ABSTRACT

Factoring large composite integers is a fundamental computational problem in number theory and has immense applications in many cryptographic protocols. Modern integer-factoring algorithms consist of two stages: the sieving stage and the linear-algebra stage. Efficient parallel implementations of both these stages have been reported in the literature. All these implementations are based on multi-core or distributed parallelization. In this thesis, we experimentally demonstrate that SIMD instructions available in many modern processors can lead to additional speedup in the computation of each core. We handle the sieving stage of the two fastest known factoring algorithms: the number-field sieve method (NFSM) and the multiple-polynomial quadratic sieve method (MPQSM). We experiment with two types of sieving: the line sieve and the lattice sieve. Although the sieving stage offers many tantalizing possibilities of data parallelism, exploiting these possibilities to get practical advantages is a challenging task. The major issue is to avoid packing and unpacking overheads to and from SIMD registers as much as possible. In this work, we suggest practical methods to effectively parallelize index calculations using SIMD features provided by Intel's SSE2 (Streaming SIMD Extensions) and AVX (Advanced Vector Extensions) instructions on a Sandy Bridge platform. This experimental improvement during index calculations is attributed to the fact that SIMD registers can be reused in multiple iterations without repacking their contents in each iteration of the sieving stage. We propose a way to reduce cache misses during index calculations. Our experiments reveal that SIMD parallelization can speed up the sieving stage by 15–40%. Since AVX supports 256-bit operations only on floating-point data and index calculations involve integer arithmetic, our implementations do not benefit from 256-bit SIMD registers. Indeed, we get similar speedup figures with SSE2 (128-bit registers) and AVX (floating-point index calculations on 256-bit registers). The recently released AVX2 features support data-parallel integer operations on 256-bit registers. Our implementations, when ported to AVX2, are expected to increase the speedup figures substantially. To the best of our knowledge, no similar SIMD-based implementation of sieving seems to have been reported in the literature.

Keywords: Integer Factorization, Sieving, Multiple-Polynomial Quadratic Sieve Method, Number-Field Sieve Method, Lattice Sieve Method, Single Instruction Multiple Data, Streaming SIMD Extensions, Advanced Vector Extensions.