Algorithms for Graph Visualization
Layered Layout – Part II

Torsten Ueckerdt
22.01.2020
Layered Layout

**Given:** directed graph $D = (V, A)$

**Find:** drawing of $D$ that emphasizes the hierarchy by positioning nodes on horizontal layers
Layered Layout

**Given:** directed graph \( D = (V, A) \)

**Find:** drawing of \( D \) that emphasizes the hierarchy by positioning nodes on horizontal layers

**Criteria:**
- many edges pointing to the same direction
- few layers or limited number of nodes per layer
- preferably few edge crossings
- nodes distributed evenly
- edges preferably straight and short
Sugiyama Framework \(\text{\cite{Sugiyama, Tagawa, Toda 1981}}\)

- **given**
- **resolve cycles**
- **layer assignment**
- **crossing minimization**
- **node positioning**
- **edge drawing**

This diagram illustrates the Sugiyama Framework for visualizing graphs, which includes steps like layer assignment, resolving cycles, and minimizing crossings. Each step is shown in a sequential manner, moving from the initial graph to the final, optimized visualization.
The Sugiyama Framework (Sugiyama, Tagawa, Toda 1981) involves several steps:

1. **Layered Layouts**
   - **given**
   - **resolve cycles**
   - **layer assignment**

2. **Crossing Minimization**
   - **crossing minimization**

3. **Node Positioning**
   - **node positioning**

4. **Edge Drawing**
   - **edge drawing**

The process begins with the given graph and proceeds through the steps of resolving cycles, layer assignment, crossing minimization, and node positioning, finally leading to the edge drawing.
Step 3: Crossing Minimization
Problem Statement

**Given:** DAG $D = (V, A)$, nodes are partitioned in disjoint layers

**Find:** Order of the nodes on each layer, so that the number of crossing is minimized
Problem Statement

Given: DAG $D = (V, A)$, nodes are partitioned in disjoint layers

Find: Order of the nodes on each layer, so that the number of crossing is minimized

Properties

• Problem is NP-hard even for two layers
  (Bipartite Crossing Number [Garey, Johnson ’83])
• No approach over several layers simultaneously
• Usually iterative optimization for two adjacent layers
• For that: insert dummy nodes at the intersection of edges with layers
One-sided Crossing Minimization (OSCM)

**Given:** 2-layered graph $G = (L_1, L_2, E)$ and ordering of the nodes $x_1$ of $L_1$

**Find:** Node ordering $x_2$ of $L_2$, such that the number of crossings is minimized
One-sided Crossing Minimization (OSCM)

**Given:** 2-layered graph \( G = (L_1, L_2, E) \) and ordering of the nodes \( x_1 \) of \( L_1 \)

**Find:** Node ordering \( x_2 \) of \( L_2 \), such that the number of crossings is minimized

**Observation:**
- The number of crossings in a 2-layered drawing of \( G \) depends only on the ordering of the nodes, not on the exact positions.
- For \( u, v \in L_2 \) the number of crossings among their incident edges depends on whether \( x_2(u) < x_2(v) \) or \( x_2(v) < x_2(u) \) and not on the ordering of other vertices.
One-sided Crossing Minimization (OSCM)

**Given:** 2-layered graph \( G = (L_1, L_2, E) \) and ordering of the nodes \( x_1 \) of \( L_1 \)

**Find:** Node ordering \( x_2 \) of \( L_2 \), such that the number of crossings is minimized

**Observation:**
- The number of crossings in a 2-layered drawing of \( G \) depends only on the ordering of the nodes, not on the exact positions.
- For \( u, v \in L_2 \) the number of crossings among their incident edges depends on whether \( x_2(u) < x_2(v) \) or \( x_2(v) < x_2(u) \) and not on the ordering of other vertices.

**Def:** \( c_{uv} := |\{(uw,vz) : w \in N(u), z \in N(v), x_1(z) < x_1(w)\}| \)

\( c_{uv} = 5 \)
One-sided Crossing Minimization (OSCM)

**Given:** 2-layered graph \( G = (L_1, L_2, E) \) and ordering of the nodes \( x_1 \) of \( L_1 \)

**Find:** Node ordering \( x_2 \) of \( L_2 \), such that the number of crossings is minimized

**Observation:**
- The number of crossings in a 2-layered drawing of \( G \) depends only on the ordering of the nodes, not on the exact positions
- for \( u, v \in L_2 \) the number of crossings among their incident edges depends on whether \( x_2(u) < x_2(v) \) or \( x_2(v) < x_2(u) \) and not on the ordering of other vertices

**Def:** \( c_{uv} := |\{(uw,vz) : w \in N(u), z \in N(v), x_1(z) < x_1(w)\}| \)

\[
\begin{align*}
  c_{uv} &= 5 \\
  c_{vu} &= 7
\end{align*}
\]
Further Properties

**Def:** Number of crossings in $G$ with orders $x_1$ and $x_2$ for $L_1$ and $L_2$ is denoted by $\text{cr}(G, x_1, x_2)$; 
$\Rightarrow$ for fixed $x_1$ we have $\text{opt}(G, x_1) = \min_{x_2} \text{cr}(G, x_1, x_2)$

**Lemma 1:** Each of the following holds:

- $\text{cr}(G, x_1, x_2) = \sum_{x_2(u) < x_2(v)} c_{uv}$
- $\text{opt}(G, x_1) \geq \sum_{\{u,v\}} \min\{c_{uv}, c_{vu}\}$
Further Properties

**Def:** Number of crossings in $G$ with orders $x_1$ and $x_2$ for $L_1$ and $L_2$ is denoted by $\text{cr}(G, x_1, x_2)$; 
⇒ for fixed $x_1$ we have $\text{opt}(G, x_1) = \min_{x_2} \text{cr}(G, x_1, x_2)$

**Lemma 1:** Each of the following holds:
- $\text{cr}(G, x_1, x_2) = \sum_{x_2(u)<x_2(v)} c_{uv}$
- $\text{opt}(G, x_1) \geq \sum_{\{u,v\}} \min\{c_{uv}, c_{vu}\}$

Efficient computation of $\text{cr}(G, x_1, x_2)$ see Exercise.
Further Properties

**Def:** Number of crossings in $G$ with orders $x_1$ and $x_2$ for $L_1$ and $L_2$ is denoted by $cr(G, x_1, x_2)$; \[ \Rightarrow \text{for fixed } x_1 \text{ we have } \text{opt}(G, x_1) = \min_{x_2} cr(G, x_1, x_2) \]

**Lemma 1:** Each of the following holds:
- $cr(G, x_1, x_2) = \sum_{x_2(u) < x_2(v)} c_{uv}$
- $\text{opt}(G, x_1) \geq \sum_{\{u,v\}} \min\{c_{uv}, c_{vu}\}$

Efficient computation of $cr(G, x_1, x_2)$ see Exercise.

Think for a minute and then share

Can you find an example where the second inequality is strict?

3 min
Iterative Crossing Minimization

Let \( G = (V, E) \) be a DAG with layers \( L_1, \ldots, L_h \).

1. compute an ordering \( x_1 \) for layer \( L_1 \)
2. for \( i = 1, \ldots, h - 1 \) consider layers \( L_i \) and \( L_{i+1} \) and minimize \( \text{cr}(G, x_i, x_{i+1}) \) with fixed \( x_i \) (→ OSCM)
3. for \( i = h - 1, \ldots, 1 \) consider layers \( L_{i+1} \) and \( L_i \) and minimize \( \text{cr}(G, x_i, x_{i+1}) \) with fixed \( x_{i+1} \) (→ OSCM)
4. repeat (2) and (3) until no further improvement happens
5. repeat steps (1)–(4) with another \( x_1 \)
6. return the best found solution
Iterative Crossing Minimization

Let $G = (V, E)$ be a DAG with layers $L_1, \ldots, L_h$.

(1) compute an ordering $x_1$ for layer $L_1$
(2) for $i = 1, \ldots, h - 1$ consider layers $L_i$ and $L_{i+1}$ and minimize $cr(G, x_i, x_{i+1})$ with fixed $x_i$ ($\rightarrow$ OSCM)
(3) for $i = h - 1, \ldots, 1$ consider layers $L_{i+1}$ and $L_i$ and minimize $cr(G, x_i, x_{i+1})$ with fixed $x_{i+1}$ ($\rightarrow$ OSCM)
(4) repeat (2) and (3) until no further improvement happens
(5) repeat steps (1)–(4) with another $x_1$
(6) return the best found solution

Theorem 1: The One-Sided Crossing Minimization (OSCM) problem is NP-hard (Eades, Wormald 1994).
Algorithms for OSCM

Heuristics:
- Barycenter
- Median

Exact:
- ILP Model
Barycenter Heuristic  (Sugiyama, Tagawa, Toda 1981)

**Idea:** Position nodes close to their neighbours.

- Set
  \[ x_2(u) = \frac{1}{\deg(u)} \sum_{v \in N(u)} x_1(v) \]

- In case of ties, break arbitrarily.
Barycenter Heuristic (Sugiyama, Tagawa, Toda 1981)

**Idea:** Position nodes close to their neighbours.

- Set

\[ x_2(u) = \frac{1}{\deg(u)} \sum_{v \in N(u)} x_1(v) \]

- In case of ties, break arbitrarily.

**Properties:**

- trivial implementation
- quick (exactly?)
- usually very good results
- finds optimum if \( \text{opt}(G, x_1) = 0 \) (see Exercises)
- there are graphs on which it performs \( \Omega(\sqrt{n}) \) times worse than optimal
Barycenter Heuristic (Sugiyama, Tagawa, Toda 1981)

**Idea:** Position nodes close to their neighbours.

- Set
  \[ x_2(u) = \frac{1}{\deg(u)} \sum_{v \in N(u)} x_1(v) \]

- In case of ties, break arbitrarily.

**Properties:**
- trivial implementation
- quick (exactly?)
- usually very good results
- finds optimum if \( \text{opt}(G, x_1) = 0 \) (see Exercises)
- there are graphs on which it performs \( \Omega(\sqrt{n}) \) times worse than optimal

Work with your neighbour and then share

Construct an example where barycenter method produces very bad results.

3 min
Median Heuristic (Eades, Wormald 1994)

Idea: Use the median of the coordinates of the neighbours.

• For a node $v \in L_2$ with neighbours $v_1, \ldots, v_k$ set
  
  $$x_2(v) = \text{med}(v) = x_1(v_{\lceil k/2 \rceil})$$

  and $x_2(v) = 0$ if $N(v) = \emptyset$.

• If $x_2(u) = x_2(v)$ and $u, v$ have different parity, place the node with odd degree to the left.

• If $x_2(u) = x_2(v)$ and $u, v$ have the same parity, break tie arbitrarily.

• Runs in time $O(|E|)$. 
Median Heuristic (Eades, Wormald 1994)

**Idea:** Use the median of the coordinates of the neighbours.

- For a node $v \in L_2$ with neighbours $v_1, \ldots, v_k$ set
  
  $x_2(v) = \text{med}(v) = x_1(v_{\lceil k/2 \rceil})$

  and $x_2(v) = 0$ if $N(v) = \emptyset$.

- If $x_2(u) = x_2(v)$ and $u, v$ have different parity, place the node with odd degree to the left.

- If $x_2(u) = x_2(v)$ and $u, v$ have the same parity, break tie arbitrarily.

- Runs in time $O(|E|)$.

**Properties:**

- trivial implementation
- fast
- mostly good performance
- finds optimum when $\text{opt}(G, x_1) = 0$
- **Factor-3 Approximation**
Approximation Factor

**Theorem 2:** Let $G = (L_1, L_2, E)$ be a 2-layered graph and $x_1$ an arbitrary ordering of $L_1$. Then it holds that
\[ \text{med}(G, x_1) \leq 3 \text{ opt}(G, x_1). \]
Theorem 2: Let $G = (L_1, L_2, E)$ be a 2-layered graph and $x_1$ an arbitrary ordering of $L_1$. Then it holds that
\[ \text{med}(G, x_1) \leq 3 \text{ opt}(G, x_1). \]
Approximation Factor

**Theorem 2:** Let $G = (L_1, L_2, E)$ be a 2-layered graph and $x_1$ an arbitrary ordering of $L_1$. Then it holds that

$$\text{med}(G, x_1) \leq 3 \text{opt}(G, x_1).$$
Integer Linear Programming

Properties:

• branch-and-cut technique for DAGs of limited size
• useful for graphs of small to medium size
• finds optimal solution
• solution in polynomial time is not guaranteed
Integer Linear Programming

Properties:
• branch-and-cut technique for DAGs of limited size
• useful for graphs of small to medium size
• finds optimal solution
• solution in polynomial time is not guaranteed

Model: see blackboard
Experimental Evaluation (Jünger, Mutzel 1997)

Results for 100 instances on 20 + 20 nodes with increasing density

Time for 100 instances on 20 + 20 nodes with increasing density
Experimental Evaluation (Jünger, Mutzel 1997)

Results for 10 instances of sparse graphs with increasing size

Time for 10 instances of sparse graphs with increasing size
Example
Example
Example
Example
Example

Layered Layouts II
Example
Example
Example
There was even an iPad game CrossingX for the OSCM problem!
Winner of Graph Drawing Game Contest 2012
Step 4: Coordinate Computation

What could be our goals?
Steightening Edges

**Goal:** For the edges with dummy nodes, minimize deviation from a straight line.

**Idea:** Use quadratic program.

- Let $p_{uv} = (u, d_1, \ldots, d_k, v)$ be $u - v$-path with $k$ dummy nodes.
- Consider the $x$-coordinate of $d_i$ when $(u, v)$ would be straight:
  
  \[ a_i = x(u) + \frac{i}{k+1} (x(v) - x(u)) \]

- Define the sum of deviations squared:
  \[ g(p_{uv}) = \sum_{i=1}^{k} (x(d_i) - a_i)^2. \]

- Minimize $\sum_{uv \in E} g(p_{uv}).$

- Subject to: $x(w) - x(z) \geq \delta$ for consecutive nodes $w, z$ on the same layer, $w$ right from $z$ for some distance parameter $\delta$. 
Steightening Edges

Goal: For the edges with dummy nodes, minimize deviation from a straight line.

Idea: Use quadratic program.

- Let $p_{uv} = (u, d_1, \ldots, d_k, v)$ be $u - v$-path with $k$ dummy nodes.
- Consider the $x$-coordinate of $d_i$ when $(u, v)$ would be straight:
  $$a_i = x(u) + \frac{i}{k+1}(x(v) - x(u)).$$
- Define the sum of deviations squared: $g(p_{uv}) = \sum_{i=1}^{k} (x(d_i) - a_i)^2$.
- Minimize $\sum_{uv \in E} g(p_{uv})$.
- Subject to: $x(w) - x(z) \geq \delta$ for consecutive nodes $w, z$ on the same layer, $w$ right from $z$ for some distance parameter $\delta$.

Properties:

- quadratic program is time-expensive
- width can be exponential
- optimization function can be adapted to optimize "verticality"
Example
Example
Step 5: Drawing edges

Possibility: Substitute polylines by Bézier curves
Example
Example
Summary

given
resolve cycles
layer assignment

crossing minimization
node positioning
edge drawing
Summary

- flexible framework to draw directed graphs
- sequential optimization of various criteria
- modelling gives NP-hard problems, which can still be solved quite well

Crossing minimization
Node positioning
Edge drawing