Prometheus: Scalable and Accurate Emulation of Task-Based Applications on Many-Core Systems

GOKCEN KESTOR, ROBERTO GIOIOSA, DANIEL CHAVARRÍA-MIRANDA

2015 Workshop on Modeling & Simulation of Systems and Applications

Exascale parallel programming models

- Exascale systems will achieve high performance through high level of parallelism
  - O(1k-10k) cores per node
  - O(billion) concurrent threads
- Task-based programming models are a promising way to program exascale applications
  - Applications are divided into a myriad of small tasks
  - The system is oversubscribed with tasks ($N_{tasks} >> N_{cores}$)
  - e.g., Cilk++, Intel TBB, Charm++, etc.

There are no tools to model these systems (hardware and software) at the level of scalability required.

Simulators/Emulators landscape

Pacific Northwest NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

1-8 Cores Error < 10%	1-O(100) Cores Error < 25%	1-1k Cores Error ?	1-O(1000) Cores Error ?
Cycle-accurate simulators	Cycle-approximate simulators	Full-system emulators	Discrete-event emulators

Prometheus

Accurat

Pros: A
Cons: S
Cores:
App: S
Ex: Ge

Emplov

Pros: F
Cons: I
Cores:
App: C
Ex: Zsi

Model f

Pros: F
Cons: I
Cores:
App: C
Ex: KV

Trace can be manipulated

Pros: Scalable
Cons: Reordering change the app behavior
Cores: 1-thousands
App: OMP, MPI
Ex: BigSim, DIMEMAS

Prometheus

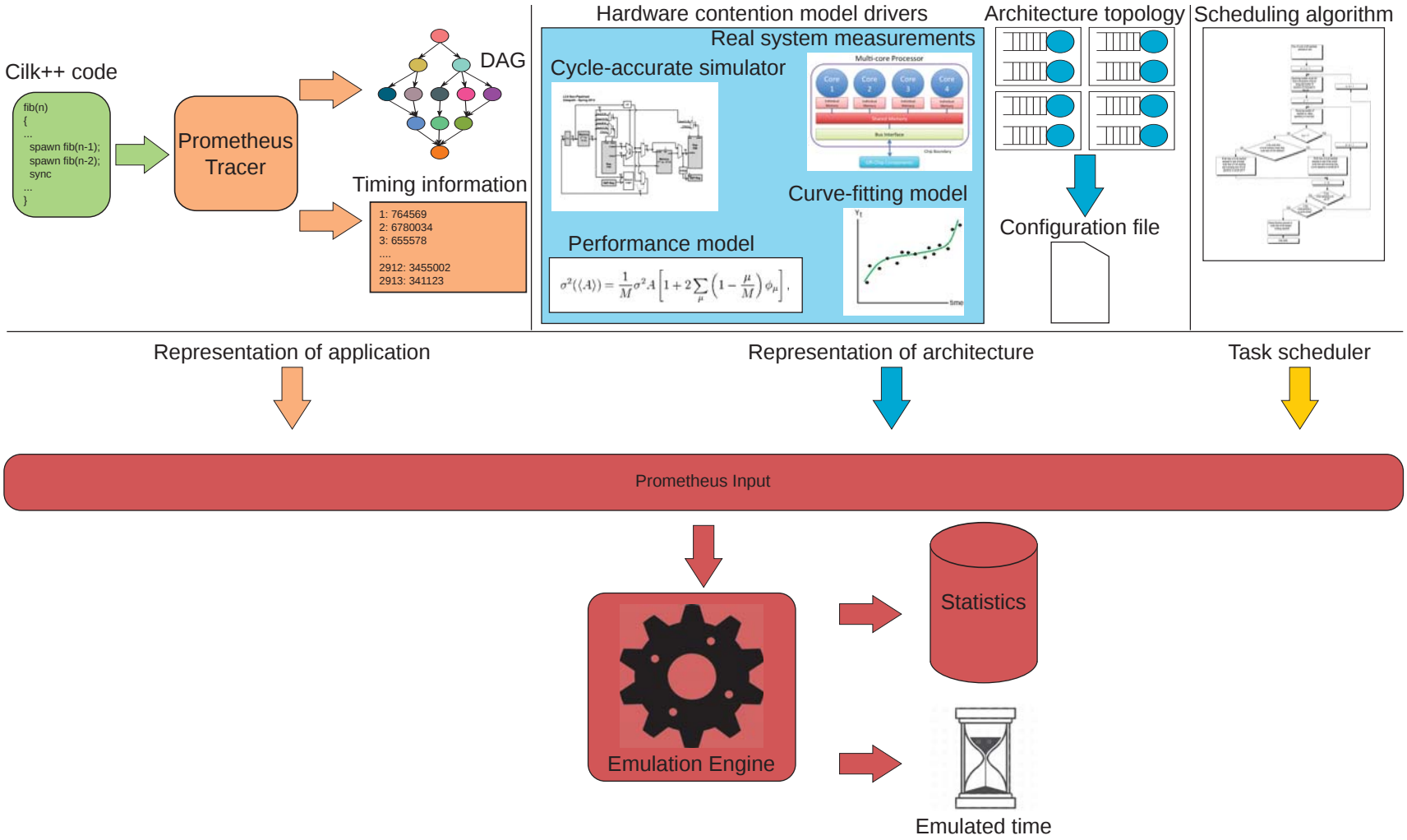
Pacific Northwest NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

Key Observation:
When modeling parallel applications on large systems, performance is largely dominated by runtime and synchronization effects

- Fast enough to emulate future exascale nodes
- Accurate and reliable results
- Modular, swap components in and out
- Model **non-determinism** of task-based applications

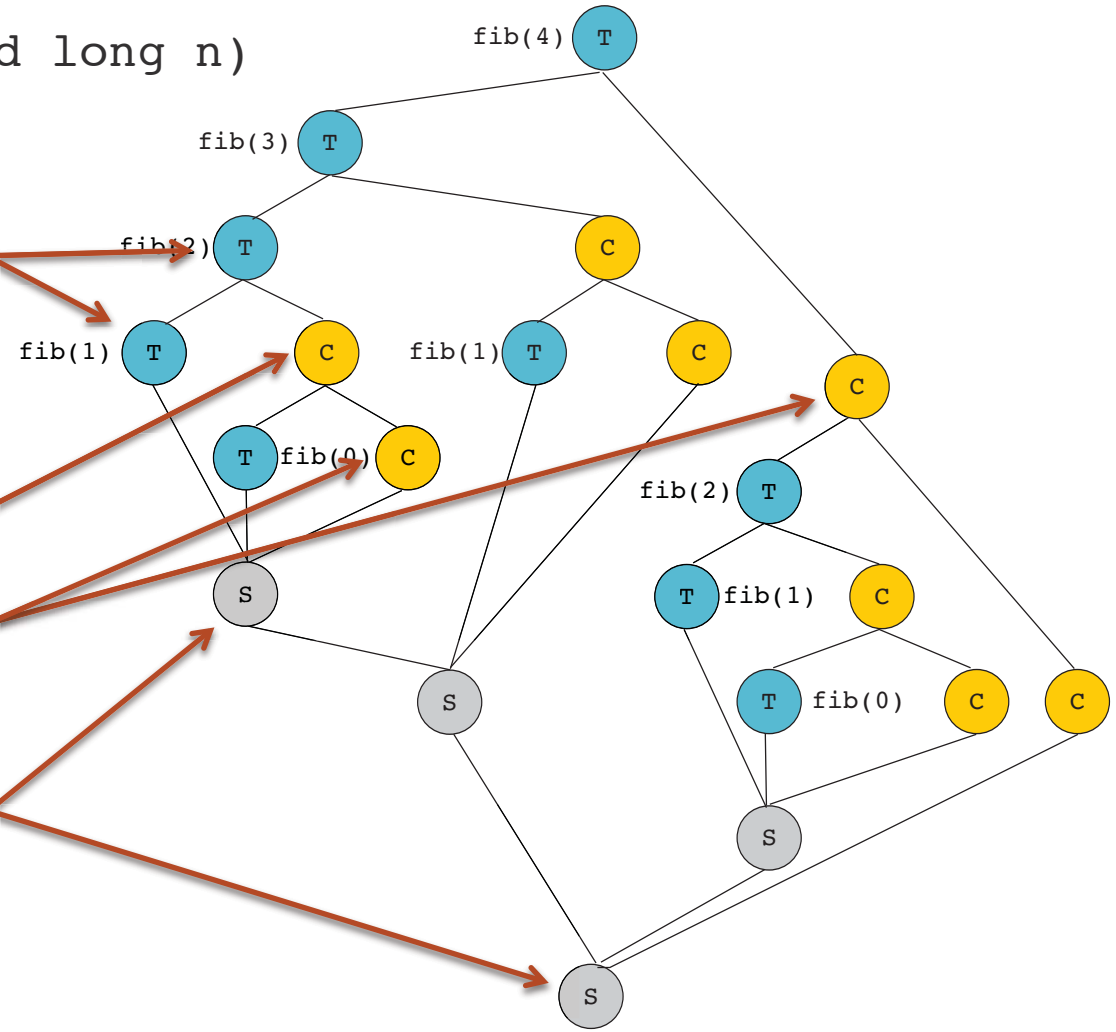
Prometheus Architecture

Pacific Northwest NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965



DAG tracer

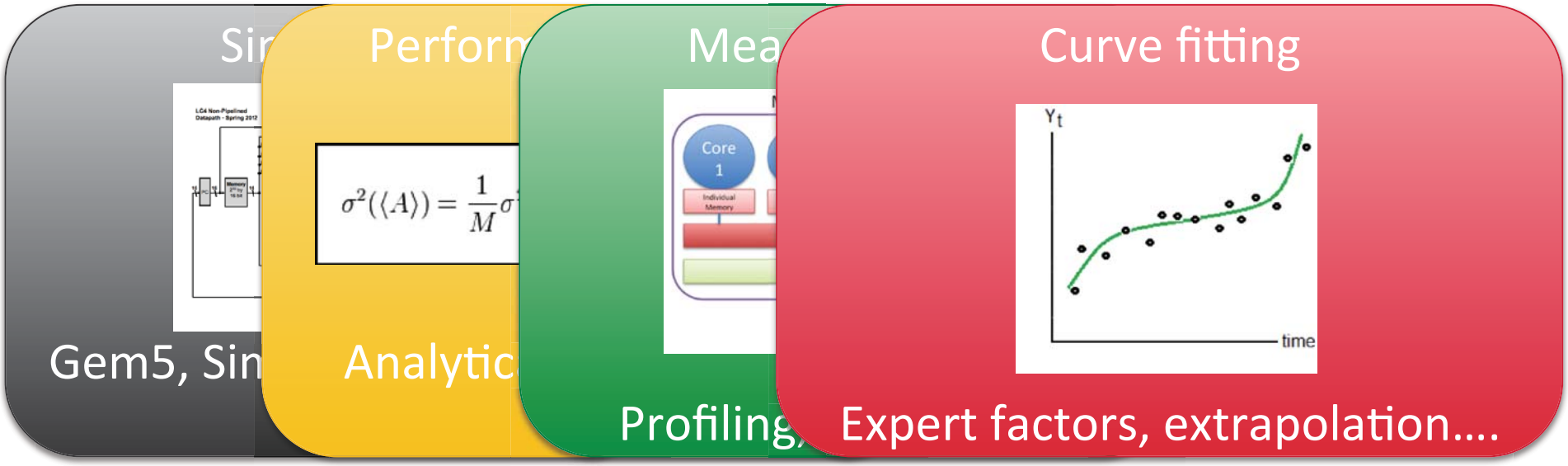

```
unsigned long fib(unsigned long n)
{
    unsigned long x, y;

    if ( n < 2 )
        return n;

    x = spawn fib(n-1);

    y = spawn fib(n-2);

    sync;

    return x + y;
}
```



Hardware contention model

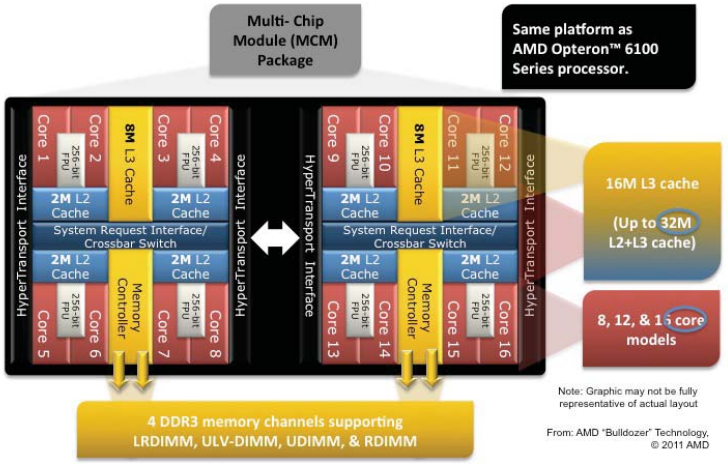
- The DAG is extracted from single thread execution
  - Difficult and not necessary to extract a DAG from parallel execution
  - DAGs do not depend on the level of execution parallelism

- Hardware contention in multi-core and multi-threaded processor is modeled by the contention module.



Validation – Experimental setup

- Modeling a AMD Interlagos systems*
  - 2 processors chips
  - 4 modules, 4 cores/module, 2 threads/core
  - 32 threads total
  - 64 GB DRAM, 4 NUMA domains
- Cilk++ task-based applications
  - Random work-stealing, work-first scheduler



Application	Configuration	Number of tasks
Fib	n = 35	74,651,756
Heat	nt = 200, nx = 4096*4, ny = 1024	1,234,945
Integrate	xMax = 5000	193,385,368
Jacobi	n =1024, steps = 100	139,809,901
MatrixMul	n = 256	71,902,351
QuickSort	n = 75000000	152,452,586

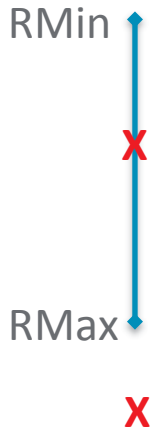
Parameter	Same core	Local NUMA	Remote NUMA
Successful steal	7555	9135	10635
Unsuccessful steal	1148	1199	1382

* Results for Intel MIC (244 hardware threads) in the paper.

Validation – Results

Pacific Northwest NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

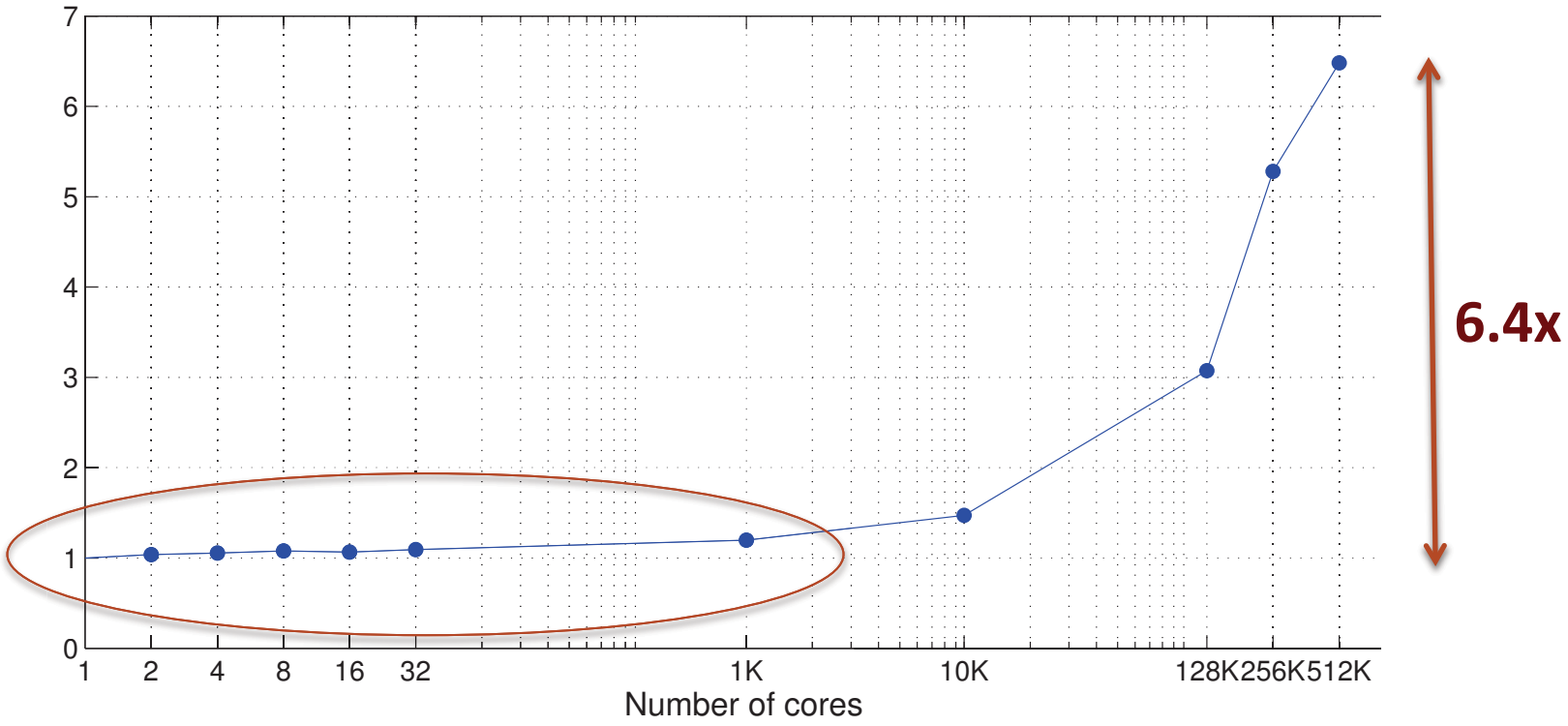
N		Fib	Heat	Integ.	Jacobi	MMul	qSort
8	RMin	53.75	29.20	53.86	24.46	104.94	5.39
	Emu	54.10	29.03	52.10	24.53	105.50	5.32
	RMax	54.88	30.01	54.40	25.21	106.24	6.17
	Error	**0.00**	**-0.58**	**-3.28**	**0.00**	**0.00**	**-1.28**
16	RMin	26.39	18.46	52.43	12.86	52.28	3.86
	Emu	26.52	18.36	50.61	12.93	52.54	3.77
	RMax	26.53	19.03	52.88	12.96	52.62	4.07
	Error	**0.00**	**-0.54**	**-3.46**	**0.00**	**0.00**	**-2**
32	RMin	17.93	17.95	37.14	8.19	29.44	4
	Emu	17.99	18.08	35.87	8.13	29.44	4
	RMax	18.15	18.89	37.40	8.36	29.54	4
	Error	**0.00**	**0.00**	**-3.43**	**-0.82**	**0.00**	**5**



Generally <4%, max 5.2%

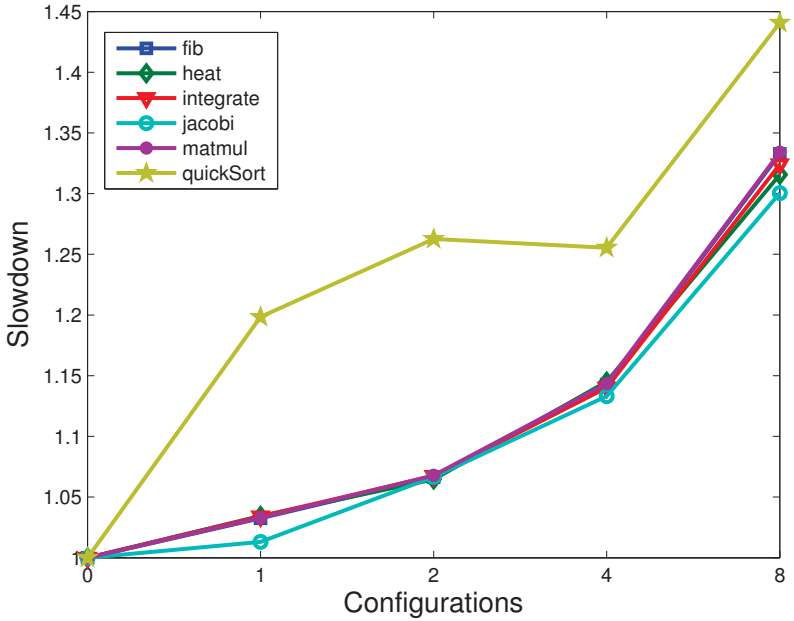
Two exceptions on MIC (6.97 and 10.16%)

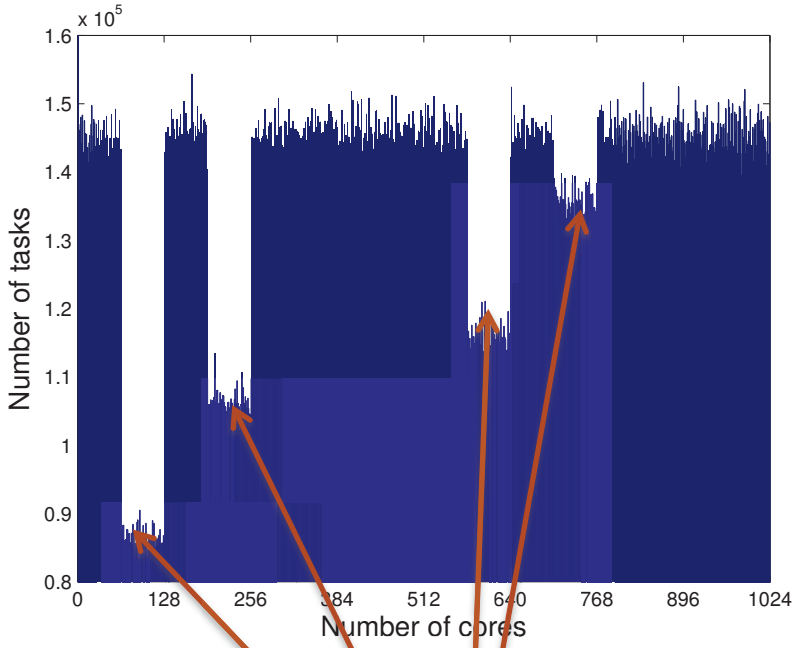
Performance



- Minimum slowdown up to 1,024 cores
- 6.4x slowdown at 512k cores
- Emulation completed in 11.5 hours
- Idle cores slow down emulation!

Case study 1: Power-constrained systems





- Emulate the behavior of heterogeneous, power-constrained exascale systems
  - 1,024 total cores, 16 voltage islands, 64 cores/island
  - Varying the number of voltage islands in low-power mode
  - Low-power mode cores run at ½ max frequency
- Automatic task balancing contains performance degradation

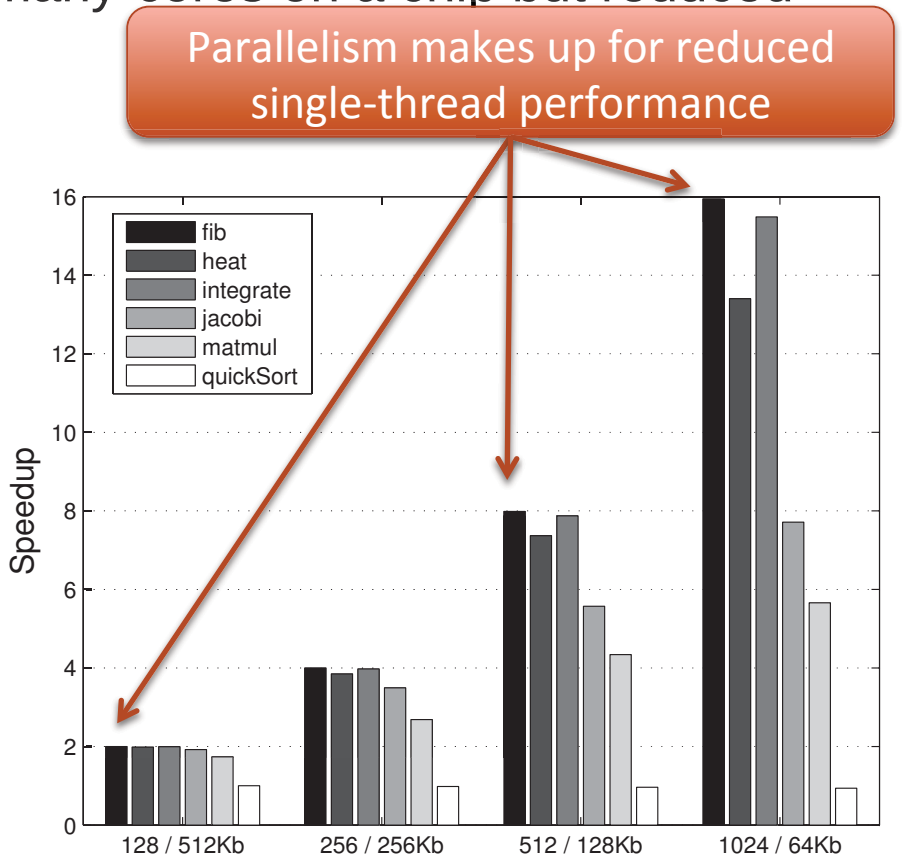
Low-power

Case study 2: Reduced per-core cache

Assume exascale architectures with many-cores on a chip but reduced per-core cache

Use Gem5 SE as hardware contention driver to model multi-core ARM processor

- 16 processor chips
- 4 MB last-level cache per processor chip
- 4 to 64 cores per processor chip



Note: we could not run Cilk++ apps on Gem5 SE

Conclusions

Pacific Northwest NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

Modeling exascale systems will require new scalable tools

Task-based programming models are non-deterministic

Prometheus: a fast, scalable, modular emulator for task-based applications

Prometheus scales up to 512k cores in 11.5h (6.4x slowdown)

Future work: add network, power, resilience…

Acknowledgements & more information

- Work sponsored by DOE ASCR under the Beyond the Standard Model (BSM) project

- Kestor G, R Gioiosa, and D Chavarría-Miranda. 2015. "Prometheus: Scalable and Accurate Emulation of Task-Based Applications on Many-Core Systems." In IEEE International Symposium on Performance Analysis of Systems and Software (**ISPASS 2015**).
- Akhmetova D, G Kestor, R Gioiosa, S Markidis, and E Laure. 2015. "On the application task granularity and the interplay with the scheduling overhead in many-core shared memory systems." To Appear in **IEEE Cluster 2015**