**Topic of degree**

In number theory we often use so-called sieving algoithms. Probably the most simple of them is the sieve of Eratosthenes. The idea behind it is that we take small prime numbers, and in the sieving table (the interval, where we are looking for primes) we mark their muliples as composite numbers. At the end of the algorithm the unmarked numbers are the primes.

If we are sieving with large primes, then there will be a lot of cache misses, therefore the number of memory I/O operations will increase rapidly, and the I/O is much slower than the sieving. The COLS algorithm divide the sieving table to segments, and read those segments only once during its execution, and sieve the whole segment with the appropriate primes.

The task of the thesis is to examine, implement and compare these algorithms in the perspective of parallel execution. The thesis will contain a sequential and a parallel implementation of the naïve sieve and the COLS, and the analysis of their execution times.

The main goal of the thesis is to demonstrate the efficiency of COLS and to implement its parallel version, because one can use the COLS not only for improving the sieve of Eratosthenes, but to other algorithms in number theory, i.e. SIQS.

For the implementation I will use the C++ programming language, and I will parallelize the algorithms with the standard C++11’s thread library.