Algorithms for Graph Visualization
Layered Layout – Part 2

Tamara Mchedlidze
11.1.2018
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 (Sugiyama, Tagawa, Toda 1981)

1. Given
2. Resolve cycles
3. Layer assignment
4. Crossing minimization
5. Node positioning
6. Edge drawing
Sugiyama Framework  
(Sugiyama, Tagawa, Toda 1981)

Layered Layout

given

resolve cycles

layer assignment

crossing minimization

node positioning

edge drawing
Step 3: Crossing Minimization

Layered Layout
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 crossing among $E$ is minimum
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 crossing among $E$ is minimum

**Observation:**
- The number of crossing in 2-layered drawing of $G$ depends only on ordering of the nodes, not from the exact positions
- for $u, v \in L_2$ the number of crossing among incident to them edges depends on whether $x_2(u) < x_2(v)$ or $x_2(v) < x_2(u)$ and not on the positions 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 crossing among $E$ is minimum

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

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

\[ c_{uv} = 5 \]
\[ c_{vu} = 7 \]
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 crossing among $E$ is minimum

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

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

for $x_2(u) < x_2(v)$

\[ c_{uv} = 5 \]
\[ c_{vu} = 7 \]
Further Properties

**Def:** Crossing number of $G$ 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 $x_1$ then $\text{opt}(G, x_1) = \min_{x_2} \text{cr}(G, x_1, x_2)$

**Lemma 1:** The following equalities hold:

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

**Def:** Crossing number of $G$ 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 $x_1$ then $\text{opt}(G, x_1) = \min_{x_2} \text{cr}(G, x_1, x_2)$

**Lemma 1:** The following equalities hold:

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

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

**Def:** Crossing number of $G$ 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 $x_1$ then $\text{opt}(G, x_1) = \min_{x_2} \text{cr}(G, x_1, x_2)$

**Lemma 1:** The following equalities hold:

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

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

Think for a minute and then share

Why the second inequality is not an equality?

1 min
Iterative Crossing Minimization

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

(1) compute a random 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 $x_i$ ($\rightarrow$ 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 $x_{i+1}$ ($\rightarrow$ 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
Iterative Crossing Minimization

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

1. compute a random 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:** few crossing when nodes are close to their neighbours

- set

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

- in case of equality introduce tiny gap
Barycenter Heuristic  
(Sugiyama, Tagawa, Toda 1981)

**Idea:** few crossing when nodes are close to their neighbours

* set

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

* in case of equality introduce tiny gap

**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 \hspace{1em} (Sugiyama, Tagawa, Toda 1981)

**Idea:** few crossing when nodes are close to their neighbours

- set

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

- in case of equality introduce tiny gap

**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

5 min

Construct an example proving that barycenter method works at least \( \sqrt{n} \) times worse than optimal
Median-Heuristic (Eades, Wormald 1994)

**Idea:** use the median of the coordinates of 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, place an arbitrary of them to the left
- Runs in time $O(|E|)$
Median-Heuristic  (Eades, Wormald 1994)

**Idea:** use the median of the coordinates of 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, place an arbitrary of them to the left

- 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

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

Modell: 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

Layered Layout
Layered Layout

Example
Example
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

Which could be the goals?
Steightening Edges

**Goal:** minimize deviation from a straight-line for the edges with dummy-nodes

**Idea:** use quadratic Program

- let $p_{uv} = (u, d_1, \ldots, d_k, v)$ path with $k$ dummy nodes between $u$ and $v$
- let $a_i = x(u) + \frac{i}{k+1}(x(v) - x(u))$ the $x$-coordinate of $d_i$ when $(u, v)$ is straight
- $g(p_{uv}) = \sum_{i=1}^{k} (x(d_i) - a_i)^2$
- minimize $\sum_{uv \in E} g(p_{uv})$
- constraints: $x(w) - x(z) \geq \delta$ for consecutive nodes on the same layer, $w$ right from $z$ ($\delta$ distance parameter)
Steightening Edges

Goal: minimize deviation from a straight-line for the edges with dummy-nodes

Idea: use quadratic Program

• let $p_{uv} = (u, d_1, \ldots, d_k, v)$ path with $k$ dummy nodes between $u$ and $v$
• let $a_i = x(u) + \frac{i}{k+1}(x(v) - x(u))$ the $x$-coordinate of $d_i$ when $(u, v)$ is straight
• $g(p_{uv}) = \sum_{i=1}^{k} (x(d_i) - a_i)^2$
• minimize $\sum_{uv \in E} g(p_{uv})$
• constraints: $x(w) - x(z) \geq \delta$ for consecutive nodes on the same layer, $w$ right from $z$ ($\delta$ distance parameter)

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

Layered Layout
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 can be solved quite well

Layered Layout

crossing minimization
node positioning
edge drawing
Practical Task

- Graph Drawing Contest holding at Graph Drawing conference each September

- Game of Thrones graph: [http://www.graphdrawing.de/contest2018/topics.html](http://www.graphdrawing.de/contest2018/topics.html)

- Create nice, readable, suitable for potential tasks visualization

- Use any software/library you like

- Think about processing the graph: clustering, finding interesting patterns

- Work in groups of 2-3

- Submit your visualization by 28/2/2018
Practical Task

• Graph Drawing Contest holding at Graph Drawing conference each September

• Game of Thrones graph:
  http://www.graphdrawing.de/contest2018/topics.html
Practical Task

• Graph Drawing Contest holding at Graph Drawing conference each September

• Game of Thrones graph:  
  http://www.graphdrawing.de/contest2018/topics.html

• Create nice, readable, suitable for potential tasks visualization
Practical Task

• Graph Drawing Contest holding at Graph Drawing conference each September

• Game of Thrones graph:
  http://www.graphdrawing.de/contest2018/topics.html

• Create nice, readable, suitable for potential tasks visualization

• Use any software/library you like
Practical Task

• Graph Drawing Contest holding at Graph Drawing conference each September

• Game of Thrones graph: http://www.graphdrawing.de/contest2018/topics.html

• Create nice, readable, suitable for potential tasks visualization

• Use any software/library you like

• think about processing the graph: clustering, finding interesting patterns
Practical Task

• Graph Drawing Contest holding at Graph Drawing conference each September

• Game of Thrones graph: http://www.graphdrawing.de/contest2018/topics.html

• Create nice, readable, suitable for potential tasks visualization

• Use any software/library you like

• think about processing the graph: clustering, finding interesting patterns

• work in groups of 2-3
Practical Task

• Graph Drawing Contest holding at Graph Drawing conference each September

• Game of Thrones graph:
  http://www.graphdrawing.de/contest2018/topics.html

• Create nice, readable, suitable for potential tasks visualization

• Use any software/library you like

• think about processing the graph: clustering, finding interesting patterns

• work in groups of 2-3

• submit your visualization by 28/2/2018