Brute-Force Factoring using Socketing

Jackson Gregory
gregoryj17@asmsa.org

Goals

- Take large numbers with only two prime factors and find their prime factors using socketing
- Split the job into pieces for multiple computers to maximize efficiency.

Approach

- Create jobs of size 30,000,000
- Pass 8 jobs to each client
- Once a client finishes a job, give it another
- Once the number's factors have been found, proceed to the next number

Challenges

- Due to timing issues with the threading, the server would originally erroneously think a number had been factored before it actually had been.
- Some computers in the computer lab used had firewall settings preventing them from hosting servers.

Run

- A server was started on one computer. Four clients were connected. Three clients used 8 threads, while the client that shared a computer with the server used only 7.
- The server was set to run for 12 hours. After the 12 hour mark, it stopped sending out jobs and told all clients to stop working.

Results

Number	Factor 1	Factor 2	Total Computation Time (Hours)
35	5	7	0
2809	53	53	0
119423	307	389	0
10581101	3137	3373	0
622049959	21433	29023	0
47099816849	209983	224303	0
2844293573123	1611223	1765301	0
213396244249259	13123081	16261139	0.01
5197575163154207	70879673	73329559	0.01
5325831044194741 03	713067571	746890093	0.20
266688517984376720 69	5149002073	5179421453	2.18
16183290277183073171 63	36869030707	43893994409	19.35
22380565083410998873337 9	408836758609	547420568531	177.81

Future Enhancements

- Debug print lines were left in the code, which slowed down the computations. Those could be removed for additional speed.
- The only technique used was skipping even number factors after 2. The factor finding algorithm could be further optimized.

Brute-Force Factoring using Socketing

Jackson Gregory
gregoryj17@asmsa.org