Programming Languages and Analyses for HPC

Stefan Muller (Illinois Tech)

Argonne-Illinois Tech Research Seminar

Mar. 23, 2022
Short Bio

• PhD 2018, CMU
  • Thesis: Responsive Parallel Computation
• Postdoc, CMU, 2018-2020
  • Static analysis for research usage, especially in CUDA
• Joined Illinois Tech CS in 2020
Cilk, Go, Parallel ML, Parallel Haskell, ...

Theorem: $T(P) \leq \frac{W}{P} + S$

So why don’t people use the abstractions?

(Short answer: user interaction)

POSIX threads
My PhD thesis

• Extend implicitly parallel languages with priorities for responsiveness
Partial order of priorities provides an intuitive, modular abstraction
We track priorities through code in **types**

order low < high

cmd[high]
{
  t <- spawn[high] { ... };
  ...
  sync(t)
}

• This thread is high-priority
• Spawn a high-priority thread
• Sync on it

constraint violated at example.prm:5.1-5.8 : high <= low
Type error: constraint violated
CUDA

• The promise: write C programs, run them on GPUs!

• The reality: complicated execution model, unexpected performance bottlenecks:
  • Warp divergences
  • Uncoalesced memory accesses
  • Bank conflicts
End-to-end static analysis for CUDA resource usage

Cost Model

Cost-annotated sequential program

Absynth [Ngo, Carbonneaux, Hoffmann]: Static resource analyzer
Static timing analysis is important for critical applications

- Predicted worst case
- Simulated worst case
- Simulated best case
- Random instances
CUDA: Current Work

• Automatically optimize programs

• Analyze and predict thread-level parallelism
Very new project: dependent array descriptors

(int **A

for (i = 0; i < N; i++) {
    for (j = 0; j < M; j++) {
        ... A[i][j] ...
    }
}

(Part of joint work with Zhiling Lan, Valerie Taylor, Mike Papka, Romit Maulik and Xingfu Wu)
Very new project: dependent array descriptors

MPI_Send

int **A

for (i = i0; i < i1; i++) {
    for (j = j0; j < j1; j++) {
        ... A[i][j] ...
    }
}

for (i = i0; i < i1; i++) {
    for (j = j0; j < j1; j++) {
        ... A[i][j] ...
    }
}
Very new project: dependent array descriptors

```c
for (i = i0; i < i1; i++) {
    for (j = j0; j < j1; j++) {
        A2[i][j] = A[i][j]
    }
}

for (i = 0; i < i1 - i0; i++) {
    for (j = 0; j < j1 - j0; j++) {
        A2[i][j] = A[i][j]
    }
}
```
Very new project: dependent array descriptors

Goal: **automatically** generate code to send minimal amount of data

```c
int **A

for (i = 0; i < N; i++) {
    for (j = 0; j < M; j++) {
        if (A[i][j] != 0) {
            if (A[i][j] % 2 == 0)
                f(A[i][j])
            ...
        }
    }
}
```
Contact

smuller2@iit.edu

cs.iit.edu/~smuller