Exact and efficient computations on circles in CGAL - Experiments

Pedro M. M. de Castro, Sylvain Pion, Monique Teillaud

European Workshop on Computational Geometry
Graz - 2007
The Computational Geometry Algorithms Library
Open Source project
www.cgal.org

> 400,000 lines of C++ code
> 3,000 pages manual
~ 12,000 downloads per year
~ 850 users on public mailing list
~ 50 developers
licenses LGPL or QPL
start-up GeometryFactory interfaces: Python, Scilab

Robustness and efficiency
Quality:
  • Editorial board
    (10 members)
  • Test-suites each night
...
Kernel =
  - elementary geometric objects
    (points, segments, . . . )
  - elementary operations on them
    (intersection tests, intersection computations, . . . )

Up to release 3.1 (Dec’04):
  essentially linear objects

Release 3.2 (May ’06):
  2D circular kernel

[P.-T.]
2D circular kernel design

[Emiris-Kakargias-P.-T.-Tsiganidas socg’04]

Guidelines:

- **code reuse:**
  - ability to reuse the CGAL kernel for points, circles, number types, ...

- **flexibility:**
  - possibility to use other implementations for points, circles, number types, ...
  - possibility to use several algebraic implementations
2D circular kernel design
[Emiris-Kakargias-P.-T.-Tsigaridas socg’04]

Guidelines:
- **code reuse:**
  - ability to reuse the CGAL kernel for points, circles, number types, ...
- **flexibility:**
  - possibility to use other implementations for points, circles, number types, ...
  - possibility to use several algebraic implementations

```cpp
template < LinearKernel, AlgebraicKernel >
class Circular_kernel
```
2D circular kernel design

[Emiris-Kakargias-P.-T.-Tsiga-
ridas socg’04]

template < LinearKernel, AlgebraicKernel >
class Circular_kernel

Types

• Must be defined by Linear_kernel
  basic number types, points, lines,…
• Must be defined by Algebraic_kernel
  algebraic numbers, polynomials
• Defined by Circular_kernel
  Circular_arc_2, Circular_arc_point_2
2D circular kernel design

[Emiris-Kakargias-P.-T.-Tsigeridas socg’04]

template < LinearKernel, AlgebraicKernel >

class Circular_kernel

Types
- Must be defined by Linear_kernel
  basic number types, points, lines,
- Must be defined by Algebraic_kernel
  algebraic numbers, polynomials
- Defined by Circular_kernel
  Circular_arc_2, Circular_arc_point_2

Predicates
e.g. intersection tests, comparisons of intersection points,
  exactness is crucial for geometric algorithms

Constructions
e.g. computation of intersection points
Representation

- **CGAL Circle_2**:  
  - center  
  - squared radius (rational)
Representation

- **CGAL Circle_2**:
  - center
  - squared radius (rational)

- **Circular_arc_2**:
  - supporting circle Circle_2
  - 2 Circular_arc_point_2 (algebraic)

- **Circular_arc_point_2**
  - root of system
  - (system = 2 equations of circles) (algebraic)
Experiments

Computation of arrangements with CGAL 3.2 package

[Wein-Fogel-Zukerman-Halperin]

* on real industrial data of VLSI models

* and synthetic models (very dense and sparse)

<table>
<thead>
<tr>
<th>VLSI</th>
<th>N</th>
<th>V</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>vlsi_1</td>
<td>11,929</td>
<td>20,649</td>
<td>26,468</td>
<td>6,385</td>
</tr>
<tr>
<td>vlsi_2</td>
<td>9,663</td>
<td>8,556</td>
<td>9,439</td>
<td>887</td>
</tr>
<tr>
<td>vlsi_3</td>
<td>35,449</td>
<td>101,758</td>
<td>163,316</td>
<td>61,887</td>
</tr>
<tr>
<td>vlsi_4</td>
<td>12,937</td>
<td>81,096</td>
<td>143,049</td>
<td>61,986</td>
</tr>
<tr>
<td>vlsi_5</td>
<td>4,063</td>
<td>40,636</td>
<td>77,598</td>
<td>36,965</td>
</tr>
<tr>
<td>vlsi_6</td>
<td>74,052</td>
<td>547,250</td>
<td>1,016,460</td>
<td>470,480</td>
</tr>
<tr>
<td>vlsi_7</td>
<td>89,918</td>
<td>495,209</td>
<td>878,799</td>
<td>383,871</td>
</tr>
<tr>
<td>vlsi_8</td>
<td>123,175</td>
<td>370,304</td>
<td>555,412</td>
<td>190,031</td>
</tr>
<tr>
<td>vlsi_9</td>
<td>139,908</td>
<td>1,433,248</td>
<td>2,690,530</td>
<td>1,257,684</td>
</tr>
</tbody>
</table>
# Experiments - Results

Pentium 4, 2.5 GHz, 1GB, Linux (2.4.20 Kernel)  
g++4.0.2, -DNDEBUG -O2  
times in seconds

<table>
<thead>
<tr>
<th>Input</th>
<th>CGAL 3.2 before</th>
<th>Specific CGAL arrangements</th>
<th>CGAL 3.3 after</th>
</tr>
</thead>
<tbody>
<tr>
<td>vlsi_1</td>
<td>28.0</td>
<td>8.55</td>
<td>4.61</td>
</tr>
<tr>
<td>vlsi_2</td>
<td>48.0</td>
<td>2.59</td>
<td>1.31</td>
</tr>
<tr>
<td>vlsi_3</td>
<td>135</td>
<td>26.7</td>
<td>21.8</td>
</tr>
<tr>
<td>vlsi_4</td>
<td>569</td>
<td>26.9</td>
<td>25.4</td>
</tr>
<tr>
<td>vlsi_5</td>
<td>125</td>
<td>14.3</td>
<td>14.8</td>
</tr>
<tr>
<td>vlsi_6</td>
<td>611</td>
<td>137</td>
<td>134</td>
</tr>
<tr>
<td>vlsi_7</td>
<td>690</td>
<td>192</td>
<td>169</td>
</tr>
<tr>
<td>vlsi_8</td>
<td>3,650</td>
<td>220</td>
<td>136</td>
</tr>
<tr>
<td>vlsi_9</td>
<td>2,320</td>
<td>581</td>
<td>492</td>
</tr>
<tr>
<td>Very dense</td>
<td>335</td>
<td>77.9</td>
<td>76.2</td>
</tr>
<tr>
<td>Sparse</td>
<td>0.91</td>
<td>0.51</td>
<td>0.21</td>
</tr>
</tbody>
</table>
Caching

- bit-field (2 bytes) in `Circular_arc_2`
  - Stores information like
    - monotone arc
    - full circle
    - ...

- caching intersections of each pair of supporting circles

memory overflow on big data sets $\rightarrow$ abandoned
First enhancements of algebraic number type

- Optimizing particular cases
  cases when algebraic numbers are in fact rational
  (intersection of tangent circles)

- Arithmetic filtering
  general idea for comparison
  - compute approximate values + error bound $\varepsilon$
  - if $\varepsilon$ small enough: conclude
  - otherwise compute exact values
Reference counting

High cost of copying coefficients of polynomials
(multiprecision numbers)

Store handles to objects instead of objects
Copying = copying the reference
Intermediate results

(some of the enhancements were already in the CGAL specific implementation for arrangements)

<table>
<thead>
<tr>
<th>Input</th>
<th>before</th>
<th>now</th>
<th>Specific CGAL arrangements</th>
</tr>
</thead>
<tbody>
<tr>
<td>vlsi_1</td>
<td>28.0</td>
<td>8.42</td>
<td>8.55</td>
</tr>
<tr>
<td>vlsi_2</td>
<td>48.0</td>
<td>3.01</td>
<td>2.59</td>
</tr>
<tr>
<td>vlsi_3</td>
<td>135</td>
<td>25.8</td>
<td>26.7</td>
</tr>
<tr>
<td>vlsi_4</td>
<td>569</td>
<td>33.4</td>
<td>26.9</td>
</tr>
<tr>
<td>vlsi_5</td>
<td>125</td>
<td>15.2</td>
<td>14.3</td>
</tr>
<tr>
<td>vlsi_6</td>
<td>611</td>
<td>135</td>
<td>137</td>
</tr>
<tr>
<td>vlsi_7</td>
<td>690</td>
<td>185</td>
<td>192</td>
</tr>
<tr>
<td>vlsi_8</td>
<td>3,650</td>
<td>445</td>
<td>220</td>
</tr>
<tr>
<td>vlsi_9</td>
<td>2,320</td>
<td>525</td>
<td>581</td>
</tr>
<tr>
<td>Very dense</td>
<td>335</td>
<td>84.5</td>
<td>77.9</td>
</tr>
<tr>
<td>Sparse</td>
<td>0.91</td>
<td>0.40</td>
<td>0.51</td>
</tr>
</tbody>
</table>
Further improvements
Representation of algebraic numbers

- Initial representation of algebraic numbers:
  root of \( ax^2 + bx + c \)
  
  - 3 rational coefficients \( a, b, c \)
    (multi-precision - GMP or CGAL::MP_Float)
  - one boolean

  Computation of approximation and error for filtering:
  
  \((−b \pm \sqrt{b^2 − 4ac})/2a\)
  
  . . .
Further improvements

Representation of algebraic numbers

- **Initial representation of algebraic numbers:**
  - root of $ax^2 + bx + c$
  - 3 rational coefficients $a, b, c$
    (multi-precision - GMP or CGAL::MP_Float)
  - one boolean

  Computation of approximation and error for filtering:
  - $(−b ± \sqrt{b^2 − 4ac})/2a$
  - ... 

- **New representation of algebraic numbers:**
  - $\alpha + \beta \sqrt{\gamma}$
  - 3 numbers $\alpha, \beta, \gamma$

  allows less efficient comparisons, but reduces the lengths of multi-precision numbers (and allows additions in easy cases)
Further improvements

Representation of algebraic numbers

Histograms (on \textit{vlsi\_8})

quantity of numbers vs. lengths (number of digits)

Initial

New
Further improvements
Geometric filtering

For **predicates** (e.g. intersection tests) use **bounding box** of objects:

- Test intersection on the boxes
- if they don’t intersect: conclude
- otherwise test intersection on the exact objects
## Results

<table>
<thead>
<tr>
<th>Input</th>
<th>CGAL 3.2 before</th>
<th>Specific CGAL arrangements</th>
<th>CGAL 3.3 after</th>
</tr>
</thead>
<tbody>
<tr>
<td>vlsi_1</td>
<td>28.0</td>
<td>8.55</td>
<td>4.61</td>
</tr>
<tr>
<td>vlsi_2</td>
<td>48.0</td>
<td>2.59</td>
<td>1.31</td>
</tr>
<tr>
<td>vlsi_3</td>
<td>135</td>
<td>26.7</td>
<td>21.8</td>
</tr>
<tr>
<td>vlsi_4</td>
<td>569</td>
<td>26.9</td>
<td>25.4</td>
</tr>
<tr>
<td>vlsi_5</td>
<td>125</td>
<td>14.3</td>
<td>14.8</td>
</tr>
<tr>
<td>vlsi_6</td>
<td>611</td>
<td>137</td>
<td>134</td>
</tr>
<tr>
<td>vlsi_7</td>
<td>690</td>
<td>192</td>
<td>169</td>
</tr>
<tr>
<td>vlsi_8</td>
<td>3,650</td>
<td>220</td>
<td>136</td>
</tr>
<tr>
<td>vlsi_9</td>
<td>2,320</td>
<td>581</td>
<td>492</td>
</tr>
<tr>
<td>Very dense</td>
<td>335</td>
<td>77.9</td>
<td>76.2</td>
</tr>
<tr>
<td>Sparse</td>
<td>0.91</td>
<td>0.51</td>
<td>0.21</td>
</tr>
</tbody>
</table>
Conclusion and future work

Combination of

- generality and re-usability of the functionality
- robustness and efficiency

Integration in CGAL 3.3.

Future work

- manipulations of spheres, circles, and circular arcs in 3D (submitted to CGAL) [dC.-T.]
- filtering of constructions? [Funke-Mehlhorn’00, Fabri-P. lcsd’06]

Work partially supported by the EU STREP Project IST-006413