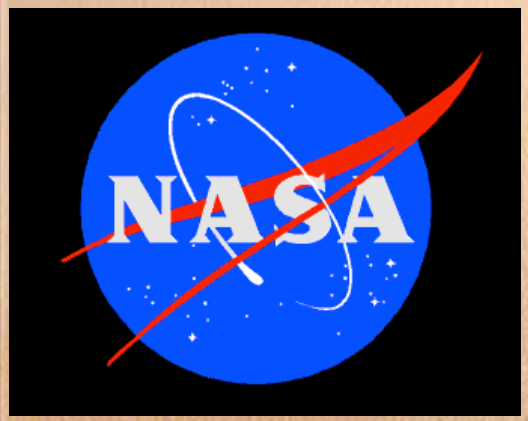
Numerical Modeling of EMRIs Accelerated by OpenCL

Dr. Gaurav Khanna

Justin McKennon



Need for Parallellization

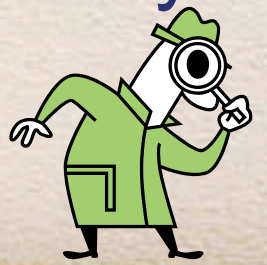
- Processor frequency stalled at around 3GHz
- Forced computer industry to move towards many-core architecture
- Science, Math, Technology getting more and more complex
- How will the computer industry keep up?

Realization of the Power of GPUs

- GPUs contain hundreds of cores (processors) -making them ideal for parallel processing
- Traditionally used for gaming
- Thousands of calculations per second - all in-game movements, actions etc are processed by the GPU
- Think of how poorly games would run on computers without graphics cards

High Performance

- Several vendor and design specific software developments available
- CUDA, ATI Stream, Cell SDK
- Very powerful languages - able to achieve significant performance gains
- Why doesn’t everyone use them?



Short Comings

- Difficult to use
- Vendor and design specific - Can only be used on specific machines and architectures - little in common between languages
- Often requires advanced hardware knowledge

A New Frontier

- OpenCL-Open Computing language
- Developed by Apple and the Khronos group
- All major processor vendors (Nvidia, ATI etc) have adopted the standard
- Completely hardware and device independent

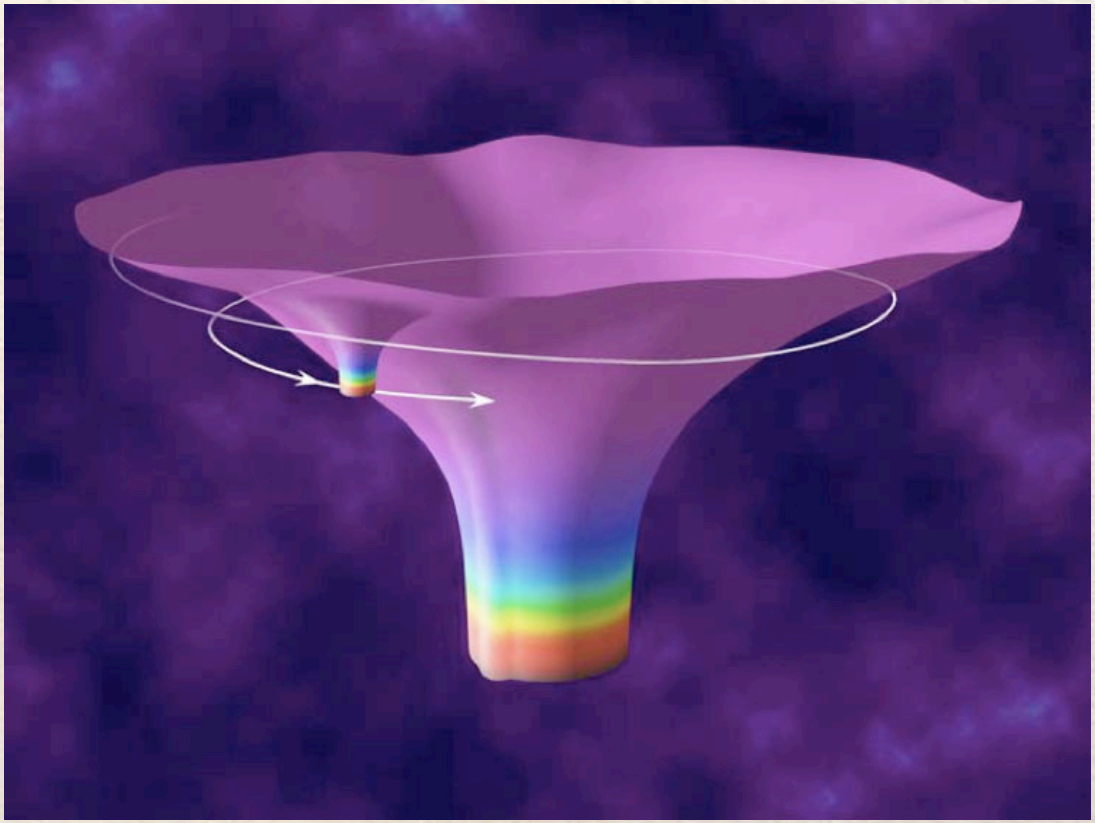
Advantages

- Saves huge amounts of time - no need to re-code for different accelerators
- Derived from the programming language C
- Easy memory management (in contrast to other parallel programming languages)

Numerical Relativity

- Study of strong sources of Gravitational Waves (GWs)
- GWs predicted by Einstein's Theory of Relativity, never directly observed - technology wasn't advanced enough until recently

Our Application



Evolution

- Evolves GWs generated by a compact object in a decaying orbit around a super massive black hole (SMBH)
- SMBH are often over 1,000,000 times more massive than our own sun
- SMBH often reside at the center of galaxies and routinely devour planets and stars
- Called EMRI's due to the ratio of the compact objects mass to that of the SMBH

Continued

- Due to the extreme mass ratio, compact object can be viewed as a structureless object or “blob”
- Problem can now be addressed under black hole perturbation theory

Teukolsky Equation

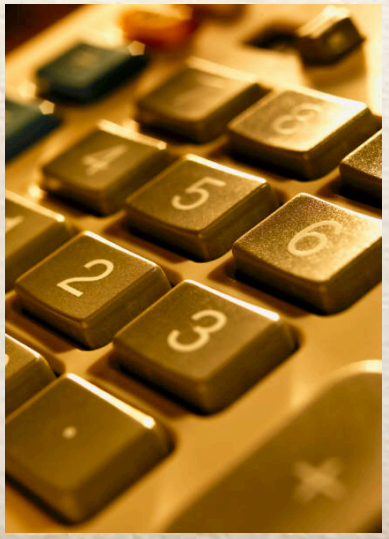
Governs Evolution of GW's

$$
\begin{aligned}
&-\left[\frac{(r^2+a^2)^2}{\Delta}-a^2\sin^2\theta\right]\partial_{tt}\Psi-\frac{4Mar}{\Delta}\partial_{t\phi}\Psi\\
&-2s\left[r-\frac{M(r^2-a^2)}{\Delta}+ia\cos\theta\right]\partial_t\Psi\\
&+\Delta^{-s}\partial_r\left(\Delta^{s+1}\partial_r\Psi\right)+\frac{1}{\sin\theta}\partial_\theta\left(\sin\theta\partial_\theta\Psi\right)+\\
&\left[\frac{1}{\sin^2\theta}-\frac{a^2}{\Delta}\right]\partial_{\phi\phi}\Psi+2s\left[\frac{a(r-M)}{\Delta}+\frac{i\cos\theta}{\sin^2\theta}\right]\partial_\phi\Psi\\
&-\left(s^2\cot^2\theta-s\right)\Psi=-4\pi(r^2+a^2\cos^2\theta)\,T,
\end{aligned}
$$

Our application deals with solving for the term T, known as the source term.

Continued

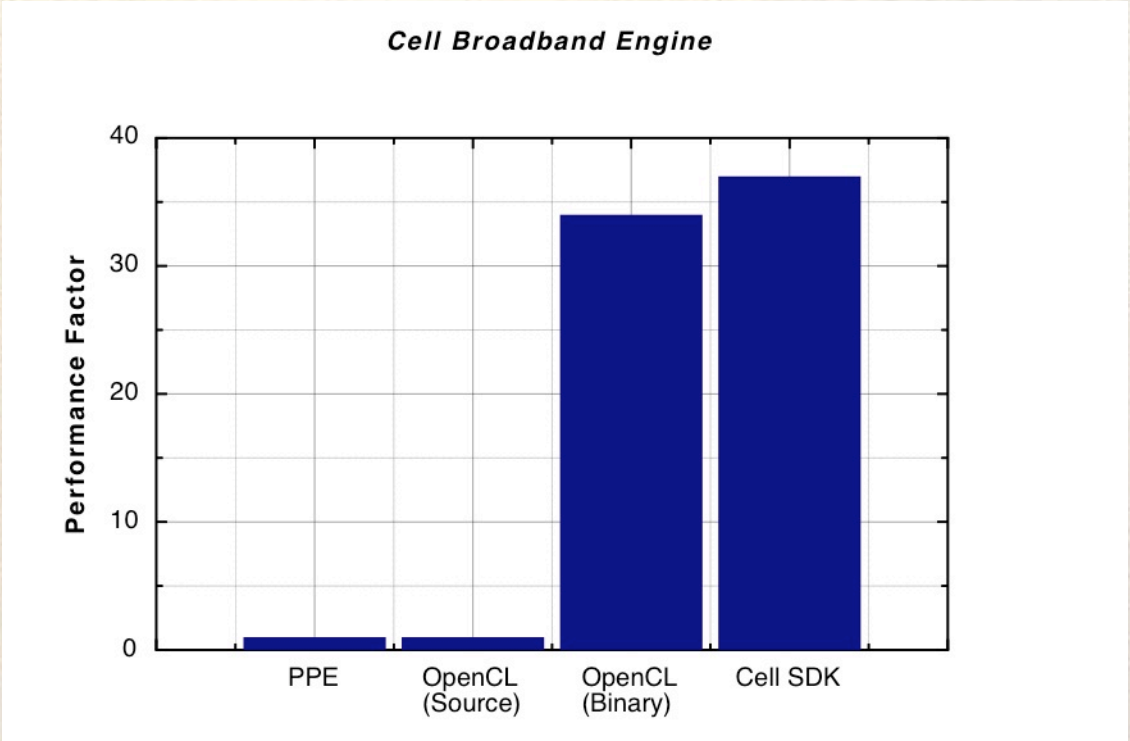
- Much too long to reproduce here
- Absolutely impossible to solve by hand
- Very time consuming for the CPU to solve for



Acceleration:Results

- First chart: Results running calculation on Cell Broadband Engine
- OpenCL(source) is with the calculation kernel NOT precompiled
- OpenCL(binary) is with kernel precompiled

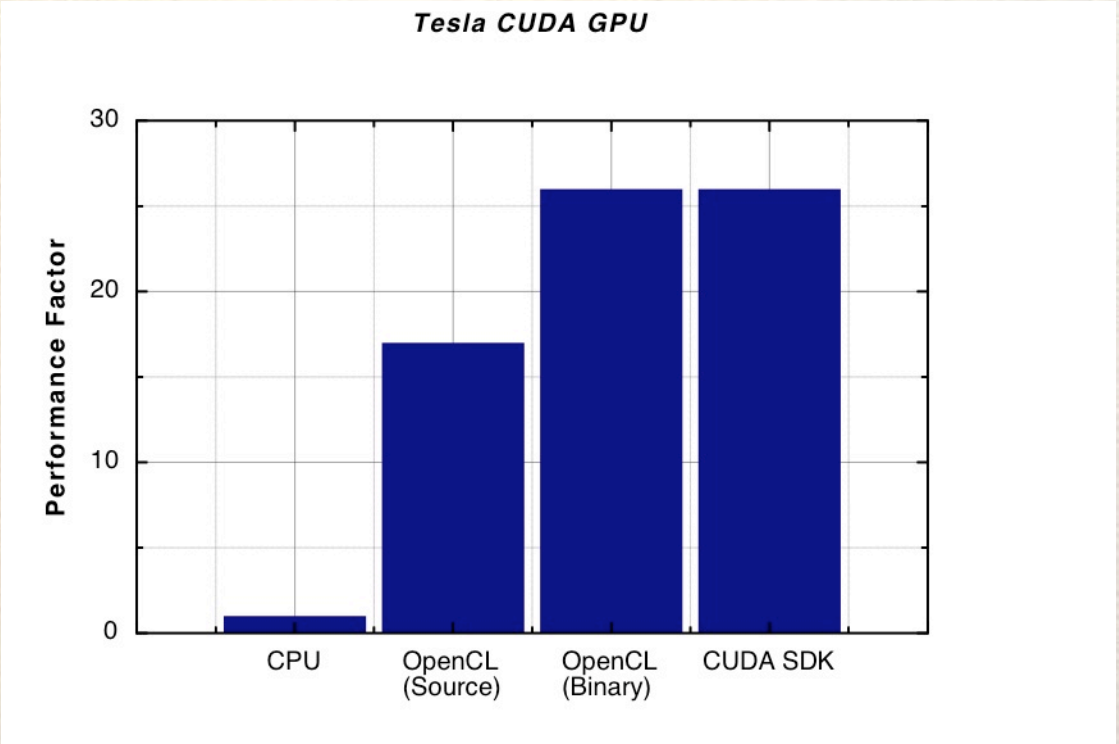
Close Comparison



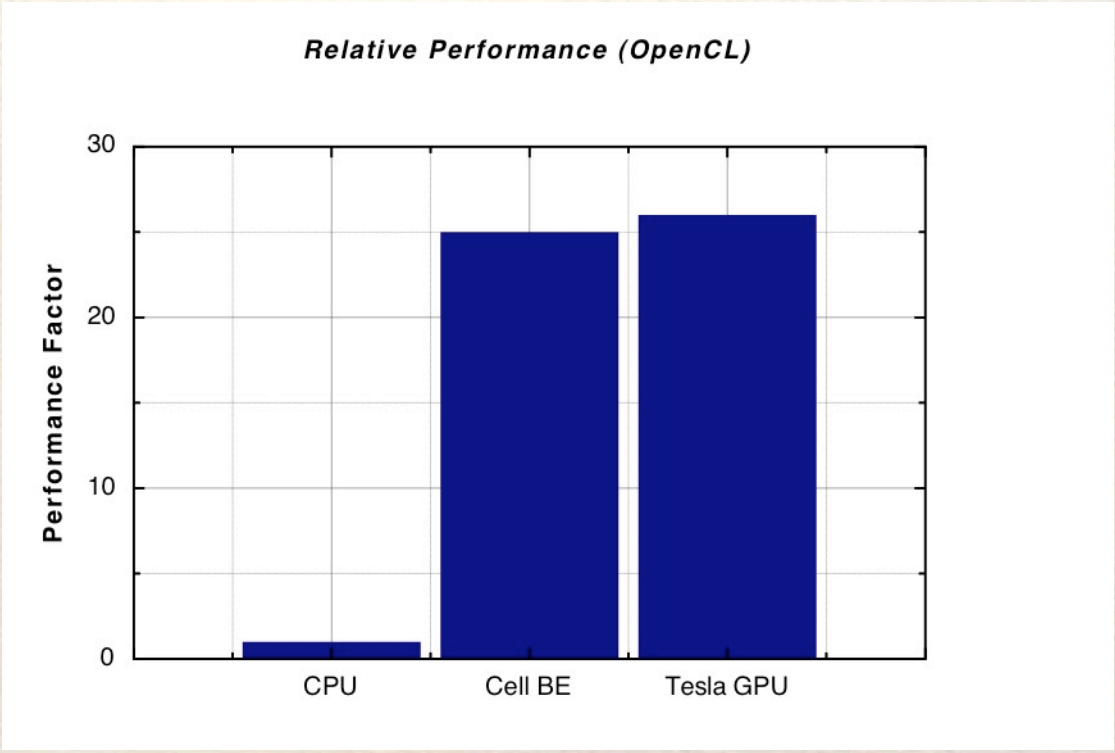
Tesla

- Simulation run on Nvidia Tesla GPU
- Baseline is AMD quad core 2.5GHz processor

Even Showing



OpenCL Relative Performance



Baseline here is the AMD Phenom 2.5GHz quad core processor

Conclusion

- OpenCL performed extremely well, regardless of the architecture
- Powerful results, and it isnt even a year old yet!
- With further updates and releases, OpenCL is expected to surpass both CUDA and the Cell SDK in performance

Difficulties

- Difficult to address problems- no experts in OpenCL yet
- Memory management is direct, but very delicate -high performance requires proper allocation of resources
- Kernel very tough to debug when run on the GPU