How to V isualize a Graph: Specification and Algorithms

Isabel F . Cruz

Tufts University

Roberto T amassia

Brown University

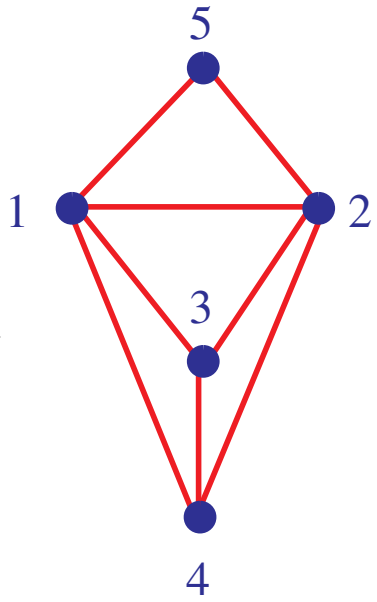
© 1994, Isabel F. Cruz and Roberto Tamassia

Introduction

Graph Drawing

- geometry → graph
  - intersection graphs
  - visibility graphs
- graph → geometry
  - graph drawing

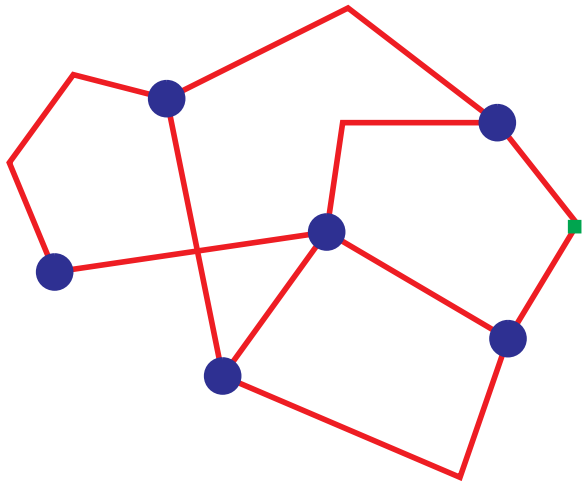
G=(V,E)
V={1,2,3,4,5}
E={(1,2), (1,3), (1,4), (1,5), (2,3), (2,4), (2,5), (3,4)}



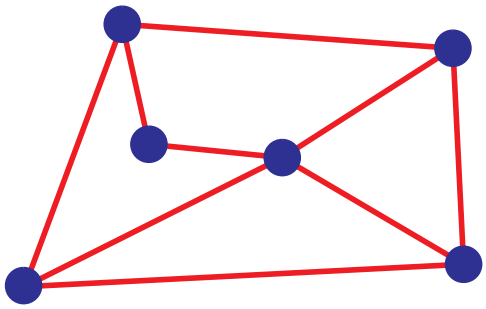
- examples of applications
  - software engineering: subroutine-call graphs, data-flow diagrams, entity-relationship diagrams
  - project management: PERT diagrams
  - knowledge representation
  - circuit schematics

Drawings of Graphs

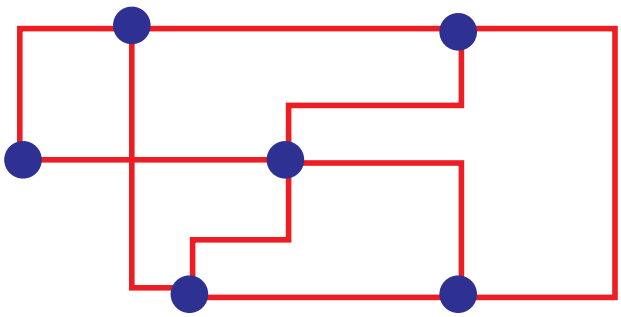
polyline drawing



planar straight-line drawing

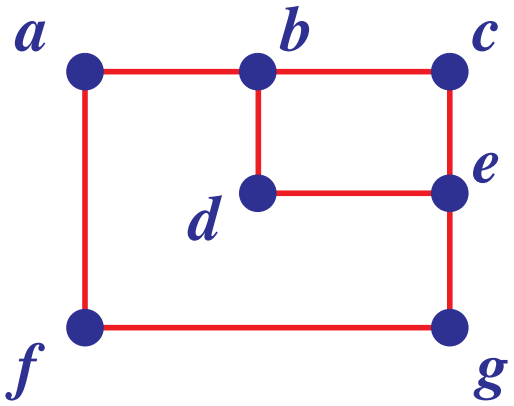


orthogonal drawing

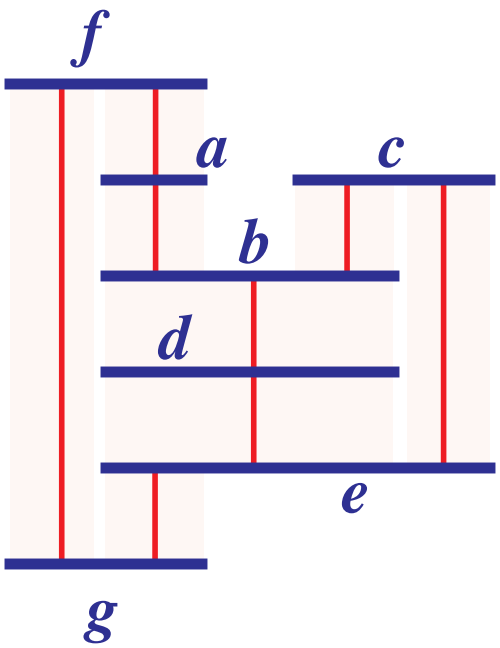


Drawings of Graphs

planar rectilinear drawing

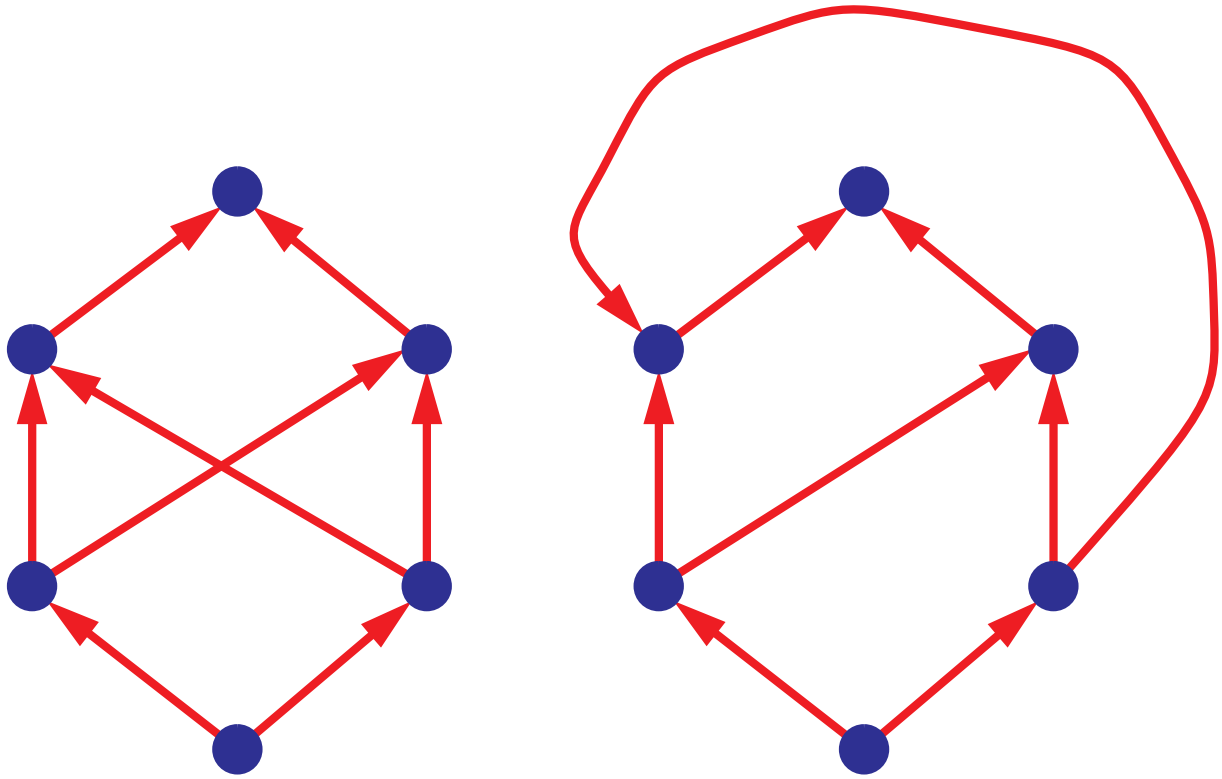


strong visibility *representation*



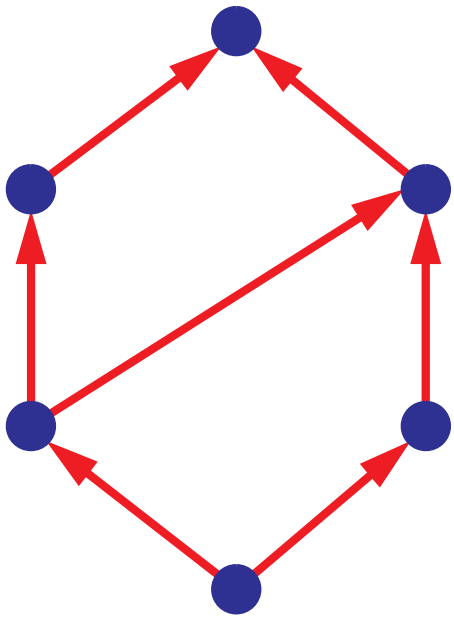
Upward Drawings

- directed acyclic graphs are usually drawn in such a way that all edges "flow" in the same direction, e.g., from left to right, or from bottom to top
- such ***upward*** drawings effectively visualize hierarchical relationships, such as covering digraphs of ordered sets
- not every planar acyclic digraph admits a planar upward drawing

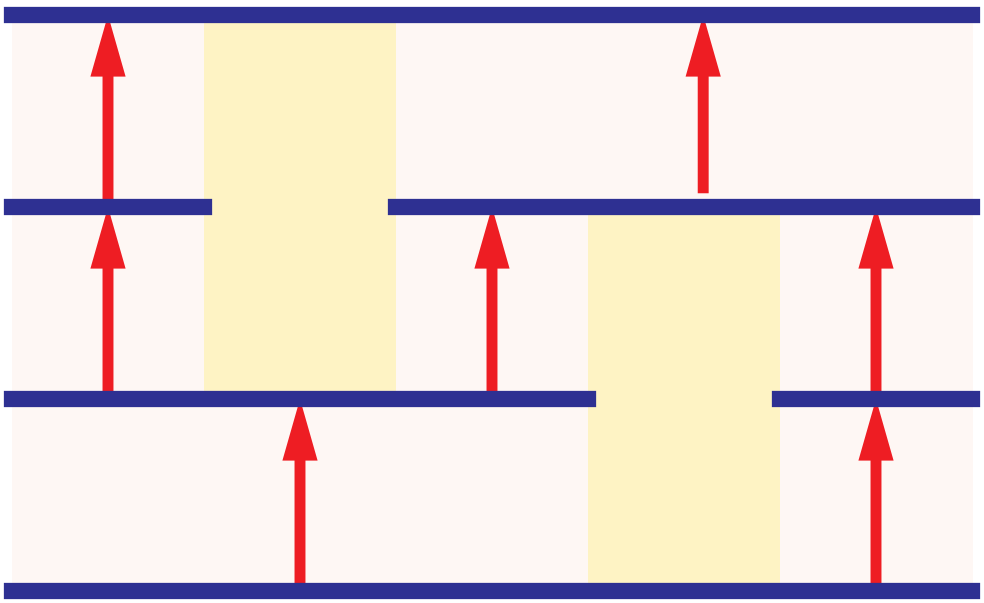


Upward Drawings of Digraphs

upward planar straight-line drawing



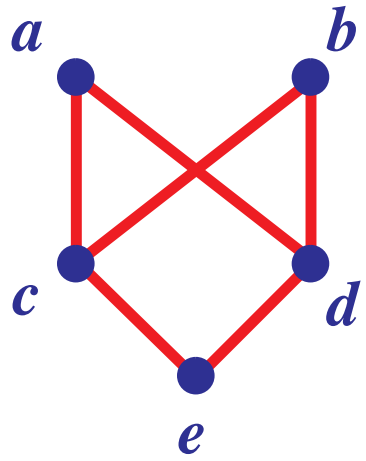
weak visibility representation



How to Draw an Ordered Set

- a "natural" geometric representation of an ordered set is an ***upward drawing*** of its covering digraph, with arrows usually omitted

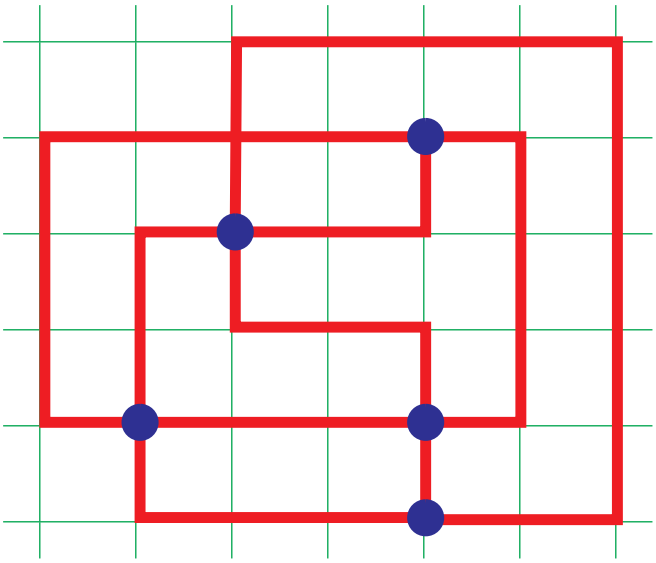
P = {(a,c), (a,d), (a,e), (b,c), (b,d), (b,e), (c,e), (d,e)}



- covering digraphs of ordered sets are acyclic digraphs without ***transitive edges*** , hence the problem of drawing ordered sets is an important special case of the problem of drawing acyclic digraphs

Resolution

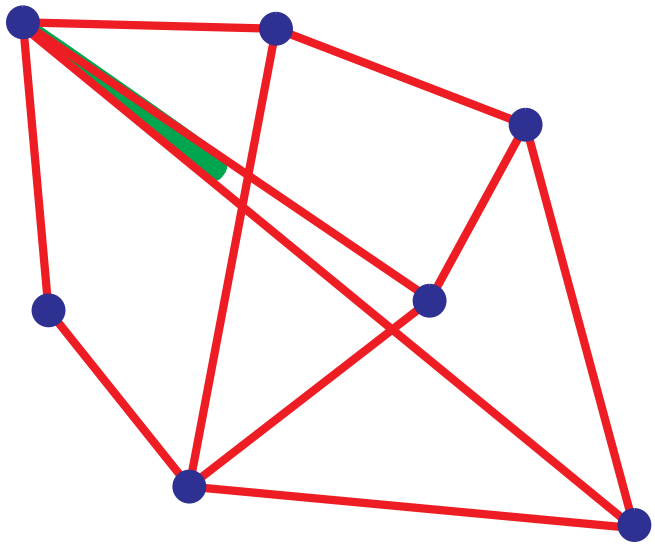
- display devices and the human eye have finite resolution
- examples of ***resolution rules*** :
  - integer coordinates for vertices and bends (***grid*** drawings)



  - prescribed minimum distance between vertices
  - prescribed minimum distance between vertices and nonincident edges
  - prescribed minimum angle formed by consecutive incident edges (***angular resolution***)

Angular Resolution

- The ***angular resolution*** ρ of a straight-line drawing is the smallest angle formed by two edges incident on the same vertex



- ***High angular resolution*** is desirable in ***visualization*** applications and in the design of ***optical communication*** networks.
- A ***trivial upper bound*** on the angular resolution is

$$\rho \leq \frac{2\pi}{d}$$

Aesthetic Criteria

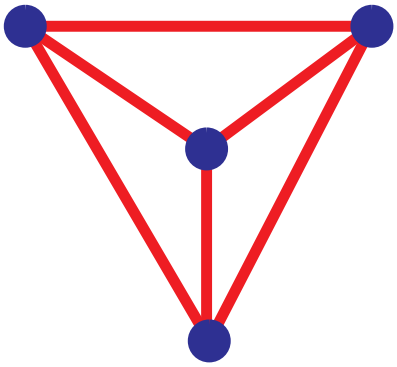
- some drawings are better than others in conveying information on the graph
- ***aesthetic criteria*** attempt to characterize readability by means of general ***optimization*** goals

Examples

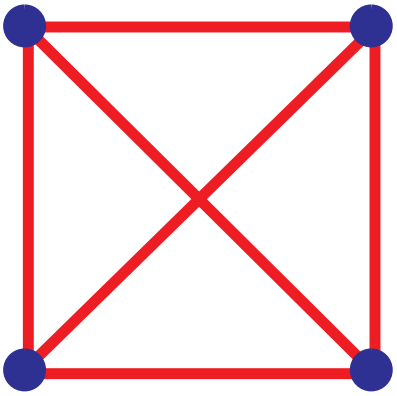
- minimize ***crossings***
- minimize ***area***
- minimize ***bends*** (in orthogonal drawings)
- minimize ***slopes*** (in polyline drawings)
- maximize ***smallest angle***
- maximize display of ***symmetries***

Trade-Offs

- in general, one cannot simultaneously optimize two aesthetic criteria



min # crossings

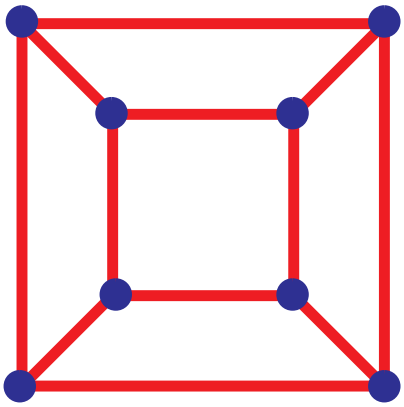


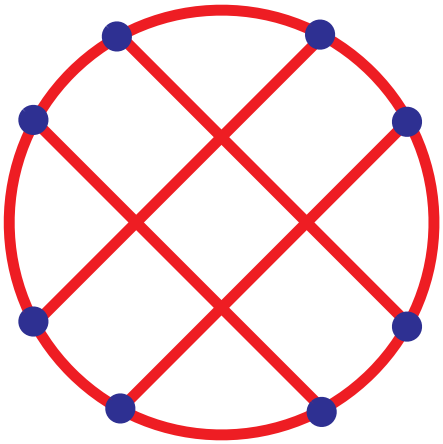
max symmetries

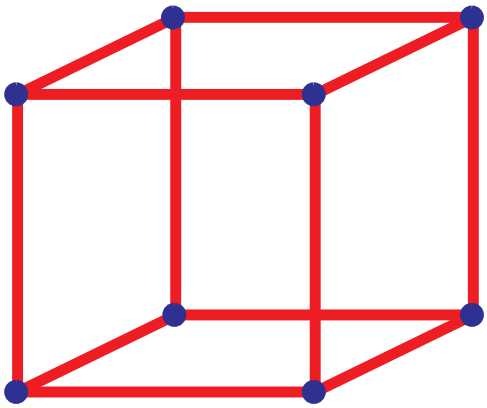
Complexity Issues

- testing planarity takes linear time
- testing upward planarity is NP-hard
- minimizing crossings is NP-hard
- minimizing bends in planar orthogonal drawing:
  - NP-hard in general
  - polynomial time for a fixed embedding

Beyond Aesthetic Criteria







Constraints

- some readability aspects require knowledge about the semantics of the specific graph (e.g., place "most important" vertex in the middle)
- ***constraints*** are provided as additional input to a graph drawing algorithm

Examples

- place a given vertex in the "middle" of the drawing
- place a given vertex on the external boundary of the drawing
- draw a subgraph with a prescribed "shape"
- keep a group of vertices "close" together

Algorithmic Approach

- Layout of the graph generated according to a prespecified set of aesthetic criteria
- Aesthetic criteria embodied in an algorithm as optimization goals . E.g.
  - minimization of crossings
  - minimization of area

Advantages

- Computational ***efficiency***

Disadvantages

- User -defined ***constraints*** are not naturally supported

Extensions

- A limited constraint-satisfaction capability is attainable within the algorithmic approach

  E.g., [T amassia Di Battista Batini 87]

Declarative Approach

- Layout of the graph specified by a *user-defined* set of *constraints*
- Layout generated by the *solution* of a *system* of constraints

Advantages

- *Expressive power*

Disadvantages

- Some natural aesthetics (e.g., planarity) need *complicated* constraints to be expressed
- General constraint-solving systems are computationally *inefficient*
- Lack of a powerful language for the specification of constraints (currently done with a detailed enumeration of facts, or with a set notation)

Getting Started with Graph Drawing

- Annotated bibliography on graph drawing
  (more than 300 entries) by Di Battista, Eades, T amassia, and T ollis.
  Computational Geometry: Theory and Applications, 4(5), 235-282 (1994). Preprint:
  `ftp://ftp.cs.brown.edu/pub/papers/compgeo/`
- Computational geometry bibliography
  (more than 5,000 BibT eX entries, including most papers on graph drawing).
  `ftp://cs.usask.ca/pub/geometry`
  `http://www.cs.ruu.nl/people/otfried/htmlgeombib.html`
- Proceedings of Graph Drawing '93
  `ftp://ftp.cs.brown.edu:pub/papers/compgeo/`
- Proceedings of Graph Drawing '94
  LNCS vol. 894, Springer -Verlag, 1995.
  Order from `order@springer.de`
- Book on graph drawing
  by Di Battista, Eades, T amassia, and T ollis, to be published by Prentice Hall in 1996.

Publishing with Graph Drawing

- The area of graph drawing has received increasing interest in the last years.
- Special issues on graph drawing, to appear in Algorithmica, CGT A, and JVLC .
- Papers on graph drawing are being presented at many conferences in combinatorics and computer science, including VL, CHI, A VI, STOC, FOCS, SODA, ESA, CG, SP AA, SPDP , WG, ORDAL, W ADS, SW AT, ...
- Annual workshop on graph drawing GD 92, 93, 94, 95, ...

How to V isualize a Graph: Specification and Algorithms

Part I: Algorithmic Approaches

Isabel F . Cruz, Tufts University

Roberto T amassia, Brown University

© 1994, Isabel F. Cruz and Roberto Tamassia

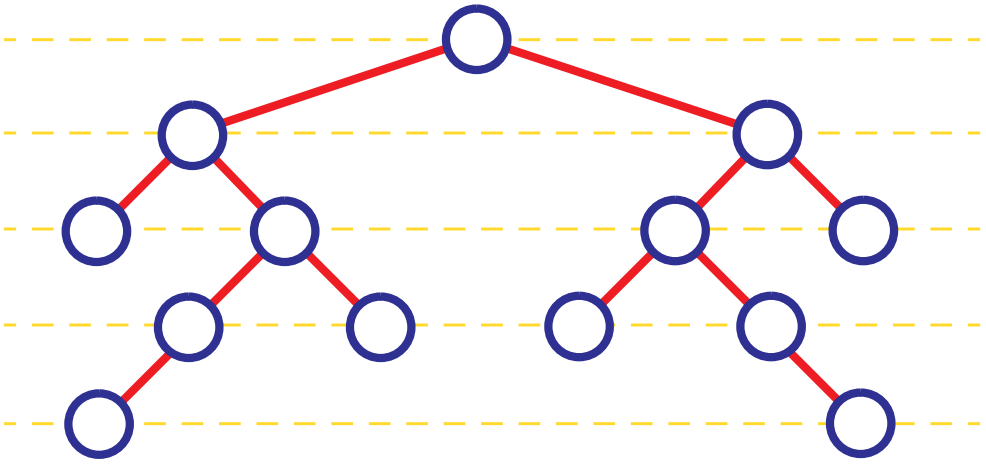
Outline of Part I

- Trees
- Planar Undirected Graphs
- Planar Directed Graphs
- General Undirected Graphs
- General Directed Graphs
- Systems
- Challenges and Open Problems

Trees

Drawings of Rooted T rees

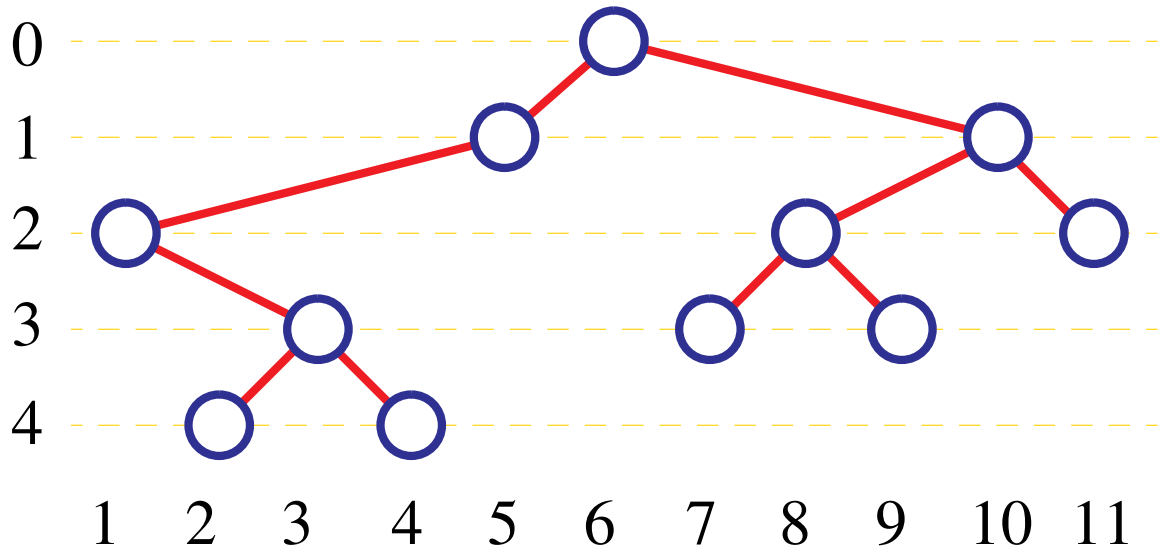
- the usual drawings of rooted trees are ***planar*** , ***straight-line*** , and ***upward*** (parents above children)
- it is desirable to minimize the ***area*** and to display ***symmetries*** and ***isomorphic subtrees***
- ***level drawing*** : nodes at the same distance from the root are horizontally aligned



- level drawings may require $\Omega(n^2)$ ***area***

A Simple Level Drawing Algorithm for Binary T rees

- y(v) = distance from root
- x(v) = inorder rank

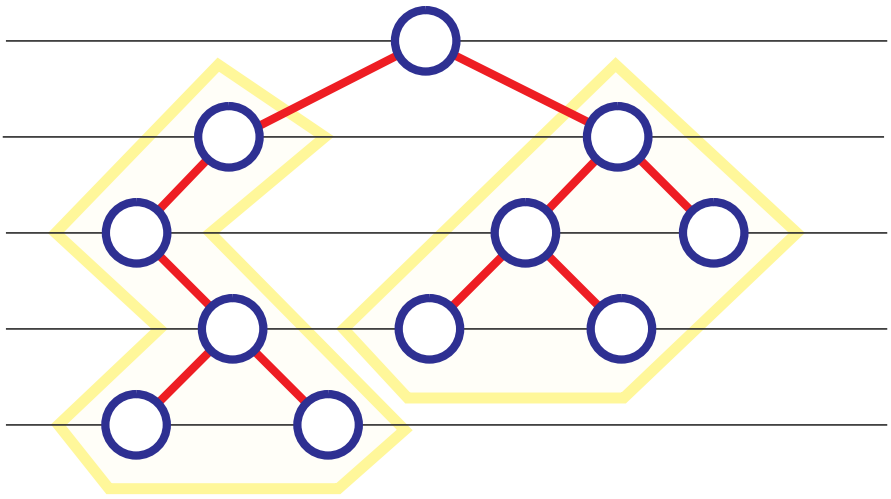


- level grid drawing
- display of symmetries and of isomorphic subtrees
- parent in between left and right child
- parents not always centered on children
- width = n − 1

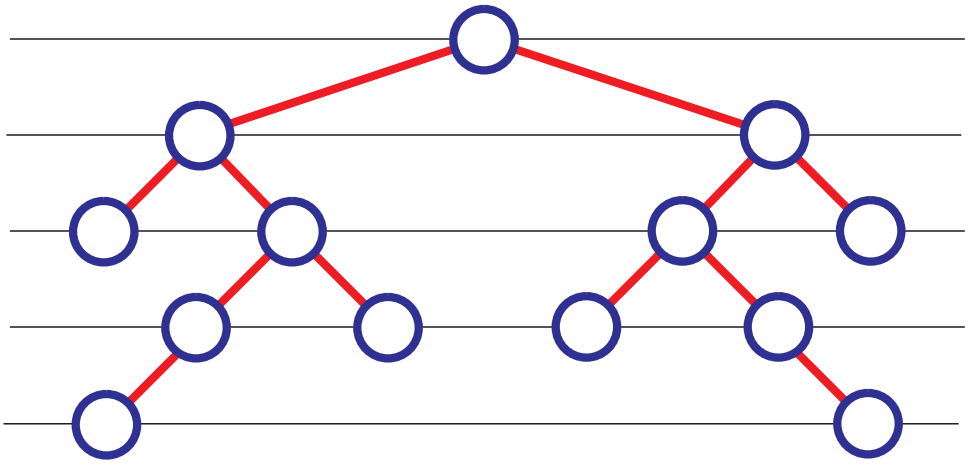
A Recursive Level Drawing Algorithm for Binary T rees

[Reingold T ilford 1983]

- draw the left subtree
- draw the right subtree
- place the drawings of the subtrees at horizontal distance 2
- place the root one level above and half-way between the children
- if there is only one child, place the root at horizontal distance 1 from the child

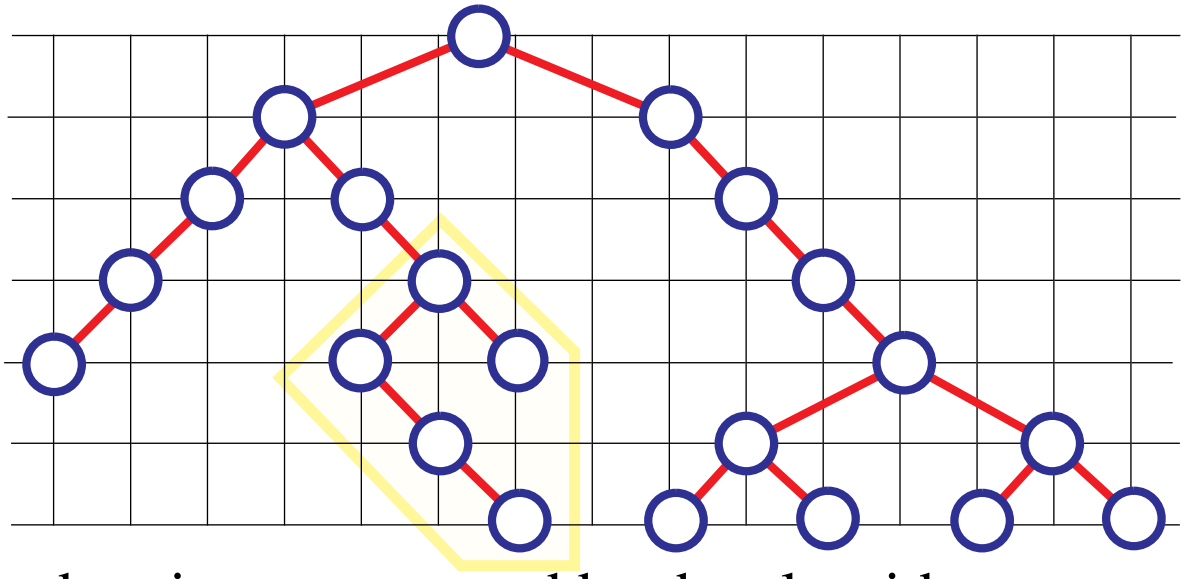


Properties of Recursive Level Drawing Algorithm for Binary T rees

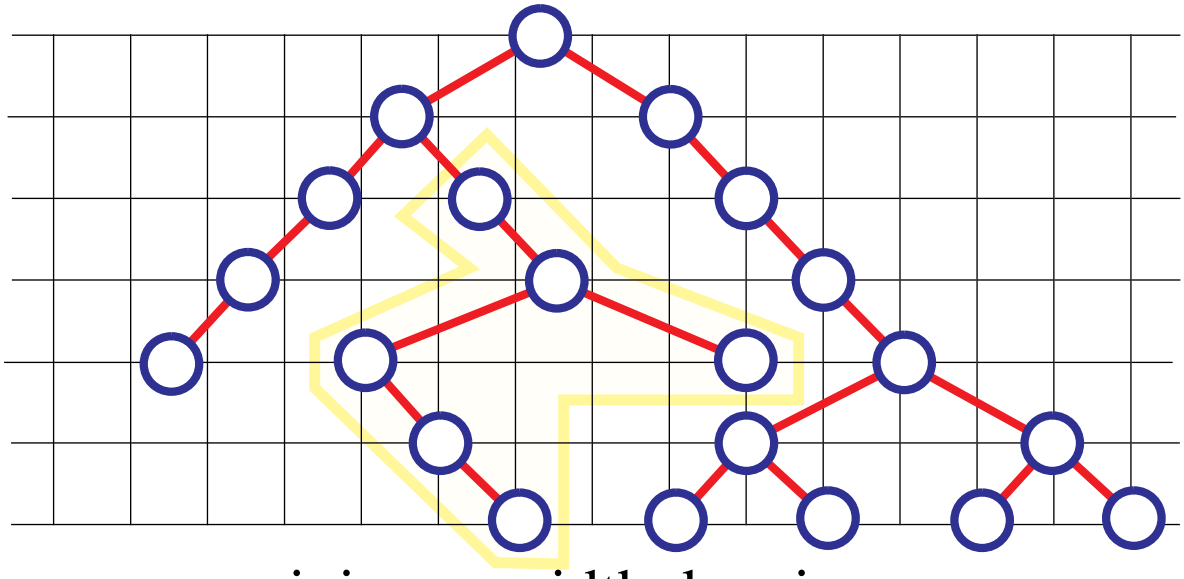


- ***centered*** level drawing
- "small" width
- display of symmetries and of isomorphic subtrees
- can be implemented to run in O(n) time
- can be extended to draw general rooted trees (e.g., root is placed at the average x-coordinate of its children)

Non Optimality of Recursive Tree Drawing Algorithm



drawing constructed by the algorithm

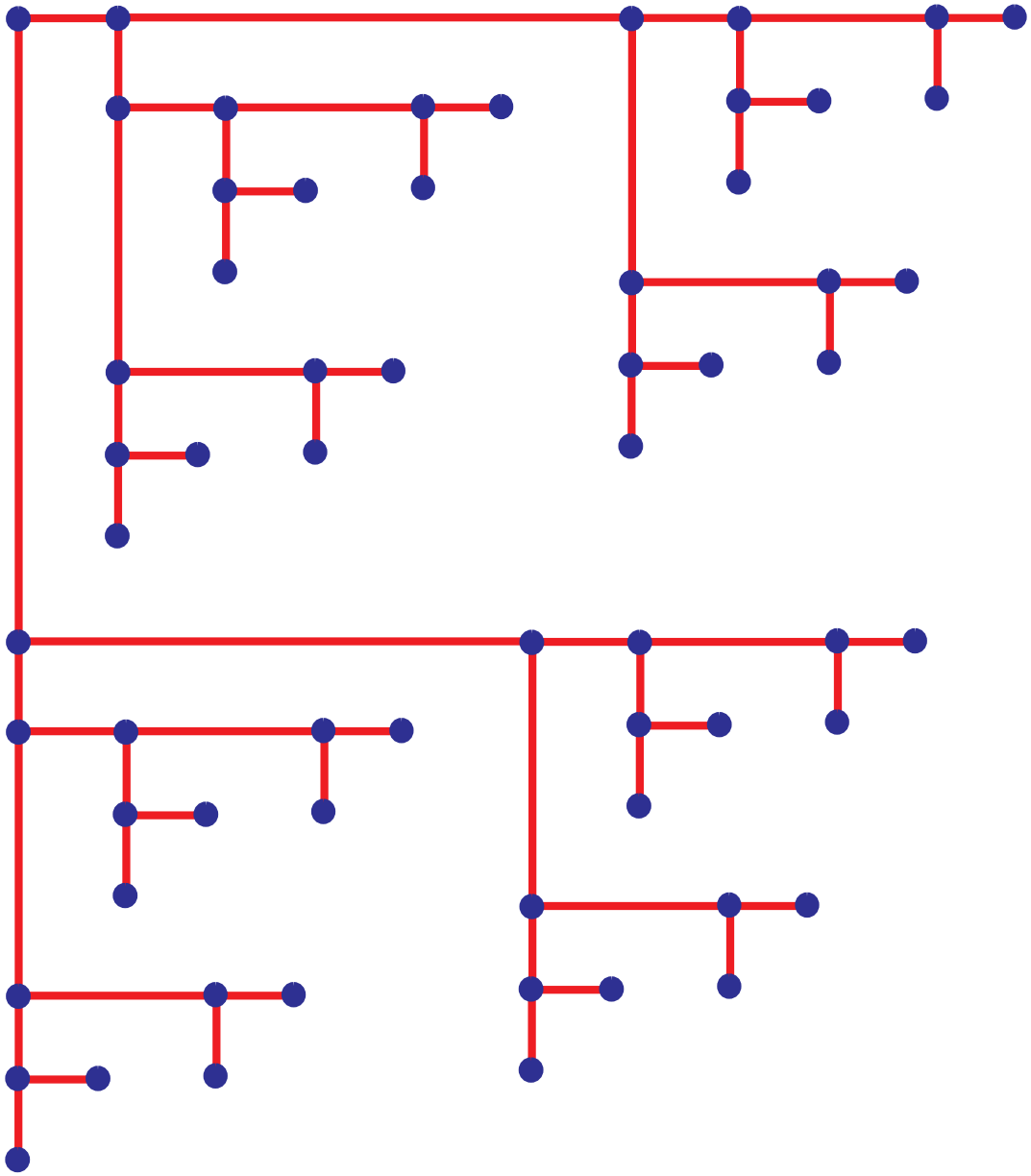


minimum width drawing

- minimizing the width is NP-hard if integer coordinates are required

Area-Efficient Drawings of T rees

- planar straight-line upward grid drawings of ***balanced trees*** with ***O(n) area*** [Crescenzi Di Battista Piperno 92]



Area-Efficient Drawings of T rees

- planar polyline upward grid drawings with ***O(n) area***
  [Garg Goodrich T amassia 93]

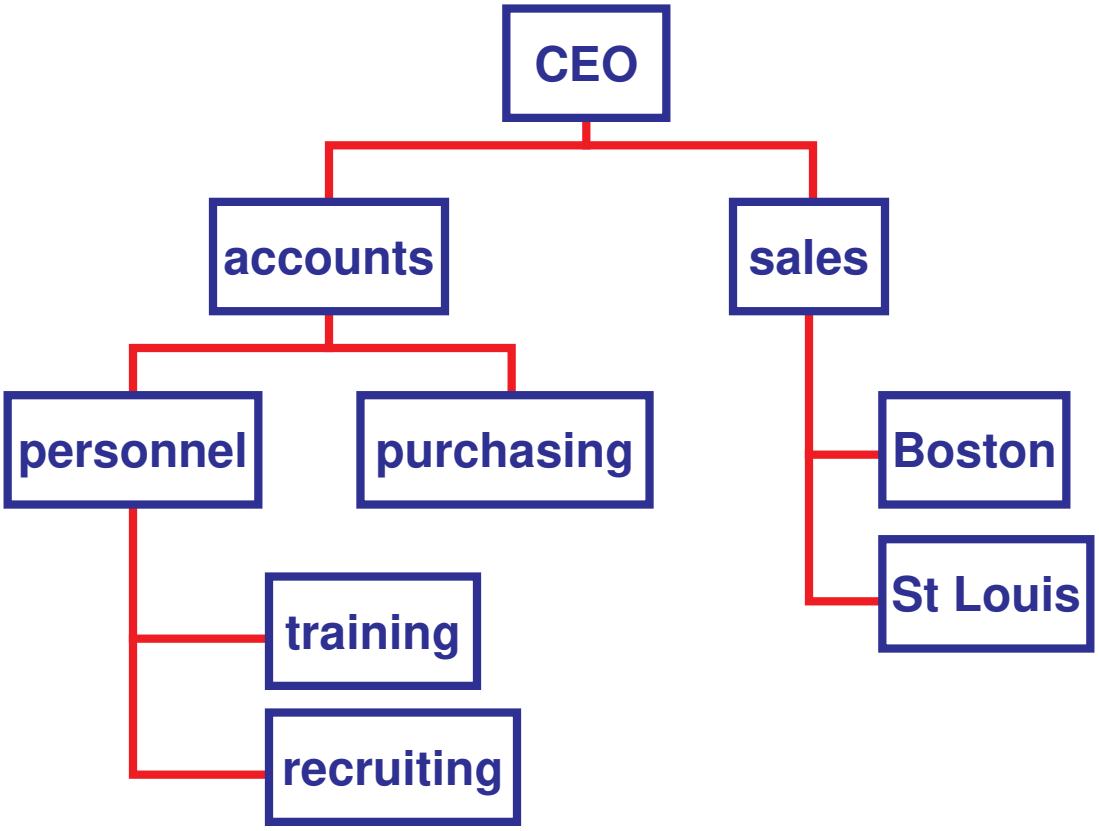
Area Requirement of Planar Drawings of T rees

upward level	$\Theta(n^2)$ [RT 83]
upward polyline	$\Theta(n)$ [GGT 93]
upward straight-line	$\Omega(n)$ $O(n \log n)$ [CDP 92]
upward orthogonal	$\Theta(n \log \log n)$ [GGT 93]
non-upward orthogonal	$\Theta(n)$ [L80, V91]
non-upward leaves-on-hull orthogonal	$\Theta(n \log n)$ [BK 80]

- ***Open Problem*** : determine the area requirement of planar upward straight-line drawings of trees

Tip-Over Drawings of Rooted T rees

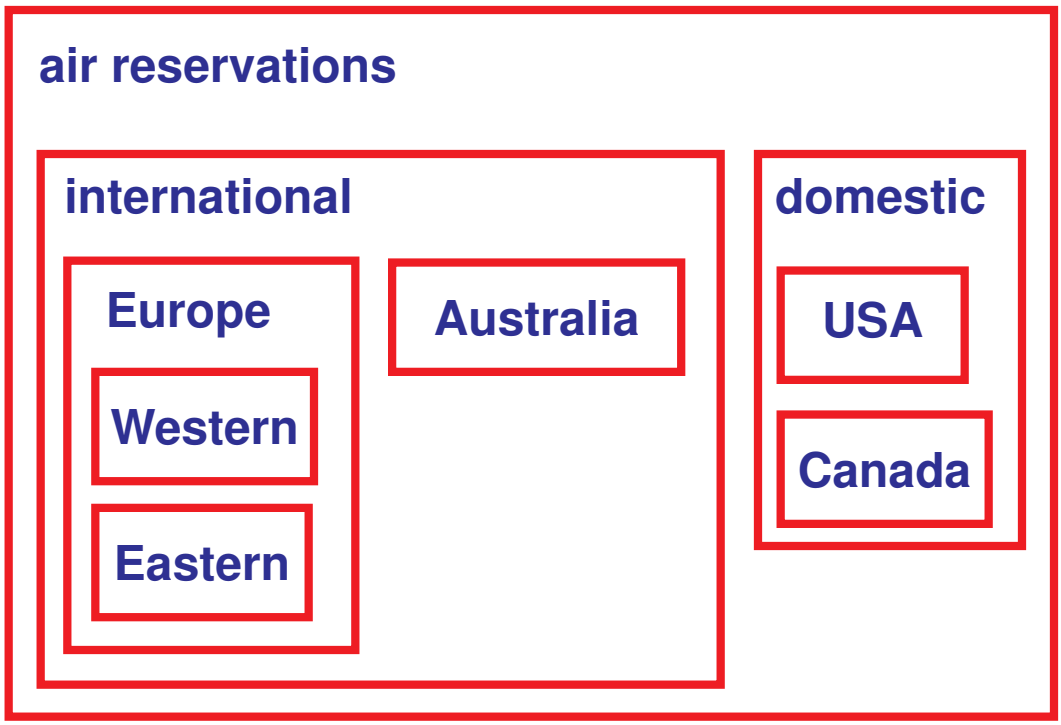
- ***Tip-over drawings*** are upward planar orthogonal drawings such that the children of a node:
  - are arranged either horizontally or vertically
  - share portions of the edges to the parent.



- Widely used in organization charts.
- Allow to better fit the drawing in a

Inclusion Drawings of Rooted Trees

- ***Inclusion drawings*** display the parent-child relationship by the inclusion between isothetic rectangles.



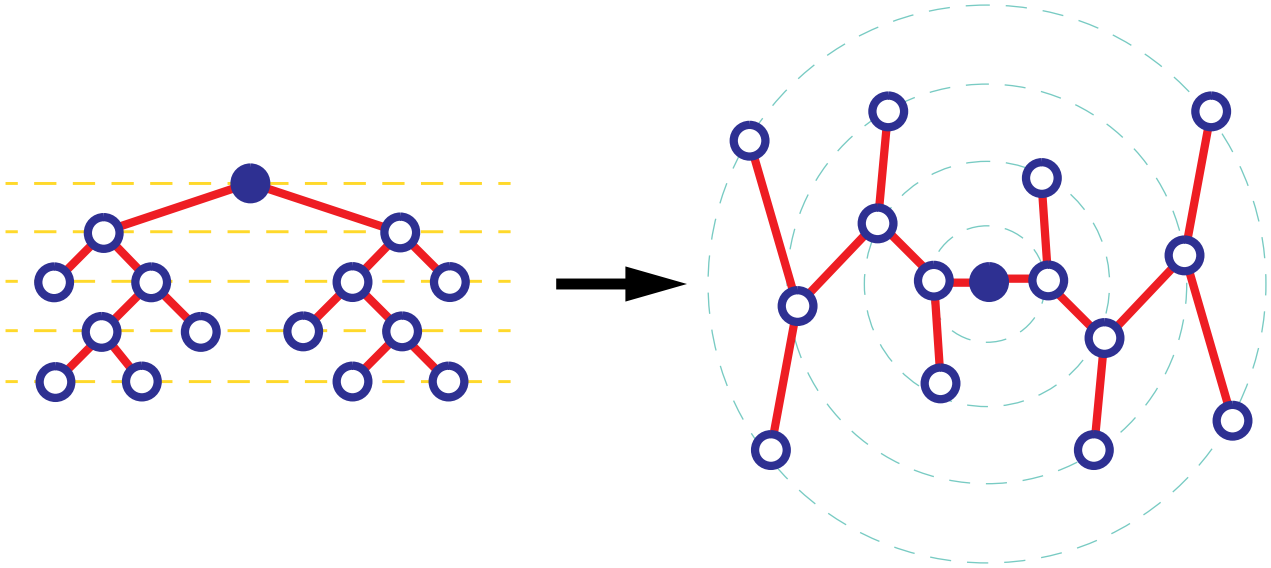
- Closely related to tip-over drawings.
- Used for displaying compound graphs (e.g., the union of a graph and a tree)
- Allow to better fit the drawing in a prescribed region

Area of T ip-Over and Inclusion Drawings

- Eades, Lin and Lin (1992) study of the area requirement of tip-over and inclusion drawings of rooted trees.
- The dimensions of the node labels are given as part of the input.
- ***Minimizing the area*** of the drawing is:
  - ***NP-hard for general trees***
  - computable in ***polynomial time*** for ***balanced trees*** with a ***dynamic programming*** algorithm
- Similar results for the following problems:
  - minimizing the ***perimeter*** of the drawing.
  - minimizing the ***width*** for a given height
  - minimizing the ***height*** for a given width

How to Draw Free T rees

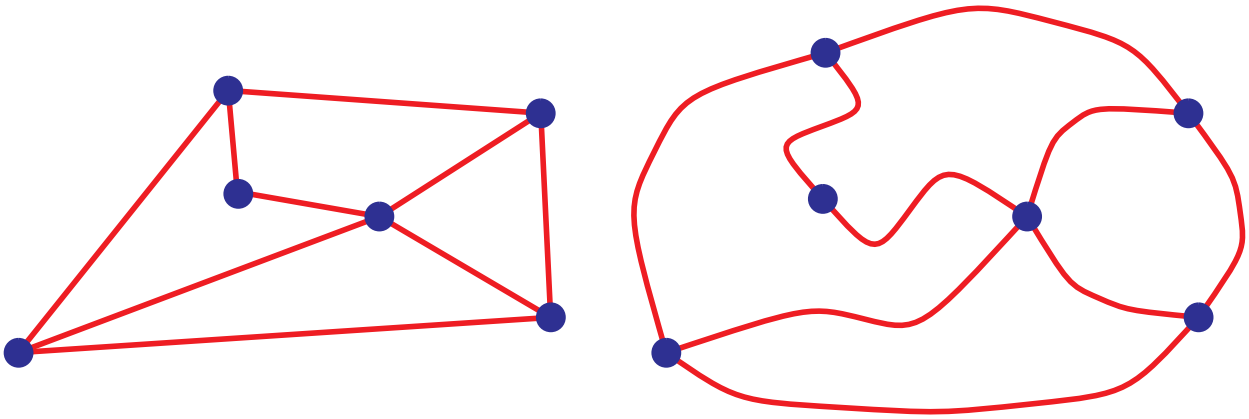
- ***Free trees*** are connected graphs without cycles and do not represent hierarchical relationships (e.g., spanning trees)
- Level drawings of rooted trees yield ***radial drawings*** of free trees:
  - root the free tree T at its ***center*** (node with minmax distance from the leaves), which gives a rooted tree T'
  - construct a level drawing Δ' of T'
  - use a geometric transformation (***cartesian*** $\rightarrow$ ***polar***) to obtain from Δ' a radial drawing Δ of T



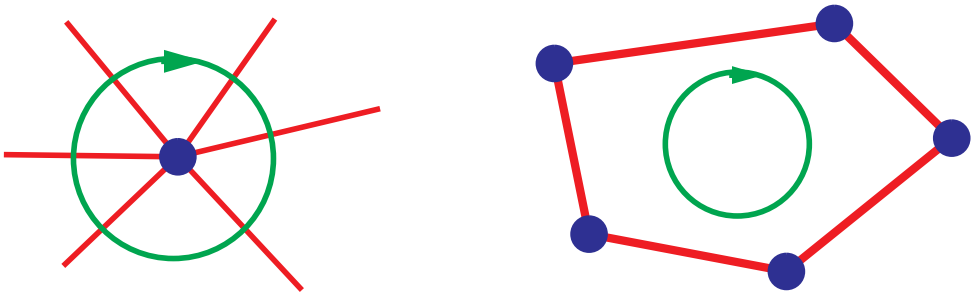
Planar Undirected Graphs

Planar Drawings and Embeddings

- a ***planar embedding*** is a class of topologically equivalent planar drawings



- a planar embedding prescribes
  - the ***star*** of edges around each vertex
  - the ***circuit*** bounding each face



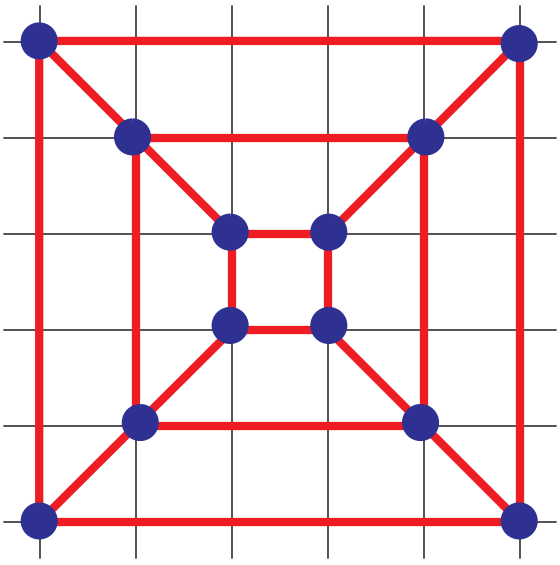
- the number of distinct embeddings is exponential in the worst case
- triconnected planar graphs have a unique embedding

The Complexity of Planarity Testing

- Planarity testing and constructing a planar embedding can be done in ***linear time*** :
  - ***depth-first-search***
    [Hopcroft T arjan 74]
    [de Fraysseix Rosenstiehl 82]
  - ***st-numbering and PQ-trees***
    [Lempel Even Cederbaum 67]
    [Even T arjan 76]
    [Booth Lueker 76]
    [Chiba Nishizeki Ozawa 85]
- The above methods are ***complicated*** to understand and implement
- ***Open Problem:***
  - devise a ***simple*** and ***efficient*** planarity testing algorithm.

Planar Straight-Line Drawings

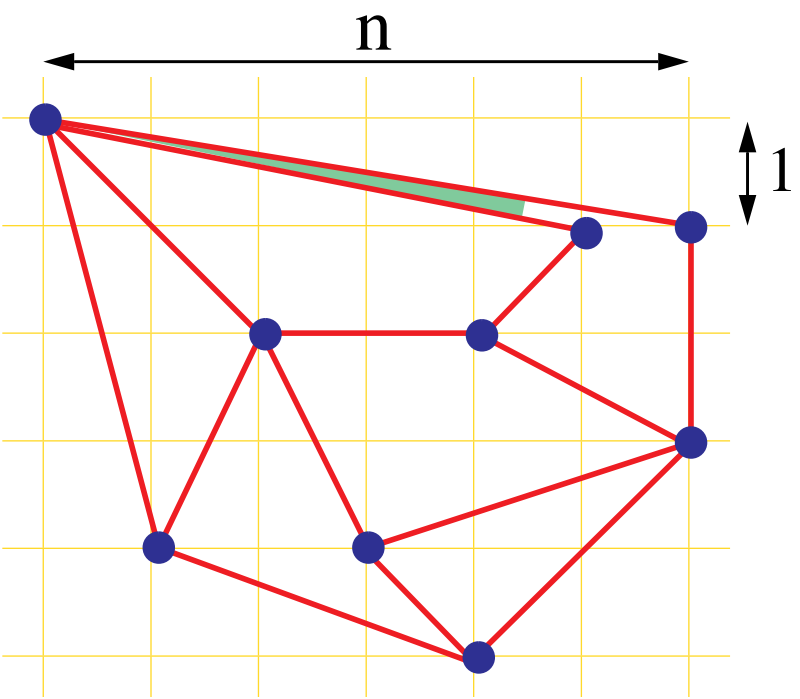
- [Hopcroft T arjan 74]: planarity testing and constructing a planar embedding can be done in O(n) time
- [Fary 48, Stein 51, Steinitz 34, W agner 36]: every planar graph admits a planar straight-line drawing



- Planar straight-line drawings may need $\Omega(n^2)$ area
- [de Fraysseix Pach Pollack 88, Schnyder 89, Kant 92]: $O(n^2)$-area planar straight-line grid drawings can be constructed in O(n) time

Planar Straight-Line Drawings: Angular Resolution

- $O(n^2)$-area drawings may have $\rho = O(1/n^2)$



- [Garg T amassia 94]:
  - ***Upper bound*** on the angular resolution:

$$\rho = O\left(\sqrt{\frac{\log d}{d^3}}\right)$$

  - ***Trade-off*** (area vs. angular resolution):

$$A = \Omega(c^{\rho n})$$

- [Kant 92] Computing the optimal angular resolution is ***NP-hard*** .

Planar Straight-Line Drawings: Angular Resolution

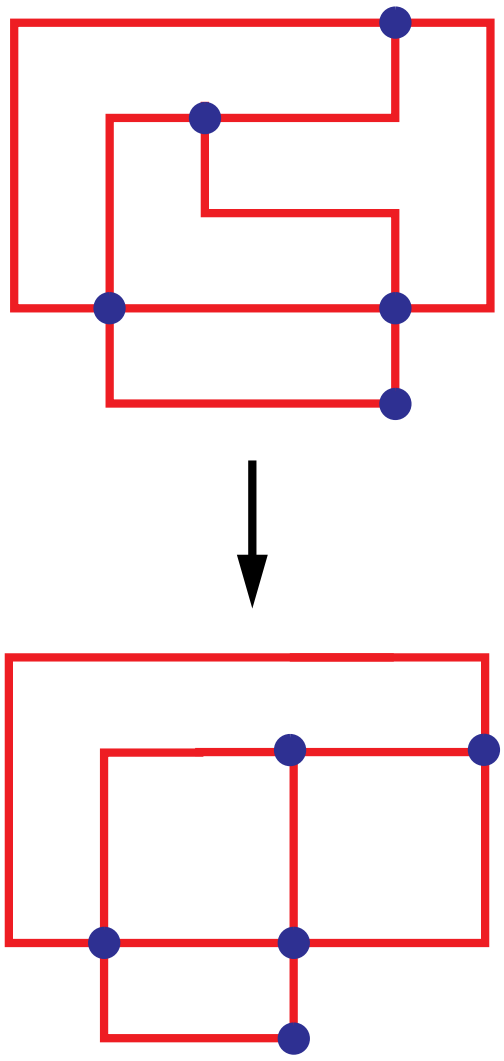
- [Malitz Papakostas 92]: the angular resolution depends on the degree only:

$$\rho = \Omega\left(\frac{1}{7d}\right)$$

- Good angular resolution can be achieved for special classes of planar graphs:
  - ***outerplanar graphs*** , $\rho = O(1/d)$ [Malitz Papakostas 92]
  - ***series-parallel graphs*** , $\rho = O(1/d^2)$ [Garg T amassia 94]
  - ***nested-star graphs*** , $\rho = O(1/d^2)$ [Garg T amassia 94]
- ***Open Problems*** :
  - can we achieve $\rho = O(1/d^k)$ (k a small constant) for all planar graphs?
  - can we efficiently compute an ***approximation*** of the optimal angular resolution?

Planar Orthogonal Drawings: Minimization of Bends

- given planar graph of degree ≤ 4, we want to find a planar orthogonal drawing of G with the minimum number of bends

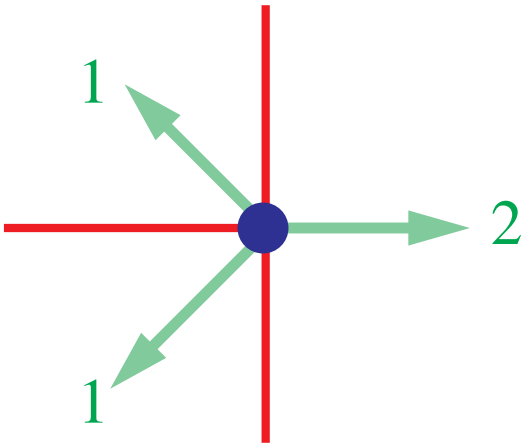


Minimization of Bends in Planar Orthogonal Drawings

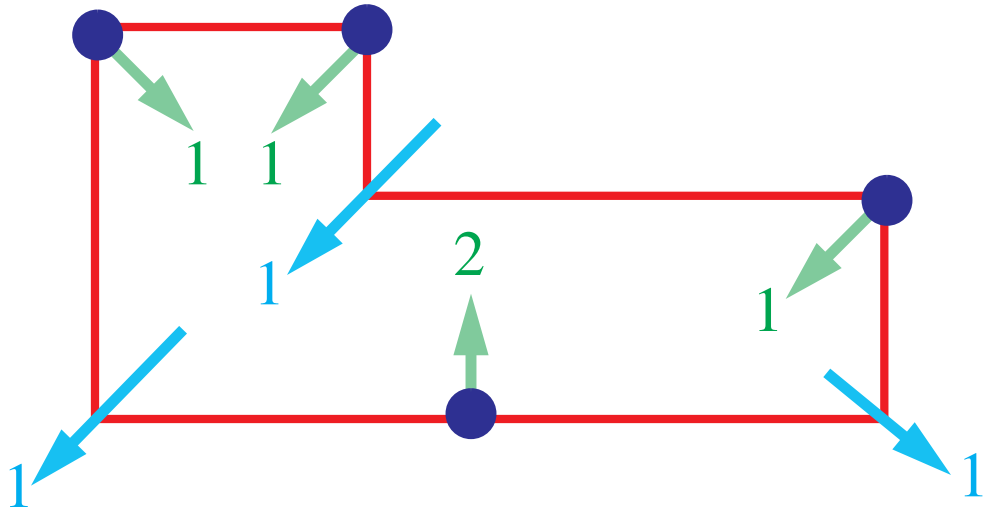
- [Tamassia 87]
  - $O(n^2 \log n)$-time bend minimization for fixed embedding
- [Di Battista Liotta V argiu 93]
  - polynomial-time bend minimization for degree-3 and series-parallel graphs
- [Tamassia T ollis 89]
  - O(n)-time approximation with O(n) bends
- [Garg T amassia 93]
  - minimization of bends is NP-hard
  - approximation with $O(opt + n^{1-\varepsilon})$ bends is NP-hard
  - rectilinear planarity testing is NP-complete

Network Flow Model

- a unit of flow is a 90 ° angle
- a vertex (source) produces 4 units



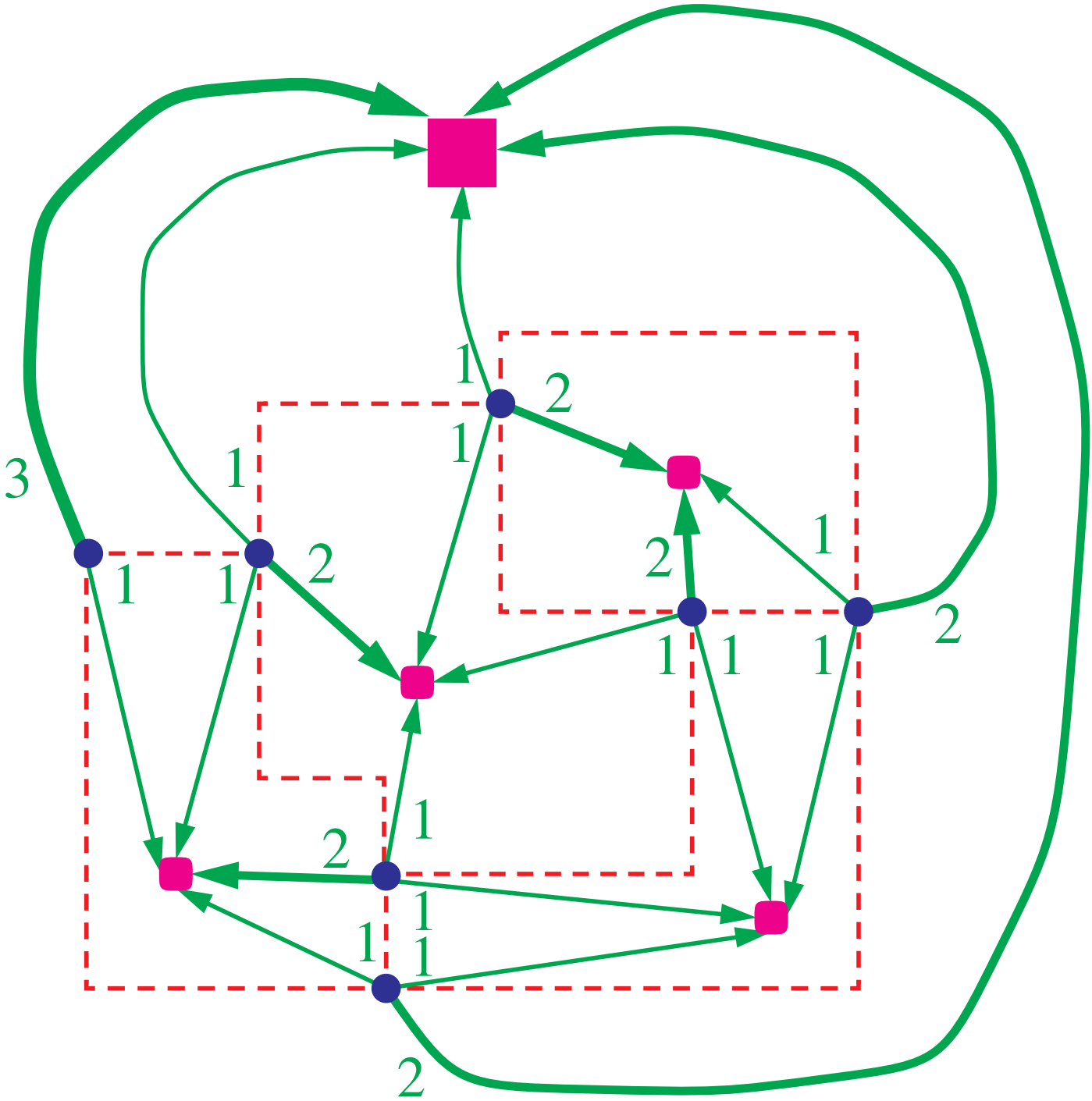
- a face f (sink) consumes 2 deg(f) – 4 units (deg(f) + 4 for the external face)



- Edges transport flow across faces

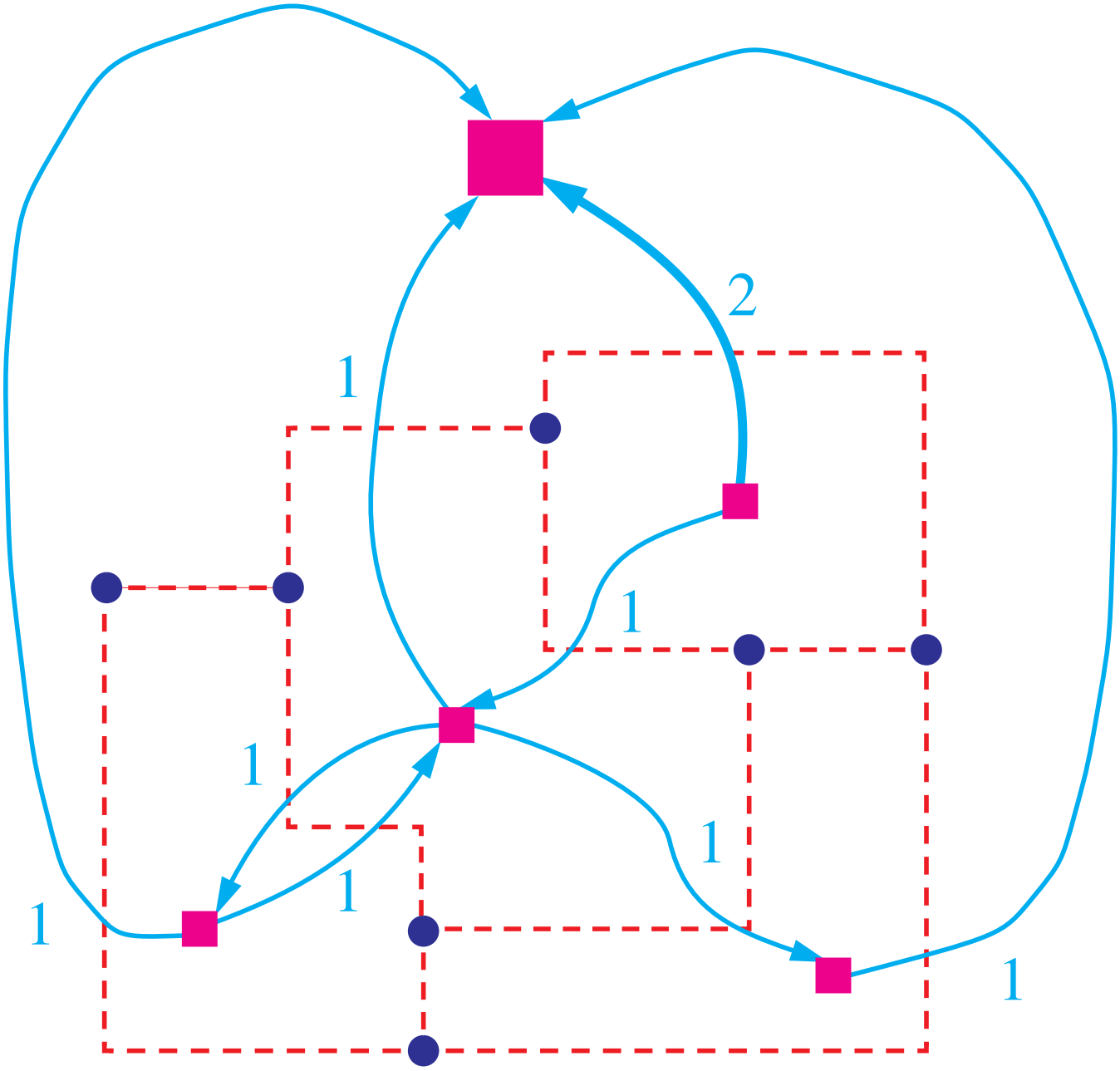
Flow Network

- vertex-face arcs: flow ≥ 1, cost = 0

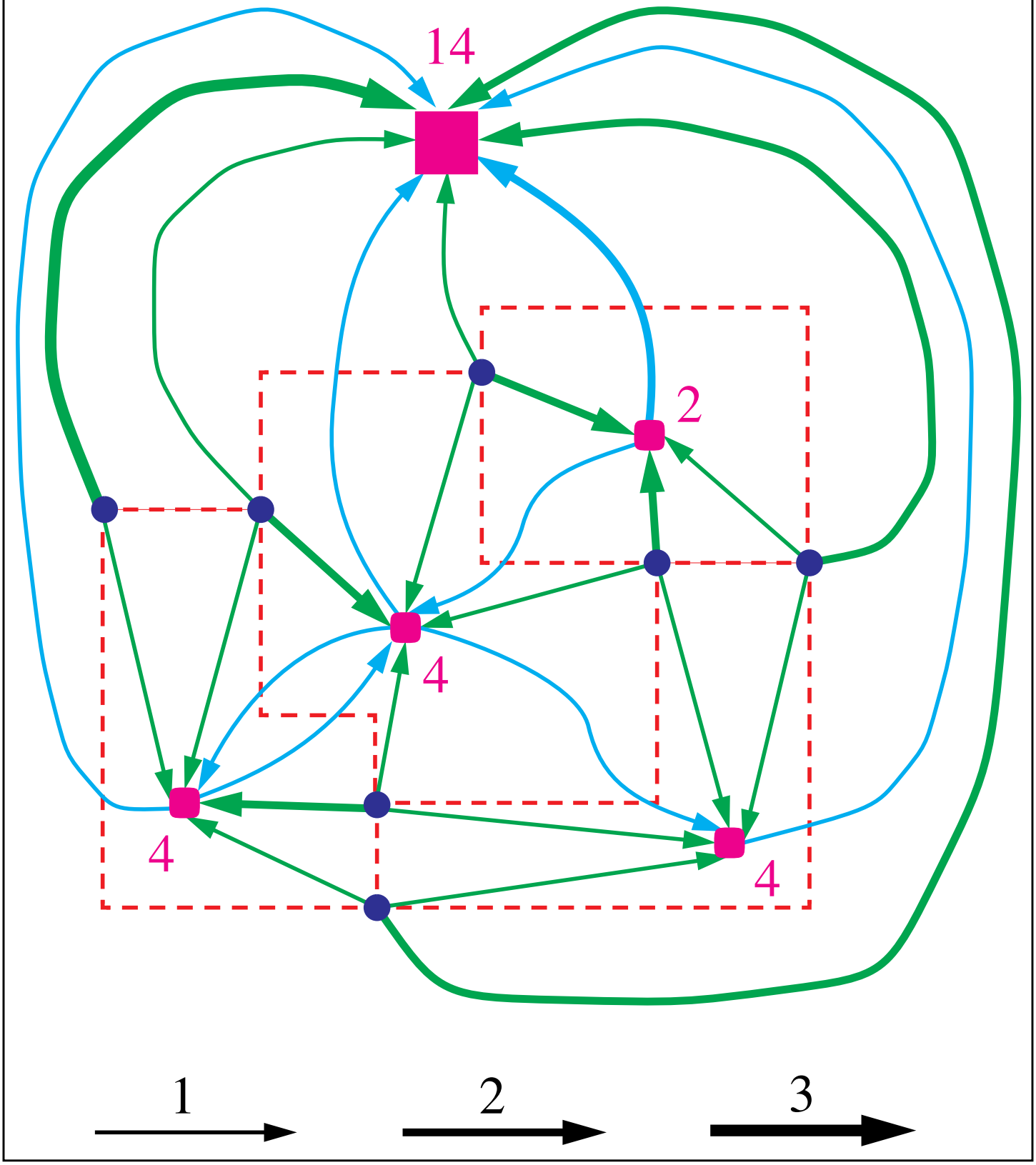


Flow Network

- face-face arcs: flow ≥ 0, cost = 1



Complete Flow Network



Correctness of Flow Model

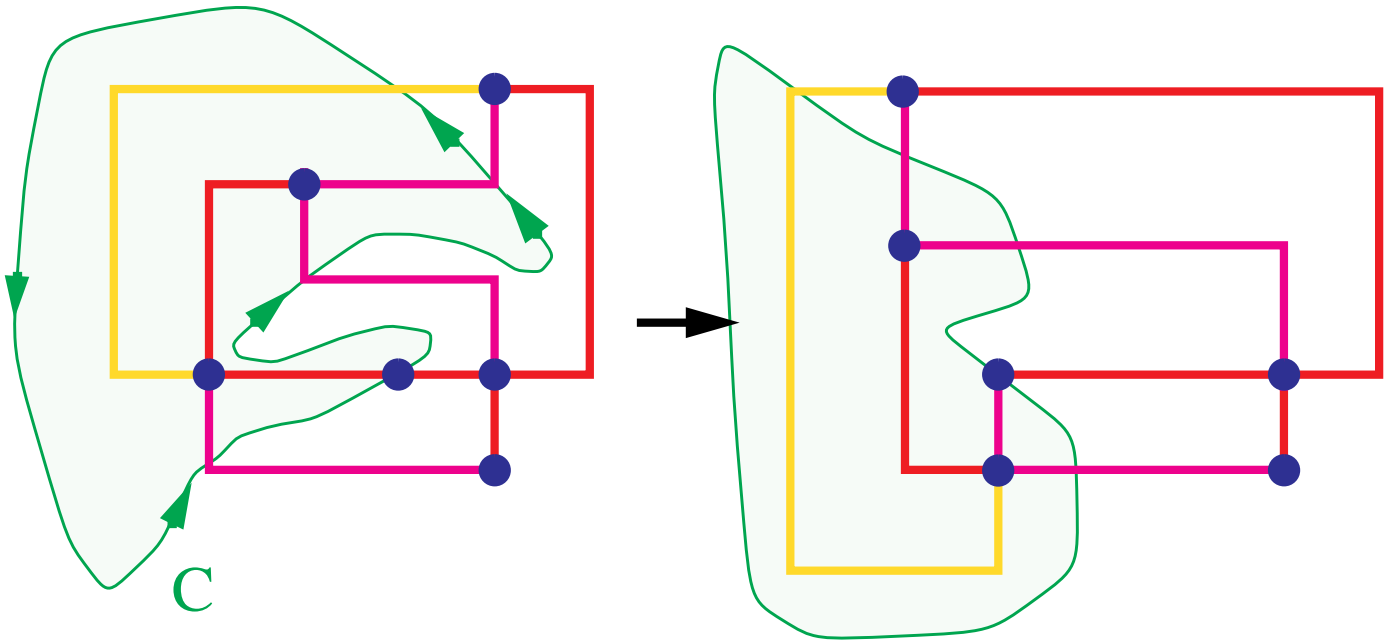
- supply of sources = demand of sinks ↔ Euler 's formula
- flow conservation at vertex ↔ ∑ angles around vertex = 360 °
- flow conservation at face ↔ (# 90 ° angles) − (# 270 ° angles) = 4
- cost of flow ↔ # bends
- flow in N ↔ drawing of G
- minimum cost flow ↔ optimal drawing

Theorem [Tamassia 87] Computing the minimum number of bends for an embedded graph G is equivalent to computing a minimum cost flow in network N, and takes $O(n^2 \log n)$ time

Open Problem: reduce the time complexity of bend minimization.

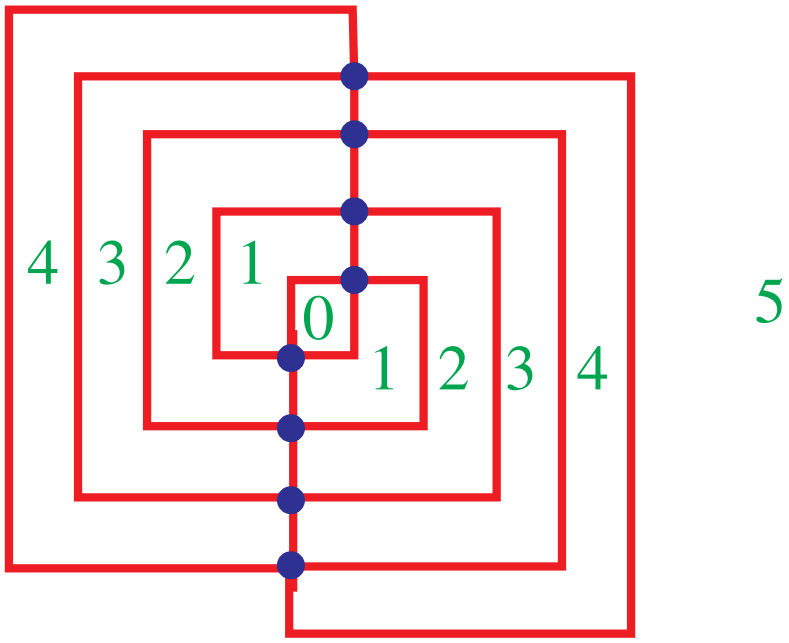
Characterization of Bend-Minimal Drawings

- A drawing has the minimum number of bends if and only if there is no oriented closed curve C such that
  - vertices are intersected by C entering from angles $\geq 180°$
  - (# edges crossed by C from $90°$ or $180°$) < (# edges crossed by C from $270°$)
- If such a curve exists, "rotating" the portion of the drawing inside C reduces the number of bends



Proving the Optimality of a Drawing

- potential Φ on each face

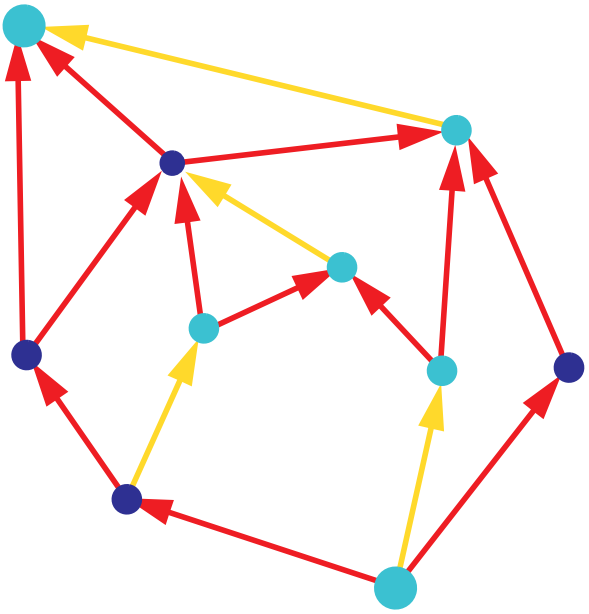


- vertices cannot be traversed by C
- C traverses edge from 270 ° $\Rightarrow$ $\Delta\Phi_i = -1$
- C traverses edge from 90 ° $\Rightarrow$ $\Delta\Phi_i = +1$
- bends removed going ‘ ‘inward’ ’ and inserted going ‘ ‘outward’ ’ $\Delta B_i + \Delta\Phi_i = 0$
- C is a closed curve $\Rightarrow$ $\sum_i \Delta\Phi_i = 0$
- Hence, $\sum_i \Delta B_i = 0$

Planar Directed Graphs

Upward Planarity T esting

- upward planarity testing for ordered sets has the same complexity as for general digraphs (insert dummy vertices on transitive edges)
- [Kelly 87, Di Battista T amassia 87]: upward planarity is equivalent to subgraph inclusion in a planar st-digraph (planar acyclic digraph with one source and one sink, both on the external face)



- [Kelly 87, Di Battista T amassia 87]: upward planarity is equivalent to upward straight-line planarity

Complexity of Upward Planarity T esting

- [Bertolazzi Di Battista Liotta Mannino 91]
  - $O(n^2)$-time for fixed embedding
- [Hutton Lubiw 91]
  - $O(n^2)$-time for single-source digraphs
- [Bertolazzi Di Battista Mannino Tamassia 93]
  - O(n)-time for single-source digraphs
- [Garg T amassia 93]
  - NP-complete

Angles in Planar Upward Drawings

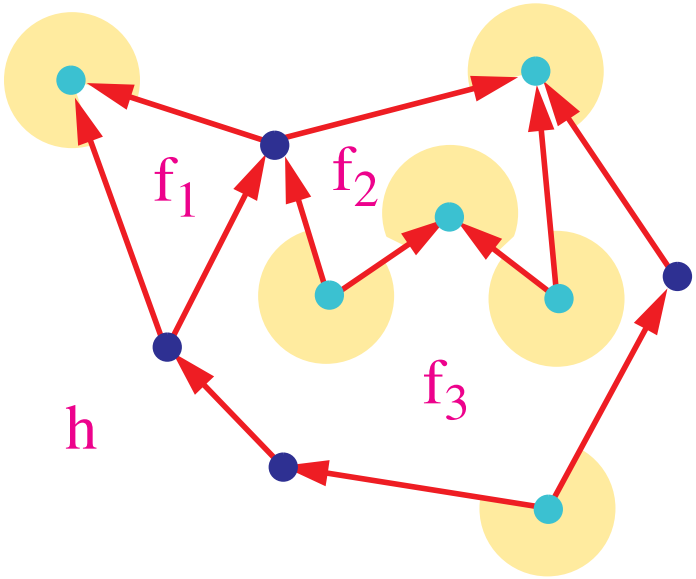
- in any upward drawing the incoming (outgoing) edges of a vertex v appear consecutively in the star of v
- s(f) = (# sources of face f) = (# sinks of face f)
- in a planar straight-line upward drawing
  - each *pole* (source or sink of the graph) has exactly one incident *large angle*
  - an internal face f has s(f) – 1 large angles

$s(f_1) = 1$

$s(f_2) = 2$

$s(f_3) = 3$

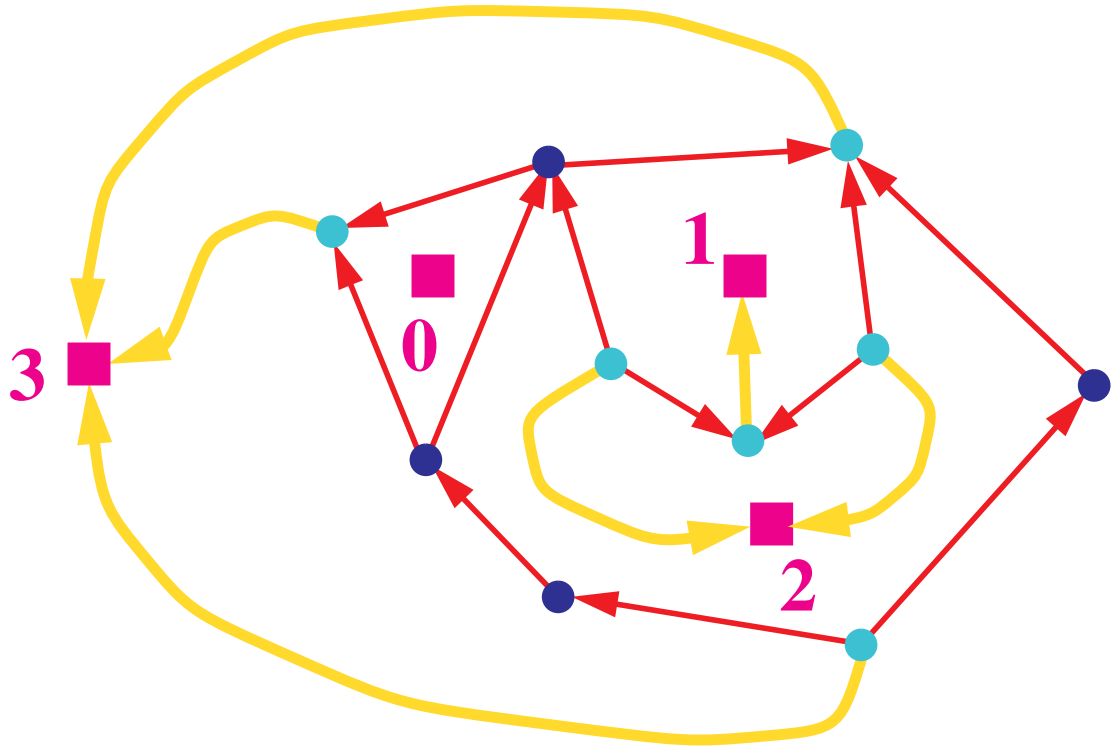
$s(h) = 2$



  - the external face h has s(h) + 1 large angles

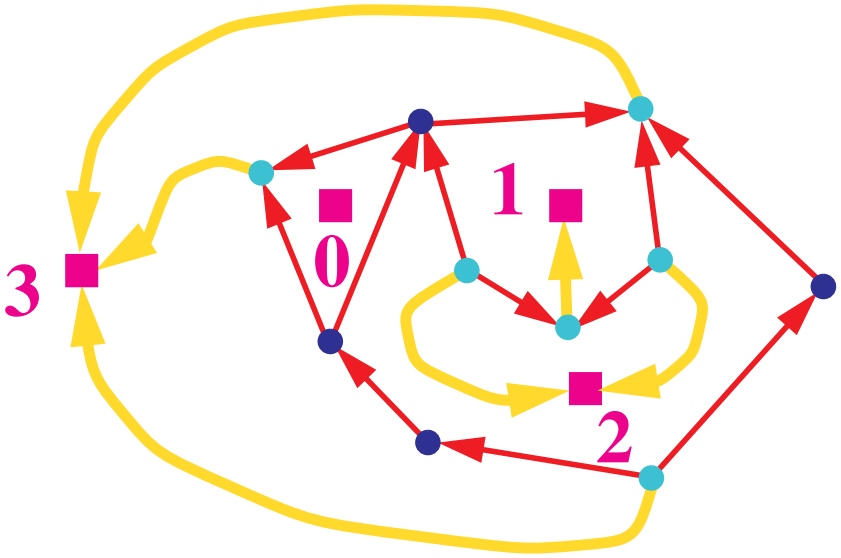
Assignment Model

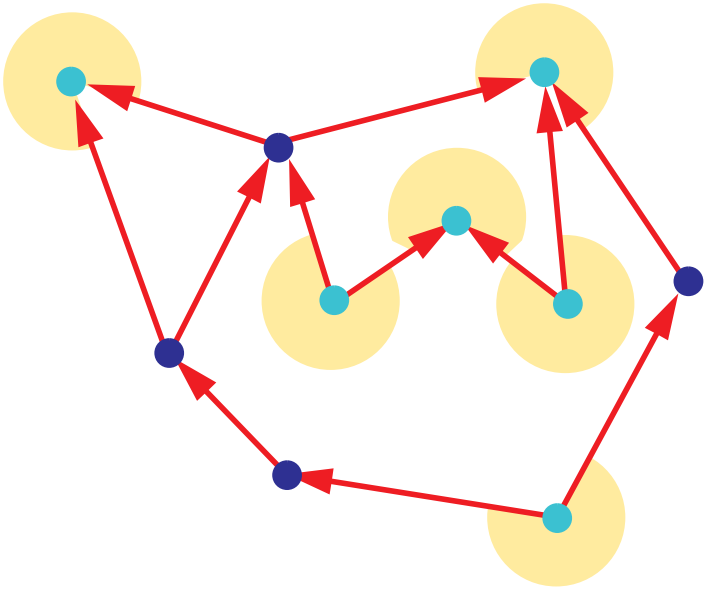
- assignment of poles (sources and sinks) to faces indicating which angles are large
- *consistent assignment* :
  - each pole v is assigned to exactly one face f incident on v
  - each internal face f has s(f) − 1 poles assigned to it
  - the external face h has s(h) + 1 poles assigned to it



Upward Planarity T esting

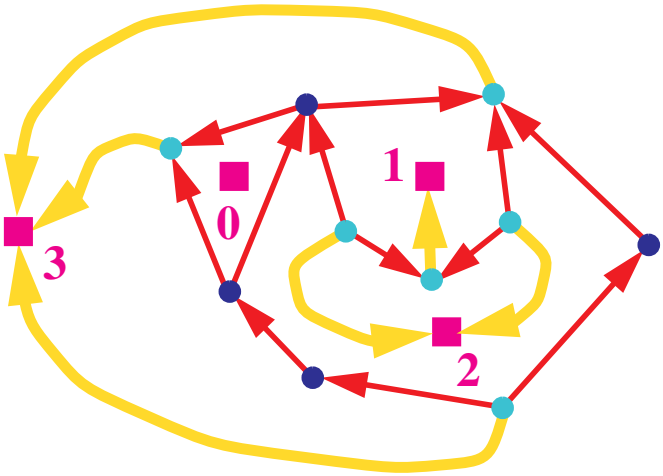
Theorem [Bertolazzi Di Battista 91]
An embedded digraph admits an upward drawing if and only if it admits a consistent assignment of poles to faces

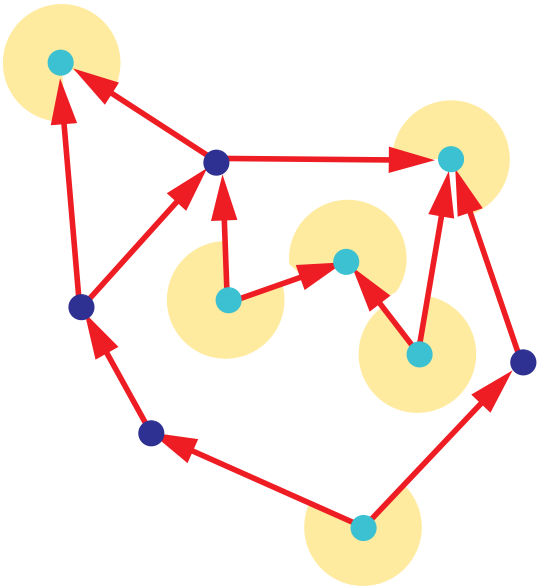


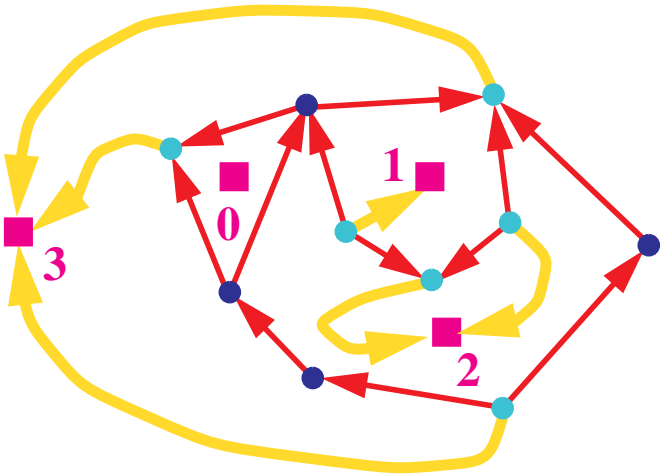


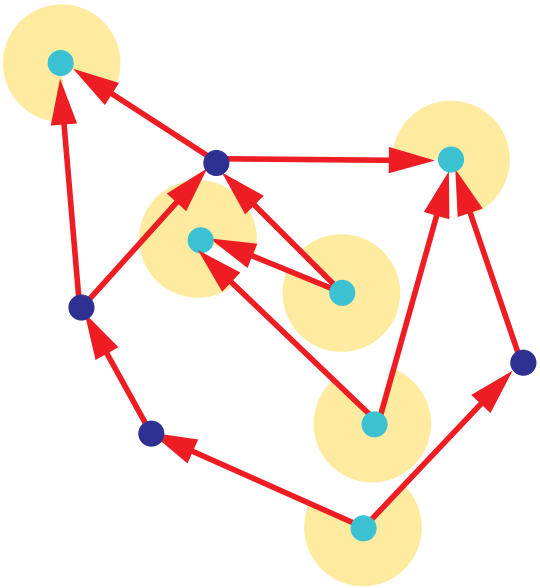
Assignments and Drawings

- distinct consistent assignments yield distinct upward embeddings









Flow Model

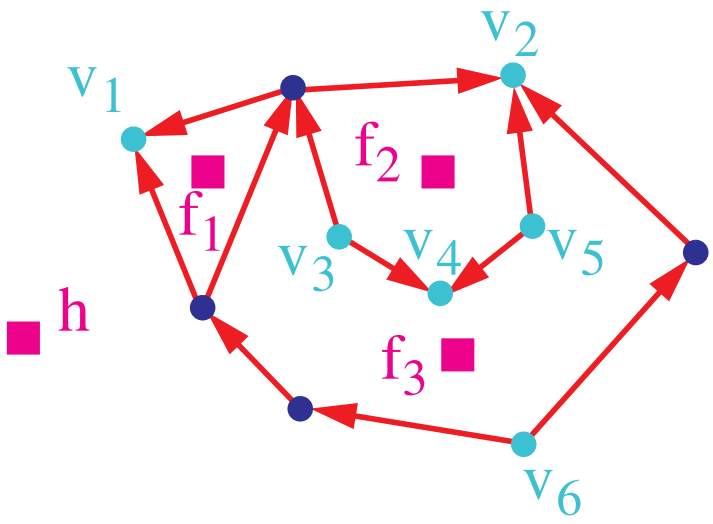
- computing a consistent assignment corresponds to finding a maximum flow in the pole-face incidence digraph, which takes $O(n^2)$ time

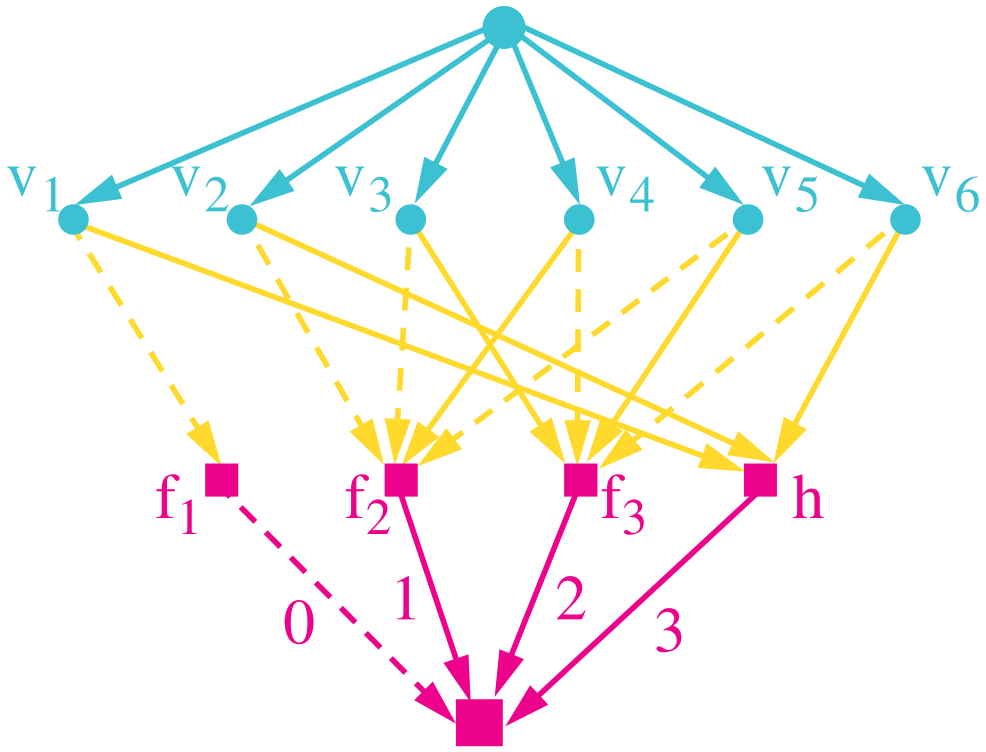
$s(f_1) = 1$

$s(f_2) = 2$

$s(f_3) = 3$

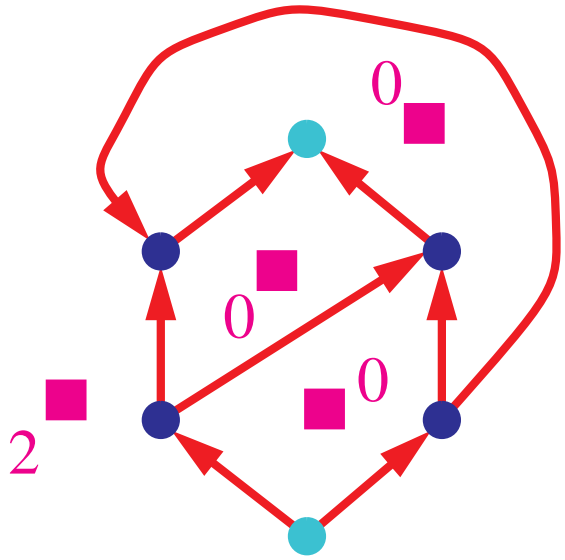
$s(h) = 2$



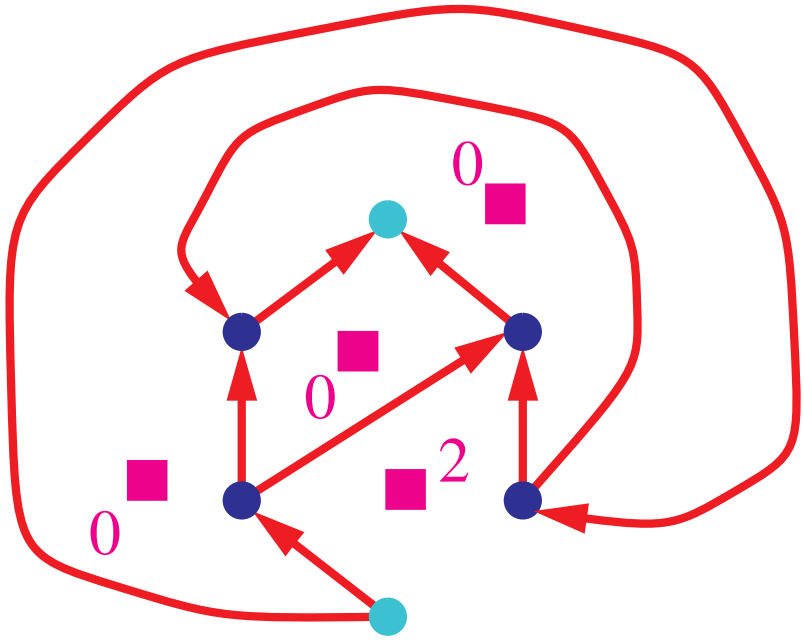


Proving that a Digraph is not Upward Planar

- the embedded digraph below has no consistent assignment ...

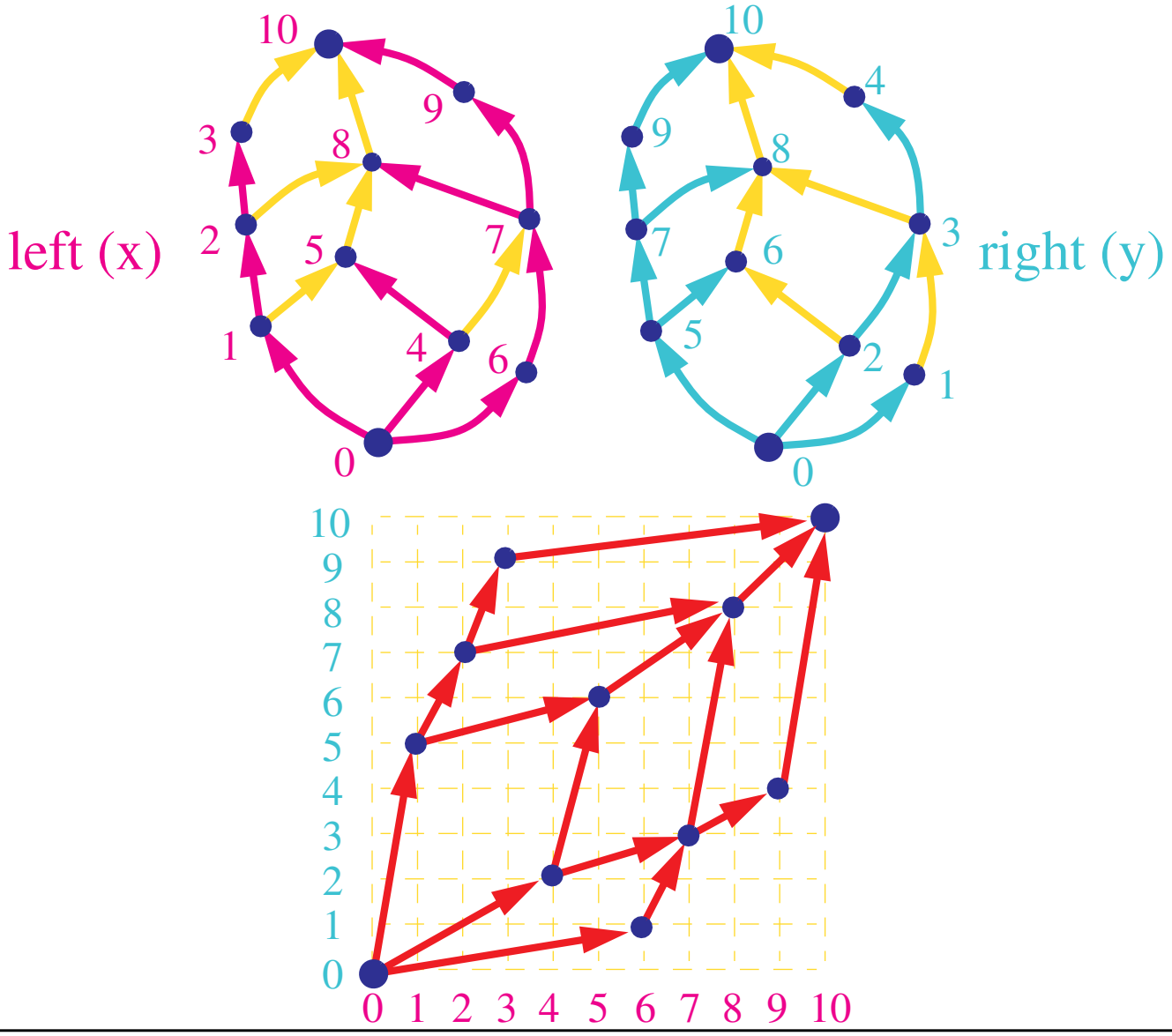


- ... even changing the external face



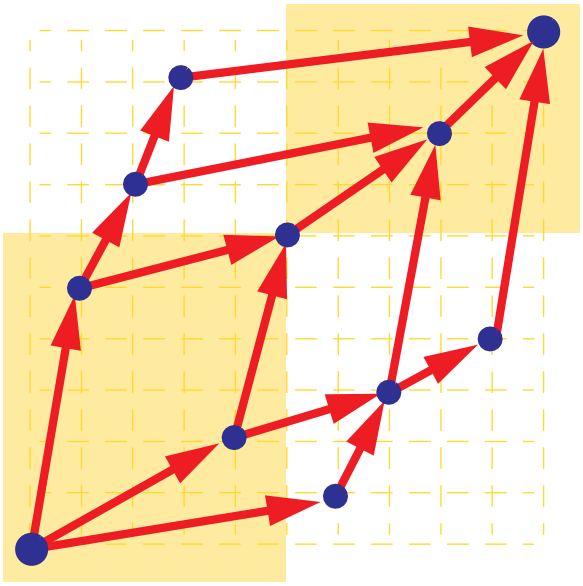
How to Construct Upward Planar Drawings

- Since an upward planar digraph is a subgraph of a ***planar st-digraph***, we only need to know how to draw planar st-digraphs
- If G is a planar st-digraph without transitive edges, we can use the ***left*** /***right*** numbering method to obtain a ***dominance drawing***:

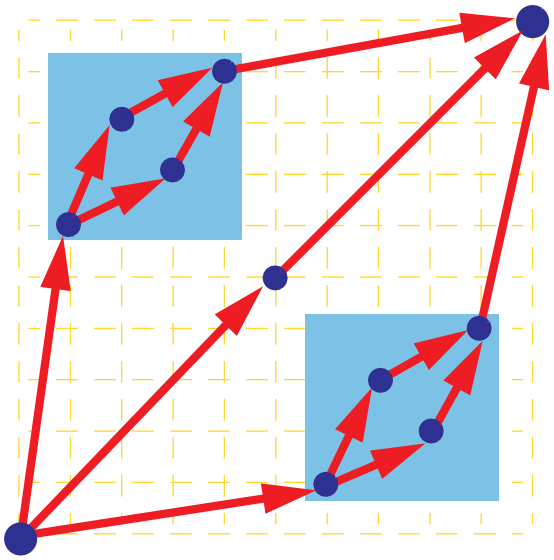


Properties of Dominance Drawings

- *Upward, planar , straight-line, O(n ^{2}) area*
- The *transitive closure* is visualized by the geometric dominace relation

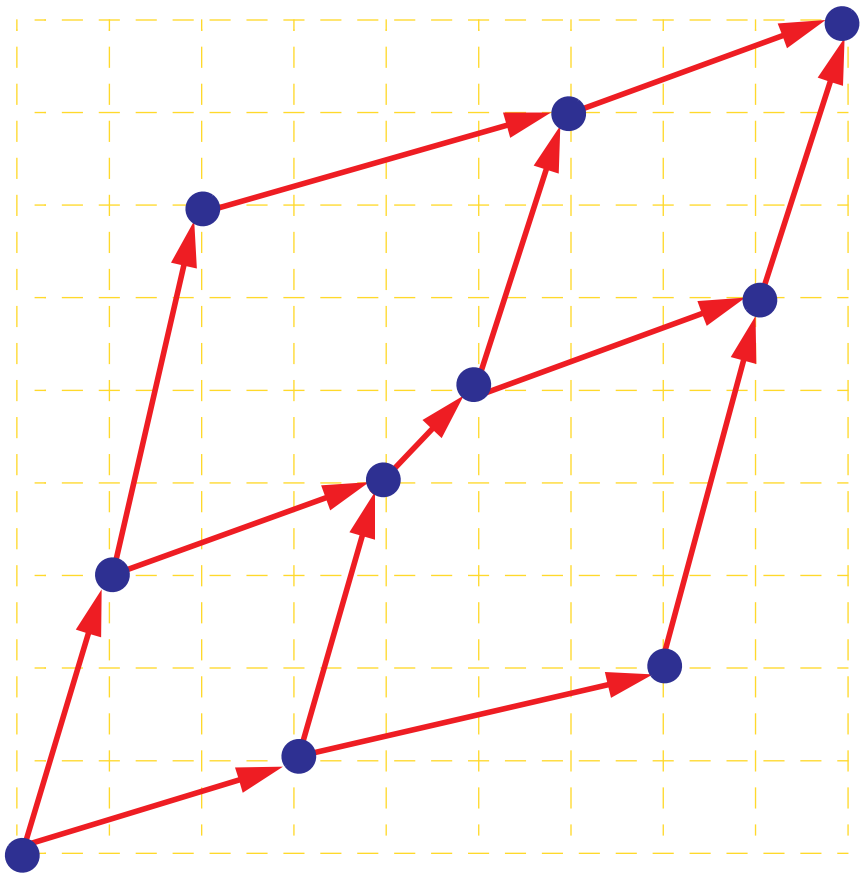


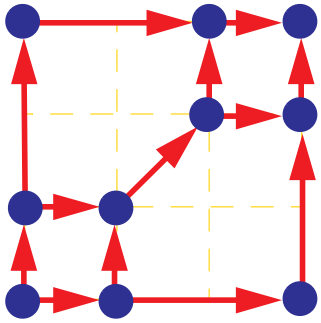
- *Symmetries* and *isomorphisms* of *st-components* are displayed



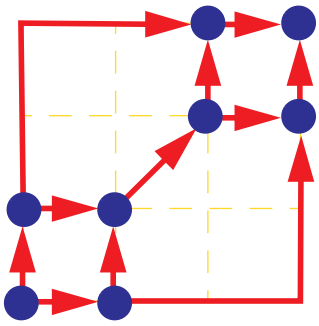
More on Dominance Drawings

- A variation of the left/right numbering yields dominance drawings with *optimal area*



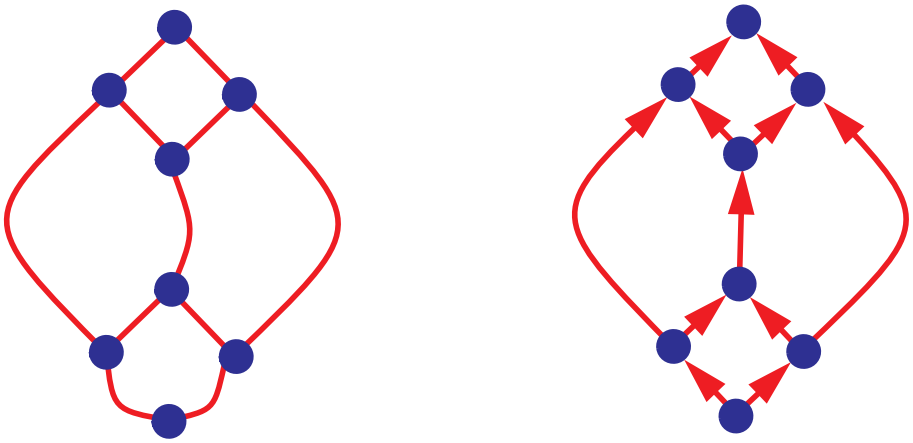


- Dummy vertices are inserted on transitive edges and are displayed as bends (upward planar polyline drawings)

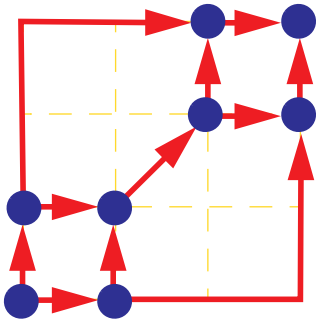


Planar Drawings of Graphs and Digraphs

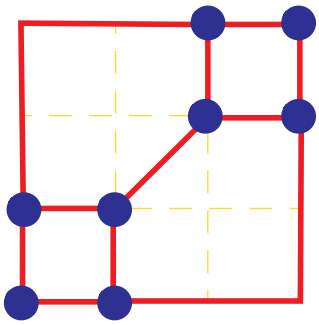
- We can use the techniques for dominance drawings also for undirected planar graphs:
  - orient G into a planar st-digraph G'



  - construct a dominance drawing of G'



  - erase arrows ...



General Undirected Graphs

Algorithmic Strategies for Drawing General Undirected Graphs

- ***Planarization method***
  - if the graph is nonplanar , ***make it planar*** ! (by placing dummy vertices at the crossings)
  - use one of the drawing algorithms for planar graphs

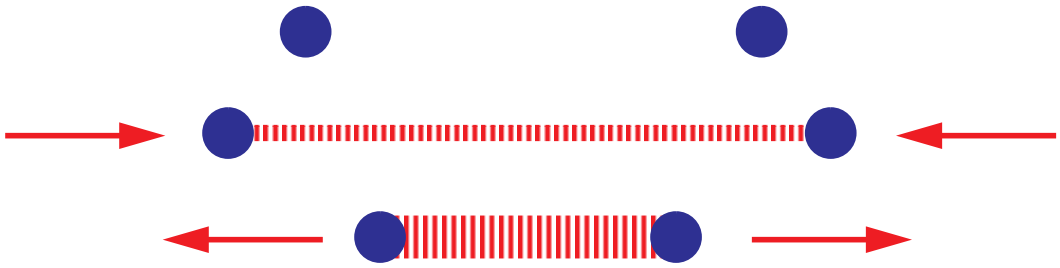
  e.g., GIOTTO [Tamassia Batini Di Battista 87]
- ***Orientation method***
  - ***orient*** the graph into a digraph
  - use one the drawing algorithms for digraphs
- ***Force-Directed method***
  - define a ***system of forces*** acting on the vertices and edges
  - find a ***minimum energy state*** (solve differential equations or simulate the evolution of the system)

  e.g., Spring Embedder [Eades 84]

The Spring Embedder

[Eades 1984]

- replace the edges by ***springs*** with unit natural length
- connect nonadjacent vertices with additional springs with infinite natural length
- recall that the springs attract the endpoints when stretched, and repel the endpoints when compressed



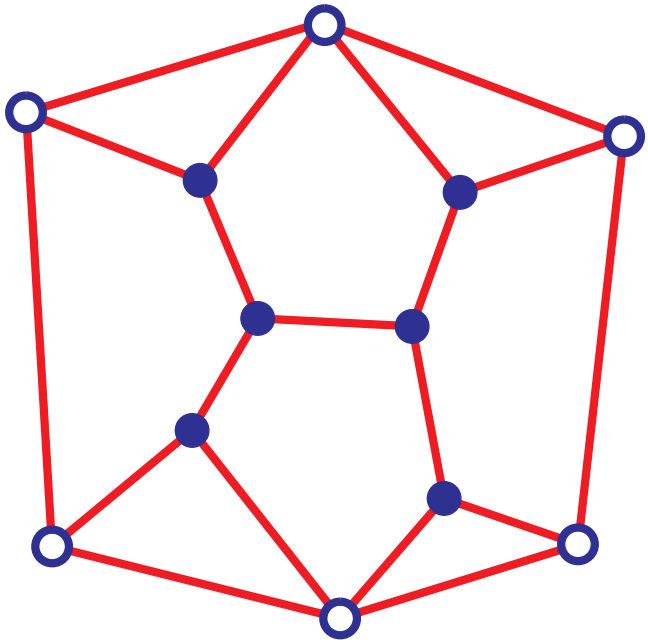
- start with an intital random placement of the vertices
- let the system go ... (assume there is ***friction*** so that a stable minimum energy state is eventually reached)

Other Force-Directed T echniques

- [Kamada Kawai 89]
  - the forces try to place vertices so that their ***geometric distance*** in the drawing is equal to their ***graph-theoretic distance***
  - for each pair of vertices (u,v) use a ***spring*** with natural length dist(u,v)
- [Fruchterman Reingold 90]
  - system of forces similar to that of ***subatomic particles*** and celestial bodies
  - given drawing region acts as wall
  - ***n-body simulation***
- [Davidson Harel 89]
  - ***energy function*** takes into account vertex distribution, edge-lengths, and edge-crossings
  - given drawing region acts as wall
  - ***simulated annealing***

Springs for Planar Graphs

- use springs with natural length 0, and attractive force proportional to the length
- pin down the vertices of the external face to form a convex polygon
- let the system go ...



- the final configuration is a state of minimum energy: min $\sum_{e} [\text{length}(e)]^2$
- equivalent to the ***barycentric mapping*** [Tutte 60]:

$$\mathbf{p}(v) = 1/\deg(v) \sum_{(v,w)} \mathbf{p}(w)$$

General Directed Graphs

Layering Method for Drawing General Directed Graphs

- ***Layer assignment:*** assign vertices to layers trying to minimize
  - *edge dilation*
  - *feedback edges*
- ***Placement:*** arrange vertices on each layer trying to minimize
  - *crossings*
- ***Routing:*** route edges trying to minimize
  - *bends*
- ***Fine tuning:*** improve the drawing with local modifications

[Carpano 80]

[Sugiyama T agawa T oda 81]

[Rowe Messinger et al. 87]

[Gansner North 88]

Systems

Some Graph Drawing Systems

- ***GraphEd***
  (University of Passau, Germany)
  - `ftp.uni-passau.de/pub/local/graphed/`
  - M. Himsolt `(himsolt@fmi.uni-passau.de)`
- ***DAG, DOT , NEA TO***
  (AT&T Bell Labs)
  - `research.att.com/dist/drawdag/`
  - S. North `(north@research.att.com)`
- ***Diagram Server***
  (University of Rome)
  - `infokit.dis.uniroma1.it/public/`
  - G. Di Battista
    `(dibattista@iasi.rm.cnr.it)`
- ***Graph Layout T oolkit***
  (Tom Sawyer Software)
  - B. Madden `(bmadden@tomsawyer.com)`

Challenges and Open Problems

- ***3D Graph Drawing***
  - [Cohen Eades Lin Ruskey 94]
  - [Reiss 94]
- ***Dynamic Graph Drawing***
  - [Cohen Di Battista T amassia T ollis 92]
  - [Hornick Miriyala T amassia 93]
- ***Aspect Ratio***
  - [Garg Goodrich T amassia 93]
  - [Chrobak Nakano 94]
- ***Planarization***
  - [Juenger Mutzel 93]
- ***Angular Resolution***
  - [Formann et al 91]
  - [Malitz Papakostas 92]
  - [Garg T amassia 94]