Zeta-search

Challenge

The Riemann zeta function is a function on the complex plane. It plays a pivotal role in analytic number theory and has applications in physics, probability theory, and applied statistics. For example, Euler has established a relation between the zeta function's zeroes and the distribution of prime numbers. Understanding it can bring us closer to the solution of the Riemann hypothesis, which is considered to be one of the most important unsolved problems in pure mathematics.

The Riemann zeta function on the critical line ($1/2 + it,\ t \in \mathbb{R}$) can be calculated using the Riemann–Siegel formula:

$$Z(t) = 2\sum_{n=1}^{N} \frac{1}{\sqrt{n}} \cos[\theta(t) - t\ln n] + R\,,$$

where

$$N = \lfloor \sqrt{t/2\pi} \rfloor,$$

and

$$R = (-1)^{N-1}\left(\frac{2\pi}{t}\right)^{\frac{1}{4}}\left[\sum_{k=0}^{m-1}\left(\frac{2\pi}{t}\right)^{\frac{k}{2}} C_k + O\left(t^{-\frac{m}{2}}\right)\right] = O\left(t^{-\frac{1}{4}}\right)$$

The goal is to find the peak values of the $Z(t)$ function.

Currently, there exists no known explicit formula to easily determine these peak values. This application uses simultaneous Diophantine approximation to find candidates to be the peak values of the Zeta function. It looks for integers that satisfy the following conditions:

$$\left| k * \frac{\log(p_i)}{\log(2)} \right| < \epsilon \quad (*),$$

where p_i is the ith prime number that is greater than 2, while ϵ is a suitably small constant. Currently, it is set to be 0.01.

The output of the application are the candidates; that is, integers that satisfy $(*)$ and the

$$F(t) = \sum_{n=1}^{\left\lfloor \ln\left(\frac{t}{2\pi}\right) \right\rfloor} \frac{1}{\sqrt{n}} \cos[\theta(t) - t\ln n]\,,\ \text{where}\ \ t = k * \frac{2\pi}{\log(2)}$$

value is close to its theoretical maximum.

Solution

We have identified the application to be a suitable candidate to be ported to desktop grid. Its parameter space (the positive integers) can be divided into arbitrarily long intervals; therefore, the problem can be solved as a parameter study application, while the execution time of the workunits can be controlled by selecting an appropriate interval length. The application uses small input (the endpoints of the interval), and produces small output (a list of a few integers). The parameter space of the application is potentially infinite, thus the number of workunits that can be generated is also potentially infinite. Therefore, using a desktop grid is particularly suitable for this application.

We started porting the application to the BOINC desktop grid middleware using the DC-API library. This library enables the easy porting of an application to several middleware at once, including BOINC.

We have acquired the application source and started porting it to the desktop grid. First, we had to enable the application to accept command line arguments, as the original version used constants built in the source code. This was required as the workunits must accept execution-time parameters. Second, we had to build the application statically, meaning that it must be able to run without external dependencies. Third, we had to identify any cross-platform portability issues, which is very important in case of—very heterogeneous—desktop grids. We have identified and solved a critical bug regarding this.

There is further work with this application before it can be executed in the desktop grid. It must be ported to Windows, as a considerable number of DG donors provide Windows systems. When the application in itself is ready to be executed on Windows *and* Linux machines, we have to extend it with DC-API library calls to make it fully DG-capable. Finally, the application must be tested, fine-tuned, and validated before deploying it in a production environment.