Toward a Data-Centric Profiler for PGAS Applications

Hui Zhang, Jeffrey K. Hollingsworth (advisor)
{hzhang86, hollings}@cs.umd.edu
Department of Computer Science, University of Maryland-College Park
Why we need a data-centric profiler?

```
1: int busy(int *x) {
2:     *x = complex();  //consumes the most time
3:     return *x;
3: }
4: int main() {
5:     for (i=0; i<n; i++) {
6:         A[i] = busy(&B[i]) + busy(&C[i]);
7:     }
8: }
```

**Code-centric Profiling**

main: 100% latency  
busy: 100% latency  
complex: 100% latency

**Data-centric Profiling**

Array A: 100% latency  
main
Array B: 50% latency  
main.busy.complex  
Array C: 50% latency  
main.busy.complex

**Figure 1.** Code-centric aggregates metrics to the different functions based on sampled lines, while data-centric can distinguish these metrics by different variables
How the tool works?

Figure 3. Our Profiling Procedure

Each barrel represents the full variable--metrics mapping information on each computing node.

*Each barrel represents the full variable--metrics mapping information on each computing node.*
A Simple Test Program

record Birthday {
    var year: int;
    var month: int;
    var day: int;
}
record Actor {
    var name: string;
    var bd: Birthday;
}

for i2 in 1..LARGE {
    mid = i%8;
    ActorA.bd.month = CPUheavier(mid);
    ActorA.bd.day = CPUheavy(mid);
}
for i in 1..LARGE/2 {
    ActorB.bd.year = CPUheavy(i);
}

Figure 4. Example Chapel code and profiling result
THANK YOU!