OOKAMI PROJECT APPLICATION

Date: April 28, 2021

Project Title: Analysis of the Linear Algebra in the Number Field Sieve Algorithm

Usage:

- Testbed

Principal Investigator: Greg Childers

- University/Company/Institute: California State University Fullerton
- Mailing address including country: 800 N. State College Blvd., Fullerton, CA 92834, USA
- Phone number: (657) 278-2159
- Email: gchilders@fullerton.edu

Names & Email of initial project users:

Greg Childers, gchilders@fullerton.edu

Usage Description:

The Number Field Sieve is currently the fastest classical algorithm for factoring a large integer into prime cofactors. Of the five steps comprising the Number Field Sieve, the linear algebra step is the most problematic as it is time-intensive yet difficult to parallelize. The matrices generated by the Number Field Sieve are much larger and much more dense than typical matrices from engineering problems, prohibiting the use of conventional sparse linear algebra solvers. I am requesting access characterize a parallelized implementation of the linear algebra in the MSIEVE software library on Ookami. This library uses the block Lanczos algorithm for the linear algebra. In collaboration with Jason Papadopoulos, the author of the MSIEVE library, a parallel version of MSIEVE utilizing MPI has been written and continues to be optimized.

The NFS linear algebra step is comprised of the common problem of finding nontrivial solution vectors, $\mathbf{x}$, of the homogenous linear system represented by the matrix equation $\mathbf{Ax} = 0$. $\mathbf{A}$ is a very large, sparse matrix of dimensions

$M \times N$, with $M < N$. Typical values of the matrix dimensions M and N range from 10 million for routine NFS factorizations to nearly 193 million for the largest NFS factorization completed to date, with approximately 100 nonzero entries per column on average. The entries of the matrix $\mathbf{A}$ are elements of the finite Galois field $\mathbb{F}_2$ and can thus be represented on a computer by the bits 0 and 1. The mathematical operations of addition and multiplication of two entries correspond exactly to the bitwise operators XOR and AND, respectively. Further, in an implementation with a w-bit word size, operations can be applied to w vectors simultaneously, represented as an array of N w-bit integers, for little additional cost over that required for one vector. (Therefore, although the literature continues to use the term "vector," in practice $\mathbf{x}$ is typically a $N \times w$ matrix.) The number of operations required to find the solution vectors is thus significantly less for algorithms adapted to take advantage of these properties than for more general algorithms. For traditional CPU implementations, the best performance is typically achieved using a 64-bit word size. However, as modern CPUs may work more efficiently with a larger word size, we have extended to library to allow tuning the of vector width.

Positive results from this project will be used to justify a larger production project on the A64FX architecture.

Computational Resources:

- Total node hours per year: 10,000
- Size (nodes) and duration (hours) for a typical batch job: Likely 4-8 nodes, 48 hours
- Disk space (home, project, scratch): 500 GB

Personnel Resources:

The library is fully written in C and thus should be straightforward to compile on the new platform.

Required software:

MPI, GMP, gcc

If your research is supported by US federal agencies:

- Agency:
- Grant number(s):
