RubyGrant 2017 メンター報告書

“Implementation of Ruby/Cumo,a CUDA-aware version of Ruby/Numo”

開発者: 瀬尾 直利

メンター: 村田　賢太

1 このプロジェクトの評価

本プロジェクトは Ruby/Numo の CUDA 対応版の開発を目的としている。担当者の瀬尾氏は、プロジェクト開始時に立てた目標をすべて達成し、さらに追加スコープとして計画していたものの一部も達成している。以下で、個々の成果について評価する。

1.1 計画の達成状況

プロジェクト開始時に設定した計画は次の通りであり、これら全てを達成している。

(1) CPU、GPU間のメモリ転送インターフェースの実装
(2) GPUメモリを高速に管理するための memory pool の実装 (glibc malloc 相当の best-fit with coalescing アルゴリズムの実装)
(3) 初期化ルーチンの実装 (empy, ones, eye, identity, zeros, fill)
(4) 基本的な elementwise function の実装
(5) ruby コードとして記述した演算を、透過的に GPU カーネル関数として実行するための仕組みの構築
(6) 基本的な broadcasting function の実装
(7) 基本的な reduction function の実装。なお、reduction (sum, inner product 演算など) は GPU のような並列演算器にとっては苦手な演算となる。
(8) cuBLAS を利用した基本的な linear algebla function の実装

余裕があったら取り組む課題として設定した追加スコープは次のとおりである。

(a) user defined kernel のサポート. nvrtc を用いてユーザが ruby 上で定義した関数を JIT コンパイル し、GPU kernel 実行する仕組みの構築
(b) apache-arrow data frame のサポート
(c) cudnn, cusolver, cusparse, thrust, curand など、その他の NVIDIA 提供ライブラリサポートの追加
(d) kernel fusion
(e) GPU aware profiler. nvprof サポート

本プロジェクトは、これらのうち (a) の JIT コンパイルと (c) の thrust を達成している。さらに、(e) は特別な対応が不要であることを明らかにした。

1.2 Ruby/Numo の CUDA 対応

Ruby/Numo の機能を CUDA カーネルへ書き換える作業は、全体で 80 ファイルあるうちの 52 ファイルが完了している。その結果、Red Chainer が使用している Numo の機能がすべて CUDA に対応したことにより、Red Chainer を CUDA を用いて高速実行できる見込みができた。

1.3 Cumo のインターフェイス

本プロジェクトは Ruby/Numo の Numo モジュールを Cumo モジュールへ置き換えるだけで、Ruby/Numo を利用したプログラムが CUDA を利用して高速計算できるよう設計している。そのため、次のコマンドによって Red Chainer プログラム中の Numo を Cumo へ置換するだけで、CUDA を利用して計算させることができる。

```
$ find . -name '*.rb' | xargs sed -i -e 's/Numo/Cumo/g' -e 's/numo/cumo/g'
```

1.4 Ruby/Numo との速度比較

Ruby/Numo との速度比較を行い、Cumo を用いた GPU で計算する場合の方が、Ruby/Numo による CPU での計算より約10倍高速である事を示した。

1.5 新たな課題の発見

本プロジェクトは Ruby/Numo の CUDA 対応をする過程で、Cumo、Ruby/Numo、CRuby における次の課題を発見している。

- GC で GPU メモリを管理することによるメモリ再利用率の低下
- Ruby/Numo の実装内部における非効率なメモリコピーによる速度低下
- Numo インターフェースとの非互換
- mkmf が複数コンパイラの使用を想定していない問題
- 拡張ライブラリが他の拡張ライブラリを参照する際の仕組みが存在しないこと

2 メンターとして果たした役割

本プロジェクトのメンターとして、開発期間中に複数回の状況確認とオンラインでのディスカッションを行った。また、Ruby 25周年イベントのポスターセッションへの応募を企画し、本プロジェクトの状況報告をする場を提供し、ポスターの製作・掲示を行った。

3 今後の期待

Cumo によって Ruby/Numo は実用的な GPGPU 対応の仕組みを手に入れたことになる。今後は、追加スコープとして計画していたものの残りに取り組み、さらに開発中に発見した課題の解決にも取り組んでいただけることを期待する。

4 まとめ

本プロジェクトを通して、瀬尾氏は Ruby/Numo の CUDA 対応を非常に高いレベルで達成した。本プロジェクトの成果物である Cumo は、すでに Red Chainer などでの利用が計画されており、将来的に Ruby を科学技術計算分野で使用する場合の計算基盤になることが期待できる。そして、Cumo によって Ruby の科学技術計算分野への対応が加速し、Ruby の将来性の拡大にも大きく寄与するはずである。