Visualizing Millions of Datapoints with GPUs using both the client and the server

Thibaud Hottelier (thibaudhottelier) thibaud@graphistry.com
GRID + SHIELD

NVIDIA
GEFORCE NOW
THE NEW WAY TO GAME.

$7.99/MONTH
FIRST 3 MONTHS FREE!*

LEARN MORE
GPUs for Live VR Streaming
GPUs for Machine Learning
GPUs to Map Cyber Attacks
1. Distributed Rendering: Dual GPUs

2. GPU Programming with Node
   General Programming with GPUs
   Using node-opencl and cl.js for easy acceleration
Dual GPU Architecture

- decoder
- encoder

Fast networking

- painting engine
- layout & geometry

Smooth visualizations
Scale to big data
We have WebGL, but…

1 GB+/minute input data

Client provides very restricted access to GPU features

Not even WebGL2 provides core performance features, e.g., memory barriers.
Why Not All Serverside?
Always Bet on JS

JS community has (inadvertently) made GPU programming awesome!

“Async function call”

OpenCL “kernels”
CPU
Few but Flexible

Execution units

GPU
Many but Rigid

Same operation
Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.

\[
\begin{array}{c}
(4 \times 0) \\
(0 \times 0) \\
(0 \times 0) \\
(0 \times 0) \\
(0 \times 1) \\
(0 \times 1) \\
(0 \times 0) \\
(0 \times 1) \\
+ (-4 \times 2) \\
\hline
-8
\end{array}
\]

Source: Apple iOS Developer Library
int centerIdx = get_global_id(0);
int sum = 0;

for (int dx = -1; dx <= 1; dx++) {
    for (int dy = -1; dy <= 1; dy++) {
        int idx = centerIdx + (4 * width * dy) + (4 * dx);
        idx = isValidImageIndex(idx, numElements) ? idx : centerIdx;
        int scaleFromMask = mask[(dy+1)*3 + (dx+1)];
        sum += ((int) imageData[idx]) * scaleFromMask;
    }
}

if (isTransparencyNotColor(centerIdx)) {
    sum = (uchar) 255;
}

newImageData[centerIdx] = make8Bit(sum);
Using Kernels From JavaScript

"Async function call"

OpenCL

"kernels"
Node-opencl

Developed by Mikael Sevenier at AMD.
Provides bindings in node to C++ driver functions.
Handles memory management, concurrency, etc., for you!

Works on Intel, Nvidia, and AMD hardware

https://github.com/mikeseven/node-opencl
// Setup Context
var ocl = require('node-openc1');
var platforms = ocl.getPlatformIDs();
var platform = platforms[0];
var devices = ocl.getDeviceIDs(platform, ocl.DEVICE_TYPE_ALL);
var clErrorHandler = function (e) { throw e;};
var context = ocl.createContext([ocl.CONTEXT_PLATFORM, platform],
    device, clErrorHandler, clErrorHandler);
var queue = ocl.createCommandQueue(context, device, 0);
CL.js

```javascript
var cl = require('cljs');
var myKernel = cl.createKernel('kernel.cl', 'kernelName');
var inputBuffer = cl.createBuffer(input);
var outputBuffer = cl.createBuffer(output);

return myKernel.run(
  [256],
  workgroup,
  [inputBuffer, outputBuffer]
).then(function () {
  return outputBuffer.read(Uint8Array);
});
```

[https://github.com/graphistry/cljs](https://github.com/graphistry/cljs)
I Will Do It Live!
Takeaways

Use the very best tool for each task!

Performance (60x with one GPU)

Productivity (95% of code)
Thanks!

Try out

CL.js  github.com/graphistry/cljs
PyGraphistry  github.com/graphistry/pygraphistry

Help us catch hackers

We are hiring
Front-end / Full-stack
UI/UX design

Email us at
thibaud@graphistry.com