Beating the Birthday Paradox: Order Computations in Generic Groups

Andrew V. Sutherland

Massachusetts Institute of Technology

April 20, 2007
Outline

1. Introduction

2. Algorithms
   - Primorial Steps
   - Multi-Stage Sieve

3. Results
Outline

1 Introduction

2 Algorithms
   • Primorial Steps
   • Multi-Stage Seive

3 Results
Hard Problems vs Easy Problems

Hard Problems
- Factoring Integers: $N = pq$
- Discrete Logarithm: $DL(\alpha, \beta)$
- Order Computation: $|\alpha|$

Easy Problems
- Multiplying: $pq = N$
- Exponentiating: $\alpha^k = \beta$
- Fast Order Computation: $|\alpha|$ given $\alpha^E = 1_G$
Generic Groups and Black Boxes

**Generic Groups**
- Isomorphic groups are equivalent.
- Algorithms work in any finite group.
- Complexity measured by group operations.

**Black Boxes**
- Opaque representation.
- Unique identifiers.
- Good software engineering.
Order Computations

Applications
- Black-box group recognition.
- Abelian group structure.
- Factoring.

Order Computation Theorem
The total cost of all order computations is at most

\[(1 + o(1)) T(\lambda(G)),\]

where \(T(N)\) is the cost of computing \(|\alpha| = N\).
Order Computations

Applications
- Black-box group recognition.
- Abelian group structure.
- Factoring.

Order Computation Theorem
The total cost of all order computations is at most

\[(1 + o(1)) T(\lambda(G)),\]

where \(T(N)\) is the cost of computing \(|\alpha| = N\).
Order Computations

Problem
- Find the least positive $N$ such that $\alpha^N = 1_G$.
- No upper bound on $N$.
- $\alpha^k = \alpha^j \iff k \equiv j \mod N$.

Solutions
- Birthday paradox.
- Shanks baby-steps giant-steps $\approx 2\sqrt{2N}$.
- Pollard rho method $\approx \sqrt{2\pi N}$. 
Order Computations

Problem

- Find the least positive $N$ such that $\alpha^N = 1_G$.
- No upper bound on $N$.
- $\alpha^k = \alpha^j \iff k \equiv j \mod N$.

Solutions

- Birthday paradox.
- Shanks baby-steps giant-steps $\approx 2\sqrt{2N}$.
- Pollard rho method $\approx \sqrt{2\pi N}$.
## Lower Bounds?

<table>
<thead>
<tr>
<th><strong>Babai</strong></th>
<th>Exponential lower bound in black-box groups.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Shoup</strong></td>
<td>$\Omega(\sqrt{N})$ lower bound for discrete logarithm in generic groups.</td>
</tr>
<tr>
<td><strong>Terr</strong></td>
<td>$\sqrt{2N}$ lower bound on addition chains.</td>
</tr>
<tr>
<td><strong>Birthday Paradox</strong></td>
<td>$\sqrt{(2 \log 2)N}$ lower bound for a random algorithm (?)</td>
</tr>
</tbody>
</table>
Outline

1. Introduction

2. Algorithms
   - Primorial Steps
   - Multi-Stage Sieve

3. Results
The Basic Idea

What if we knew $|\alpha|$ were odd?

What if we knew $|\alpha| \perp 6$?

What if we knew $|\alpha| \perp \prod_{p \leq L} p$?
The Basic Idea

What if we knew $|\alpha|$ were odd?

What if we knew $|\alpha| \perp 6$?

What if we knew $|\alpha| \perp \prod_{p \leq L} p$?
The Basic Idea

What if we knew $|\alpha|$ were odd?

What if we knew $|\alpha| \perp 6$?

What if we knew $|\alpha| \perp \prod_{p \leq L} p$?
Key Fact #1

Orders Can Be Factored

For any $\beta = \alpha^k$:

$$|\beta| = N_1 \quad \text{and} \quad |\alpha^{N_1}| = N_2 \quad \implies \quad |\alpha| = N_1 N_2.$$
Primorial Steps Algorithm

1. Let $E = \prod p^h$ for $p \leq L$, $p^h \leq M < p^{h+1}$, and let $P = \prod p$.

2. Compute $\beta = \alpha^E$.

3. Use baby-steps $\perp P$ and giant-step multiples of $P$ to find $N_1 = |\beta|$.

4. Use a fast order algorithm to find $N_2 = |\alpha^{N_1}|$ given $E$.

5. Return $N_1 N_2$. 
### Primorials

<table>
<thead>
<tr>
<th>$w$</th>
<th>$p_w$</th>
<th>$P_w$</th>
<th>$\phi(P_w)$</th>
<th>$\phi(P_w)/P_w$</th>
<th>$P_w/\phi(P_w)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.5000</td>
<td>2.0000</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>0.3333</td>
<td>3.0000</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>30</td>
<td>8</td>
<td>0.2667</td>
<td>3.7500</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>210</td>
<td>48</td>
<td>0.2286</td>
<td>4.3450</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>2310</td>
<td>480</td>
<td>0.2078</td>
<td>4.8125</td>
</tr>
<tr>
<td>6</td>
<td>13</td>
<td>30030</td>
<td>5760</td>
<td>0.1918</td>
<td>5.2135</td>
</tr>
<tr>
<td>7</td>
<td>17</td>
<td>510510</td>
<td>92160</td>
<td>0.1805</td>
<td>5.5394</td>
</tr>
<tr>
<td>8</td>
<td>19</td>
<td>9699690</td>
<td>1658880</td>
<td>0.1710</td>
<td>5.8471</td>
</tr>
<tr>
<td>9</td>
<td>23</td>
<td>223092870</td>
<td>36495360</td>
<td>0.1636</td>
<td>6.1129</td>
</tr>
<tr>
<td>10</td>
<td>29</td>
<td>6469693230</td>
<td>1021870080</td>
<td>0.1579</td>
<td>6.3312</td>
</tr>
</tbody>
</table>

**Table:** The First Ten Primorials
Complexity

Worst Case

\[ O\left(\sqrt{\frac{N}{\log \log N}}\right) \]

Best Case

\[ O(L) \]

Average Case

???
Complexity

Worst Case

\[ O\left(\sqrt{\frac{N}{\log \log N}}\right) \]

Best Case

\[ O(L) \]

Average Case

???
## Complexity

<table>
<thead>
<tr>
<th>Case</th>
<th>Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Worst Case</strong></td>
<td>$O\left(\sqrt{\frac{N}{\log \log N}}\right)$</td>
</tr>
<tr>
<td><strong>Best Case</strong></td>
<td>$O(L)$</td>
</tr>
<tr>
<td><strong>Average Case</strong></td>
<td>???</td>
</tr>
</tbody>
</table>
Number Have Smooth Parts and Coarse Parts

Let $\sigma_y(x)$ be the largest $y$-smooth divisor of $x$. Define $\kappa_y(x) = x/\sigma(x)$ to be the $y$-coarse part of $x$,

$$x = \sigma_y(x) \kappa_y(x).$$

Typically $y = x^{1/u}$.

How Big is the Coarse Part?

Few numbers are $y$-smooth, but for most numbers, $\kappa_y(x) \ll x$. 
The Multi-Stage Sieve

Factoring in the Dark
Problem: We don’t know any factors until we find them all.

Play the Odds
Solution: Alternate sieving and searching until we do.
The Multi-Stage Sieve

Factoring in the Dark
Problem: We don’t know any factors until we find them all.

Play the Odds
Solution: Alternate sieving and searching until we do.
**Introduction**

**Multi-Stage Sieve**

## How Numbers Are Made

### Random Bisection Model

How to generating random integers with known factorizations (Bach).

### Distribution of Smooth Numbers

\[ \Psi(x, x^{1/u}) \sim \rho(u)x. \]

### Distribution of Semismooth Numbers

Semismooth probability function: \( G(r, s) \).
Multi-Stage Sieve

Complexity

Median Complexity

$O(N^{0.344})$, assuming uniform distribution of $N = |\alpha|$. Typically better.

More generally...

$$Pr \left[ T(N) \leq cN^{1/u} \right] \geq G(1/u, 2/u)$$
Semismooth and Smooth Probabilities

<table>
<thead>
<tr>
<th>$u$</th>
<th>$G(1/u, 2/u)$</th>
<th>$\rho(u)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2</td>
<td>0.8958</td>
<td>0.2203</td>
</tr>
<tr>
<td>2.5</td>
<td>0.7302</td>
<td>0.1303</td>
</tr>
<tr>
<td>2.9</td>
<td>0.5038</td>
<td>0.0598</td>
</tr>
<tr>
<td>3.0</td>
<td>0.4473</td>
<td>0.0486</td>
</tr>
<tr>
<td>4.0</td>
<td>0.0963</td>
<td>0.0049</td>
</tr>
<tr>
<td>6.0</td>
<td>1.092e-03</td>
<td>1.964e-05</td>
</tr>
<tr>
<td>8.0</td>
<td>3.662e-06</td>
<td>3.232e-08</td>
</tr>
<tr>
<td>10.0</td>
<td>5.382e-09</td>
<td>2.770e-11</td>
</tr>
</tbody>
</table>
Outline

1. Introduction

2. Algorithms
   - Primorial Steps
   - Multi-Stage Seive

3. Results
What does it all mean?

Reference Problem for Generic Algorithms - Ideal Class Groups

Compute the ideal class group of $\mathbb{Q}[\sqrt{D}]$ for negative $D$. Interesting problem for number theorists, and cryptographers.

Comparison to Generic Algorithms: $D = -4(10^{30} + 1)$

Rho algorithm: 200 million gops, 15 days (Teske 1998).
Multi-stage sieve: 200,000 gops, 6 seconds.

Comparison to Non-Generic Algorithms: $D = -4(10^{54} + 1)$

Subexponential MPQS algorithm: 9 hours (Buchmann 1999).
Multi-stage sieve: 800,000 gops, 25 seconds.
What does it all mean?

Reference Problem for Generic Algorithms - Ideal Class Groups

Compute the ideal class group of $\mathbb{Q}[\sqrt{D}]$ for negative $D$. Interesting problem for number theorists, and cryptographers.

Comparison to Generic Algorithms: $D = -4(10^{30} + 1)$

Rho algorithm: 200 million gops, 15 days (Teske 1998).
Multi-stage sieve: 200,000 gops, 6 seconds.

Comparison to Non-Generic Algorithms: $D = -4(10^{54} + 1)$

Subexponential MPQS algorithm: 9 hours (Buchmann 1999).
Multi-stage sieve: 800,000 gops, 25 seconds.
What does it all mean?

Reference Problem for Generic Algorithms - Ideal Class Groups
Compute the ideal class group of $\mathbb{Q}[\sqrt{D}]$ for negative $D$. Interesting problem for number theorists, and cryptographers.

Comparison to Generic Algorithms: $D = -4(10^{30} + 1)$
Rho algorithm: 200 million gops, 15 days (Teske 1998).
Multi-stage sieve: 200,000 gops, 6 seconds.

Comparison to Non-Generic Algorithms: $D = -4(10^{54} + 1)$
Subexponential MPQS algorithm: 9 hours (Buchmann 1999).
Multi-stage sieve: 800,000 gops, 25 seconds.
**Recipe for Subexponential Algorithms**

**Lottery Problem**

Given a random sequence of problems, how long does it take to solve one? You only have to win once.

**Subexponential Approach**

Choose $u$ so that $cN^{1/u} G(1/u, 2/u) \approx 1$. Running time is "asymptotically" $L(1/2, \sqrt{2})$ or $L(1/2, 1)$.

**Generic Solution**

Works for any problem that can be reduced to random order computations.
Subexponential Result

Example: $D = -(10^{80} + 1387)$

Computed using $2 \times 10^9$ gops ($u = 6.7$). $L(1/2, 1)$ bound would predict $10^{13}$ gops.
Points to Ponder...

What is the right bound for order computation?

Known: $\Omega \left( N^{1/3} \right)$ \quad $O \left( \sqrt{N / \log \log N} \right)$

Unknown: $\Omega \left( N^{1/2} / \log N \right)$? \quad $O \left( \sqrt{N / \log N} \right)$?

Space efficient worst case?

$o \left( \sqrt{N} \right)$ algorithm using polylogarithmic space?

Subexponential Applications

Which problems can be reduced to random order computations in finite groups?