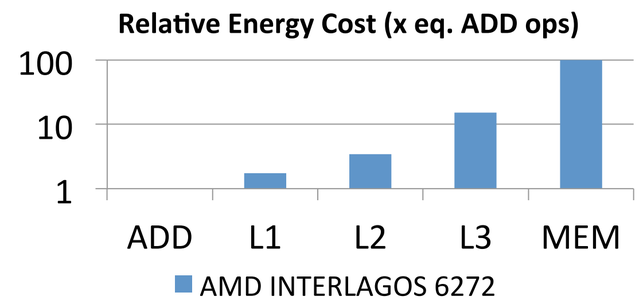
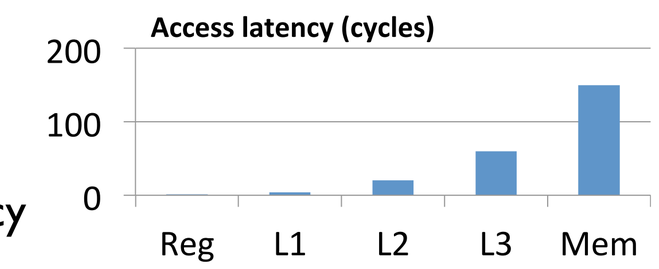
Data Profiler: Exposing Data Movements

William Wang, Chris Emmons, Nigel Paver

Motivation and Goals

- Data movements dominate and cost 2~100x more energy than computations
- Data profiler addresses the rising cost of communication by exposing data movements in the memory hierarchy to enable
  - ✓ Optimizing data structures and data access patterns for data locality
  - ✓ Optimizing memory hierarchies
  - ✓ Studying the potential of heterogeneous memories to improve system efficiency





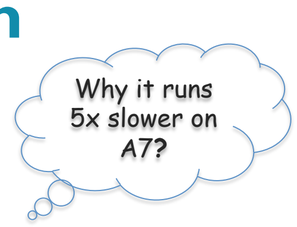
Source: Kestor, Gokcen, et al. "Quantifying the energy cost of data movement in scientific applications."

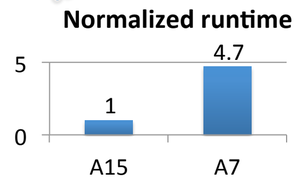
Example Problem

```
#define M = 2048;   // stride distance
#define N = 64;     // num of elements
#define IREP = 200; // iterations

double x[M*N], y[M*N];

for (int j = 0; j < IREP; ++j) {
  for (int i = 0; i < N*M; i += M) {
    y[i] += x[i];
}}
```





The run environment is on Linux with normal page size 4KB on TC2. IREP is set to 2M.

Reference: Gutierrez, et al. "Sources of Error in Full-System Simulation."

- **Profile** the code through DataProf and visualize the results in Streamline Data View

Data variable	Size	Accesses	Read	L1$	Compulsory	Conflict	Capacity
y[131072]	8	25600	12800	12800	64	12736	0
x[131072]	8	12800	12800	12800	64	12736	0

L2ReadMisses	Compulsory	Conflict	Capacity	L2WriteMisses	Compulsory	Conflict	Capacity
10333	64	10269	0	0	0	0	0
10426	64	10362	0	0	0	0	0

A15 L2$

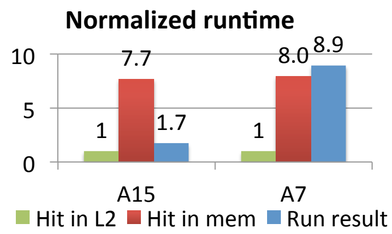
L2ReadMisses	Compulsory	Conflict	Capacity	L2WriteMisses	Compulsory	Conflict	Capacity
19215	64	19151	0	0	0	0	0
12795	64	12731	0	0	0	0	0

A7 L2$

- **Analyze** data accesses in D$. All L1D$ reads miss due to *conflict* apart from cold misses, only one cache set utilized. In A7, most L2$ reads miss due to *conflict*, and most reads go to memory. In A15, more reads hit in L2.

```
L1D$ A15:   32KB, 2-way, LRU,
     A7 :   32KB, 4-way, Pseudo random

L2$  A15:    1MB,16-way, Random,
     A7 : 512KB, 8-way, Pseudo random
```



Optimize for Data Locality

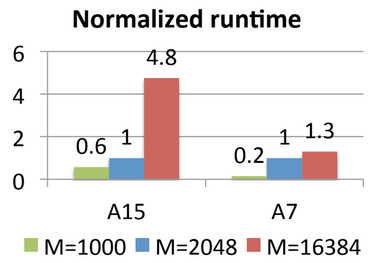
- **Optimize** software and hardware for data locality

Software optimizations

- Don't stride at the D$ set size
- Reorganize array elements

Hardware optimizations

- Hashed cache indexing
- Increase A7 L2 associativity



DataProf Exposes Data Movements

- Data profiler *identifies data hotspots* in user space, including static, local and dynamic data variables
- Data profiler *correlates data hotspots with cache misses*, breakdown into compulsory / conflict / capacity misses, and true / false sharing misses
- *Optimize for data locality* based on analysis of data profile including temporal and spatial access patterns, sharing and movement of data at the system level

Data Optimization During Its Lifetime

Definition: Structure padding, splitting, field reordering

Allocation: Customized allocator for collocation of objects

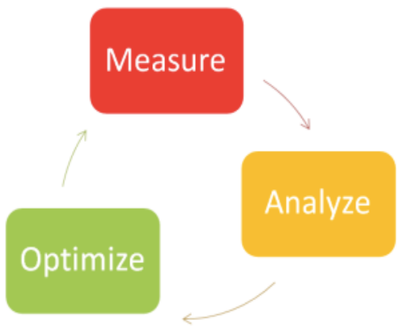
Reorganization: Topology or profile-guided

Conclusions

NIKLAUS WIRTH, 1976

Algorithms + Data Structures = Programs

- DataProf helps measure and optimize data locality
- Data locality reduces data movements and leads to better system energy efficiency and performance



Data Profiling → Data Locality Optimization → Energy Efficiency & Performance