Hastega: Elixir プログラミングにおける超並列化を実現するための GPGPU 活用手法

山崎 進　森 正和　上野 嘉大　高瀬 英希

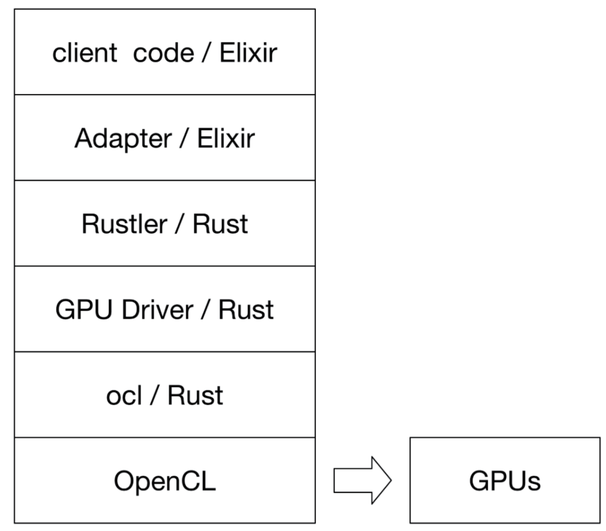
Elixir では Flow を用いた簡潔な表現で
マルチコア CPU の並列性を活用できる

Flow によるプログラム記述が
GPGPU にも容易に適用できると着想

→ Hastega として実装

```
0..10000          単純で均質で大量にあるデータ
|> Flow.from_enumerable
|> Flow.map(foo)          容易にSIMD命令に変換できる
|> Flow.map(bar)          同じような命令列
|> Enum.to_list
```

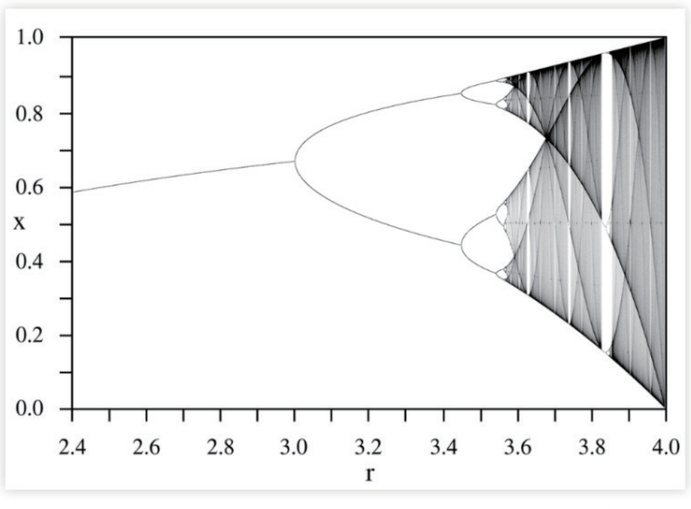
```
__kernel void calc(
  __global long* input,
  __global long* output) {
  size_t i = get_global_id(0);
  long temp = input[i];
  temp = foo(temp);       # 実際には関数呼出しをインライン展開する
  temp = bar(temp);       # ことで,さらなる最適化を促進する
  output[i] = temp;
}
```



Hastega アーキテクチャ

性能に関する比較と評価

整数演算で負荷のかかる計算として、素体のロジスティック写像を用いたベンチマークプログラムを開発し評価した。



$$X_{i+1} = \mu_p X_i (X_i + 1) \bmod p$$

本手法で求めた値をグラフにプロットしても図のようにならない。
本手法はオリジナルの方法と幾何学的性質が異なるためである。

評価環境

Mac Pro (Mid 2010)
Processor: 2.8 GHz Quad-Core Intel Xeon
(プロセッサ数 1，物理コア数 4，論理コア数 8)
Memory: 16 GB 1066 MHz DDR3
Graphics: ATI Radeon HD 5770 1024MB

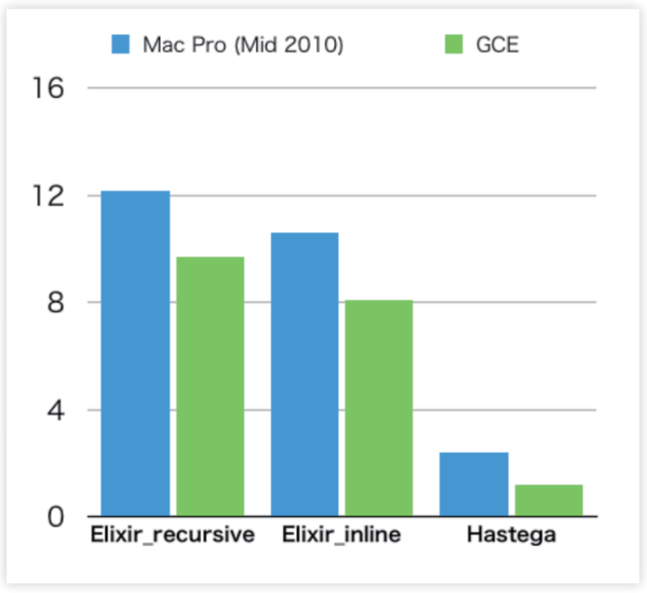
GCE
Machine type：custom
(8 vCPUs、16GB メモリ)
CPU platform” Intel Broadwell
GPU：NVIDIA Tesla K80 (x1)
Zone: us-west1-b

参考文献：Miyazaki, T. et al.: A Study of an Automorphism on the Logistic Maps over Prime Fields, Proc. of ISITA2014.

	Mac Pro (Mid 2010)	GCE
OS	macOS Sierra 10.12.6	ubuntu 16.04
Elixir	1.6.6 (OTP 21)	1.6.6 (OTP 21)
Flow	0.14	0.14
Rust	1.27.0	1.27.0
OpenCL	1.2	1.2
Rustler	0.17.1	0.17.1
ocl	0.18	0.18
rayon	1.0	1.0
scoped-pool	1.0.0	1.0.0
Python	3.6.0 (Anaconda 4.3.0)	3.5.2
CUDA	N/A	9.0 (CuPy), 9.2 (その他)
NumPy	1.11.3	1.14.3
CuPy	N/A	4.1.0

Elixir からの速度向上

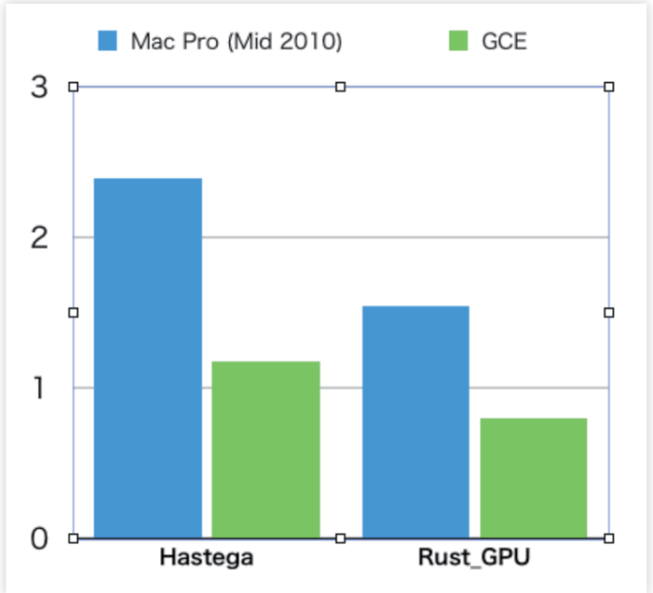
提案手法は Elixir 単体のコードと比べて
4.43〜8.23 倍高速になった



	Mac Pro	GCE
Elixir_recursive	12.177	9.674
Elixir_inline	10.579	8.075
Rustler_GPU (Hastega)	2.388	1.176

ネイティブコードとの比較

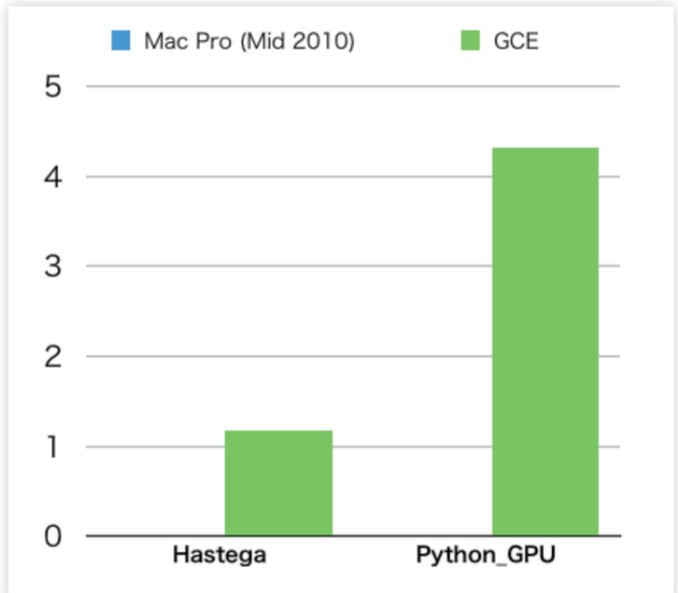
提案手法は GPU を使用するネイティブコードと
比べ，1.48〜1.54 倍遅くなっただけである



	Mac Pro	GCE
Rustler_GPU (Hastega)	2.388	1.176
Rust_GPU	1.546	0.797

Python からの速度向上

提案手法は GPU を使用する Python の
コードと比べ，3.67 倍高速である



	GCE
Rustler_GPU (Hastega)	1.176
Python_GPU	4.316

まとめと将来課題

機械学習のデファクトスタンダードである Python と比べて Elixir の潜在的優位性が示された。
今後，提案手法を用いて Python より性能優位な機械学習のライブラリと新しい処理系 ZEAM を開発したい。