Routage dans graphes géométriques

Nicolas Bonichon
Qu’est-ce qu’un bon réseau ?

Etirement : $\max_{a,b} \frac{\text{distance dans le réseau}_{a \rightarrow b}}{\text{distance à vol d’oiseau}_{a \rightarrow b}}$

Nombre d’arêtes ou somme des longueurs des arêtes

Degré maximal : taille du plus grand rond-point

Planaire : ni pont ni tunnel

Facilement Routable
Routing ratio

• Routing algorithm

• Routing in Delaunay
  – Upper & lower bounds

• Routing Theta-Graphs
Routing Algorithm

\[ \text{NextHop}(p_i, N(p_i), \text{mem}, t) \]

\[ \text{(…)} \]

\[ \text{update(mem)} \]

\[ \text{return } p_{i+1} \]

\[ \text{GreedyNextHop}(p_i, N(p_i), \emptyset, t) \]

\[ \text{return } \text{neighbor } p_{i+1} \text{ of } p_i \text{ such that } |p_{i+1}t| \text{ is min.} \]
Routing ratio of an algorithm

\( \text{Route}_A(s, t) \): path from \( s \) to \( t \) computed by \( A \).

\(|\text{Route}_A(s, t)|\): sum of length of edges of \( \text{Route}_A(s, t) \)

\( \text{routing ratio} \) of an algorithm \( A \): \( \max_G \max_{s, t} \frac{|\text{route}_A(s, t)|}{||st||} \)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Routing ratio</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greedy routing</td>
<td>( \infty )</td>
<td>0</td>
</tr>
<tr>
<td>Compass routing</td>
<td>( \infty )</td>
<td>0</td>
</tr>
<tr>
<td>shortest path (non local routing) [Xia 14]</td>
<td>&lt;1.998</td>
<td>( \infty )</td>
</tr>
<tr>
<td>shortest path (non local routing) [Xia Zhang 11]</td>
<td>( &gt; 1.59 )</td>
<td>( \infty )</td>
</tr>
<tr>
<td>Parallel Voronoi routing [Bose Morin 99]</td>
<td>45,749</td>
<td>O(1) R/W</td>
</tr>
<tr>
<td>Searching on the Voronoi path [Bose De Carufel et al 14]</td>
<td>15.48</td>
<td>O(1) R/W</td>
</tr>
<tr>
<td>Generalized Chew's algorithm [Bose B. de Carufel Perkovic van Rensen 16]</td>
<td>5.90</td>
<td>O(1) Ronly</td>
</tr>
</tbody>
</table>

- Chew's algorithm [Chew'89]: designed for L1-Delaunay triangulation (empty squares).
- Generalization: quite straightforward
- Analysis of the generalization: not trivial/new techniques
Generalized Chew's algorithm

Consider only triangles that intersect \([st]\)

\[
\text{ChewNextHop}(p_i, N(p_i), s, t) \\
T_i \leftarrow \text{rightmostTriangle}(p_i) \\
C_i \leftarrow \text{emptyCircle}(T_i) \\
\text{if } p_i \in \text{upperArc}(C_i) \\
\quad \text{move clockwise on } C_i \\
\text{else} \\
\quad \text{move counterclockwise on } C_i \\
\text{return next visited vertex of } T_i
\]
Lower Bound on Chew's routing ratio

\[ \text{routing ratio} \geq 5.7282 > 1 + \frac{3\pi}{2} \]
Analysis of the routing ratio

Let's forget irrelevant details:
- my route doesn't depend on the « non-rightmost » triangles
- my route doesn't depend on the position of the third point of the rightmost triangles

Let's be pessimistic

$C'_i$: the worst circle $C_i$ that goes from $p_i$ to $p_{i+1}$.

$C'_i$ must intersect $[st]$
$p_i$ must be on the good arc (upper or lower) of $C'_i$

$$|\text{Route}_{Chew}(s, t)| \leq |\text{Route}_{red}(s, t)|$$
Analysis of the routing ratio

Proof by induction on $i$:

First point after $p_i$ on the other side of $[st]$

\[
\forall i \quad \min \ + +0.18 \leq +0.18
\]

\[i = 0, \ |\text{Route}_{Chew}(s,t)| \leq (1 + \frac{3\pi}{2} + 0.18)|st| = 5.90|st|\]
Routing ratio of an algorithm

Route\(_A(s, t)\): path from \(s\) to \(t\) computed by \(A\).

\(|Route\(_A(s, t)\)|: sum of length of edges of Route\(_A(s, t)\)

routing ratio of an algorithm \(A\): \(\max_G \max_{s,t} \frac{|route\(_A(s, t)\)|}{||st||}\)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Routing ratio</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greedy routing</td>
<td>(\infty)</td>
<td>0</td>
</tr>
<tr>
<td>Compass routing</td>
<td>(\infty)</td>
<td>0</td>
</tr>
<tr>
<td>shortest path (non local routing) [Xia 14]</td>
<td>&lt;1.998</td>
<td>(\infty)</td>
</tr>
<tr>
<td>shortest path (non local routing) [Xia Zhang 11]</td>
<td>&gt; 1.59</td>
<td>(\infty)</td>
</tr>
<tr>
<td>Parallel Voronoi routing [Bose Morin 99]</td>
<td>45.749</td>
<td>O(1) R/W</td>
</tr>
<tr>
<td>Searching on the Voronoi path [Bose De Carufel et al 14]</td>
<td>15.48</td>
<td>O(1) R/W</td>
</tr>
<tr>
<td>Generalized Chew’s algorithm [Bose B. de Carufel Perkovic van Rensen 16]</td>
<td>5.90</td>
<td>O(1) Ronly</td>
</tr>
<tr>
<td>MixedChordArc algorithm [Bose B. Despré Hill Smid 17+]</td>
<td>3.56</td>
<td>O(1) Ronly</td>
</tr>
</tbody>
</table>
MixedChordArc algorithm

MixedChordArcNextHop\((p_i, N(p_i), s, t)\)

\(T_i \leftarrow \text{rightmostTriangle}(p_i)\)

\(C_i \leftarrow \text{emptyCircle}(T_i)\)

\[
\begin{align*}
\text{if } |\text{ArcCW}(p_i, t_i)| &< |p_i l_i| + |\text{ArcCW}(p_i, t_i)| \\
\text{return } u_i \\
\text{else} \\
\text{return } l_i
\end{align*}
\]

Consider only triangles that intersect \([st]\)
Spanning ratio vs routing ratio
*(the cost of locality)*

<table>
<thead>
<tr>
<th>Graph family</th>
<th>Spanning ratio (shortest path)</th>
<th>Routing ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>TD-Delaunay triangulations</td>
<td>2 [Chew 86]</td>
<td>2.89 [Bose Fagerberg van Renssen Verdonshot 12]</td>
</tr>
<tr>
<td>L1-Delaunay triangulations</td>
<td>2.61 [B. Gavoille Hanusse Perković 14]</td>
<td>2.71 &lt;c &lt; 3.16 [Chew 89]</td>
</tr>
<tr>
<td>Delaunay triangulations</td>
<td>[Xia Zhang 11] 1.59&lt;c&lt;1.998 [Xia 14]</td>
<td>[B. Bose De Carufel Perkovic van Rensen 14] 1.70&lt;c&lt;3.56 [Bose B. Despré Hill Smid 17+]</td>
</tr>
</tbody>
</table>
You see, there are two kinds of people in the world, my friend: those who route through \( q \) and those who route through \( q' \).

\[
\text{dist}_G(s,q) + \text{dist}_G(q,t) = 1.70
\]

There is no deterministic \( k \)-local routing algorithm with routing ratio < 1.70.
Routing ratio

• Routing algorithm

• Routing in Delaunay
  – Upper & lower bounds

• Routing Theta-Graphs
Compass routing

- Select the neighbour that is in the direction of the destination.
### Routing ratio of compass algorithm

<table>
<thead>
<tr>
<th>papier</th>
<th>k</th>
<th>etirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>[El Molla 09]</td>
<td>2,3</td>
<td>∞</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>∞</td>
</tr>
<tr>
<td>[Clarkson 87][Keil 88]</td>
<td>≥ 7</td>
<td>$\frac{1}{1-2\sin(\pi/k)}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>k</th>
<th>stretch</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>7.56</td>
</tr>
<tr>
<td>8</td>
<td>4.26</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>12</td>
<td>2.07</td>
</tr>
<tr>
<td>13</td>
<td>1.91</td>
</tr>
</tbody>
</table>
Quand le nombre de points devient grand...

[Marckert B. 11]

\[ c_1 = \frac{1}{2} \left( \frac{1}{\cos(\pi/k)} + \frac{\text{arcsinh}(\tan(\pi/k))}{\tan(\pi/k)} \right) \]
\[ c_2 = \frac{1}{2} \left( \frac{1}{\cos^2(\pi/k)} + \frac{\text{arcsinh}(\tan(\pi/k))}{\sin(\pi/k)} \right) \]

\[ \text{chemin}(s, t) \rightarrow [s, I] \cup [I, t] \]
\[ |\text{chemin}(s, t)| \rightarrow 1.05||s, I|| + 1.22||I, t|| \]
\[ |\text{chemin}(s, t)| \rightarrow c_1||s, I|| + c_2||I, t|| \]

\[ \Pr \left( \sup_{s, t} \left( |\text{chemin}(s, t)| - (c_1||s, I|| + c_2||I, t||) \right) > \frac{1}{n^\alpha} \right) < \frac{1}{n^\beta} \]

Valable aussi pour des densités de points non-uniformes
Decomposition of the trajectory in 3 phases

- **Phase 1:**
  - $\Rightarrow$ only steps of type 1.

- **Phase 2:**
  - Only steps of type 1 and 2

- **Phase 3:**
  - Final approach
  - Steps of any type
Compass routing algorithm
*(straight)*

- Compute a path by selecting at each step the neighbor in the direction of the target.