ImPrEd
An Improved Force-Directed Algorithm that Prevents Nodes from Crossing Edges

Authors: Paolo Simonetto
Daniel Archambault
David Auber
Romain Bourqui

LaBRI, Université Bordeaux 1
INRIA Bordeaux Sud-Ouest
University College Dublin
PrEd - Introduction

• PrEd is a force-directed algorithm that optimises an existing layout.

• PrEd preserves edge crossing properties:
  - no new crossings created,
  - no old crossings undone.

• PrEd can be used for:
  - planar graph drawing,
  - interactive layout.

PrEd – The principle

• When moving, nodes must not cross edges:
  - edge crossings properties preserved,
  - node position can be constrained using edges.

• PrEd can be used to generate Euler diagrams.

PrEd – The Algorithm

• At each iteration:
  
  – Max movement $M$ computation:
    • According to 8 sectors.
  
  – Force $F$ computation:
    • node-node repulsion,
    • adjacent node attraction,
    • edge-node repulsion.
  
  – Node displacement:
    • along vector $F$,
    • max distance according to $M$. 
Improving PrEd

• PrEd limitations:
  1. computationally expensive,
  2. overrestrictive in terms of node movements,
  3. sub-optimal in terms of layout quality,
  4. not very flexible.

• ImPrEd mainly features:
  - surrounding edges (1),
  - application of QuadTrees (as in previous work) (1),
  - new max movement rules (2),
  - force system cooling (3),
  - crossable and flexible edges (3-4).
ImPrEd – Surrounding Edges

Max movement needs to be computed for every node/edge?

\[ S_v = B^e(F) \cup B^e(H) \]

In the paper: how to identify a face with disconnected boundaries and crossings.
ImPrEd – Surrounding Edges

• What:
  - improve the running time.

• How:
  - compute for each node its surrounding edges.

• Where:
  - computation as a pre-processing step,
  - used in max movement and in node-edge repulsion.
ImPrEd – Application of QuadTrees

• What:
  - improve the running time.

• How:
  - efficiently detect the nearby elements of a node,
  - distant elements are ignored.

• Where:
  - computation at the beginning of each iteration,
  - used in node-node and node-edge repulsion,
  - used in max movement, because of global max movement.
ImPrEd – Surrounding Edges and QuadTrees

• What if we have a tree?
  - One face → Surrounding edges useless,
  - Nodes lightly constrained → QuadTrees are efficient.

• What if the graph is very intricate?
  - Many faces → Surrounding edges efficient,
  - Nodes heavily constrained → QuadTrees lose efficiency.

• Surrounding edges and QuadTrees combines well:
  - Sparse graphs: SE low, QT high
  - Dense graphs: QT low, SE high
ImPrEd – New Rules for Max Movement

ImPrEd maximises the sectors when computing the max movement.
ImPrEd – New Rules for Max Movement

• What:
  - improves the running time / drawing quality.

• How:
  - increase the node mobility, accelerating convergence.

• Where:
  - changes the max movement computation rules.
ImPrEd – Force System Cooling

- We gradually adapt the forces through the iterations:
  - increase the exponent of repulsive forces,
  - decrease the exponent of attractive forces.
ImPrEd – Force System Cooling

- $\delta$ is the optimal edge length, $\Upsilon$ is the optimal node-edge distance.

\[
\begin{align*}
\delta &= 5 \\
\Upsilon &= 2 \\
\delta &= 2 \\
\Upsilon &= 5 \\
\delta &= 5 \\
\Upsilon &= 2 \\
\delta &= 2 \\
\Upsilon &= 5
\end{align*}
\]
ImPrEd – Force System Cooling

• What:
  – improves the quality of the drawing,
  – improves the reliability of the parameters.

• How:
  – gradually shifts the effect of the forces from global to local.

• Where:
  – exponents at force computation.
ImPrEd – Crossable and Flexible Edges

• Input class expanded to polyline multigraphs.

• Each edge can be labelled as:
  
  – **Uncrossable/Crossable**: crossable edges influence forces but not max movement,

  – **Rigid/Flexible**: flexible edges can expand and contract (change the number of bends) according to their stress.
ImPrEd – Crossable and Flexible Edges

• What:
  - improve the quality of the drawing,
  - improve the flexibility of the algorithm.

• How:
  - allow polyline edges,
  - define uncrossable/crossable and rigid/flexible edges.

• Where:
  - pre and post processing to handle polyline edges,
  - every x iterations, flexible edges are updated.
Results – Speed-Up

We executed PrEd and ImPrEd on:
- 10 planar + 10 non-planar graphs of 50, 70, 100, 150 and 200 nodes randomly generated,
- 2 intersection and grid graphs of real Euler diagrams, with 100, 250 and 500 iterations.

We obtained that:
- ImPrEd performed consistently faster,
- flexible edges introduce a quality/time tradeoff,
- the overhead introduced is negligible even for small graphs.

The worst case complexity is unchanged.
Results – Speed-Up

Random planar

Graph dimension
Time (s)
PrEd
ImPrEd (R)
ImPrEd (F)

Random non-planar

Graph dimension
Time (s)
PrEd
ImPrEd (R)
# Results – Speed-Up

<table>
<thead>
<tr>
<th>Graph</th>
<th>iGraphA</th>
<th>gGraphA</th>
<th>iGraphB</th>
<th>gGraphB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes</td>
<td>149</td>
<td>2601</td>
<td>23</td>
<td>220</td>
</tr>
<tr>
<td>Edges</td>
<td>169</td>
<td>507</td>
<td>22</td>
<td>362</td>
</tr>
<tr>
<td>PrEd</td>
<td>100</td>
<td>17.50</td>
<td>2721.79</td>
<td>0.51</td>
</tr>
<tr>
<td></td>
<td>250</td>
<td>40.91</td>
<td>6792.53</td>
<td>1.28</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>80.46</td>
<td>13585.36</td>
<td>2.53</td>
</tr>
<tr>
<td>ImPrEd (R)</td>
<td>100</td>
<td>4.90</td>
<td>131.55</td>
<td>0.30</td>
</tr>
<tr>
<td></td>
<td>250</td>
<td>11.81</td>
<td>308.83</td>
<td>0.73</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>23.34</td>
<td>597.55</td>
<td>1.42</td>
</tr>
<tr>
<td>ImPrEd (F)</td>
<td>100</td>
<td>6.68</td>
<td>97.22</td>
<td>0.47</td>
</tr>
<tr>
<td></td>
<td>250</td>
<td>22.02</td>
<td>215.62</td>
<td>1.73</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>35.29</td>
<td>409.18</td>
<td>3.79</td>
</tr>
<tr>
<td>ImPrEd (PP)</td>
<td>—</td>
<td>0.13</td>
<td>5.64</td>
<td>0.02</td>
</tr>
</tbody>
</table>

| ImPrEd (R) | Gain | 3.49 | 21.81 | 1.75    | —       |
| ImPrEd (F) | Gain | 2.25 | 30.90 | 0.73    | —       |
Results – Drawing Quality – iGraph

ImPrEd generates a more precise and regular node spacing.

PrEd

ImPrEd (R)
Results – Drawing Quality – iGraph

Flexible edges help improving the angular resolution.

ImPrEd (R)  ImPrEd (F)
Results – Drawing Quality – gGraph

PrEd

ImPrEd (R)
Results – Drawing Quality – gGraph

ImPrEd (R)    ImPrEd (F)
Results – Drawing Quality and Flexibility
Conclusions

• We designed ImPrEd to improve PrEd in terms of:
  - running time (surrounding edges and QuadTrees),
  - drawing quality (force cooling and new max move),
  - flexibility (polyline, crossable and flexible edges).

• We presented evidences of improvements based on:
  - 100 randomly generated planar and non-planar graphs,
  - 4 Euler diagram graphs from real world data,
  - over 900 algorithm executions.

• Improve the complexity to scale to larger graphs.
Thank you for your attention.