Hashcash Parallelization on GPGPU using OpenCL

Etienne Servais

MINES ParisTech, Centre de Recherche en Informatique

April 19, 2011
Table of Contents

1 Hashcash overview
   • The hashcash protocol
   • Stamp format in current version
   • Current CPU implementation performances

2 OpenCL

3 My implementation of Hashcash
   • source code survey

4 My results
   • Performances
   • Economical analyze
   • Stamp record!
Asymmetric *proof-of-work* algorithm based on burning CPU cycles.

Goals:

- **Spam-killer:**
  
  **Email header:**

  ```
  From: Calvin <calvin@comics.net>
  To: Hobbes <hobbes@comics.net>
  Subject: Suzy Derkins
  Date: 19 Jan 2038 11:59:59 +0000
  X-Hashcash: 1:24:380119:hobbes@comics.net::000000000FE4EA5E9
  ```

- **DoS defense.**
Stamp example

1:24:110309:bob@comics.net::M/42qrTP4ANgmSSs:003oMpI
Stamp format in current version

ver:bits:date:resource:rand:counter

Where

- \( \text{ver} = 1 \)
- \( \text{bits} = \text{Number of bits of the partial-preimage} \)
- \( \text{date} = \text{YYMMDD} \)
- \( \text{resource} = \text{IP, email adress...} \)
- \( \text{rand} = \text{random string, avoids getting twice the same stamp.} \)
- \( \text{counter} = \text{string used to find the preimage.} \)
### Hashcash overview

#### OpenCL

- **My implementation of Hashcash**

#### My results

**Current CPU implementation performances**

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Collision tests per sec.</th>
<th>estimate for a 20 bits stamp</th>
<th>estimate for a 30 bits stamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMD Turion(tm) X2 Ultra Dual-Core Mobile ZM-82 at 2.2GHz</td>
<td>4508800</td>
<td>255 ms</td>
<td>235 s</td>
</tr>
<tr>
<td>Intel Xeon at 2.00GHz</td>
<td>4131200</td>
<td>253 ms</td>
<td>260 s</td>
</tr>
<tr>
<td>Intel Atom N270 at 1.60GHz</td>
<td>2224000</td>
<td>603 ms</td>
<td>460 s</td>
</tr>
<tr>
<td>Intel Pentium M at 1.70GHz</td>
<td>3478400</td>
<td>305 ms</td>
<td>310 s</td>
</tr>
<tr>
<td>AMD Phenom(tm) II X4 955 Processor at 3.20GHz</td>
<td>6144000</td>
<td>171 ms</td>
<td>175 s</td>
</tr>
</tbody>
</table>

**Figure:** Hashcash calculation time on different computers.
Cross platform API to C.

Specifications written by the Khronos Group regrouping:

- Apple
- AMD
- Intel
- NVIDIA
- IBM
-...

Specifications first published December 8, 2008.
2 parts in an OpenCL program:
3 levels of parallelism:

- Work-items (an instance of a kernel, a *thread* in CUDA) contained inside
- Work-groups (*thread-blocks* in CUDA) sharing data contained inside
- NDRanges which is a 1-to-3 dimensional container.
Host code:

**SHA-1 stamp construction:**

```
1:24:380119:hobbes@comics.net::00000000FE4EA5E9
```
Host code:

Device code:

SHA-1 stamp construction:

1:24:380119:hobbes@comics.net::0000000000 FE4E A5E9
Based on FIPS PUB 180-1

for(i = 20; i < 40; i++)
{
    W[i & 0x0f] = rotateLeft(W[(i-3) & 0x0f] ^ W[(i-8) & 0x0f] ^ W[(i-14) & 0x0f] ^ W[(i-16) & 0x0f], 1);
    temp = rotateLeft(A, 5) + (B ^ C ^ D) + E + W[i & 0x0f] + K1;
    E = D;
    D = C;
    C = rotateLeft(B, 30);
    B = A;
    A = temp;
}
Full loop unrolling

//i=25
temp = rotateLeft(A, 5) +
       (B ^ C ^ D) + E + W[9] + K1;
E = D;
D = C;
C = rotateLeft(B, 30);
B = A;
A = temp;
Array scalarization

// i=25
W9 = rotateLeft(W6 ^ W1 ^ Wb ^ W9, 1);
temp = rotateLeft(A, 5) +
    (B ^ C ^ D) + E + W9 + K1;
E = D;
D = C;
C = rotateLeft(B, 30);
B = A;
A = temp;
Copy propagation

// i=25
W9 = rotateLeft(W6 ^ W1 ^ Wb ^ W9, 1);
t1 = rotateLeft(t0, 5) +
    (t2 ^ C1 ^ C0) + C2 + W9 + K1;
C2 = rotateLeft(t2, 30);
Standard implementations has about 3000 basic operations per SHA-1:

- 656 assignations

In my version, only 1200 per SHA-1!

- 228 assignations
Standard implementations has about 3000 basic operations per SHA-1:

- 224 rotations

In my version, only 1200 per SHA-1!

- 220 rotations
Standard implementations has about 3000 basic operations per SHA-1:

- 1886 logical and arithmetical operations

In my version, only 1200 per SHA-1!

- 752 logical and arithmetical operations
Standard implementations has about 3000 basic operations per SHA-1:

- 432 array accesses

In my version, only 1200 per SHA-1!
<table>
<thead>
<tr>
<th>Operations</th>
<th>Initially</th>
<th>In my implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignations</td>
<td>656</td>
<td>228</td>
</tr>
<tr>
<td>Rotations</td>
<td>224</td>
<td>220</td>
</tr>
<tr>
<td>Logical and arithmetical operations</td>
<td>1886</td>
<td>752</td>
</tr>
<tr>
<td>Array Accesses</td>
<td>432</td>
<td>0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>3198</strong></td>
<td><strong>1200</strong></td>
</tr>
</tbody>
</table>

**Figure:** Comparison of official and optimised version of SHA-1.
<table>
<thead>
<tr>
<th>Architecture</th>
<th>Operations per sec.</th>
<th>SHA-1 per sec.</th>
<th>Prix en euros</th>
</tr>
</thead>
<tbody>
<tr>
<td>NVIDIA Tesla C2050 14 GPU at 1147 MHz (448 CUDA cores)</td>
<td>513 G $2^{38.90}$</td>
<td>424 M $2^{28.66}$</td>
<td>2350</td>
</tr>
<tr>
<td>NVIDIA GeForce 8800 GTX 16 GPU at 1350 MHz (128 CUDA cores)</td>
<td>173 G $2^{37.33}$</td>
<td>142 M $2^{27.08}$</td>
<td>40 (ebay.fr)</td>
</tr>
<tr>
<td>AMD Phenom(tm) II X4 955 Processor at 3.20 GHz</td>
<td>13 G $2^{33.58}$</td>
<td>(20 M) $2^{24.26}$</td>
<td>150</td>
</tr>
</tbody>
</table>

**Figure:** SHA-1 performances on different devices.
1 response per 12,500,000 emails (November 2008);

\[ P = 365 \times 24 \times 3600 \times 2^{27.08 - b} \times \frac{10}{12500000} - 40 - 365 \times 24 \times 0.280 \times 0.12 \]

\( b \) : number of required bits.
• Selling products 10€ each;

You earn per year about:

\[ P = 365 \times 25 \times 24 \times 3600 \times 2^{27.08-b} \times \frac{10}{12500000} - 40 - 365 \times 25 \times 24 \times 0.280 \times 0.12 \]

\( b \): number of required bits.
Buying a GeForce 8800 GTX (40€);

You earn per year about:

\[ P = 365 \times 24 \times 3600 \times 2^{27.08-b} \times \frac{10}{12500000} - 40 - 365 \times 24 \times 0.280 \times 0.12 \]

\( b \) : number of required bits.
Computer power is about 280W (dev configuration);

You earn per year about:

\[ P = 365,25 \times 24 \times 3600 \times 2^{27.08-b} \times \frac{10}{12500000} - 40 - 365,25 \times 24 \times 0.280 \times 0.12 \]

\[ b \]: number of required bits.
In France, 1 kWh costs about 0,12€;

You earn per year about:

\[
P = 365 \times 25 \times 24 \times 3600 \times 2^{27.08-b} \times \frac{10}{12500000} - 40 - 365 \times 25 \times 24 \times 0.28 \times 0.12
\]

\(b\): number of required bits.
1 response per 12,500,000 emails (November 2008);
- Selling products 10€ each;
- Buying a GeForce 8800 GTX (40€);
- Computer power is about 280W (dev configuration);
- In France, 1 kWh costs about 0.12€;

You earn per year about:

\[ P = 365.25 \times 24 \times 3600 \times 2^{27.08-b} \times \frac{10}{12500000} - 40 - 365.25 \times 24 \times 0.280 \times 0.12 \]

\( b \) : number of required bits.
Spammer income per year according to the minimal number of bits to zero required in hashcash

Number of required bits

Incomes in euro

Spammer income per year

Economical analyse
A 48 bits stamp!

1:48:110416:etienne@cri.fr:::000A2F00000063BF012

 Obtained in 1266748 seconds (about 14 days, 10 hours)
Thank you for your attention!