A Parallel Deconvolution Algorithm in Perfusion Imaging

SINAPSE
Scottish Imaging Network - A Platform for Scientific Excellence

Fan Zhu[1,2], David Rodríguez González[1,2,3], Trevor Carpenter[3], Malcolm Atkinson[2] and Joanna Wardlaw[1,3]

[1] SINAPSE [2] Data-Intensive Research Group, School of Informatics, University of Edinburgh. [3] SFC Brain Imaging Research Centre, Division of Clinical Neuroscience.

THE UNIVERSITY of EDINBURGH

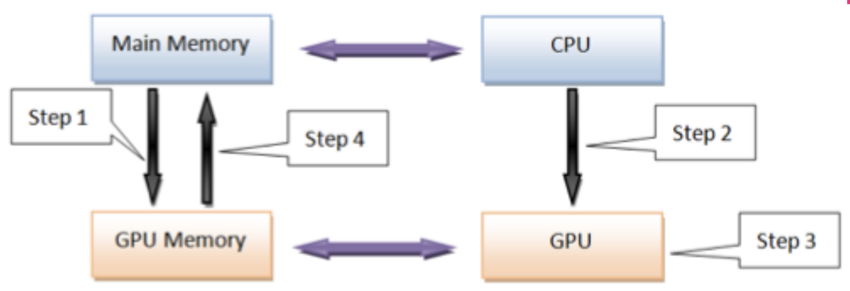
Abstract:

The objective of brain perfusion quantification is to generate parametric maps of relevant hemodynamic quantities such as Cerebral Blood Flow (CBF), Cerebral Blood Volume (CBV) and Mean Transit Time (MTT) that can be used in diagnosis of conditions such as stroke or brain tumors. These calculations involve deconvolution operations that in the case of using local Arterial Input Functions (AIF) can be very expensive computationally.

GPUs originated as graphics generation dedicated co-processors, but the modern GPUs have evolved to become a more general processor capable of executing scientific computations. They provide a highly parallel computing environment due to their huge number of computing cores and constitute an affordable high performance computing method.

We present the serial and parallel implementations of such algorithm and the evaluation of the performance gains on GPGPU (General Purpose Graphics Processor Units) using the CUDA (developed by NVIDIA) programming model.

NVIDIA GPU



CUDA Data and Control Flow

ALGORITHM - PARALLEL PERFUSION IMAGING ANALYSIS

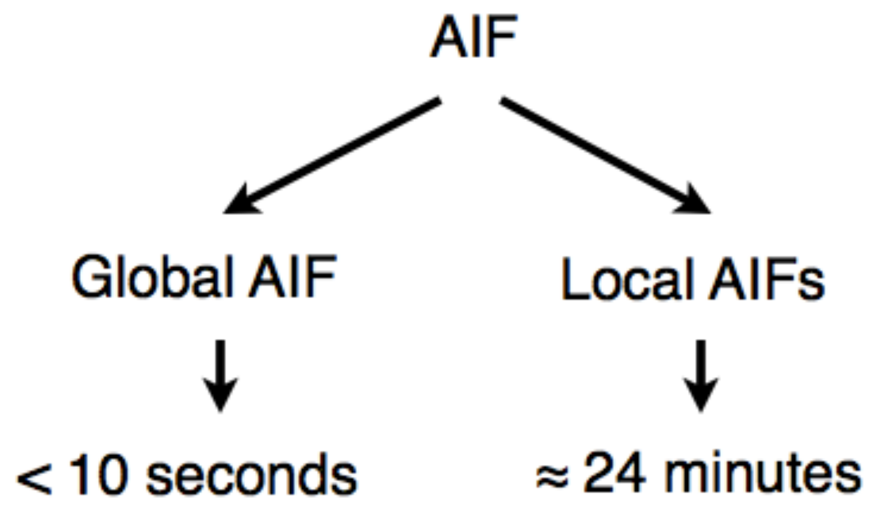
```
CPU.A(1 : Time, 1 : Size) ← 4D MR or CT image data
GPU.A(1 : Time, 1 : Size) ← CPU.A(1 : Time, 1 : Size)
GPU: Parallel do, shared(A, A', A")
if DoImageDenoising = true
   then GPU.A'(1 : Size, 1 : Time) ← reorganise GPU.A(1 : Time, 1 : Size)
        GPU.A"(1 : Size, 1 : Time) = GPU.A'(1 : Size, 1 : Time)
   else GPU.A"(1 : Size, 1 : Time) ← Denoise and reorganise GPU.A(1 : Time, 1 : Size)
GPU: Parallel do, private(localAIF, i, IRF), shared(A, CBF, CBV, MTT)
for n ← 1 to Dim3
    do for i ← 1 to Dim1 × Dim2
           do Generate localAIF(1 : Time)
              IRF ← Deconvolution result (GPU.A"(i+n × Dim1 × Dim2,1:Time) & localAIF(1:Time))
              GPU.CBF(i + n × Dim1 × Dim2) ← Max(IRF)
              GPU.CBV(i + n × Dim1 × Dim2) ← Sum(IRF)
              GPU.MTT(i + n × Dim1 × Dim2) ← GPU.CBV/GPU.CBF
        CPU.CBF(slice n) ← GPU.CBF(slice n)
        CPU.CBV(slice n) ← GPU.CBV(slice n)
        CPU.MTT(slice n) ← GPU.MTT(slice n)
        CBF colored map ← CPU.CBF(slice n)
        CBV colored map ← CPU.CBV(slice n)
        MTT colored map ← CPU.MTT(slice n)
```

Memory Copy

Runs on GPU

Memory Copy

Parallel Perfusion Imaging Analysis Algorithm



Four Intel(R) Xeon(R) CPUs
- Dual cores each
- 3.0 GHz each core
- 8.0 GB global memory
- 4 MB cache each

Versus

Two Tesla C1060 GPUs
- 480 GPU cores in total
- 1.44 GHz each core
- 2.0 GB global memory
- 8 KB shared memory
- 2.073 TFLOPS

The most expensive part in perfusion imaging analysis is calculating inverse of the AIF matrices. Given that there is no dependency or communication between tasks, these calculation can be ideally parallelized using data parallelism, and no big effort is required to separate work into parallel tasks.

Abbreviations:
AIF: Arterial Input Function
CBF: Cerebral Blood Flow
CBV: Cerebral Blood Volume
CT: Computed Tomography
GPGPU: General Purpose Computing on GPU
GPU: Graphics Processing Unit
MTT: Mean Transmit Time
SINAPSE: Scottish Imaging Network - A Platform for Scientific Excellence

PERFORMANCE OF EACH STEP

Step	Serial Running Time (s)	Parallel Running Time (s)	Speedup Factor
Brain data load	0.10	0.10	Not Applied
Data copying (CPU to GPU)	Not Applied	0.17	Not Applied
Data reorganization	1.1	0.01	110
Reorganization & denoising	4.3	0.01	430
Deconvolution	2.1×10^3	8.2×10^2	2.5
Data copying (GPU to CPU)	Not Applied	0.01	Not Applied
Draw parametric maps	0.20	0.20	Not Applied
Overall	2.1×10^3	8.2×10^2	2.5

OVERALL PERFORMANCE

Data Size ($Dim1 \times Dim2 \times Dim3 \times time$)	Serial Running Time (min)	Parallel Running Time (min)	Speedup Factor
128*128*11*44	6.0	1.25	4.8
128*128*22*80	35	13.5	2.6

Performance Improvement

Conclusion:

We introduced an implementation of perfusion imaging analysis which provides considerable speed improvement and equivalent quality of results than current serial implementations.

The improvement over previous methods is a factor of 2.6 to 4.8 depending on data size. As resolution and number of time steps used for brain imaging increases, this approach will show great benefits.

In clinical diagnosis, time is vitally important especially for acute stroke cases, the earlier we deliver the result for diagnosis, the higher the possibility that patients will be cured. Therefore, performance is as important as accuracy in perfusion imaging and our implementation can be used to help clinical diagnosis.

In conclusion, using GPGPU is a desirable approach in perfusion imaging analysis.

edinburgh data-intensive research
research.nesc.ac.uk

THE UNIVERSITY of EDINBURGH informatics

Brain Research Imaging Centre,
University of Edinburgh,
www.bric.ed.ac.uk

F.Zhu@ed.ac.uk