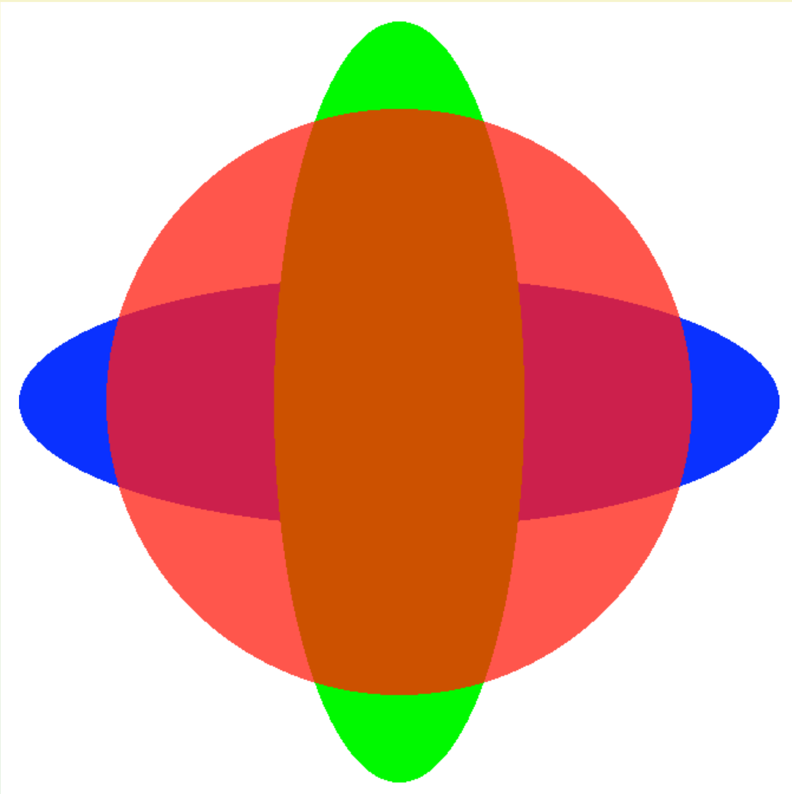
- A ***slice*** connects a location on a boundary to a point within the bounded surface.
- Similar to a cut.

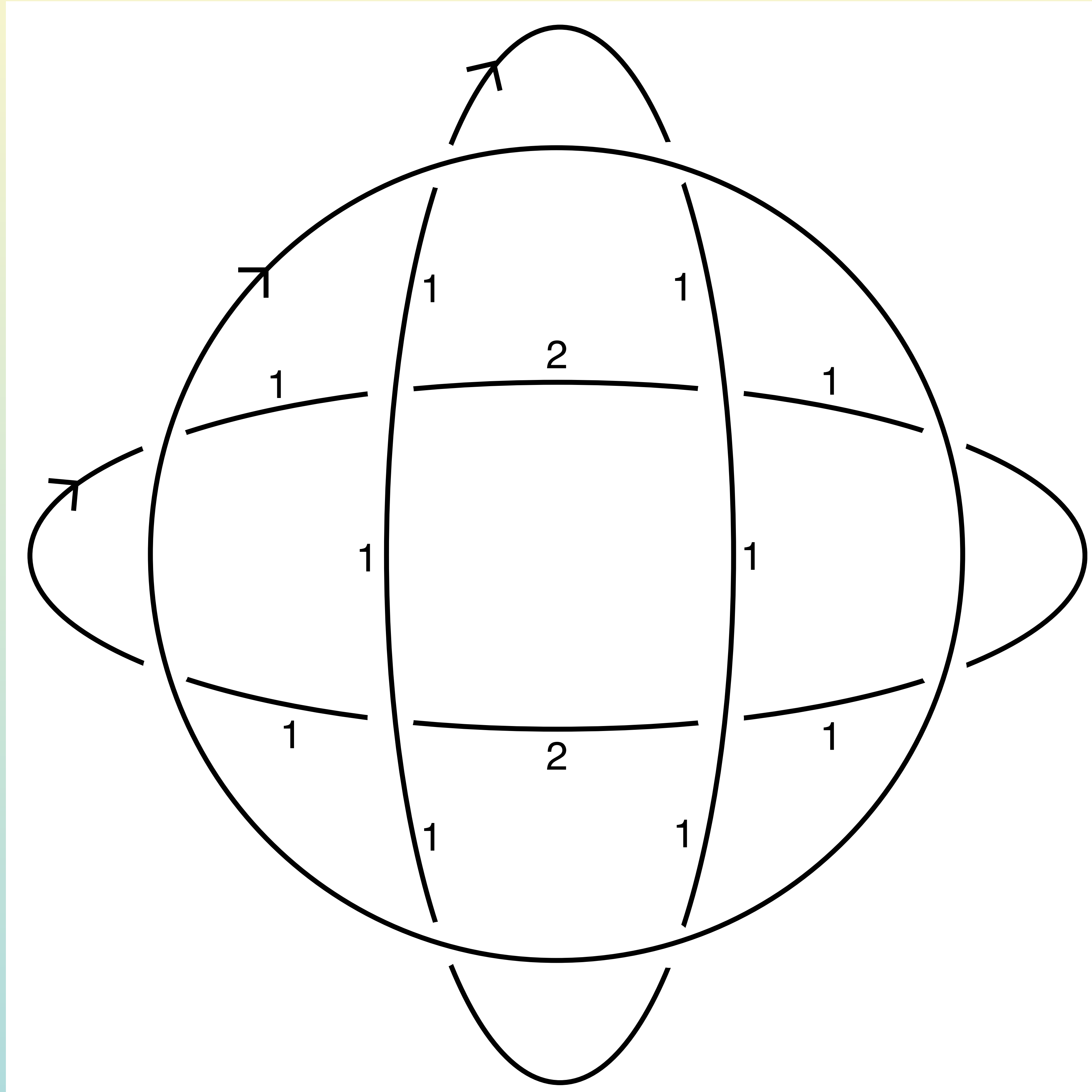


Slices are a geometric device. Needn't be horizontal or straight.

35

Using Slices to Find Region Coverings

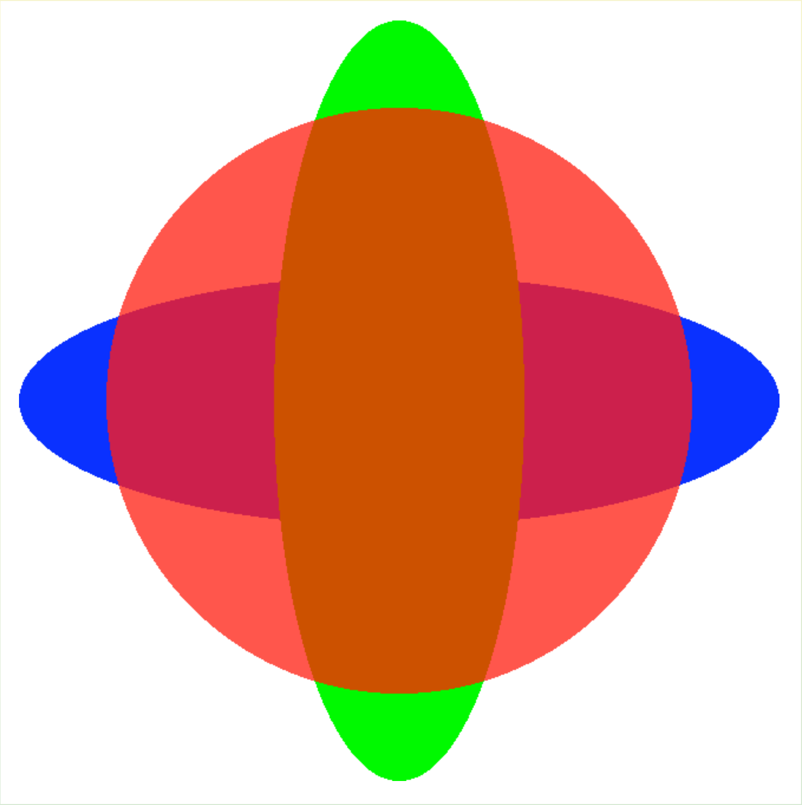




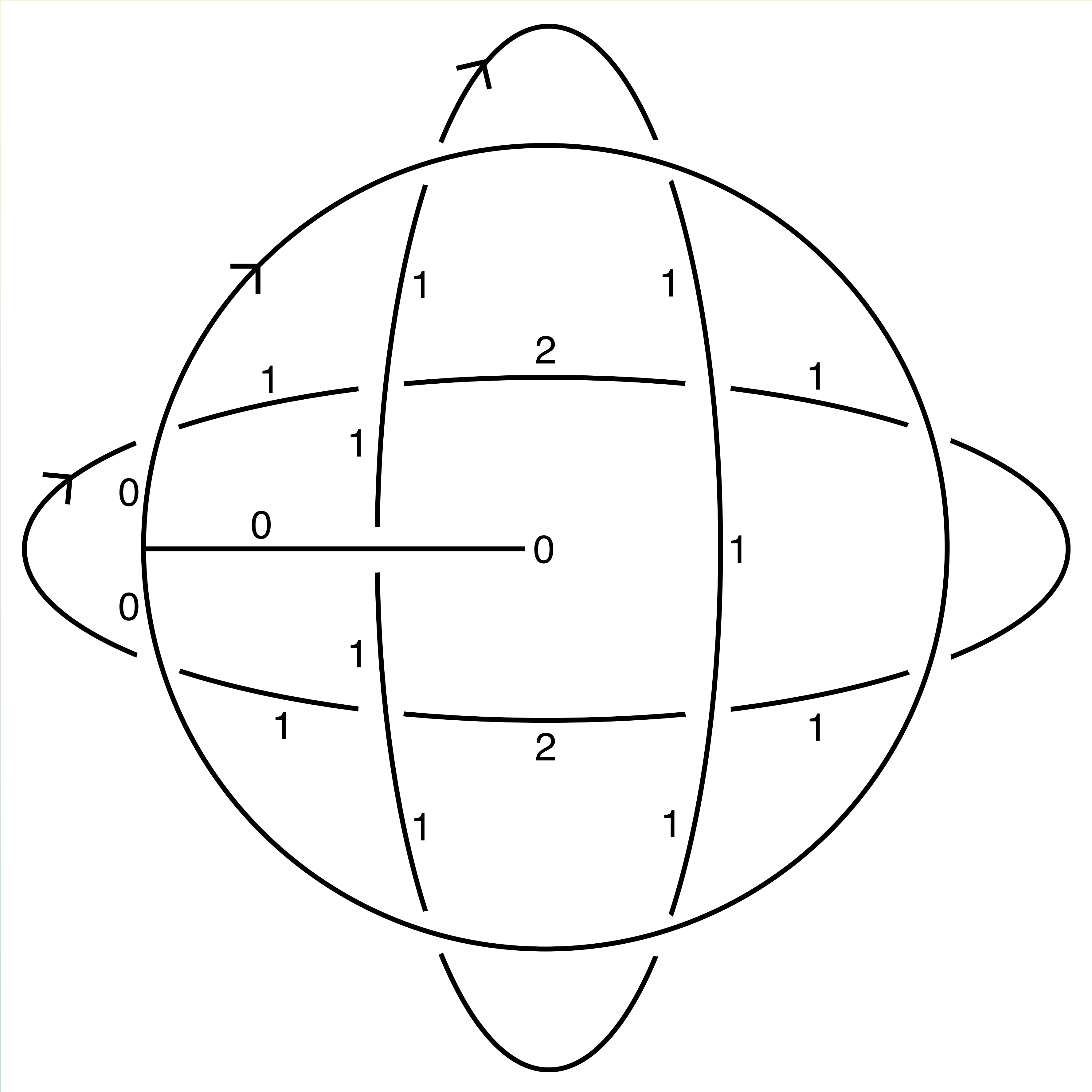
Red is above **green**, which is above blue.

36

Using Slices to Find Region Coverings

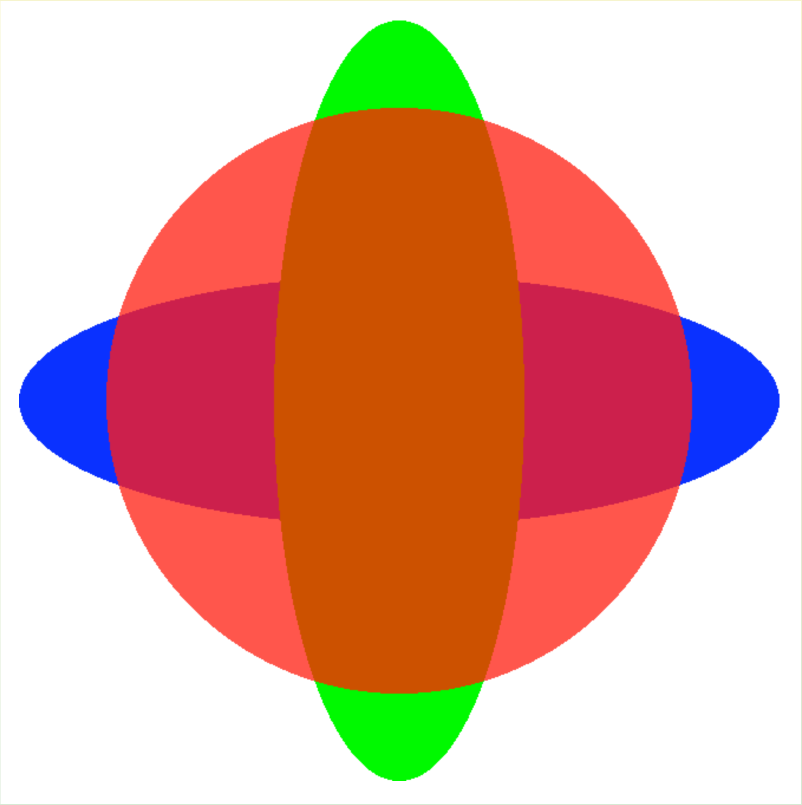


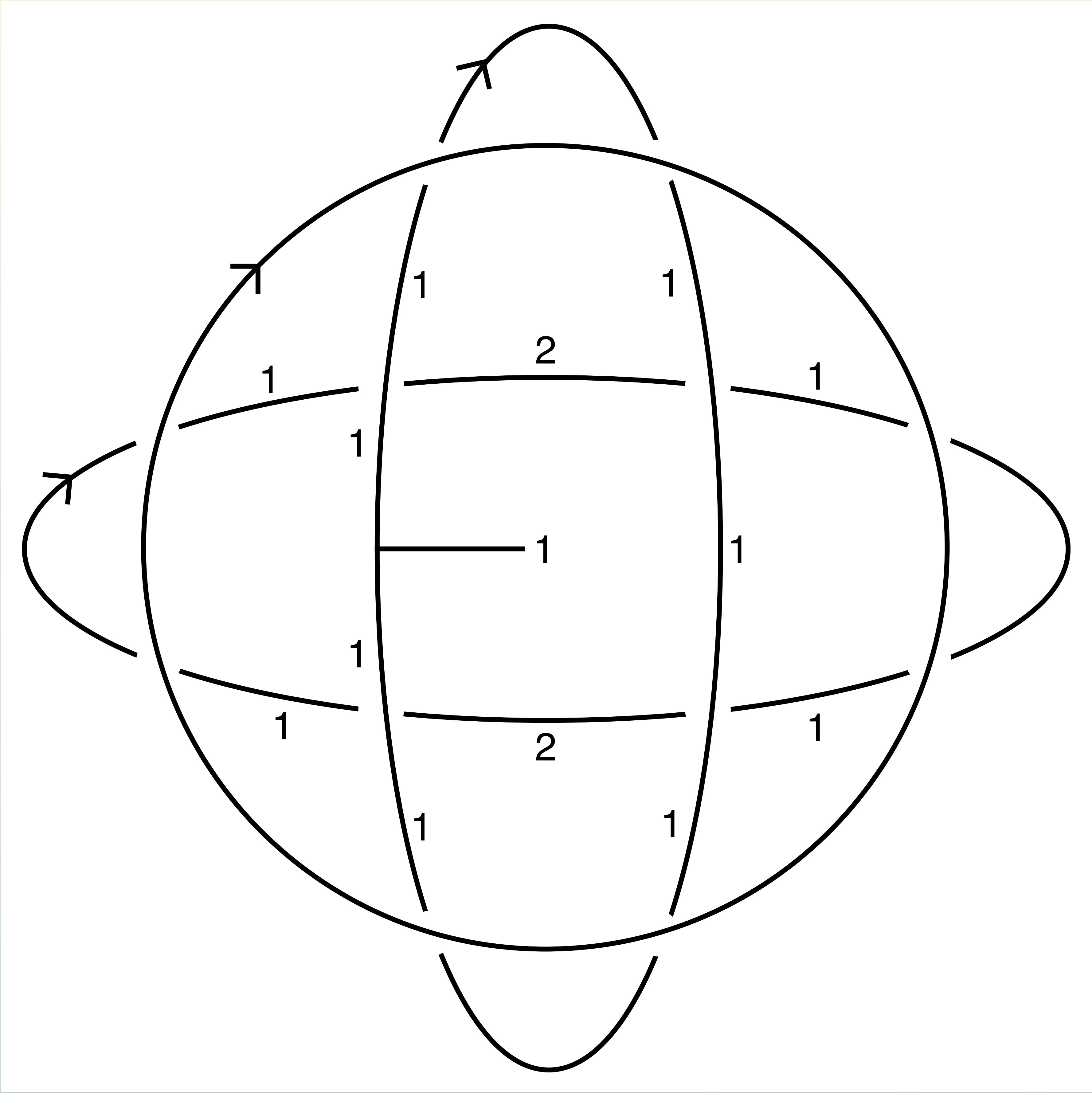
Red is above **green**, which is above blue.



36

Using Slices to Find Region Coverings

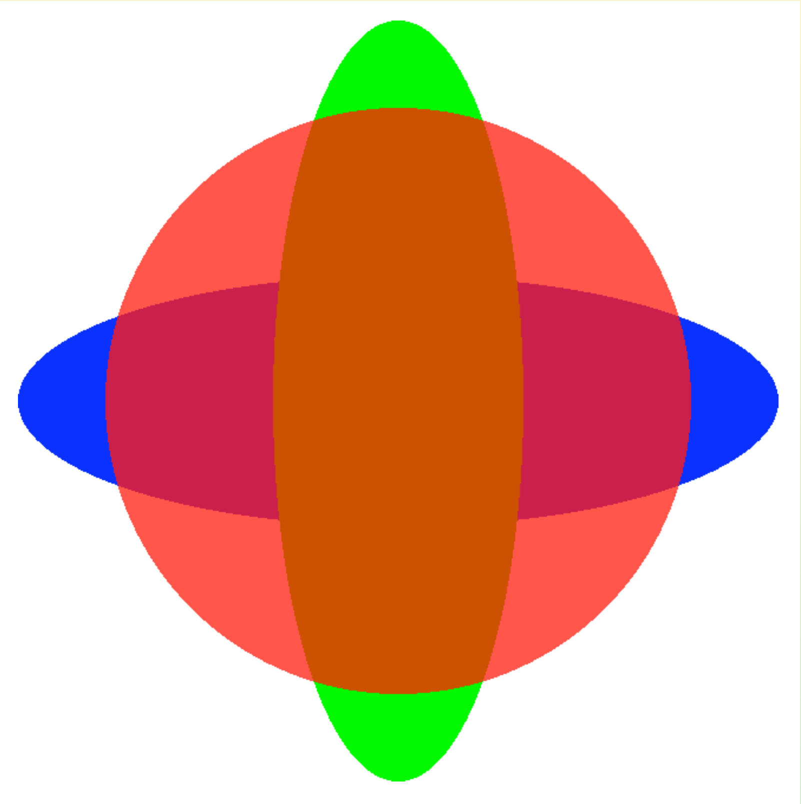


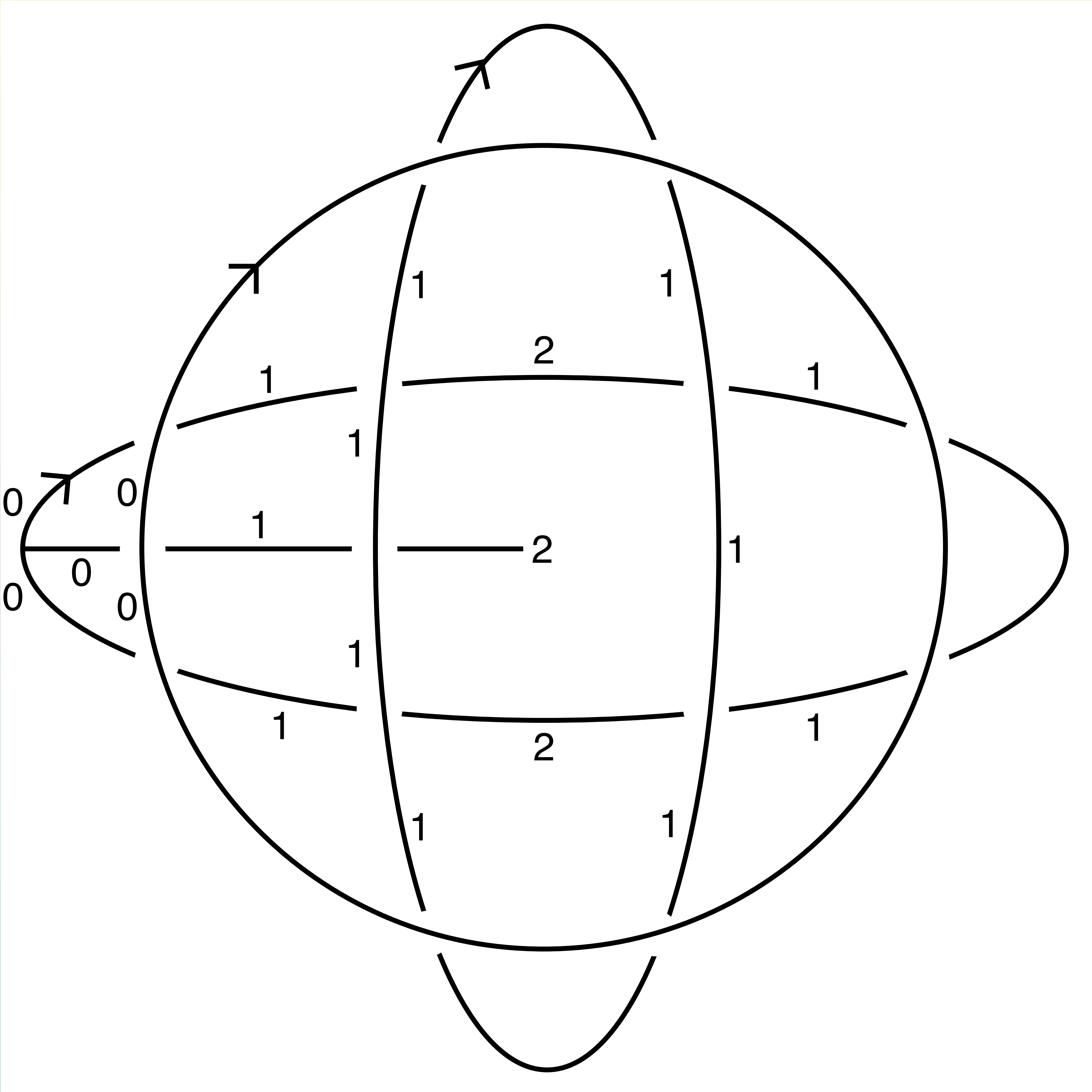


Red is above **green**, which is above blue.

36

Using Slices to Find Region Coverings

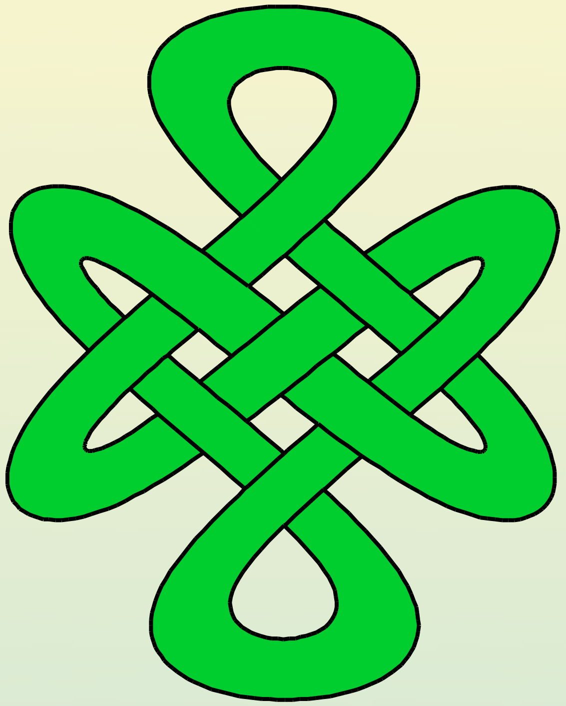


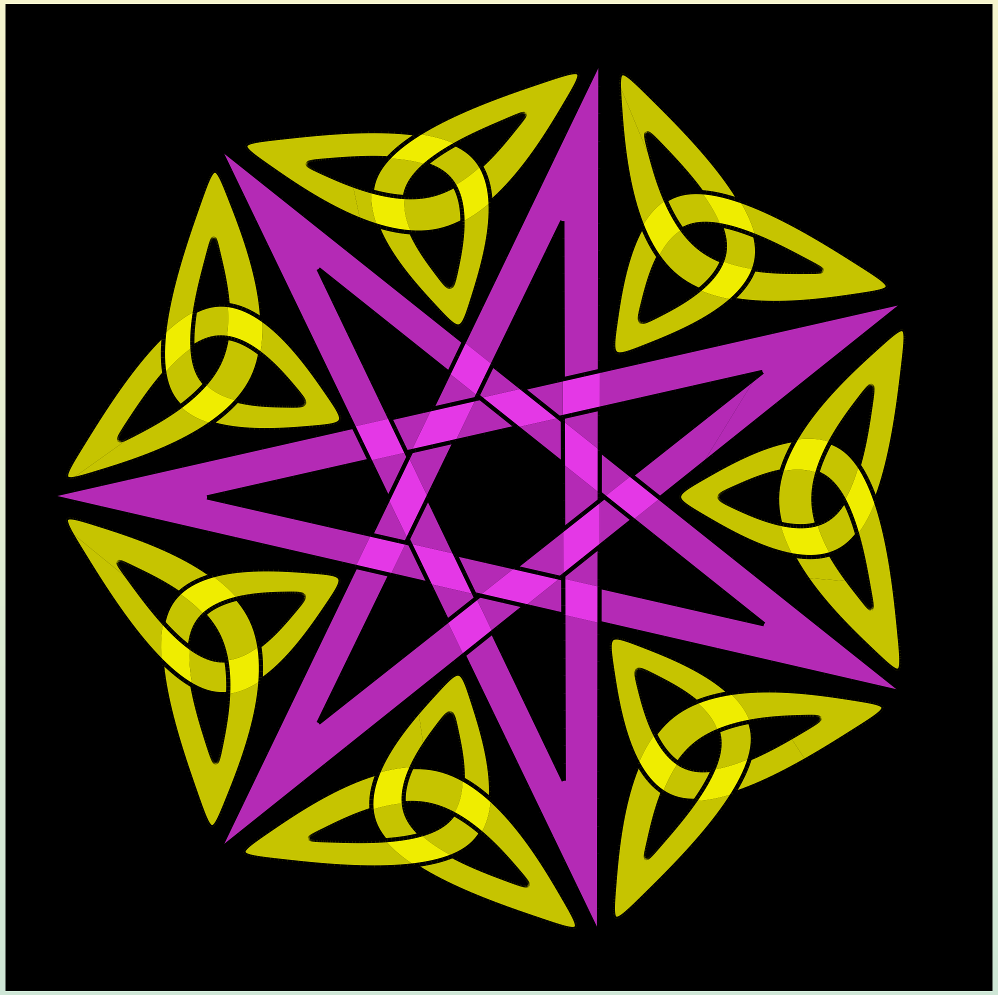


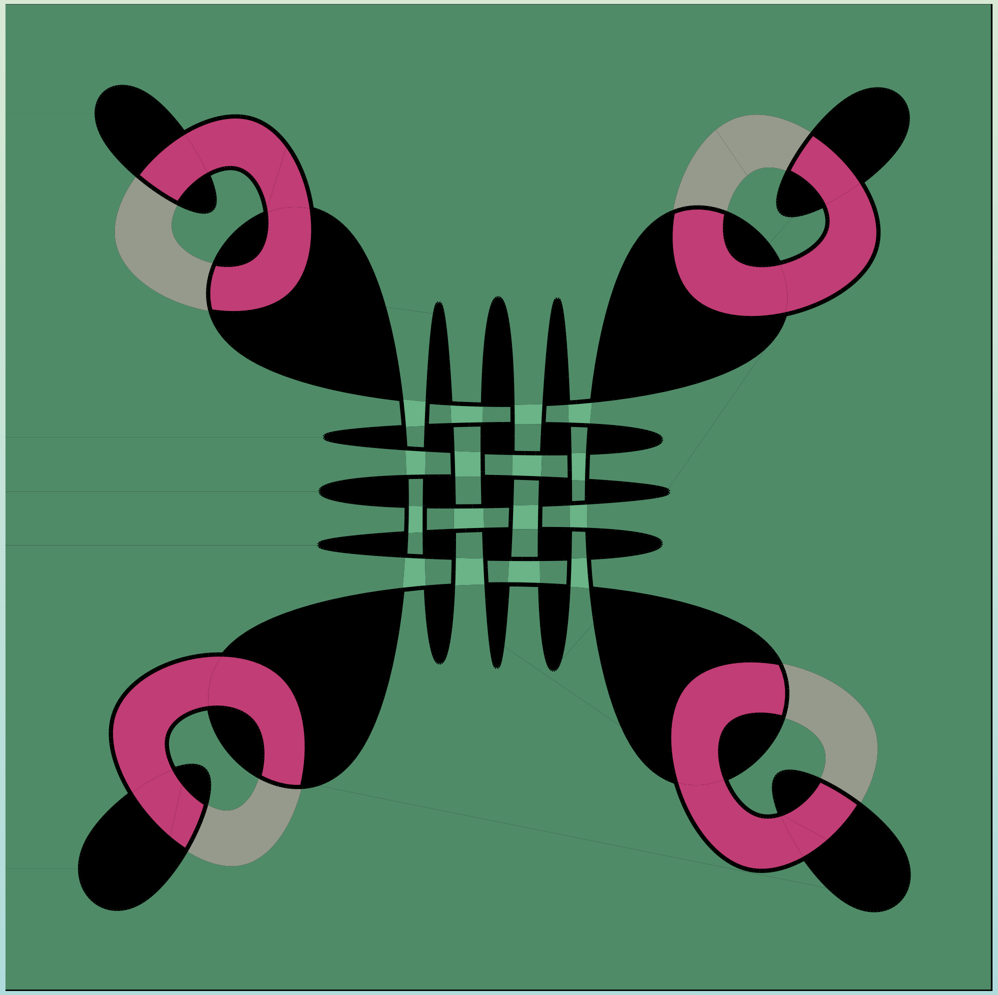
Red is above **green**, which is above blue.

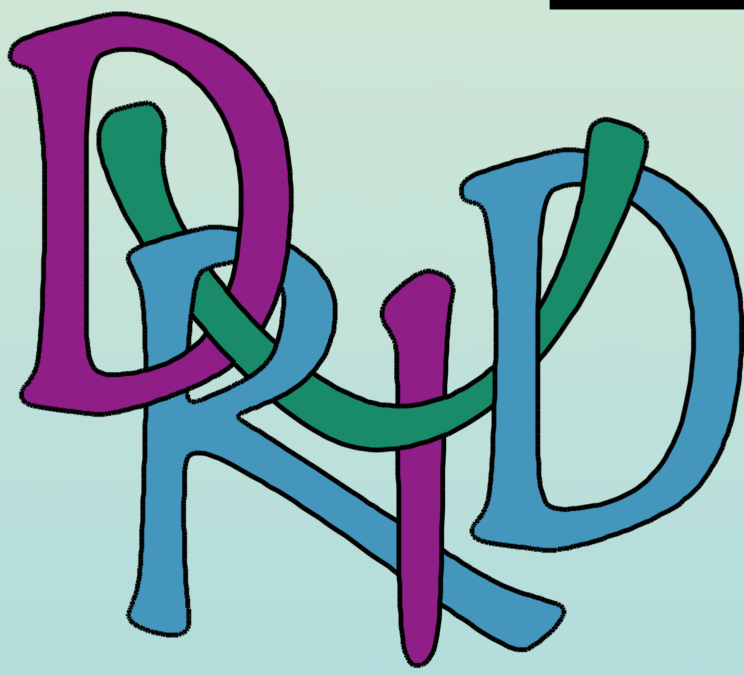
36

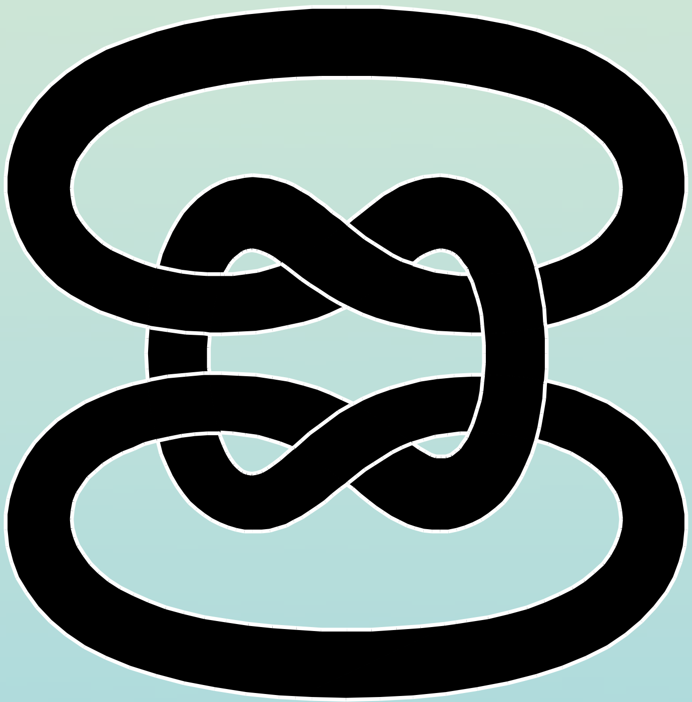
Druid Examples











37

Talk Overview

- Introduction, Current State-of-the-Art
- Druid Description, Usage
- Finding Legal Labelings
- **Crossing-State Equivalence Classes**
- Conclusions

38

A Problem with the Search

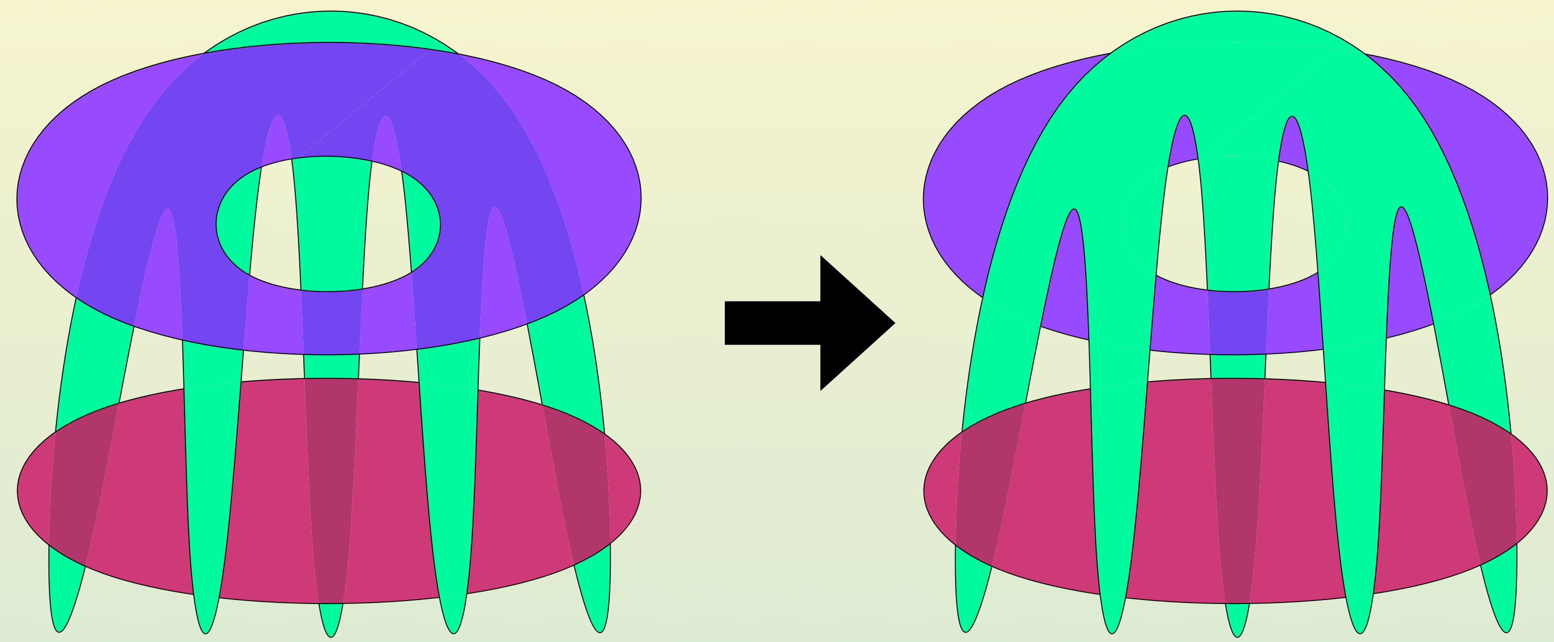
- Search space size: 2^C for *C* crossings
- A drawing can have hundreds of crossings.
- The search takes too long for complex drawings.
- Thus, *Druid* as described in (Wiley and Williams '06a) was limited.

Wiley, K. B., Williams, L., 2006. Representation of Interwoven Surfaces in 2 1/2 D Drawing. *Proc. of CHI*, Conference on Human Factors in Computing Systems, Montreal, Canada, 2006.

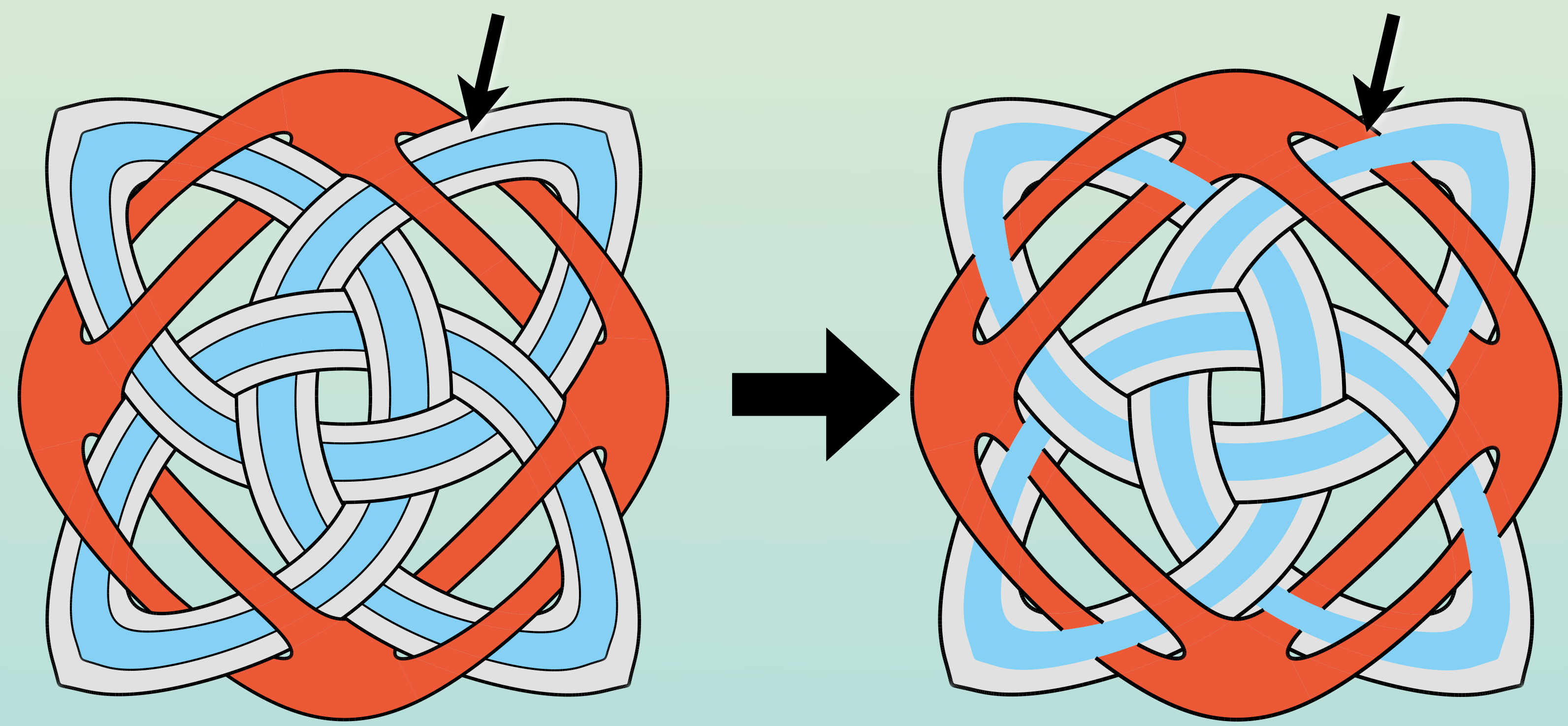
39

A Problem with the Search (contd.)

Druid fails to label this flip in under 120 seconds in 50% of tests



Druid takes 35 seconds on average to perform one of these flips (and fails in 2% of tests)



40

Crossing-State Equivalence Class Rule

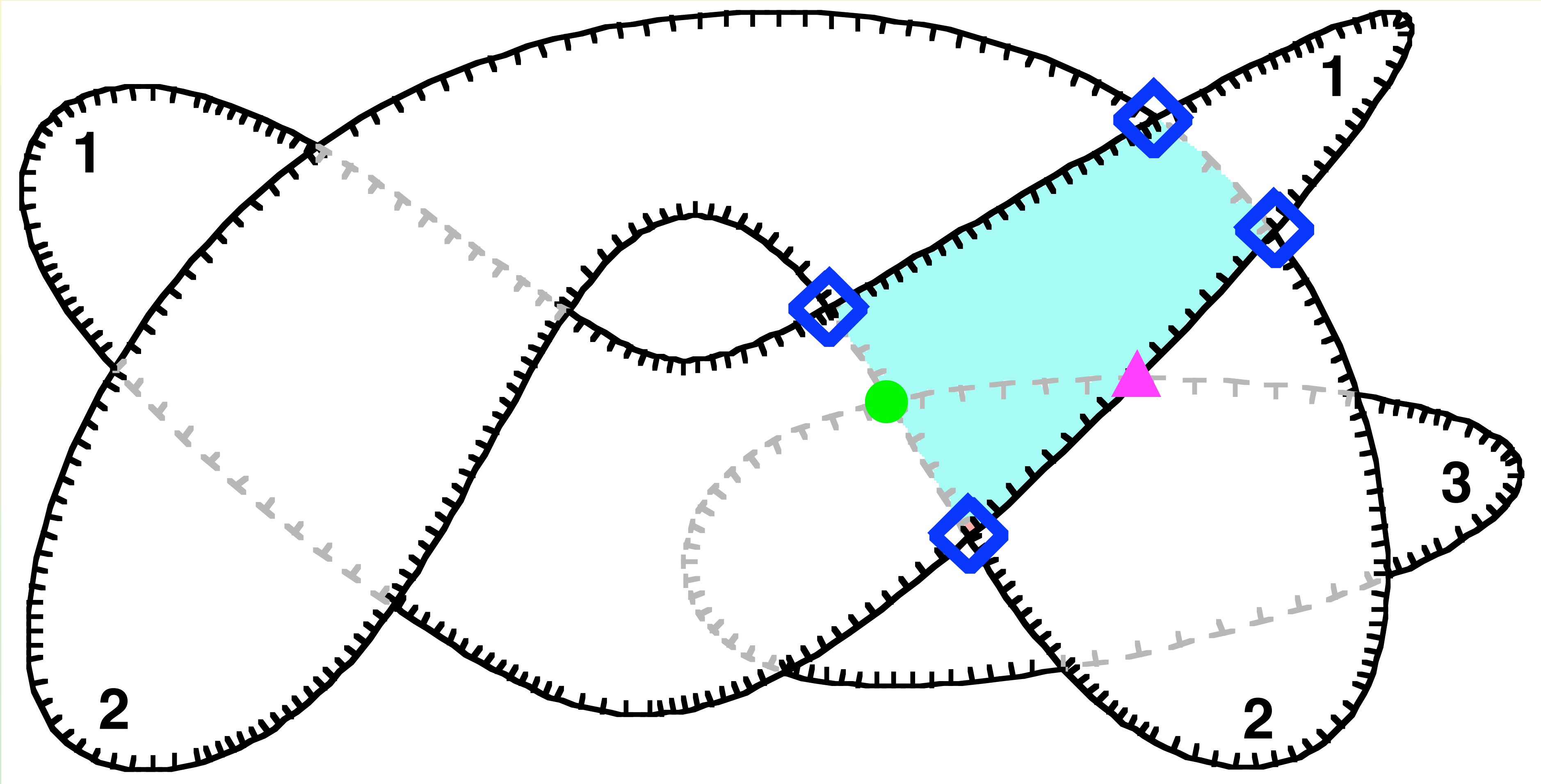
- Discovered a property of 2½D scenes, the ***crossing-state equivalence class (CSEC) rule***.

- Use this property to improve performance.

41

Area of Overlap

Numbers label unique surfaces

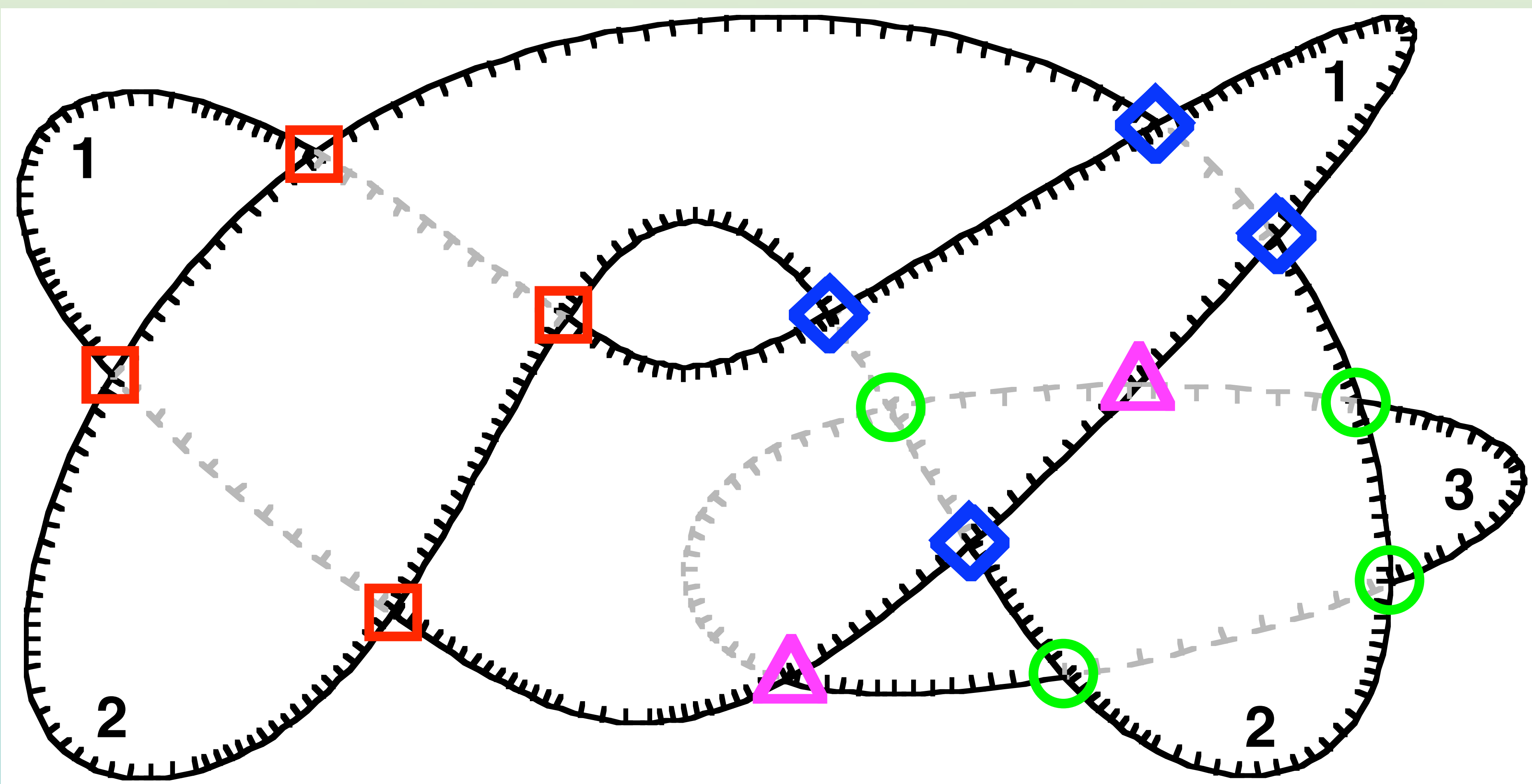


- ***Area-of-overlap***: The maximum contiguous area where two surfaces overlap, *e.g.*, the shaded area for surfaces *1* and *2*
- ***Corner***: A crossing where a traversal of an *area-of-overlap's* border switches boundaries, *e.g.*, the blue diamonds for the shaded area

42

Crossing-State Equivalence Class (CSEC)

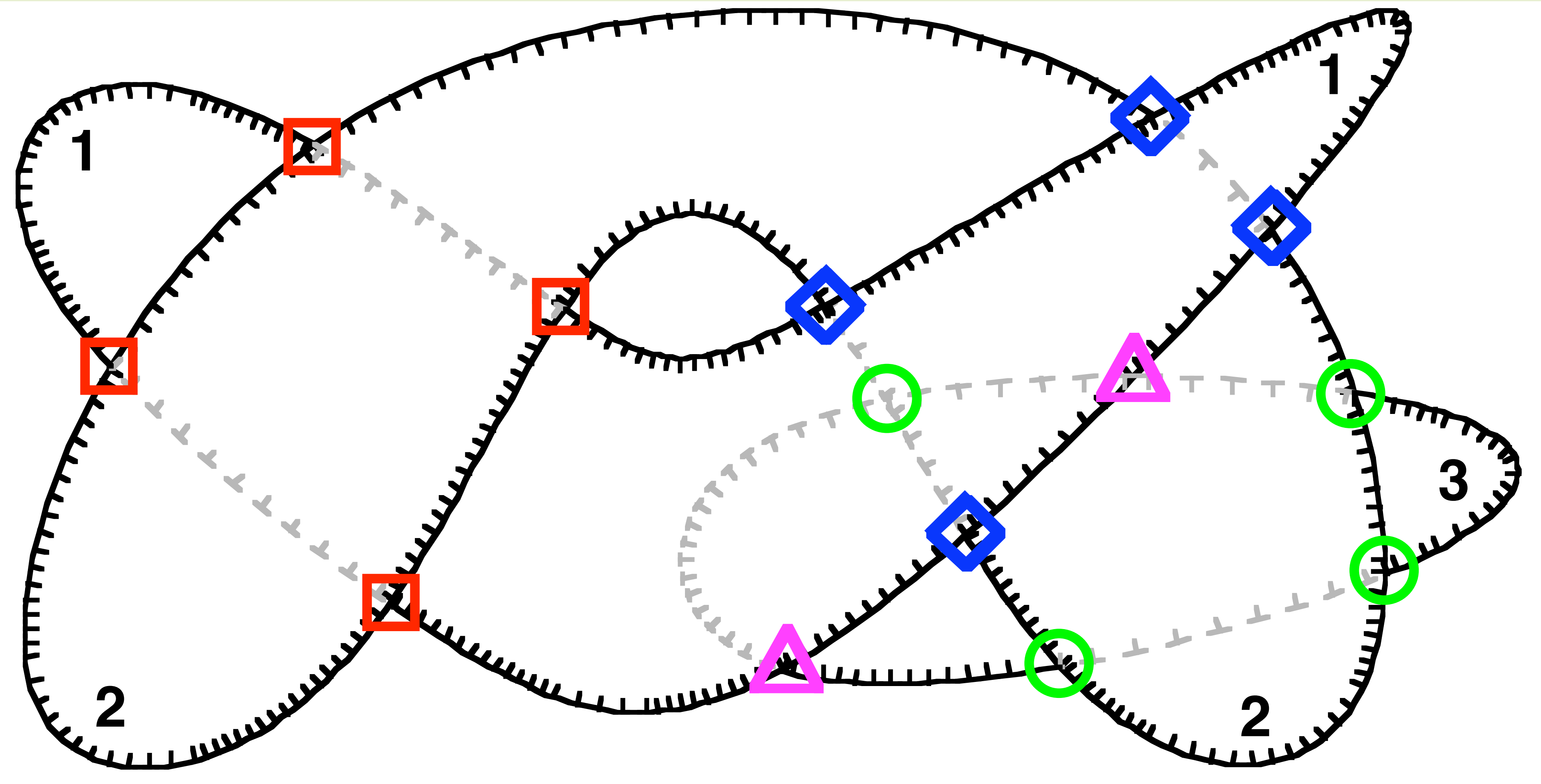
The *corners* of an *area-of-overlap* comprise a *CSEC*.



Unique shapes/colors indicate CSECs

Crossing-State Equivalence Class Rule

All members of a crossing-state equivalence class must be in the same state.

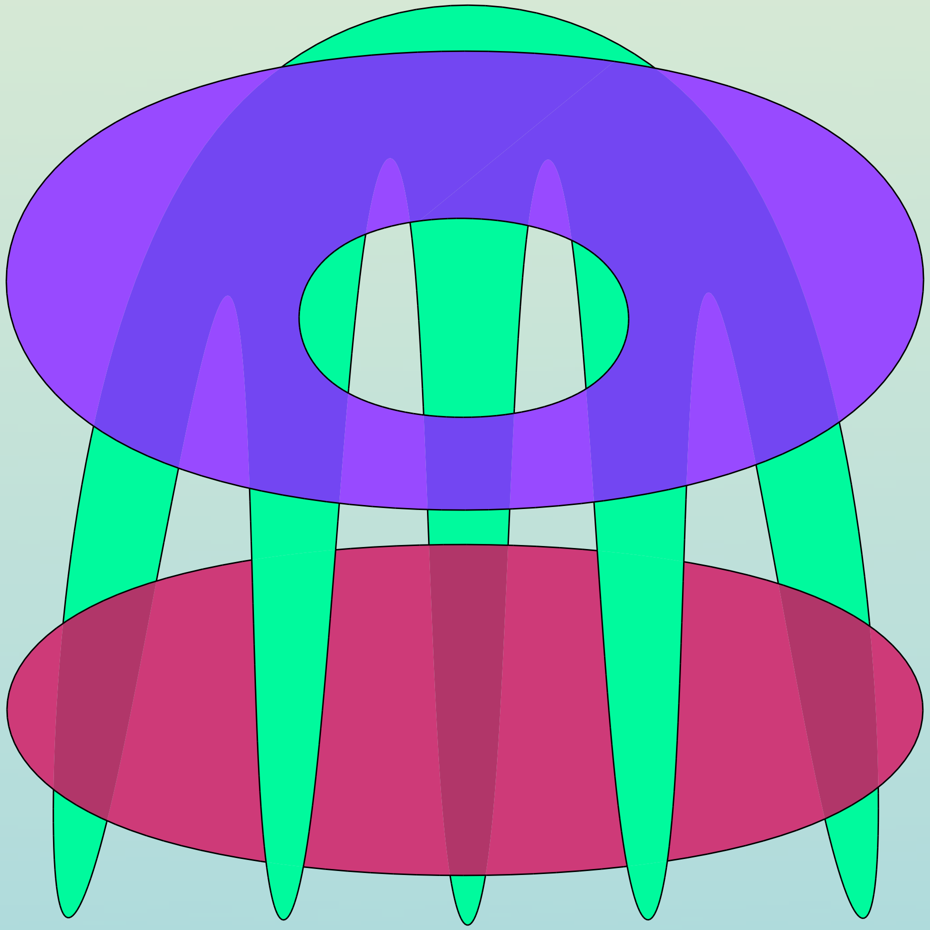


e.g., for surfaces *2* and *3* all corners of the green circle CSEC must be in the same state, *i.e.*, either *2* is above *3 or vs/va*.

44

Labeling with CSECs

- CSECs have a profound effect on the search space size.
- e.g., this drawing has 40 crossings but only 7 CSECs, an improvement by a factor of 2^{33}, or 8.5 billion.



CSECs Used	Labeling space size
No	2^{40} (for 40 crossings)
Yes	2^{7} (for 7 CSECs)

Relabeling with CSECs

1. *Druid (OLD)*: (Wiley and Williams '06a)

    - ***Labeling*** and ***relabeling*** both perform a tree search of size 2^C (C = num crossings).

2. *Druid (NEW)*: (Wiley and Williams '06b)

    - ***Relabeling*** performed by maintaining the CSECs ***without a search***. Segment depth changes are directly deduced.

    - ***Labeling*** searches a space of size 2^E (E = num CSECs).

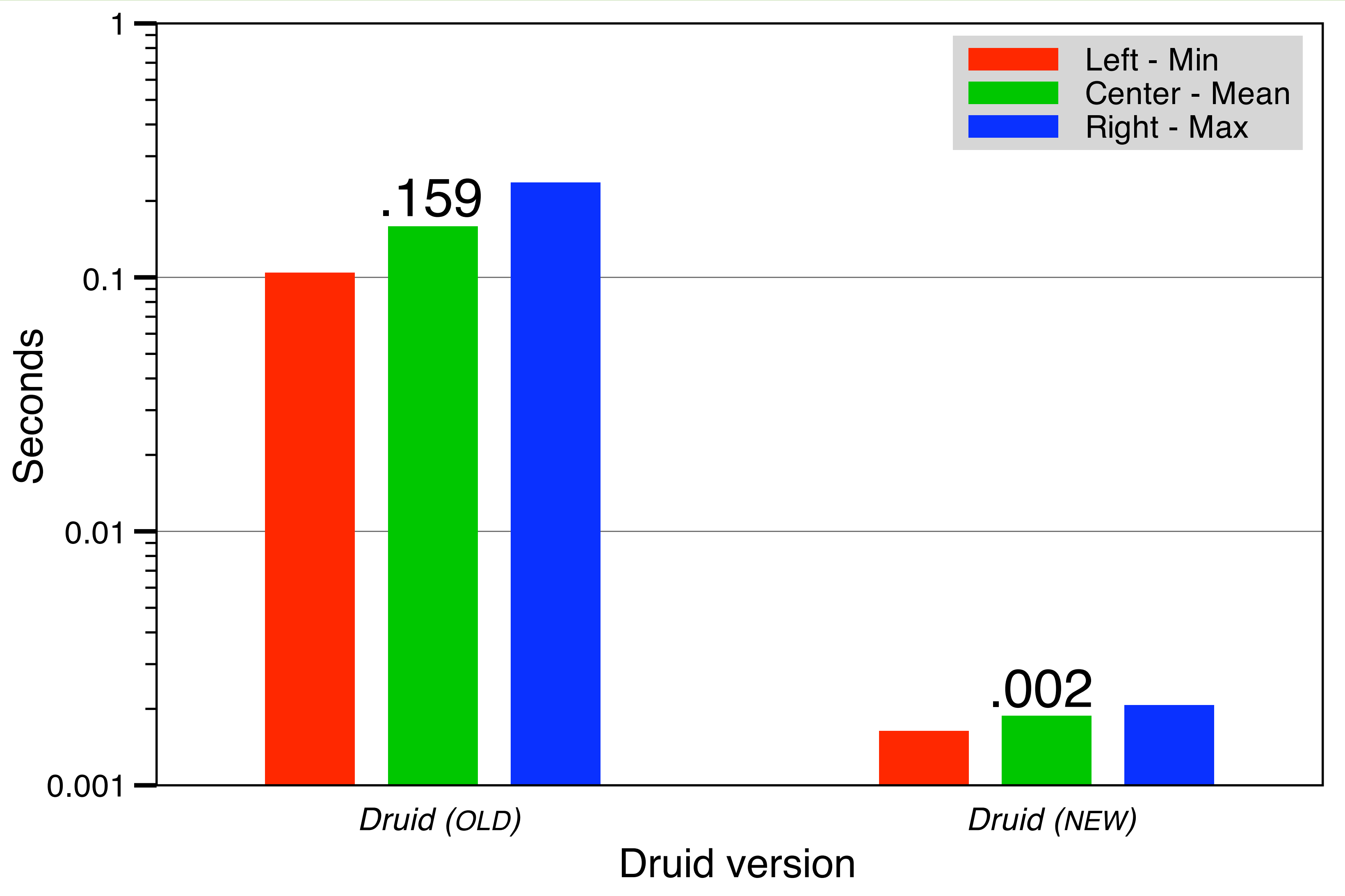
Wiley, K. B., and L. R. Williams, 2006. Representation of Interwoven Surfaces in 2 1/2 D Drawing. *Proc. of CHI*, Conference on Human Factors in Computing Systems, Montreal, Canada, 2006.

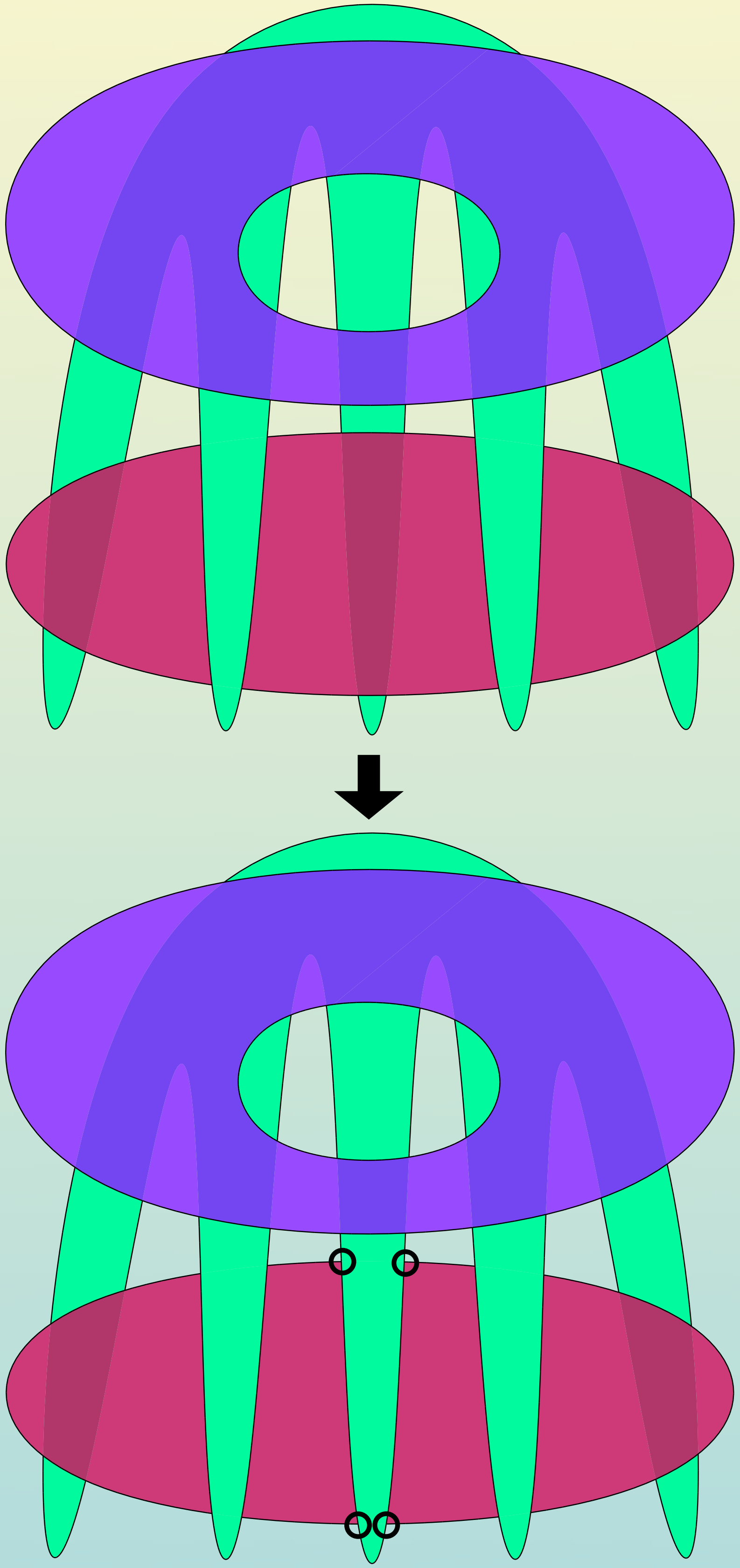
Wiley, K. B., and L. R. Williams. Use of Crossing-State Equivalence Classes for Rapid Relabeling of Knot-Diagrams Representing 2 1/2 D Scenes. Tech Report, UNM, Dept of Computer Science, TR-CS-2006-08, 2006.

46

Relabeling Results: A Small CSEC Flip

- Size 4, indicated with circles
- Running times on 1.6GHz G5 PowerMac
- *Druid* (NEW) performs ~100 times faster than *Druid* (OLD)



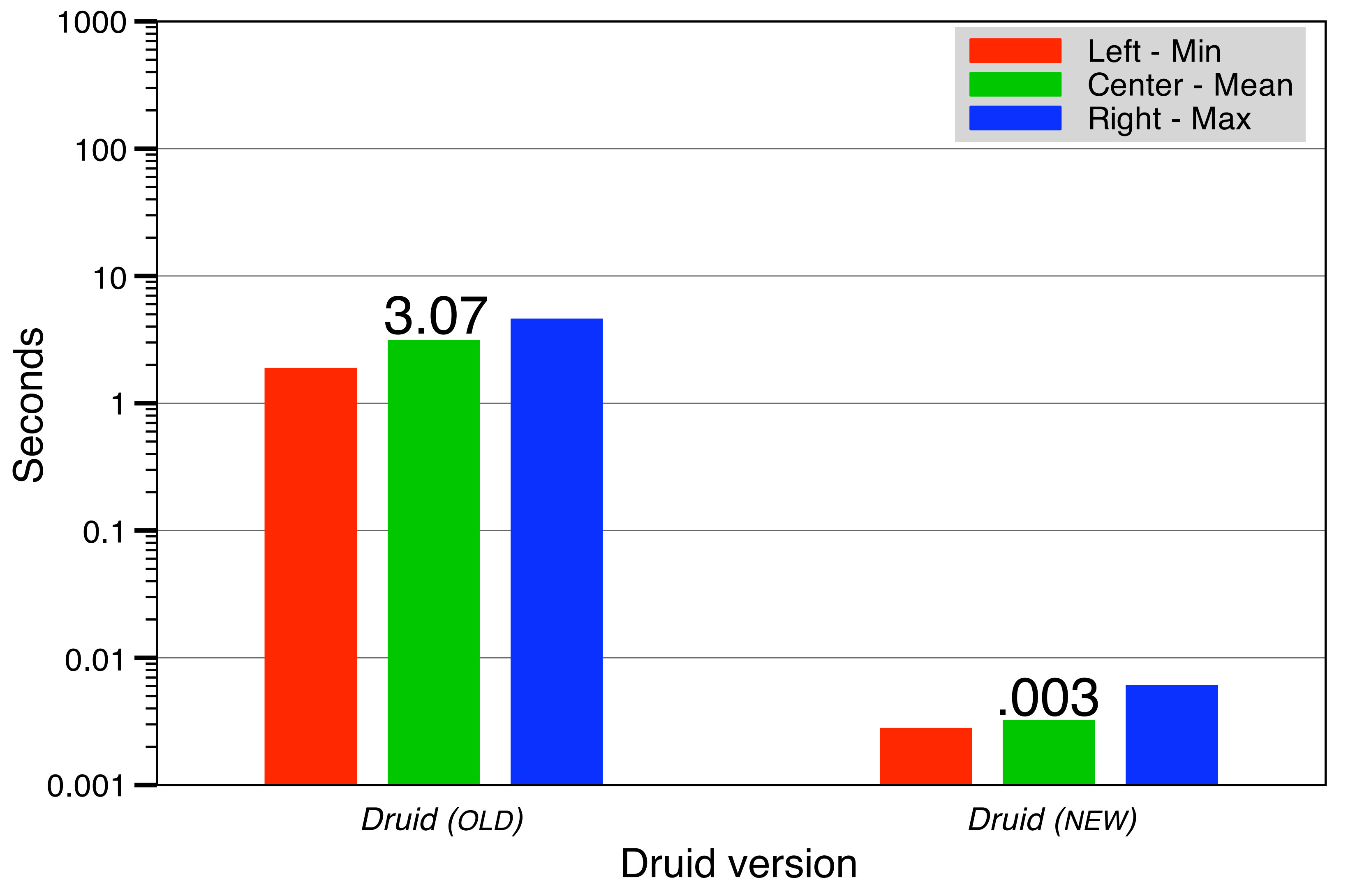


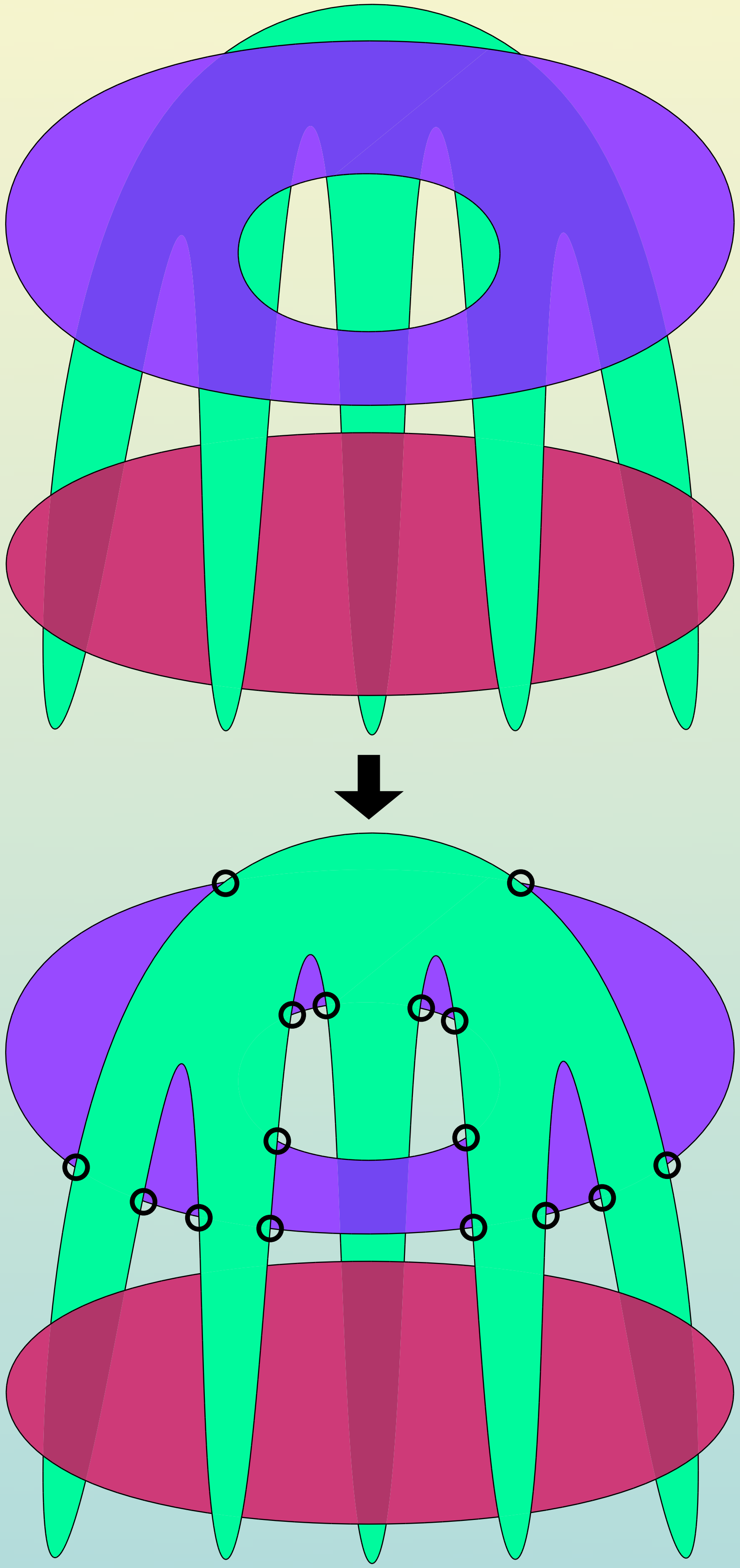
Min, mean, max with respect to a crossing-flip performed independently on each corner

47

Relabeling Results: A Large CSEC Flip

- Size 16, indicated with circles
- *Druid* (OLD) cannot relabel in a reasonable time.
- *Druid* (NEW) performs ~1000 times faster.
- Note: *Druid* (OLD) failed 50% of the time.

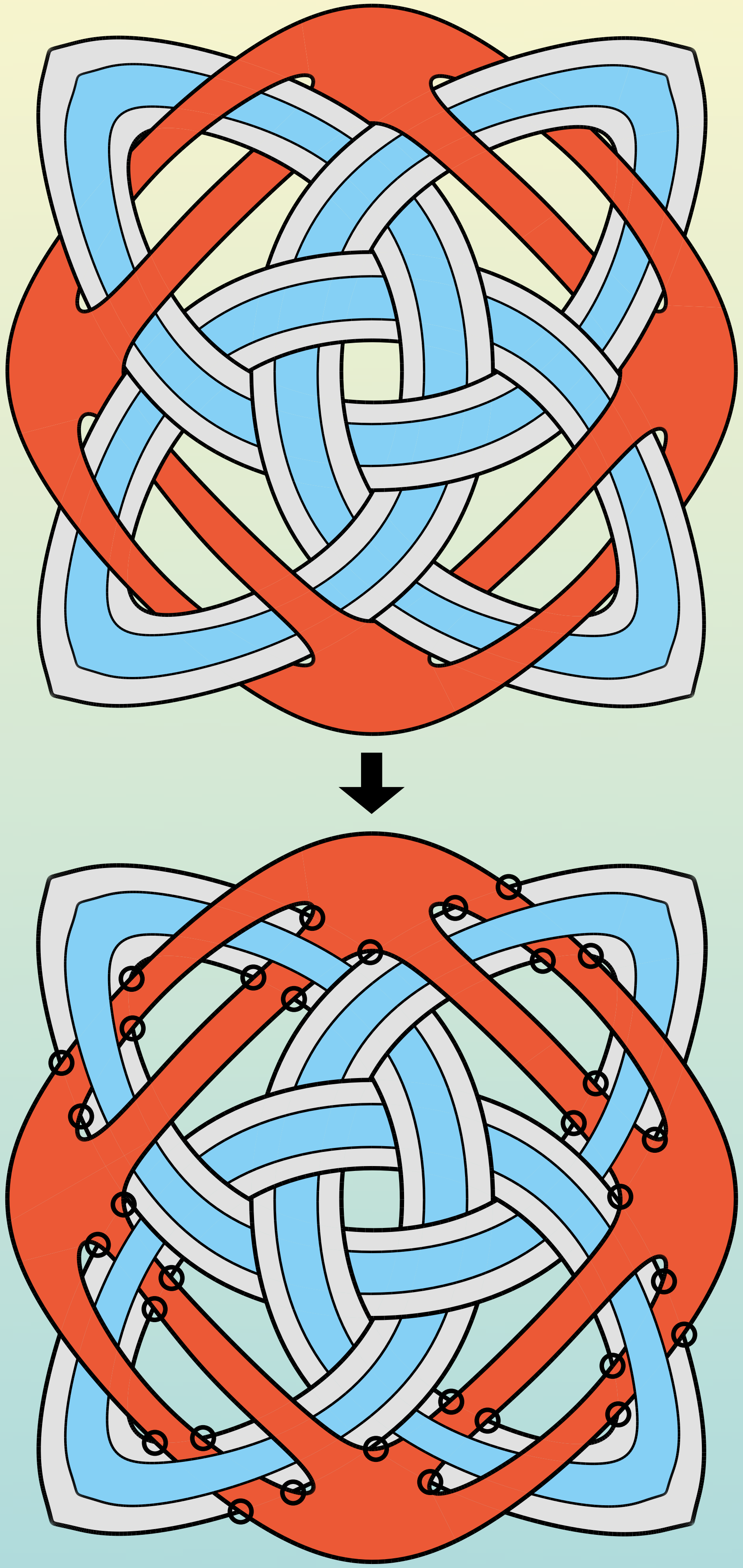




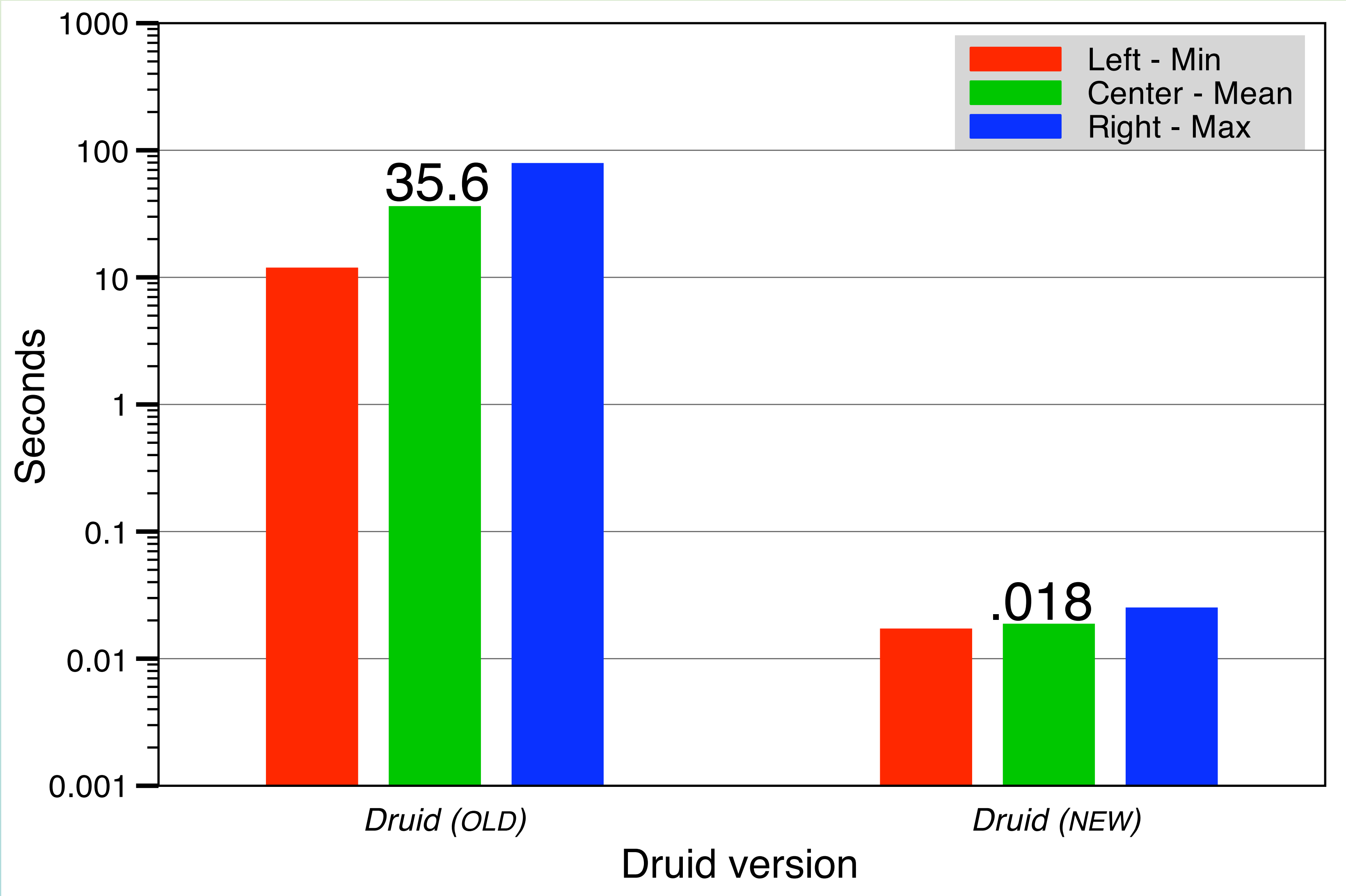
Min, mean, max with respect to a crossing-flip performed independently on each corner

48

Relabeling Results: A Complex Figure

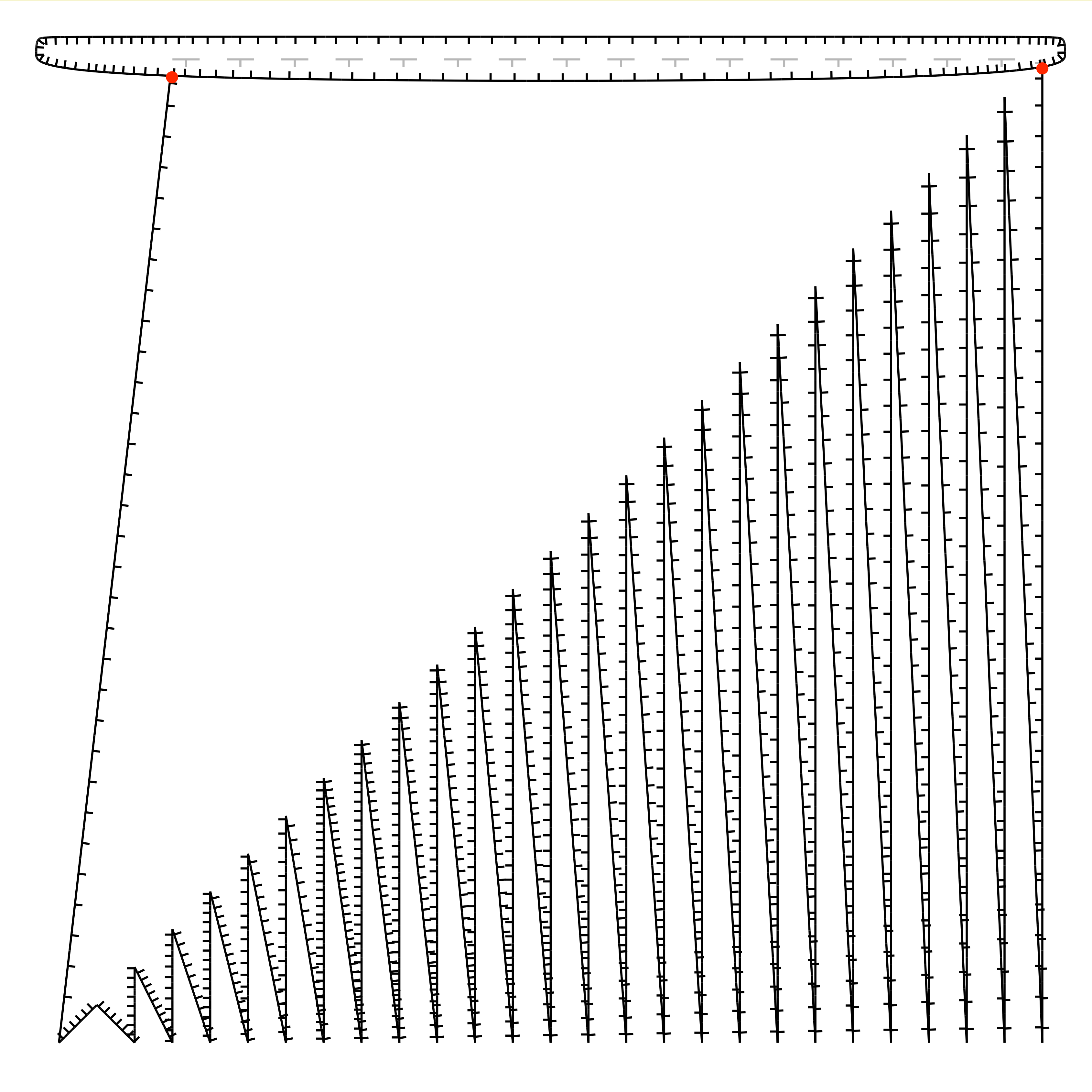


- 256 crossings, 64 CSECs
- *Druid* (OLD) cannot relabel this small CSEC flip in a reasonable time.
- *Druid* (NEW) relabels in .02 seconds, ~2000 times faster.
- Note: *Druid* (OLD) failed 2% of the time.



49

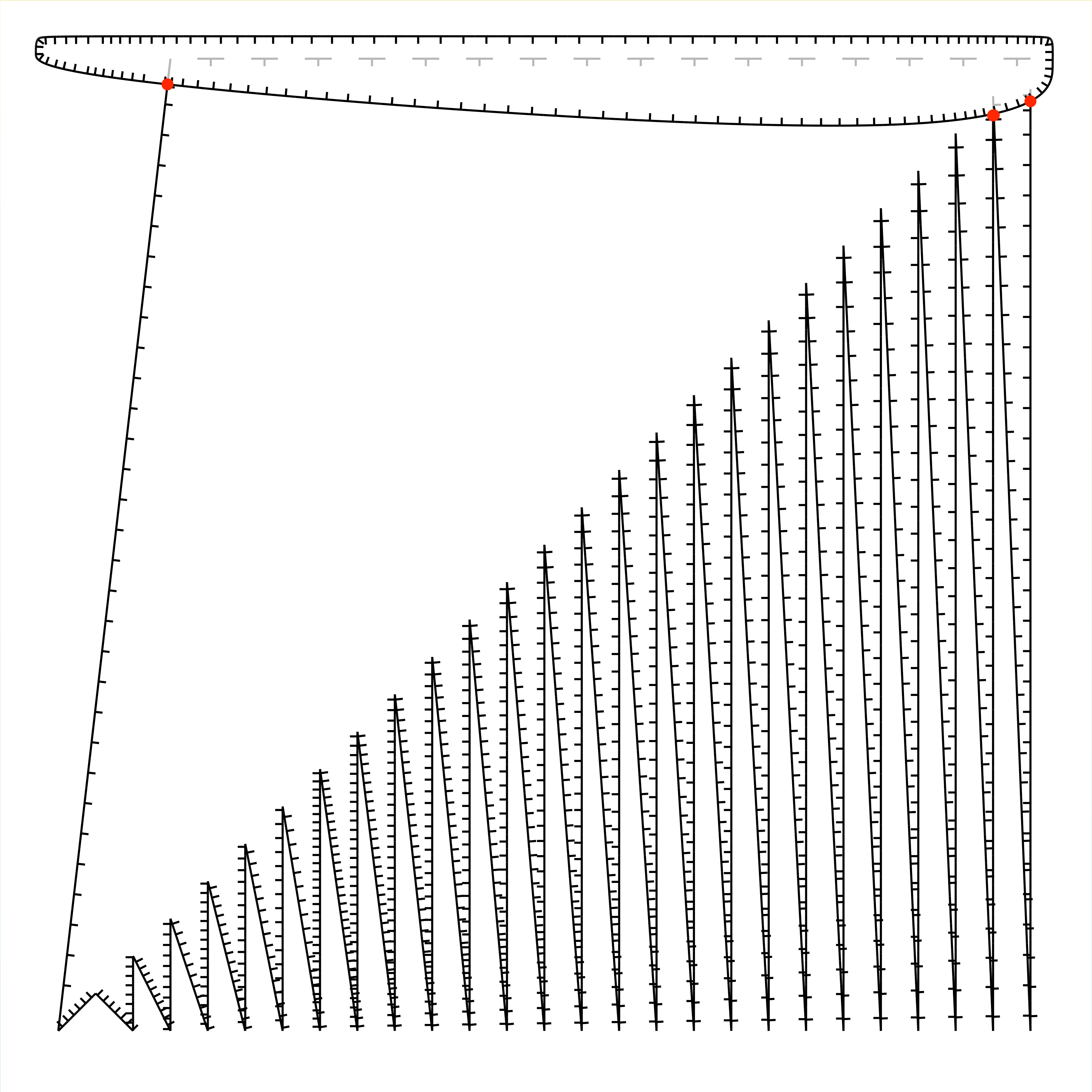
CSEC Flip Performance



Flipped CSEC size: linear in the total number of crossings

50

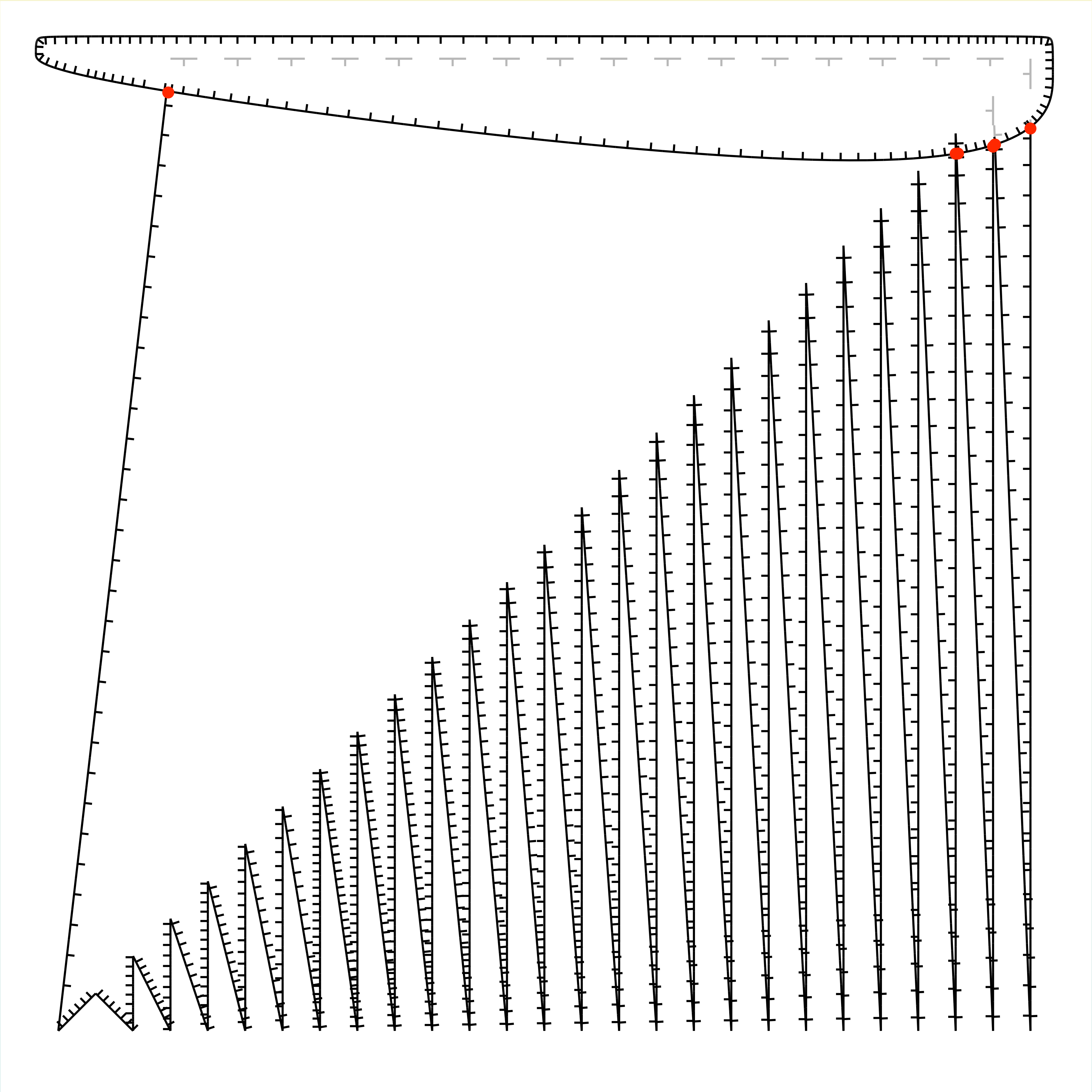
CSEC Flip Performance



Flipped CSEC size: linear in the total number of crossings

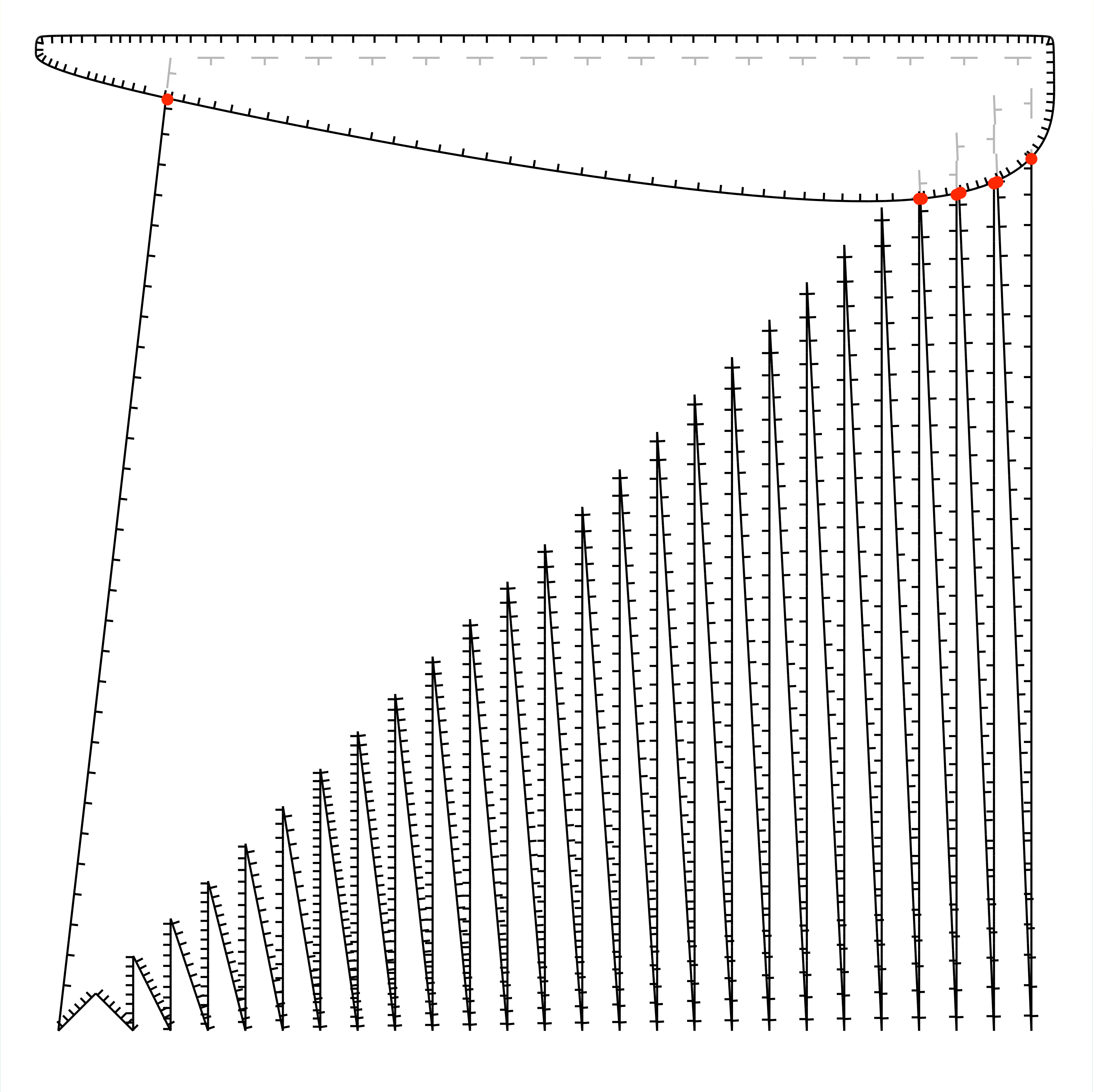
50

CSEC Flip Performance



Flipped CSEC size: linear in the total number of crossings

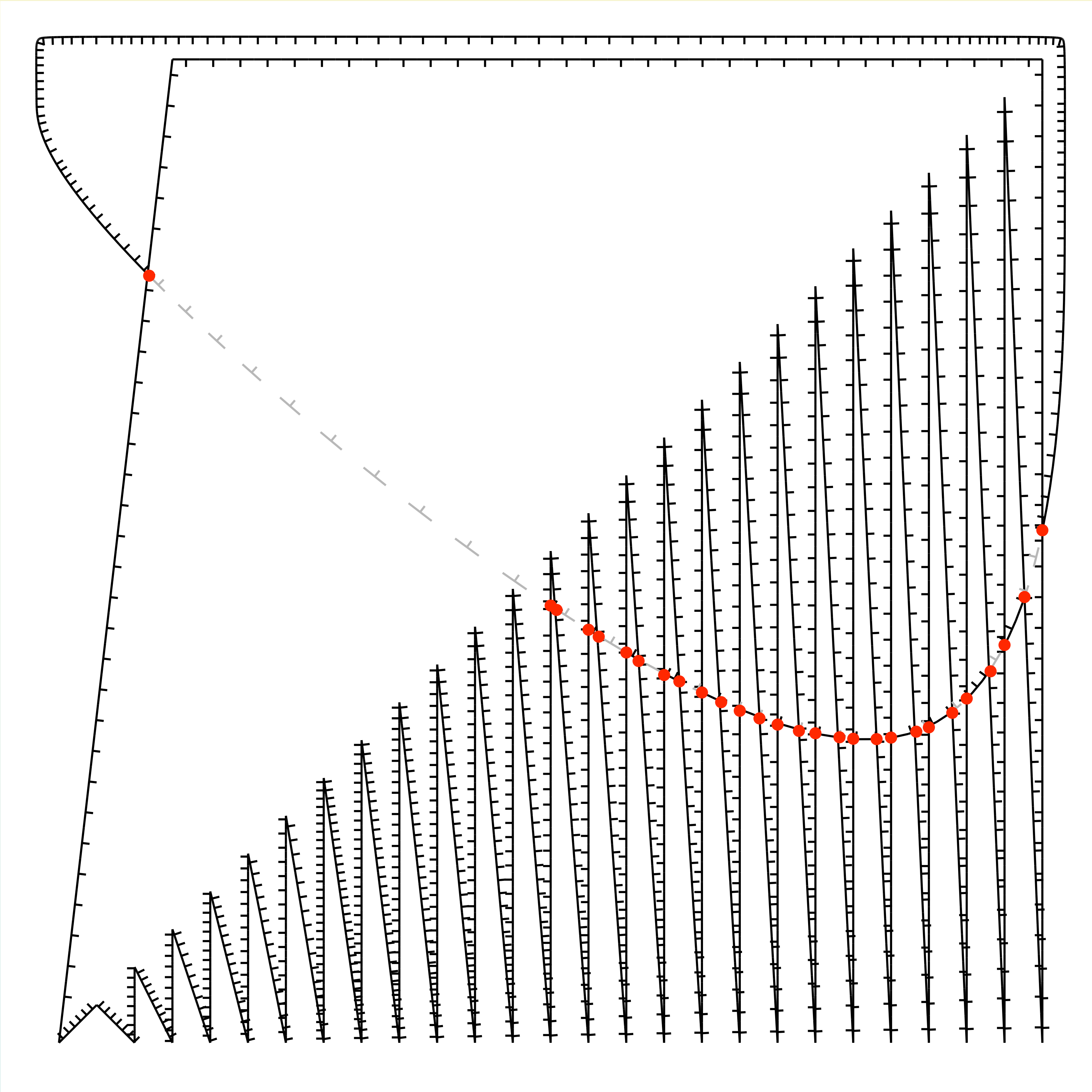
CSEC Flip Performance



Flipped CSEC size: linear in the total number of crossings

50

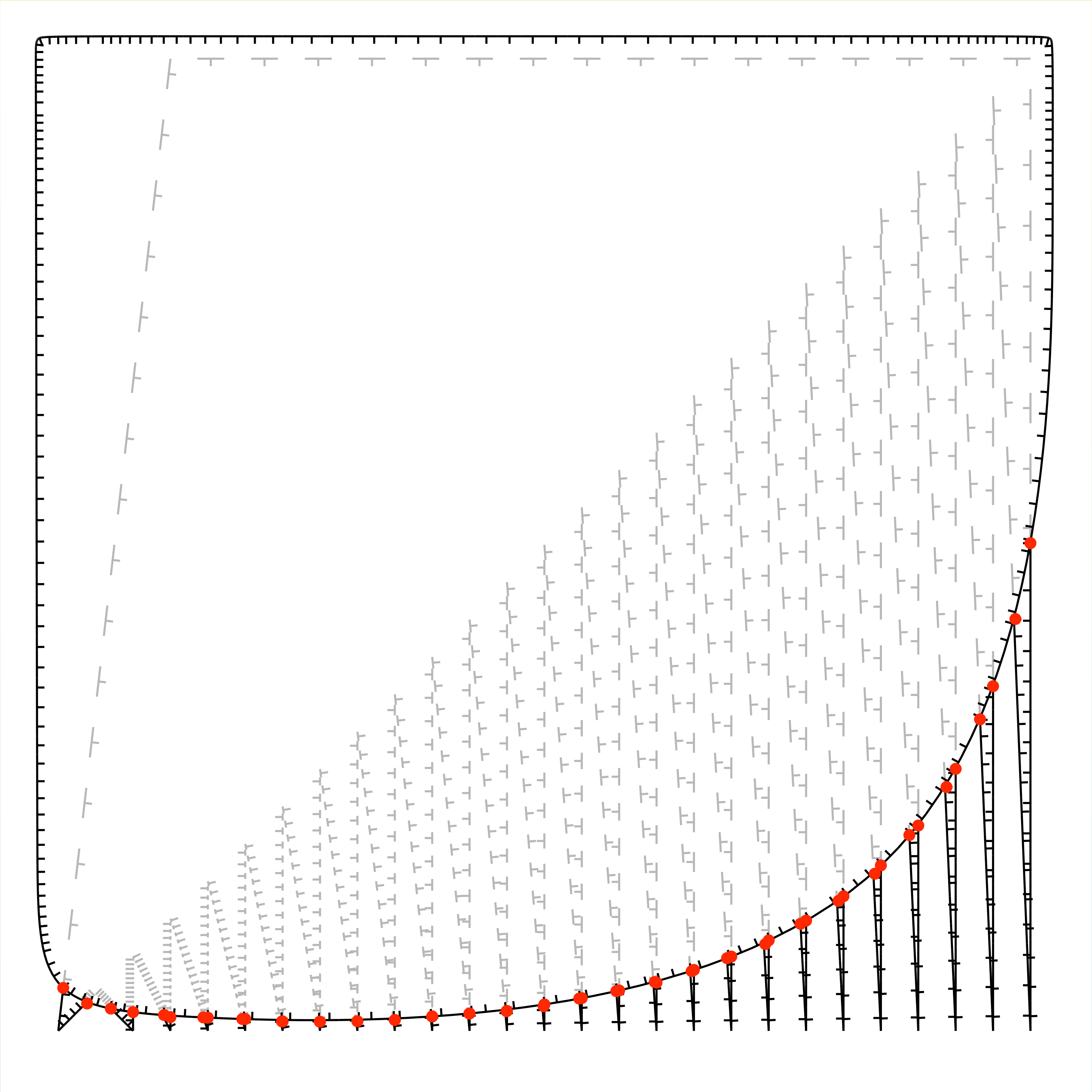
CSEC Flip Performance



Flipped CSEC size: linear in the total number of crossings

50

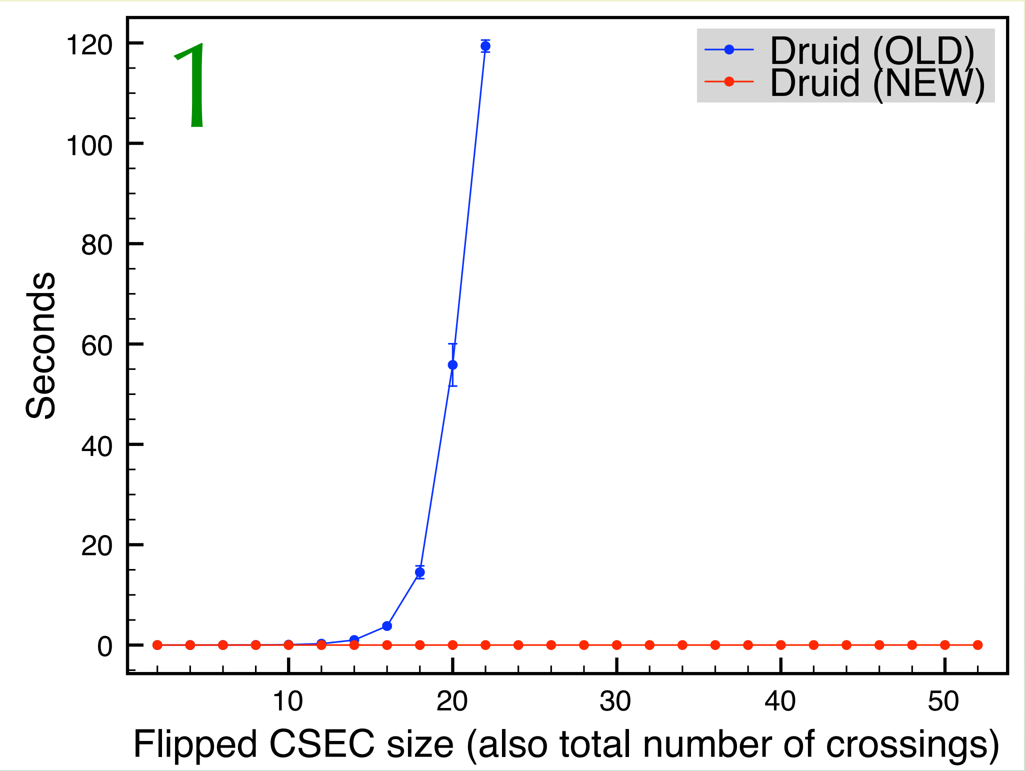
CSEC Flip Performance

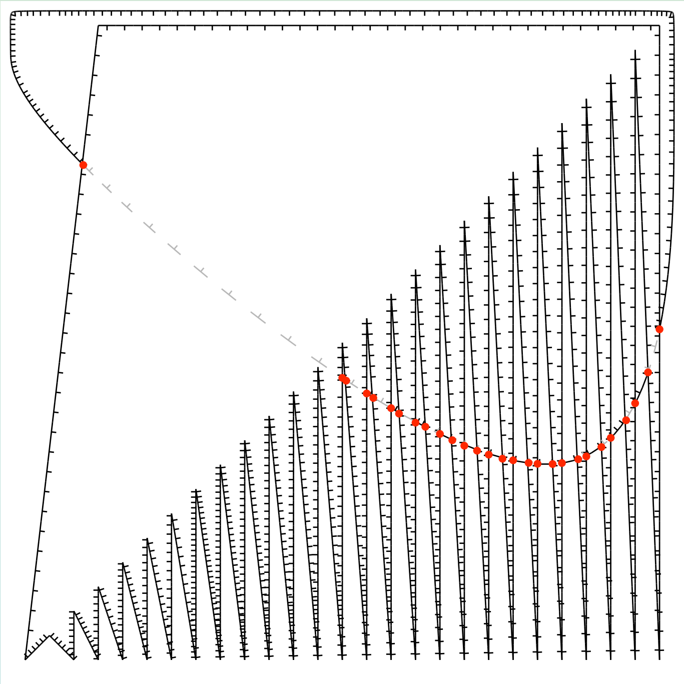


Flipped CSEC size: linear in the total number of crossings

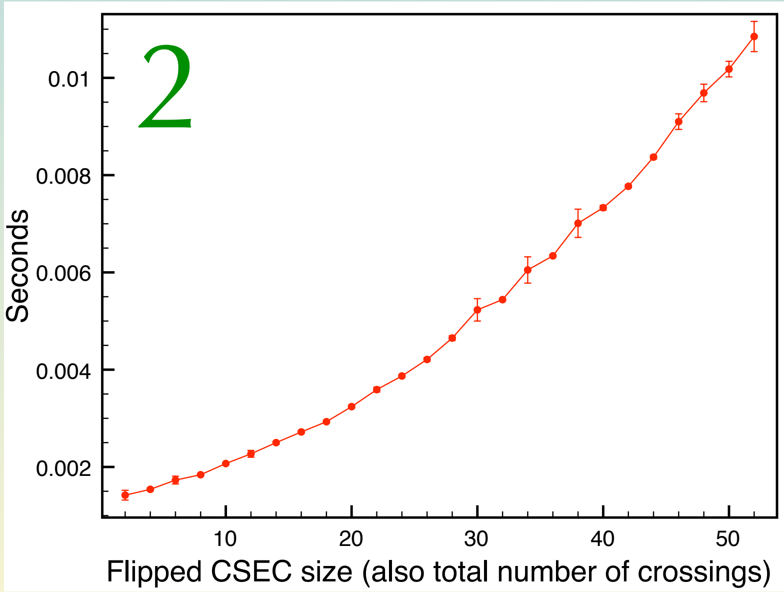
CSEC Flip Performance

Running time vs. CSEC size

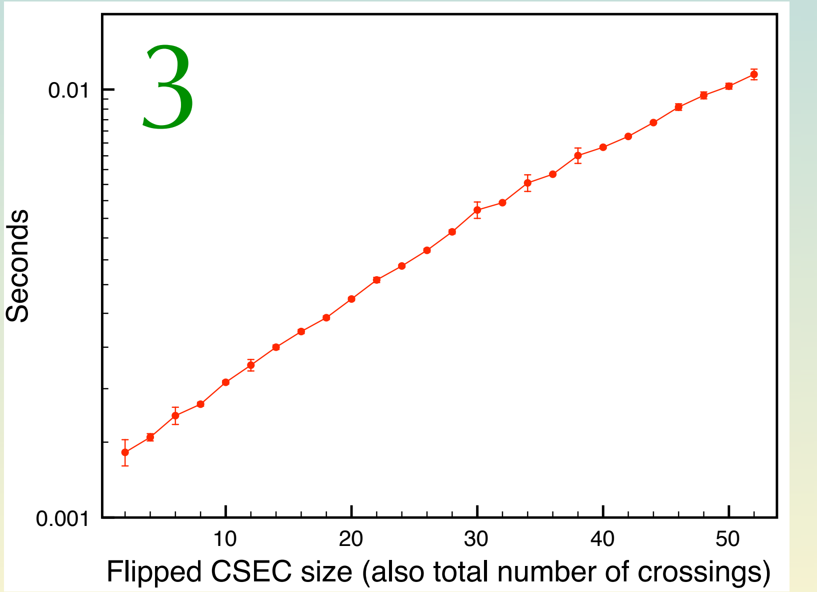




CSEC Flip Performance (*Druid* (NEW))

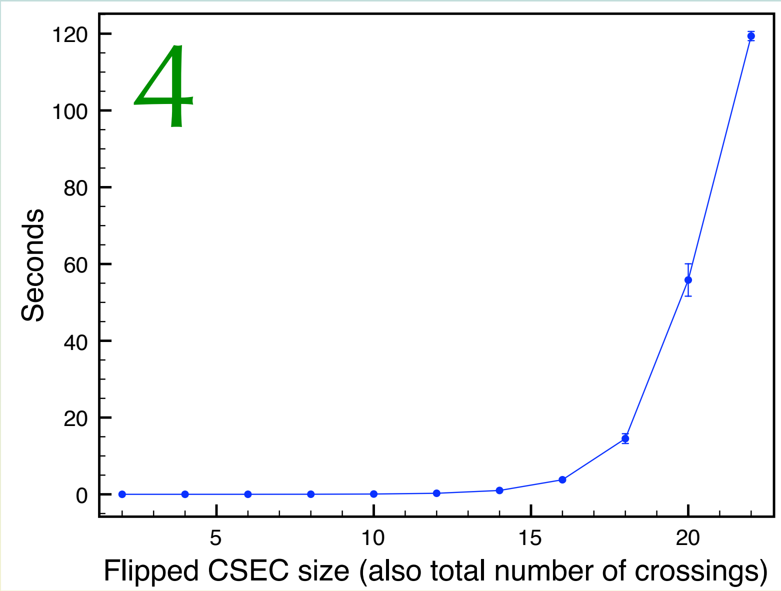


CSEC Flip Performance (*Druid* (NEW))
(log Y axis)

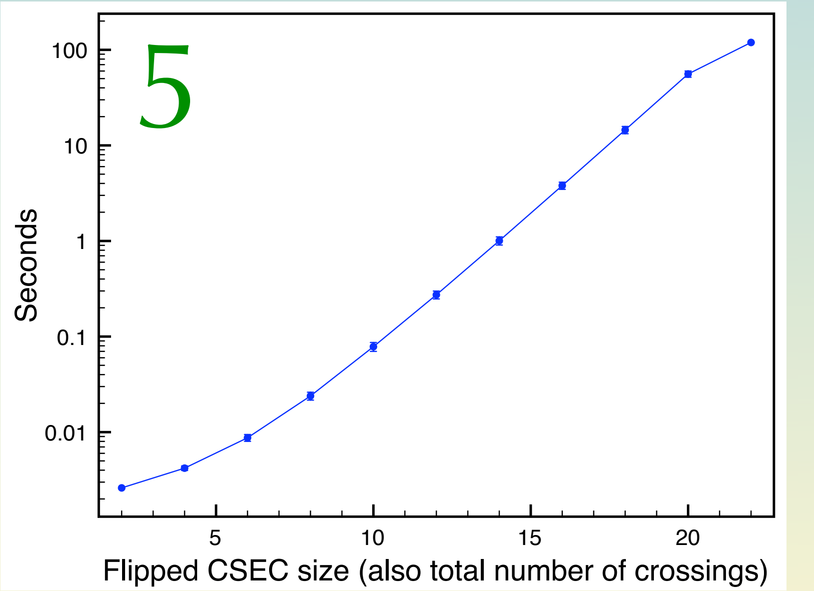


Performance is polynomial w.r.t. CSEC size

Search Performance (*Druid* (OLD))



Search Performance (*Druid* (OLD))
(log Y axis)

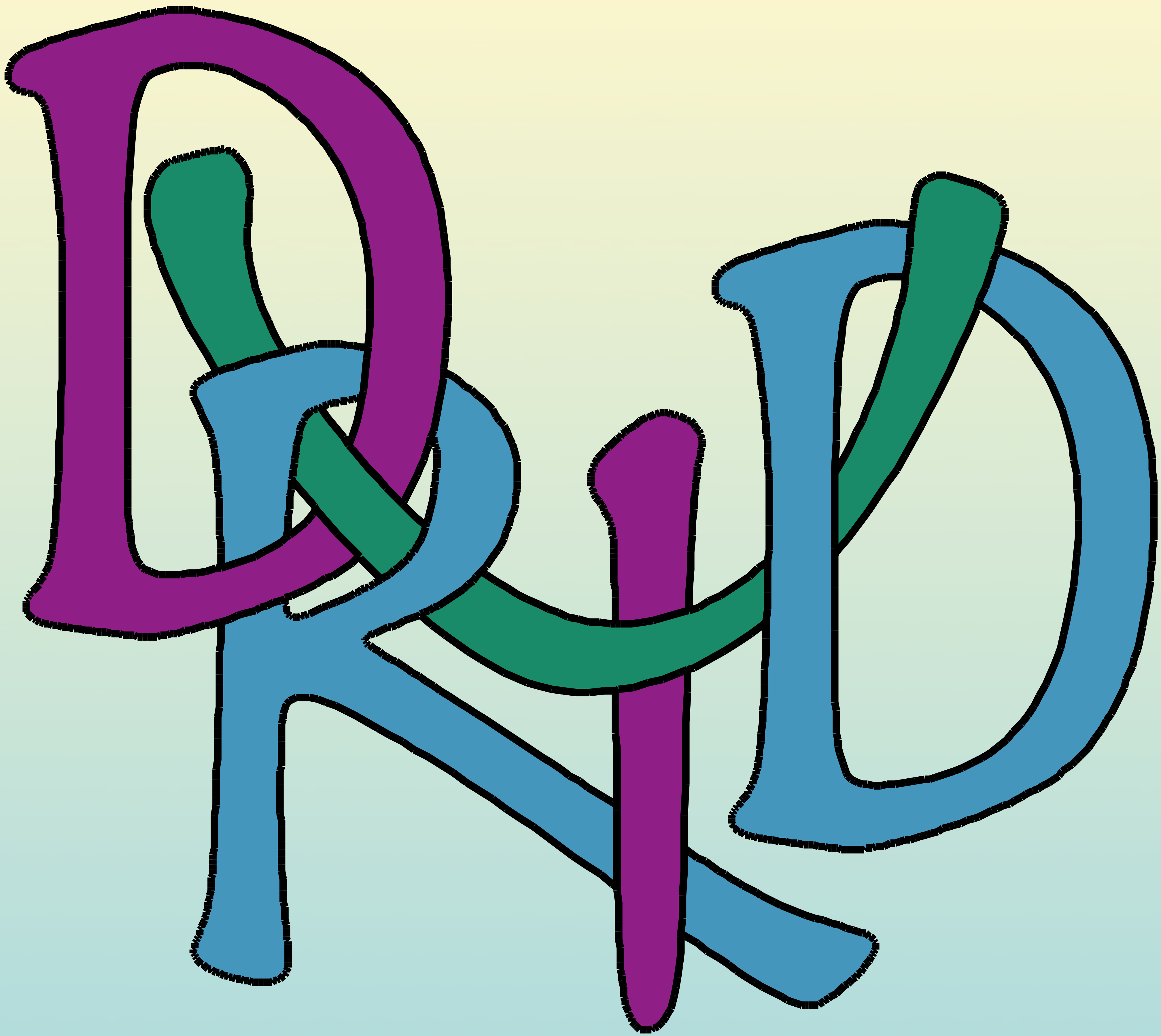


Performance is exponential w.r.t. CSEC size

Conclusions

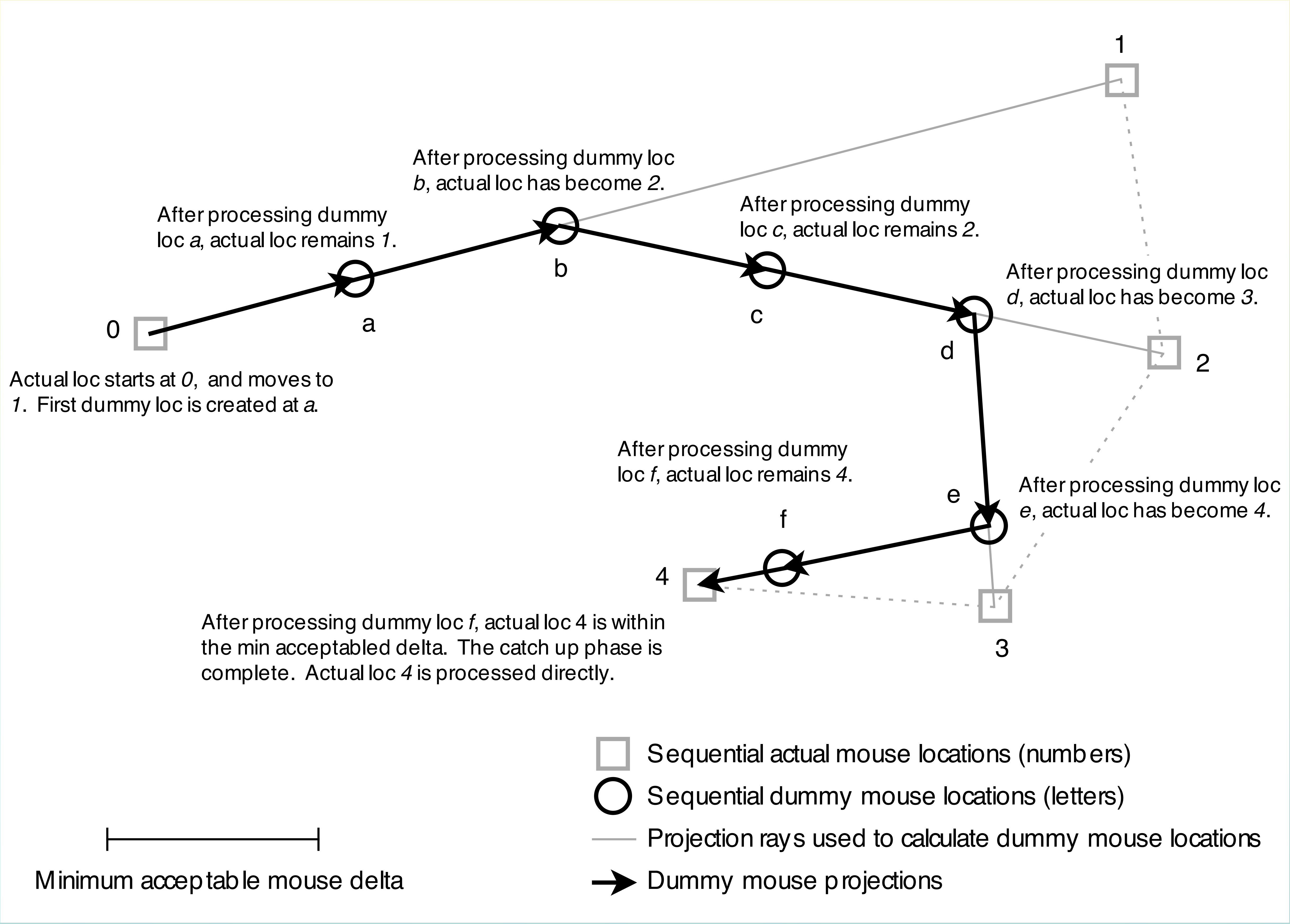
- Developed *Druid*, a system for constructing interwoven 2½D scenes.
- Use of branch-and-bound search to label; gives the user the experience of interacting directly with idealized physical surfaces.
- Search hinders *Druid's* scalability.
- Discovered a topological property of 2½D scenes, the *crossing-state equivalence class rule*.
- Exploitation of this property can alleviate the need to search in some situations and can dramatically reduce the search space in remaining situations.
- Vastly extended the complexity of drawings that users of *Druid* can construct.

52



53

Min. Acceptable Mouse Delta



Back