Javascript-GPU

By
Aparna Tiwari
Saurabh Malviya
Motivation & Background

- Till recently Javascript executed sequentially
- Not leveraging from parallel client hardware
- Resulting in not so good user experience
Recent works

- Intel’s Rivertrail project
- Modern browsers like Chrome & IE9
Our approach

- To keep it old & simple
- Applicable to any computation intensive algorithm
- Power in the hands of Javascript programmer
Flow – Sequential - JavaScript

Code Block 1

Code Block 2

Code Block 3
Flow – Hybrid – JavaScript+GPU

Code Block 1

Parse Code Block 2

Convert to LLVM

Code Block 3

Convert to ptx and execute
Example

```javascript
Components.utils.import("resource://gre/modules/ctypes.jsm");
var problemsize = 512 * 100
var bufferA = new ArrayBuffer(problemsize);
var bufferB = new ArrayBuffer(problemsize);
var bufferC = new ArrayBuffer(problemsize);
var N = problemsize

var float32ViewA = new Float32Array(bufferA);
var float32ViewB = new Float32Array(bufferB);
var float32ViewC = new Float32Array(bufferC);

/* Open the library */
try {
    var klib = ctypes.open("/home/salviyas/JS-GPU/libklib.so");
}
catch (c) {
    alert("Error : Unable to load klib");
}

var execute_kernel = klib.declare("execute_kernel",
    ctypes.default_abi,
    ctypes.void_ptr_t,
    ctypes.void_ptr_t,
    ctypes.void_ptr_t,
    ctypes.int32_t
);

var ret = execute_kernel(float32ViewA.buffer, float32ViewB.buffer, float32ViewC.buffer, problemsize);
```
Benchmarking in progress

- N-body
- Ray tracing
Thank You