Project Development Proposal

Project

Project Request

The name our project is "Enumeration Library". This will be a numerical Math library for C and C++ programmers. Firstly, we will start with Number Theory, and subsequently, explore other areas of mathematics.

Our project will be an open source one, which will be published under GNU General Public License. Our intention is to follow the open source business model. So, it will be really useful for the scientific computing community. As it will be developed in an open source environment, the development cost will be minimal. Again, our main profit will be through the donations, and through supporting corporations,and institutions. So, there won't be any particular hard benefit option for our project. Every improvement will move towards the soft benefits.

Scope of Effort

Critical Priority:

"Enumeration Mathematical Library" will support :

1. Arbitrary Large Integers
2. Arbitrary Precision Numbers
3. Basic Number Theoretic Functions
4. Complex, Quaternions and Higher Fields
5. P-adic Fields
6. Algebraic Number Theory
7. Combinatorial Objects and Functions
8. Matrix Algebra

It will depend on Glibc C Standard Library but will be strictly implemented in ANSI C format, for portability. The library will have eight main parts which are listed above. For each part of the library there will be a single header file which will contain all function definitions, type definitions, structure definitions and constants. The part of the library could be used only by importing the Header file.

First of all, we will provide the library only in Linux operating system and the architecture support will be limited to x86-64 processor types. It will compiled as 64-bit

and 32-bit binaries for some specific operating systems such as Ubuntu or other Debian based OS. A tar file will also be provided for all other distributions.

Nice To Have :

It will be nice to if we can a standard basic math library of our own, highly optimized with assembly code for x86-64. This is because that we want our math library to be as fast as possible.

Future Updates:

We will like to support more architectures, especially ARM, ARM64, Itanium-64, POWERPC, etc. We will support parallel programming for these so that cluster computation can be achieved. We will like to write some heavy duty parts in assembly for different architectures for a more fast and robust library. We will be adding many more math modules and bindings for different programming languages such as Java, Rust, etc. We also intend to support other development platforms such as WIN32 an Mac OSX developer platform.

Approach

Our library will follow the following structure, and each library will be implemented in the order shown below. The details are of each library or module is followed by the name of the library or module.

Arbitrary Large Integers:

The library will have a separate package for arbitrary large integers and possible operations with it.

Arbitrary Precision Numbers:

The library will have a separate package for arbitrary precision floating point numbers and operations on it.

Complex, Quaternions and Higher Fields:

The library will have a package for different fields starting with complex numbers and quaternions, p-adic fields. It will include axioms and operations for those fields.

Combinatorial Objects:

The library will support combinatorial objects like combinations, permutations, power series, recurrences, tabuleax, q-analogues, etc.

Graphical Library:

It will support a graphical library for plotting, drawing mathematical objects and other graphical renderings and data.

Performance matters for this library and hence we will like our library to have the following features:

Fast Library:

EML will be carefully designed to be as fast as possible, both for small operands and for huge operands. The speed is achieved by using full words as the basic arithmetic type, by using fast algorithms, with highly optimized C code for the most common inner loops for a lot of CPUs, and by a general emphasis on speed. Fast Algorithms : The mathematical algorithms will be as fast as possible.

Parallel Programming:

The library will support parallel programming by using OpenMP and MPI libraries.

Easy User Interface:

In C programs, EML library can be accessed by including the header files. For example, to include arbitrary precision library from EML, #include <eml/arb.h>. Then while compiling use the proper flags for EML like -leml in GCC. For C++, you need to #include <eml/arb>. Then while compiling use the proper flags for EML like -leml in GCC.

Major Deliverable

By the spring term, we are planning to deliver a stable working library as our alpha release. We will make our source code available to public as an archive .zip or .tar of source code per open source ideology. We will also provide a dynamic compiled library as .so file. with complete documentation and system manual.

List of deliverables:

Static Library:

A .a file that has no dependencies but cannot be shared and can be used for development purposes that choose to distribute this library. Library of object code which is linked with, and becomes part of the application.

Dynamic Library:

A .so file or dynamic library that is dynamically linked at run time but statically aware. The libraries must be available during compile/link phase. The shared objects are not included into the executable component but are tied to the execution.

Source Code:

We will deliver the source as a .zip or tar archive. The source can be used to modify the library and build upon different platforms.