RubyGrant 2017 Mentor Report
“GPU-accelerated Libraries for Ruby to handle very large datasets”

Developer: Prasun Anand
Mentor: Kenta Murata

1 The evaluation of this project

The goal of this project is the development of Ruby binding library for CUDA libraries such as CUDA runtime and cuBLAS. The name of the library is RbCUDA. Mr. Prasun Anand, who is the developer of this project, has finished the implementation of APIs that is necessary for processing double-precision numerical arrays on GPU. Moreover, comparing with the existing gem libraries, such as ArrayFile and NMatrix, he has clarified that RbCUDA is most efficient on matrix multiplication.

1.1 The target libraries

This project targeted the following CUDA libraries.

- CUDA driver
- CUDA runtime library
- CUDA profiler
- cuBLAS, cuBLASXT
- cuSolver
- cuRand

In addition, it’s also the goal of this project to add CUDA support into the following gem libraries.

- nmatrix
- numo-narray

1.2 The achievements

The achievements of this project is:

- CUDA driver and CUDA runtime APIs
- cuBLAS and cuBLASXT APIs only for double-precision arrays
- cuSolver API for performing singluar value decomposition of double-precision matrices
- cuRAND APIs for creating random arrays following some probabilistic distributions
- CUDA profiler APIs for focus profiling
- CUDA support for nmatrix

The benchmark results illustrated that RbCUDA is 24x faster than ArrayFire gem at matrix multiplication. It shows that RbCUDA is best solution for writing CUDA applications in Ruby.

1.3 Future works

Although the benchmarking result illustrated the usefulness of RbCUDA, there are a lot of remaining works for completing the implementation. The following things are still remaining:

- cuBLAS and cuBLASXT APIs for single- and half-precisions arrays

- cuSolver APIs except for SVD of double-precision matrices
- cuRAND APIs, not covered by this project
- CUDA support for numo-narray
- Writing tests

2 My role as a mentor

As a mentor of this project, I’ve had online discussions with Prasun several times during the grant period. And I’ve arranged and create the poster for the poster session in Ruby 25th event.

3 The hope for the future development

At first, I hope Prasun to continue the development of RbCUDA after this grant period. The following things is important for the future of RbCUDA as the remaining works.

- Scaling up the implementation of the remaining APIs
    - E.g. call for the contributor from Ruby community
- Considering the integration with Cumo.
- Writing more examples of RbCUDA application
- Writing tutorials for users

4 Conclusion

In this project, Prasun succeeded to make the first step of RbCUDA. Although there are a lot of remaining works, his achievement that has illustrate the usefulness of RbCUDA by benchmarking is very big. After RbCUDA will have been completed, we will be able to make proofs-of-concept of CUDA applications not by C language, but Ruby. Therefore RbCUDA has the potential to widen the capability of Ruby in the numerical analysis field.