Computing Betti Tables with HTCondor

Jay Yang

May 18, 2016
What are Betti Tables

- A tool in Algebraic Geometry
- A shape determines a “barcode” of integers known as a Betti table.

Example

```
Example

0 1 2 3
0 1 0 0 0
1 0 6 8 3
```

- Unfortunately we don’t understand the dictionary between shapes and “barcodes”
- The goal is to understand how this correspondence encodes geometry
Role of Computation in Pure Math

Examples

- Twin Primes Conjecture
- Riemann Hypothesis
- Average Rank of Elliptic Curves
Our Problem

- Betti tables of the projective plane of degree $d$

<table>
<thead>
<tr>
<th>$d$</th>
<th>Pen and Paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Minutes</td>
</tr>
<tr>
<td>3</td>
<td>Hours</td>
</tr>
<tr>
<td>4</td>
<td>Impractical</td>
</tr>
</tbody>
</table>
Old Algorithm

Gröbner Basis

- Based on polynomial algebra
- Developed in the 60s
- Implemented in the 80s
- Advantages
  - Already implemented
  - Well optimized
- Disadvantages
  - Difficult to distribute
  - Doubly exponential
<table>
<thead>
<tr>
<th>$d$</th>
<th>Pen and Paper</th>
<th>Gröbner Basis</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Minutes</td>
<td>0.0005s</td>
</tr>
<tr>
<td>3</td>
<td>Hours</td>
<td>0.007s</td>
</tr>
<tr>
<td>4</td>
<td>Impractical</td>
<td>115s</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>Out of Memory</td>
</tr>
</tbody>
</table>
New Algorithm

▶ Based on linear algebra
▶ Advantages
  ▶ Based on well known linear algebra algorithms
  ▶ Easily Distributable
▶ Disadvantages
  ▶ Lose exactness
Workflow

For $d=5$
6 matrices

600 submatrices

1 betti table (3x19 array)

Construct Matrix

Split Matrix

HTCondor

Find Rank (QR)...

Find Rank (QR)

Post Processing
## With Our Algorithm

<table>
<thead>
<tr>
<th>$d$</th>
<th>Pen and Paper</th>
<th>Gröbner Basis</th>
<th>Our Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Miniutes</td>
<td>0.0005s</td>
<td>$\sim 20s$</td>
</tr>
<tr>
<td>3</td>
<td>Hours</td>
<td>0.007s</td>
<td>$\sim 1m$</td>
</tr>
<tr>
<td>4</td>
<td>Impractical</td>
<td>115s</td>
<td>$\sim 2m$</td>
</tr>
<tr>
<td>5</td>
<td>Out of Memory</td>
<td></td>
<td>$\sim 11m$</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td>??</td>
</tr>
</tbody>
</table>
## Timing Data

<table>
<thead>
<tr>
<th>$d$</th>
<th>Construct Matrices</th>
<th>(Wall Time)</th>
<th>(CPU time)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.1s</td>
<td>~20s</td>
<td>~10s</td>
</tr>
<tr>
<td>3</td>
<td>0.8s</td>
<td>~1m</td>
<td>~20s</td>
</tr>
<tr>
<td>4</td>
<td>30s</td>
<td>~2m</td>
<td>~5m</td>
</tr>
<tr>
<td>5</td>
<td>8m</td>
<td>~11m</td>
<td>~40m</td>
</tr>
<tr>
<td>6</td>
<td>4h</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
executable = wrapper.sh
output = outdir/single_entry_14_1.$(CLUSTER).$(PROCESS).out
error = outdir/single_entry_14_1.$(CLUSTER).$(PROCESS).err
log = single_entry_14_1.$(CLUSTER).log

universe = vanilla

arguments=$(infile) ./out_14_1/

request_memory = 6G

queue infile matching files ./matrices/map_14_1/*.dat
What’s Next

- Run $d = 6$
  - Largest matrices for $d = 5$ use 5GB of ram
  - Largest matrices for $d = 6$ use 10-100GB? of ram
  - Dynamic Memory Requests in Condor
  - Flock to CHTC’s HTCondor pool on campus
- Obtain partial tables for $d > 6$
- Investigate other rank algorithms other than QR
- More complex shapes
- Create a database of Betti tables
Thanks to

- Thanks to my collaborators Daniel Erman and David Bruce
- Steve Goldstein for introducing us to HTCondor and helping us understand and use it
- Steve Wright for his advice on matrix rank for sparse matrices
- The organizers for giving us the opportunity to give this talk