\textbf{\textit{\textlambda}-platform:}

Life and Medical Biology Data Accelerator (Lambda)

Guangming Tan

Institute of Computing Technology, Chinese Academy of Sciences
Biological Imaging Data Challenge

High-Throughput Image Data Analysis is Required!

Higher Resolution

GAP: O(years)!

High Spatiotemporal Resolution Two-Photon Microscope Imaging System

- In vivo
- High Dimension

Peking University
Event Detection at Cellular Level

Elementary Events of Calcium Signals

**Calcium Spark**

Cheng, H
*Science* 1993

Sparks and Transients

**Superoxide Flash**

Cheng, H
*Cell* 2008

Visualization of Reactive oxygen species (ROS)

Animal’s dynamic neural signals

**Dendrite Calcium Imaging**

(Zhuang Zhou, Xiaowei Can)
Life and Medical Biology Data Accelerator (Lambda, \(\lambda\))

**Data**
- PostgreSQL
- Bio-Format

**Engine**
- Domain-Specific Accelerator
- Auto-tuning library

**Pipeline**
- Built-in modules
- Customizable framework
λ-Image

Software/Hardware Stack

- High-dimension & multi-mode biological image data system
- Data analysis pipeline for massive biological image
- Accelerating data-intensive algorithms for biological image analysis

Biological Data Analysis Pipeline (cell event detection, segmentation)

Biological Data Analysis Algorithm Toolkit
- deconvolution
- denoising
- stencil
- machine learning

Cardiovasology
- Mouse embryo heart image cell lineage

Brain
- Mice brain cell Ca2+ spark detection

Endocrinology
- Islet forming in pancreatic and imaging in vivo

Database
- MPI
- Spark
- CUDA
- OpenCL
- RDMA
- Accelerator
High-throughput Image Processing Algorithm

Preprocess
- normalization
- subtract background
- flip

3D Deconvolution

O(N*P^3)

Mutual Information Based Registration

Match
Find the minimum mutual information

Powell
- get trans
- PV interpolation
- mutual information

Registration
O(S * P^2)

Content Based Fusion

O(S * P^2)

Fusion result

Current Computing Systems (Software/Hardware): O(Years)

Interactive → High accuracy

High Performance Computing Platform: O(Minutes)

Unbiased Analysis of Events

Machine Learning
Parallelization with in-memory Computing Model

Preprocess
- Raw Data
- 3D Deconvolution
- Intensity Normalization
- Subtract Background
- Preprocessed Data

Registration
- Left Side
- Right Side
- Match
- Powell
- Mutual Information
- Preprocessed Data

Fusion
- Left Side
- Right Side
- Wavelet Decomposition
- Activity Measure
- Fusion Decomposition
- Fused Data

Segmentation
- Fused Data
- Planarity Enhancement
- Tensor Voting
- 3D Watershed
- Labelmap Image Data

Raw Data

Image L1
- Preprocess
- Registration
- Registered Image L1
- Fusion
- Fused Image1
- Segmentation
- Fused Image Stack
- Final Result For Visualization Process

Image L2
- Preprocess
- Registration
- Registered Image L2

Image R1
- Preprocess
- Registration
- Registered Image R1

Image R2
- Preprocess
- Registration
- Registered Image R2

Spark
GPU Acceleration of Algorithm Modules

- Deconvolution
- Median Filter
- Objectness Filter
- Iterative Closing

Comparison between CPU and GPU acceleration.
Image Processing & Analysis Pipeline

- **Segmentation**
  - Watershed segmentation
  - Labelmap selection
  - Particle analysis

- **Registration**
  - Mutual Information is derived from Information Theory and its application to image registration has been proposed in different forms

- **Fusion**
  - Use global five-level wavelet decomposition

- **Deconvolution**
  - 2D and 3D iterative deconvolution.

- **Analysis**
  - Mutual Information
  - Wavelet based
  - Machine Learning
  - Machine Learning (Event detection or Pattern Analysis)

- **Pipeline Tools**
  - GPU
  - Spark
  - RL/Sparse

- **Applications**
  - Mouse Brain Cell Ca^{2+} Spike Detection
Deconvolution of Pancreas Islet Images

Terabyte EM Images

<table>
<thead>
<tr>
<th>name</th>
<th>pixel (XYZ)</th>
<th>#ite rs</th>
<th>size</th>
<th>Fiji (JAVA)</th>
<th>GPU</th>
<th>speed up</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta.tif</td>
<td>1024x2048x51</td>
<td>100</td>
<td>408MB</td>
<td>60m</td>
<td>22s</td>
<td>163</td>
</tr>
<tr>
<td>glucose_sequential2.tif</td>
<td>512x512x400</td>
<td>50</td>
<td>200MB</td>
<td>30m</td>
<td>10s</td>
<td>180</td>
</tr>
</tbody>
</table>

4.7YEARS 4 GPUs (K20) 2 DAYS
Extracting Cells from Mouse Embryos Images

- Light-sheet microscopes images
- 200 Time points
- 2x500 images
- 2048x2048 pixels
- 4GB*2*200 = 1.6TB

1.5 DAYS

- Fast, two-side, 3D, duel-color imaging
- Reconstruction
Blitz: High Performance Machine Learning Toolkit

**NVIDIA DIGITS (Customized)**

- **Classification**
  - SVM
  - KNN

- **Clustering**
  - K-means

- **Dimensionality**
  - PCA

- **DNN**
  - CNN

**Algorithm Interface**

- **Layer Operation**

- **Virtual Backend**

**Operator Language**
(Linear Algebra / Tensor Primitives)

- **Automatic Performance Tuning**

- **Programming Hardware**
  - Accelerator:
    - vectorization
    - multithread
  - RDMA

**Sugon Xmachine**
Convolutional Nets 2012 (AlexNet)

**Hardware Environment**
CPU: Dual Intel(R) Xeon(R) CPU E5-2680 v3, 28
CPU-Memory: 128GB
GPU: Tesla K20
GPU-Memory: 6GB

**13-layer architecture**

<table>
<thead>
<tr>
<th>Layer</th>
<th>Type</th>
<th>Maps and neurons</th>
<th>Kernel size</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Input</td>
<td>1 map of 224*224 neurons</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Convolution</td>
<td>64 maps of 55*55 neurons</td>
<td>11*11</td>
</tr>
<tr>
<td>2</td>
<td>Pooling</td>
<td>64 maps of 27*27 neurons</td>
<td>3*3</td>
</tr>
<tr>
<td>3</td>
<td>Convolution</td>
<td>192 maps of 27*27 neurons</td>
<td>5*5</td>
</tr>
<tr>
<td>4</td>
<td>Pooling</td>
<td>192 maps of 13*13 neurons</td>
<td>3*3</td>
</tr>
<tr>
<td>5</td>
<td>Convolution</td>
<td>384 maps of 13*13 neurons</td>
<td>3*3</td>
</tr>
<tr>
<td>6</td>
<td>Convolution</td>
<td>256 maps of 13*13 neurons</td>
<td>3*3</td>
</tr>
<tr>
<td>7</td>
<td>Convolution</td>
<td>256 maps of 13*13 neurons</td>
<td>3*3</td>
</tr>
<tr>
<td>8</td>
<td>Pooling</td>
<td>256 maps of 6*6 neurons</td>
<td>3*3</td>
</tr>
<tr>
<td>9</td>
<td>Fully-connected</td>
<td>4096 neurons</td>
<td>1*1</td>
</tr>
<tr>
<td>10</td>
<td>Dropout</td>
<td>4096 neurons</td>
<td>1*1</td>
</tr>
<tr>
<td>11</td>
<td>Fully-connected</td>
<td>4096 neurons</td>
<td>1*1</td>
</tr>
<tr>
<td>12</td>
<td>Dropout</td>
<td>4096 neurons</td>
<td>1*1</td>
</tr>
<tr>
<td>13</td>
<td>Fully-connected</td>
<td>1000 neurons</td>
<td>1*1</td>
</tr>
</tbody>
</table>

**batch size=128**
1 epoch running time

<table>
<thead>
<tr>
<th>framework</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>blitz</td>
<td>1310s</td>
</tr>
<tr>
<td>caffe</td>
<td>1960s</td>
</tr>
</tbody>
</table>

1 batch size running time

<table>
<thead>
<tr>
<th>framework</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>blitz</td>
<td>125ms</td>
</tr>
<tr>
<td>caffe</td>
<td>196ms</td>
</tr>
</tbody>
</table>
Flash Detection

E.Coli, time series, 512X512X(100 frames).

Intensity increases rapidly

Intensity declines obviously

averaged intensities change continuously

A nonstandard flash is not found by either expert or threshold-based method
Automated Flash Detection based on Blitz

F value:
\[ F = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \]
where
- precision = (no. returned flashes)/(no. returned peaks)
- recall = (no. returned flashes)/(no. all the flashes).

- Use cross validation to find parameters to train a model which can get better accuracy and F value.
- Use MPI + CUDA parallelization to reduce training time.
Membrane Segmentation based on Blitz

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Thresholding</td>
<td>445</td>
<td>15522</td>
<td>222</td>
<td></td>
</tr>
<tr>
<td>NIPS 2012</td>
<td>48</td>
<td>434</td>
<td>60</td>
<td>7 Days (Four GPUs)</td>
</tr>
<tr>
<td>Our approach</td>
<td>116</td>
<td>2865</td>
<td>95</td>
<td>2 Days (One GPUs)</td>
</tr>
</tbody>
</table>

Deep Neural Network

<table>
<thead>
<tr>
<th>Layer</th>
<th>Type</th>
<th>Maps and neurons</th>
<th>Kernel size</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>input</td>
<td>1 map of 95x95 neurons</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>convolutional</td>
<td>48 maps of 92x92 neurons</td>
<td>4x4</td>
</tr>
<tr>
<td>2</td>
<td>max pooling</td>
<td>48 maps of 46x46 neurons</td>
<td>2x2</td>
</tr>
<tr>
<td>3</td>
<td>convolutional</td>
<td>48 maps of 42x42 neurons</td>
<td>5x5</td>
</tr>
<tr>
<td>4</td>
<td>max pooling</td>
<td>48 maps of 21x21 neurons</td>
<td>2x2</td>
</tr>
<tr>
<td>5</td>
<td>convolutional</td>
<td>48 maps of 18x18 neurons</td>
<td>4x4</td>
</tr>
<tr>
<td>6</td>
<td>max pooling</td>
<td>48 maps of 9x9 neurons</td>
<td>2x2</td>
</tr>
<tr>
<td>7</td>
<td>convolutional</td>
<td>48 maps of 6x6 neurons</td>
<td>4x4</td>
</tr>
<tr>
<td>8</td>
<td>max pooling</td>
<td>48 maps of 3x3 neurons</td>
<td>2x2</td>
</tr>
<tr>
<td>9</td>
<td>fully connected</td>
<td>200 neurons</td>
<td>1x1</td>
</tr>
<tr>
<td>10</td>
<td>fully connected</td>
<td>2 neurons</td>
<td>1x1</td>
</tr>
</tbody>
</table>
Conclusion

- Develop a Spark-based parallelization framework for high throughput image analysis pipelines

- Optimizations on GPU
  - Core algorithms in image processing (3x-10x)
  - SGEMM in deep learning (↑30%)

- Achieve significant speedups for image processing
  - Years → days
Thanks!