The Future of Graph Drawing
and a rhapsody

Peter Eades

Question: What is the future of Graph Drawing?
Answer: ... I’ll tell you later ...

But first: some constraints, and a brief history ....

Two different motivations for research

Extrovert
- I do research to make some impact on the world
  - I want to solve problems posed by others
  - I want to make the world a better place

Introvert
- I do research because I really want to know the answer
  - I am driven to uncover the truth
  - I am driven by my personal curiosity

Constrain: This talk mostly aimed at extroverts

What is the relationship between extroverted research and the real world?

Research that is inspired by the real world

Research that is useful for the real world

Note: Fundamental research ("theory") is usually inspired by the real world

A brief history

1970s

An idea emerges:
- Visualise graphs using a computer!
- Inspired by the need for better human decision making
- Implementations aimed at business decision making, circuit schematics, software diagrams, organisation charts, network protocols, and graph theory

Some key ideas defined
- Aesthetic criteria defined intuitively
- Key scientific challenge defined: layout to optimise aesthetic criteria
1980s

Exciting algorithms and geometry:

- Many fundamental graph layout algorithms designed, enunciated, implemented, and analysed
- Extra inspirational ideas from graph theory, geometry, and algorithmics
- Planarity becomes a central concept

1990s

Maturity

- Graph Drawing matures as a discipline
- The Graph Drawing Conference begins
- The academic Graph Drawing “community” emerges

More demand

- High data volumes increase demand for visualization
- Small companies appear

More communities

- Information Visualization discipline appears

2000s

Even more demand

- Data volumes become higher than ever imagined, and demand for visualization increases accordingly
- New customers: systems biology, social networks, security, ...

Better engineering

- More usable products, both free and commercial
- More companies started, older companies become stable

Invisibility

- Graph drawing algorithms become invisible in vertical tool

Graph Drawing is successful

- Algorithms for graph drawing are used in many industries:
  - Biotechnology
  - Software engineering
  - Networks
  - Business intelligence
  - Security

- Graph drawing software is an industry:
  - Employs about 500 FTE people
  - Market worth up to $100,000,000 per year
  - About 100 FTE researchers

- Graph Drawing is scientifically significant
  - Graph Drawing has provided an elegant algorithmic approach to the centuries-old interplay between combinatorial and geometric structures
  - GD2010 is a "rank A" conference
  - Graph drawing papers appear in many top journals and conference proceedings
“Graph Drawing is *the* big success story in information visualization”

Stephen North, September 2010

Subjective Observation:

Graph Drawing is connected to many other disciplines.

Keith Nesbitt 2003: use metro map metaphor for abstract data connections

Unsolved problem: make a really good metro map of graph drawing connections
- Based on real data
- Each metro line represents a community, as instantiated by a conference
- Each station joining different lines indicates that there are papers co-authored by people in different communities

$10^*$
Unsolved problem: Create algorithms and systems to draw (ordered) hypergraphs in the metro-map metaphor.

Interview with a Very Experienced Industry Researcher in a Telco, Sept 14, 2010

Interviewer: “What are the most useful results from the Graph Drawing researchers in the last ten years?”
Industry Researcher: <thinking>
...<60 seconds of silence>
<suddenly> “Scale! Those force directed algorithms run much faster now than they did around 2000, using the Koren/Quigley/Walshaw methods!”
<more thinking>
“For force directed methods, visual complexity is now a problem...”

Interview with the CEO and CTO of a Graph Drawing software company, Sept 15, 2010

CTO: “Yes. We implemented some of them. We had to fix them a bit, but they gave us much better runtimes.”
Interviewer: “Any other useful results from the Graph Drawing researchers in the last ten years?”
Industry people: <thinking in silence> ....
CEO: “I know one thing: nested drawings. This models the data better than previously.”
Interviewer: <Smiles, knowing that nested drawings have been around much longer> “Anything else?”

Interview with the CEO and CTO of a Graph Drawing software company, Sept 15, 2010

CEO: “Well, our tools are much better engineered than ten years ago. We’ve spent a lot of energy...”
Interviewer: <interrupts> “Yes, but I guess that kind of thing didn’t come from the Graph Drawing research community?”
CEO: “Oh, I guess not.”
<thinking> ...
CTO: “I don’t think we have used any of the other results. They are certainly interesting, but...”
GD Conference: Citations

Data source: citeceer

Unsubstantiated claim: the quality of graph drawings isn’t getting any better

Data source: none

Open Problem: Is it worth the money?

Average cost of Graph Drawing Researcher per year (assuming $100Kpa, 2.5 oncost multiplier) = $250K. There are probably about 100 FTE GD researchers in the world.

The cost of the past ten years of Graph Drawing Research is about $250M

Open Problem: Has the world made a profit from this investment yet? If not, how long will it take to get some ROI?

Open Problem: What kind of community is GD?
The GD community
- Every node is a paper at the GD conference
- Edge from A to B if A cites B

A high (academic) impact community
- Large indegree
- Influences other fields
- Fundamental area of research

An engineering community
- Large outdegree
- Uses many other fields to produce solutions for problems
- Perhaps has commercial impact?

An island community
- Not much connection to the outside
- Perhaps has no impact?
- Introspective community?

Open Problem:
- What kind of community is GD?
- Has its character changed over time?
- Can you show this in a picture (or a time-lapse animation) of citation network(s)?

Unsupported conjecture:
- Planarity has about 5 years to either live or die
1930s: Fary Theorem
- Straight-line drawings exist

1960s: Tutte’s algorithm
- A straight-line drawing algorithm

1970s: Read’s algorithm
- Linear time straight-line drawing

1984: Tamassia algorithm
- Minimum number of bends

1987: Tamassia-Tollis algorithms
- Visibility drawing, upward planarity

1989: de Frassieux - Pach - Pollack Theorem
- Quadratic area straight-line drawing

Conversation at GD1994, between an academic delegate and an industry delegate

Industry delegate: “Why are you guys so obsessed with planarity? Most graphs that I want to draw aren’t planar.”

Academic delegate: “Well, planarity is a central concept even for non-planar graphs. To be able to draw general graphs, we find a topology with a small number of edge crossings, model this topology as a planar graph, and draw that planar graph.”

Industry delegate: “Sounds good. But I don’t know how to solve these sub-problems, for example, how to find a topology with a small number of crossings”

Academic delegate: “These problems are fairly difficult and we don’t have perfect solutions. But we expect a few more years of research and we will get good results”

Industry delegate: “cheerily: “Sounds good ... I guess I’m looking forward to it”

Mid 1990s: Mutzel’s thesis
- Crossing reduction by integer linear programming

1995: Purchase experiments
- Crossings really do inhibit understanding

Late 1990s:
- Many beautiful papers on drawing planar graphs
- Good crossing reduction methods

Early 2000s:
- Many more beautiful papers on planar graphs

Subjective Observation: The graph drawing community is obsessed with planarity.

Interview with the CEO and CIO of a Graph Visualization Company, Sept 15 2010

Interviewer: “Do you use planar graph drawing algorithms?”

CEO: “No.”

Interviewer: “Why not?”

CEO: “Too much white space. Stability is a problem. Too difficult to integrate constraints. Incremental planar drawing doesn’t work well.”

CIO: “laughs”

Interview with a Very Senior Software Engineer in a Very Large Company that has a Very Small Section that produces visualization software, Sept 23 2010

Interviewer: “Do your graph visualization tools use planar graph drawing algorithms?”

Software Engineer: “Our customers want hierarchical layout first, and hierarchical layout second, and then they want hierarchical layout. We also have spring algorithms, but I’m not sure whether they use them.”

Interviewer: “Yes, but maybe deep down in your software there is a planarity algorithm, or maybe a planar graph drawing algorithm?”

Software Engineer: “No. No planarity.”

Note: Yworks does use planar graph drawing algorithms

Interview with the CEO and CIO of a Graph Visualization Company, Sept 15 2010

Interviewer: “Do you use planar graph drawing algorithms?”

CEO: “No.”

Interviewer: “Why not?”

CEO: “Too much white space. Stability is a problem. Too difficult to integrate constraints. Incremental planar drawing doesn’t work well.”

CIO: “laughs”

Subjective observation: The graph drawing community is obsessed with planarity.

Interview with a Very Senior Software Engineer in a Very Large Company that has a Very Small Section that produces visualization software, Sept 23 2010

Interviewer: “Do your graph visualization tools use planar graph drawing algorithms?”

Software Engineer: “Our customers want hierarchical layout first, and hierarchical layout second, and then they want hierarchical layout. We also have spring algorithms, but I’m not sure whether they use them.”

Interviewer: “Yes, but maybe deep down in your software there is a planarity algorithm, or maybe a planar graph drawing algorithm?”

Software Engineer: “No. No planarity.”

Note: Yworks does use planar graph drawing algorithms

Interview with a Very Senior Software Engineer in a Very Large Company that has a Very Small Section that produces visualization software, Sept 23 2010

Interviewer: “Do your graph visualization tools use planar graph drawing algorithms?”

Software Engineer: “Our customers want hierarchical layout first, and hierarchical layout second, and then they want hierarchical layout. We also have spring algorithms, but I’m not sure whether they use them.”

Interviewer: “Yes, but maybe deep down in your software there is a planarity algorithm, or maybe a planar graph drawing algorithm?”

Software Engineer: “No. No planarity.”

Note: Yworks does use planar graph drawing algorithms

Interview with a Very Senior Software Engineer in a Very Large Company that has a Very Small Section that produces visualization software, Sept 23 2010

Interviewer: “Do your graph visualization tools use planar graph drawing algorithms?”

Software Engineer: “Our customers want hierarchical layout first, and hierarchical layout second, and then they want hierarchical layout. We also have spring algorithms, but I’m not sure whether they use them.”

Interviewer: “Yes, but maybe deep down in your software there is a planarity algorithm, or maybe a planar graph drawing algorithm?”

Software Engineer: “No. No planarity.”

Note: Yworks does use planar graph drawing algorithms

Interview with the CEO and CIO of a Graph Visualization Company, Sept 15 2010

Interviewer: “Do you use planar graph drawing algorithms?”

CEO: “No.”

Interviewer: “Why not?”

CEO: “Too much white space. Stability is a problem. Too difficult to integrate constraints. Incremental planar drawing doesn’t work well.”

CIO: “laughs”

Interview with a Very Senior Software Engineer in a Very Large Company that has a Very Small Section that produces visualization software, Sept 23 2010

Interviewer: “Do your graph visualization tools use planar graph drawing algorithms?”

Software Engineer: “Our customers want hierarchical layout first, and hierarchical layout second, and then they want hierarchical layout. We also have spring algorithms, but I’m not sure whether they use them.”

Interviewer: “Yes, but maybe deep down in your software there is a planarity algorithm, or maybe a planar graph drawing algorithm?”

Software Engineer: “No. No planarity.”

Note: Yworks does use planar graph drawing algorithms

Interview with the CEO and CIO of a Graph Visualization Company, Sept 15 2010

Interviewer: “Do you use planar graph drawing algorithms?”

CEO: “No.”

Interviewer: “Why not?”

CEO: “Too much white space. Stability is a problem. Too difficult to integrate constraints. Incremental planar drawing doesn’t work well.”

CIO: “laughs”

Interview with a Very Senior Software Engineer in a Very Large Company that has a Very Small Section that produces visualization software, Sept 23 2010

Interviewer: “Do your graph visualization tools use planar graph drawing algorithms?”

Software Engineer: “Our customers want hierarchical layout first, and hierarchical layout second, and then they want hierarchical layout. We also have spring algorithms, but I’m not sure whether they use them.”

Interviewer: “Yes, but maybe deep down in your software there is a planarity algorithm, or maybe a planar graph drawing algorithm?”

Software Engineer: “No. No planarity.”

Note: Yworks does use planar graph drawing algorithms

Interview with the CEO and CIO of a Graph Visualization Company, Sept 15 2010

Interviewer: “Do you use planar graph drawing algorithms?”

CEO: “No.”

Interviewer: “Why not?”

CEO: “Too much white space. Stability is a problem. Too difficult to integrate constraints. Incremental planar drawing doesn’t work well.”

CIO: “laughs”

Interview with a Very Senior Software Engineer in a Very Large Company that has a Very Small Section that produces visualization software, Sept 23 2010

Interviewer: “Do your graph visualization tools use planar graph drawing algorithms?”

Software Engineer: “Our customers want hierarchical layout first, and hierarchical layout second, and then they want hierarchical layout. We also have spring algorithms, but I’m not sure whether they use them.”

Interviewer: “Yes, but maybe deep down in your software there is a planarity algorithm, or maybe a planar graph drawing algorithm?”

Software Engineer: “No. No planarity.”

Note: Yworks does use planar graph drawing algorithms

Interview with the CEO and CIO of a Graph Visualization Company, Sept 15 2010

Interviewer: “Do you use planar graph drawing algorithms?”

CEO: “No.”

Interviewer: “Why not?”

CEO: “Too much white space. Stability is a problem. Too difficult to integrate constraints. Incremental planar drawing doesn’t work well.”

CIO: “laughs”
Outlandish Claim:
The graph drawing community can contribute a lot to solving the scale problem.

The scale problem currently drives much of Computer Science:
- Data sets are growing at a faster rate than the human ability to understand them.
- Businesses (and sciences) believe that their data sets contain useful information, and they want to get some business (or scientific) value out of these data sets.

For Graph Drawing, there are two facets of the scale problem:
1. Computational complexity
   - Efficiency
   - Runtime
     - We need more efficient algorithms
2. Visual complexity
   - Effectiveness
   - Readability
     - We need better ways to untangle large graphs

Graph Drawing has proposed three approaches to the scale problem:
1. Use 3D: spread the data over a third dimension
2. Use interaction: spread the data over time
3. Use clustering: view an abstraction of the data
Well supported claim: 3D is almost dead.

Conversation between an Australian academic and a Canadian IBM research manager, 1992

IBM Research Manager: “Graphics cards capable of fast 3D rendering will soon become commodity items. They will give an unprecedented ability to draw diagrams in 3D. This is completely new territory.”

Academic: “What?”

IBM Research Manager: “Every PC will be able to do 3D. In IBM we do lots of graph drawing to model software. We think that 3D graph drawing will become commonplace.”

Academic: “Really?”

IBM Research Manager: “There’s more space in 3D, edge crossings can be avoided in 3D, navigation in 3D is natural.”

Academic: “Really?”

IBM Research Manager: “If you do a 3D graph drawing project, then IBM will fund it.”

Academic: “Let’s do it!”

1980s 2D Theorem: Every planar graph with maximum degree 4 admits a 2D orthogonal drawing with no edge crossings and at most 4 bends per edge.

1996 3D Theorem: Every graph (even if it is nonplanar) with maximum degree 6 admits a 3D orthogonal drawing with no edge crossings and a constant number of bends per edge.

1990 – 2005:
- Many theorems on 3D
- Many metaphors
- Many research grants
- Many experiments
- A start-up company

Mostly, 3D failed.
2004*: success with 2.5D?

- Colin Ware: “use 3D with a 2D attitude”
- Tim Dwyer: “use the third dimension for a single simple parameter (eg time)”
- Seokhee Hong: “Multiplane method”

Multiplane method
1. Partition the graph
2. Draw each part on a 2D manifold in 3D
3. Connect the parts with inter-manifold edges

 unsupported conjecture: There is some hope of life for 2.5D graph drawing

 unsupported conjecture: There are many interesting algorithmic and geometric problems for graph drawing in the multiplane style.
Graph Drawing has proposed three approaches to the scale problem:

1. **Use 3D**: spread the data over a third dimension
2. **Use interaction**: spread the data over time
3. **Use clustering**: view an abstraction of the data

In practice . . . . . . .

**Many iterations**

Interaction flow

1. The human looks at key frame $F_i$.
2. The human thinks.
3. The human clicks on something.
4. System computes new key frame $F_{i+1}$.
5. System computes in-betweening animation from key frame $F_i$ to key frame $F_{i+1}$.
6. System displays animated transition from $F_i$ to $F_{i+1}$.
7. $i++$
8. Go to 1.

**Outrageous claim**: all graph drawing algorithms need to be designed with interaction flow in mind

Interaction can solve both problems:

- **Computational complexity**:
  - The layout is only computed for the key frame (a relatively small graph)
  - The "user think time" can be used for computation

- **Visual complexity**:
  - At any one time, only a small graph is on the screen

Interaction also raises some problems:

- **Cognitive complexity**:
  - The user must remember stuff from one key frame to the next
  - "mental map" problem

Unsupported conjecture: Interaction flow poses many interesting problems for graph drawing
Graph Drawing has proposed three approaches to the scale problem:

1. **Use 3D**: spread the data over a third dimension
2. **Use interactor**: spread the data over time
3. **Use clustering**: view an abstraction of the data

A **clustered graph** $C=(G,T)$ consists of:
- a classical graph $G$, and
- a tree $T$ such that the leaves of the tree $T$ are the vertices of $G$.

The tree $T$ defines a clustering of the vertices of $G$.

---

We can only draw a part of a huge graph at a time.

What part shall we draw?

- A **précis**: a graph formed from an antichain in the cluster tree.

A précis forms an abstraction of the data set.
A précis is a graph, and can be drawn with the usual graph drawing algorithms.

Drill down interaction

The basic human interaction with a clustered graph is drill down:
- "Open" a node to see what it contains, that is, i.e., replace a node in the antichain with its children.

Also, we need drill up:
- "Close" a set of nodes to make the picture simpler i.e., replace a set of siblings in the antichain with their parent.

Note: In practical systems:
- the human performs drill-down
- the system performs drill-up

For interaction:
- "Drill down" on node X changes the size of X
- "Drill up" should be performed by the system, not by the user
- Nodes must move to accommodate change in size of node X
- The new picture must be nice in the usual graph drawing sense
- The mental map must be preserved:
  - Preserve orthogonal ordering
  - Preserve proximity
  - Preserve topology

Partially supported conjecture:
Graph drawing researchers can design algorithms to give good drill-down/up interaction.
Totally planar clustered graph drawing

Say \( C \) is a clustered graph with cluster tree \( T = (V, E) \) and underlying graph \( G = (U, F) \).

A drawing is a mapping \( p : V \rightarrow \mathbb{R}^2 \) (edges are straight lines).

A drawing is totally planar if

- For every node \( u \) of \( T \), all vertices inside the convex hull of the descendents of \( u \) are descendents of \( u \).
- And every précis is planar.

Open problem: Does every c-planar graph have a totally planar drawing?

Equivalently: Almost equivalently:

- Open problem: Does every c-planar graph have straight-line multilevel drawing in which:
  - Every level is planar
  - The projection of a \( i \)-th level vertex onto level \( i-1 \) lies within the convex hull of its children

Unsubstantiated claim:
The only chance for a solution to the scale problem for graph drawing lies in the algorithmics and geometry of interaction and clusters.
A personal timeline

- Early 1980s: I used intuition and introspection to evaluate graph drawings
- Late 1980s: I read Shneiderman’s book
  - Quality of an interface is a scientifically measurable function
  - Task time, error rate, etc
- Now: I think graph drawings need to be beautiful as well as useful.
- Katy Borner, 2009: “In order to change behaviour, data graphics have to touch people intellectually and emotionally.”
Outlandish claim:
Graph drawings researchers have the skills needed to create graph drawing art

George Birkoff, 1933:
- Beauty can be measured as a ratio $M = O/C$
  - $O =$ “order” $\rightarrow$ Kolmogorov complexity
  - $C =$ “complexity” $\rightarrow$ Shannon complexity

Graph drawing problem as an optimization problem
- A number of objective functions $f: \text{Drawings} \rightarrow \mathbb{R}$
- Given a graph $G$, find a drawing $p(G)$ that optimizes $f$

Unsupported conjecture: Using Birkoff-style functions, graph drawing algorithms can produce art.

Size of Star represents the amount of email
Each person is a star
Distance between two stars represents closeness

Social circle
Currently:
Graph drawing aims for
- 100% information display, and
- 0% art

Outlandish suggestion: Graph drawing should have a range of methods, aiming for \( x\% \) information display and \((100-x)\%\) art, for all \(0 \leq x \leq 100\)

Open problem: Use integer linear programming to produce valuable graph drawing art.

Force directed methods
- There have been many experiments
- A few more theorems would be good

One theorem has been proved: if a graph has the right automorphisms, then there is a local minimum of a spring drawing that is symmetric.

Some theory exists
- Combinatorial rigidity theory
- Theory of multidimensional scaling

But there are many questions that I don’t know the answer

Warning: this is introspective research!

Open problem:
- Why do force directed methods work?

Open problem:
- What is the time complexity of a spring algorithm?

Open problem:
- How many local minima are there?

Open problem:
- How close are Euclidean distances to the graph theoretic distances?

Open problem:
- Do force directed methods give bounded crossings most of the time?

This is the end of the rhapsody of unsupported conjectures, subjective observations, outlandish claims, a few plain lies, and open problems.
Back to the main question:
* What is the future of Graph Drawing?
* Where does the road lead?
Question: What is the future of Graph Drawing?

......

Sorry, I’ve run out of time.

Question: What is the future of Graph Drawing?

Answer: it’s up to you.

Many thanks