Co-Design, Domain-Specific Languages and the Road to Exascale

Patrick M\textsuperscript{c}Cormick

Applied Computer Science Group
Los Alamos National Laboratory
Background and History

• Los Alamos & DOE have been involved in GPU computing from the “dark ages”
  • Register combiners/OpenGL pipeline
• Home of the first petaflop computer (Roadrunner) – an accelerator-based architecture... (in top 10 of Green500)
• Funded portions of early work in academia that directly influenced what would become CUDA and OpenCL
• Funded the development of NVIDIA Linux GPU drivers
• Active R&D activities continue...
Emerging Architectures

• New architectures present a significant challenge to our mission
  • “We’ve been here before... Before...”
  • Or have we?
Approach #1

"I want you to find a bold and innovative way to do everything exactly the same way it’s been done for 25 years."

Copyright 2004 by Randy Glasbergen.
www.glasbergen.com
Approach #2

[Science is a] "series of peaceful interludes punctuated by intellectually violent revolutions" [where] "one conceptual world view is replaced by another".

-- Thomas Kuhn
What is Co-Design?

• A cross-discipline approach to addressing the path to exascale (and emerging architectures)
  • Solutions are not possible without broad participation
• LANL co-design summer school:
  • [http://codesign.lanl.gov](http://codesign.lanl.gov)
• DOE ASCR/NNSA Efforts
• More efforts on the way???
The Programming Challenge

- Multiple platforms, architectures and programming models
- Common approaches:
  - Emulate MPI (“MPI everywhere”)
  - MPI + “X” (CUDA/OpenCL/OpenMP /...)
- Nice short-term solution(s)
- Potentially scary in terms of exascale...
Domain-Specific Languages

• Definition: A language that exploits domain knowledge for both productivity and efficiency

• MATLAB, R, SQL, LaTeX

• By definition, well suited for co-design...

• What if our domain was a computational mesh?

Unstructured Mesh in C

```
for(int ifa=0; ifa < nfa; ifa++) {

double x_fa_approx[3] = {0.,0.,0.};
for(int nof = nofa_[ifa];
    nof < nofa_[ifa+1]; nof++) {
    for(int i = 0; i < 3; i++) {
        x_fa_approx[i] += x_no[nofa_v[nof]][i];
    }
    for(int i = 0; i < 3; i++)
        x_fa_approx[i] /= (double)(noofa_i[ifa+1] – noofa_i[ifa]);
.
```

Unstructured Mesh in Liszt

```
for( c <= cells(mesh) ) {
    val center = avg(pos(c.vertices))
    for( f <= faces(c) ) {
        val face_dx= avg(pos(f.vertices)) – center
        .
```

Built-in features:
• mesh, cells, faces, edges, vertices
• fields
• linear operators as matrices

See paper in this year’s proceedings...
Embedded DSLs: *DSL is hosted within another language*

- Leverages host language’s compiler to implement *portions of the language*
- E.g., no assumptions should be made by the programmer about data are laid out in memory
- These abstractions are first-class language constructs that the compiler can reason about

Scout DSL embedded in C++

```cpp
#include <ostream>
using namespace std;

uniform mesh MyUMesh {
    cells:
        float t0, t1;
    vertex:
        float3 v;
};

int main(int argc, char *argv[])
{
    MyUMesh m[512,512];
    ...
}
```

*vector values at cell vertices.*
*t0 and t1 at cell centers*
Abstractions & Flexibility

- Extend mesh-based abstractions to support end-user programmable in situ visualization
  - Modeled after graphics shaders
  - Exploring expanded support for other operations and representations

```c
renderall cells c of m {
    ...
    float norm = t / MAX_TEMPERATURE;
    float hue  = 240.0 – 240.0 * norm;
    color = hsv(hue, 1.0, 1.0);
}
```
Summary

• Co-Design is the critical path for success at Exascale
• We need to explore new ways of programming
  • That lift the level of abstraction (further) away from the hardware complexities