THE RELENTLESS EXECUTION MODEL

Lucas A. Wilson, Ph.D.

Dept. of Computer Science

The University of Texas at San Antonio

MOTIVATION AND PROBLEM STATEMENT

- Current parallel execution models assume guaranteed hardware availability
- Future systems (many-petaflop/exaflop) will likely be unable to provide the level of reliability

- **Existing execution and programming models are not well suited to future distributed memory parallel systems where hardware reliability cannot be guaranteed.**

- Develop an execution model where:
  - Tasks can be added to and remove from a running computation *(Elastic)*
  - Necessary data can be recomputed on demand if lost

THE RELENTLESS EXECUTION MODEL (REM)

- REM is a dataflow-inspired execution model which relies on two main features:
  - A set of uncoordinated parallel processes
  - A distributed, eventually-consistent key-value store (dictionary) where:
    - Values are the data operated on by the program
    - Keys are metadata which are used to inform the runtime task scheduler which operations to schedule next

REM TASK SCHEDULING

- Traditional dataflow task scheduling is *eager*, building a work queue of presently executable tasks which is amended as tasks are completed
- REM task scheduling is *lazy*, seeking only to compute the tasks necessary to complete the end goal (generate the key-value pairs which are part of the result set
  - ***Compute-by-need***

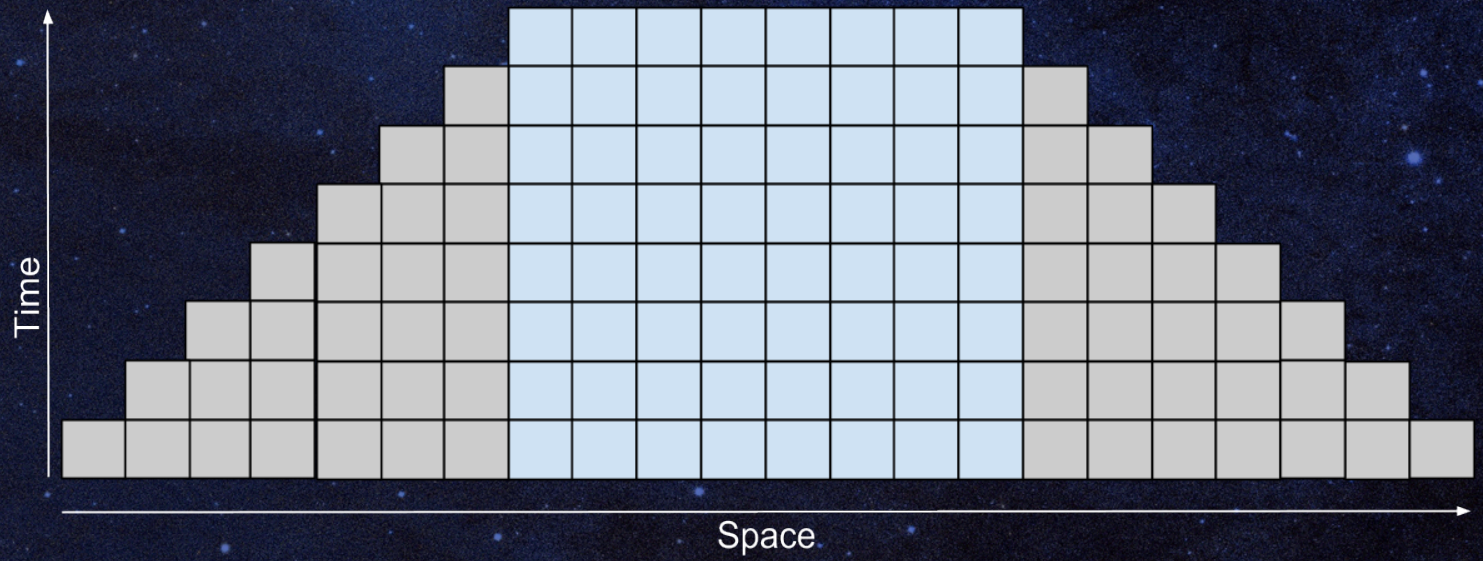
REM TASK SCHEDULING

- Independent tasks each perform their own ***compute-by-need*** scheduling, without directly interacting with other tasks
  - ***Task-uncoordinated***
- In parallel, these uncoordinated tasks work independently to compute the nodes of the dataflow graph, using the keys in the dictionary to inform them of what has already been completed
  - ***Cooperative Pruning***

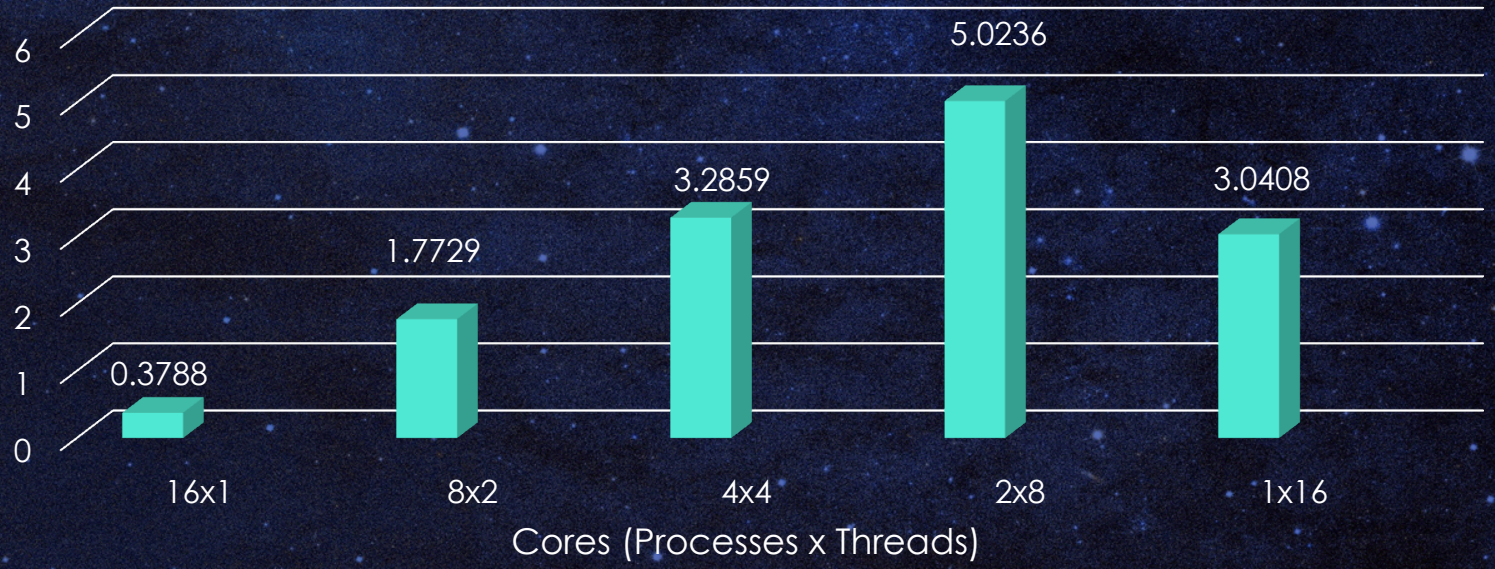
STENSAL

- StenSAL (Stencils in a Single Assignment Language) is a domain-specific language which allows for fast development of explicit stencil algorithms and targets REM
- The StenSAL compiler takes StenSAL code, expressing single element operations on a fixed grid, and converts it into a REM program which has large, coarser-grained ***coalesced*** tasks
  - Caching
  - Register Reuse
  - Vectorization (OpenMP 4.0)
  - Thread-based work sharing (OpenMP 4.0)

COALESCING – COMMUNICATION AVOIDANCE

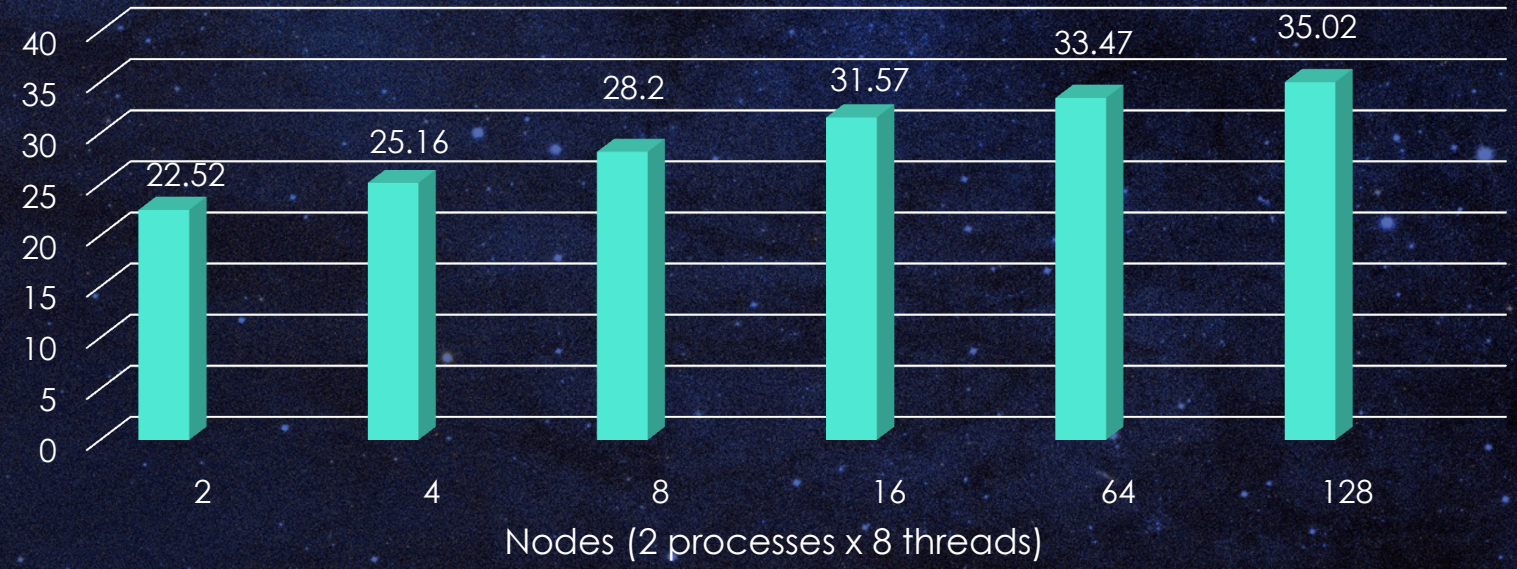


SINGLE NODE % OF THEORETICAL PEAK



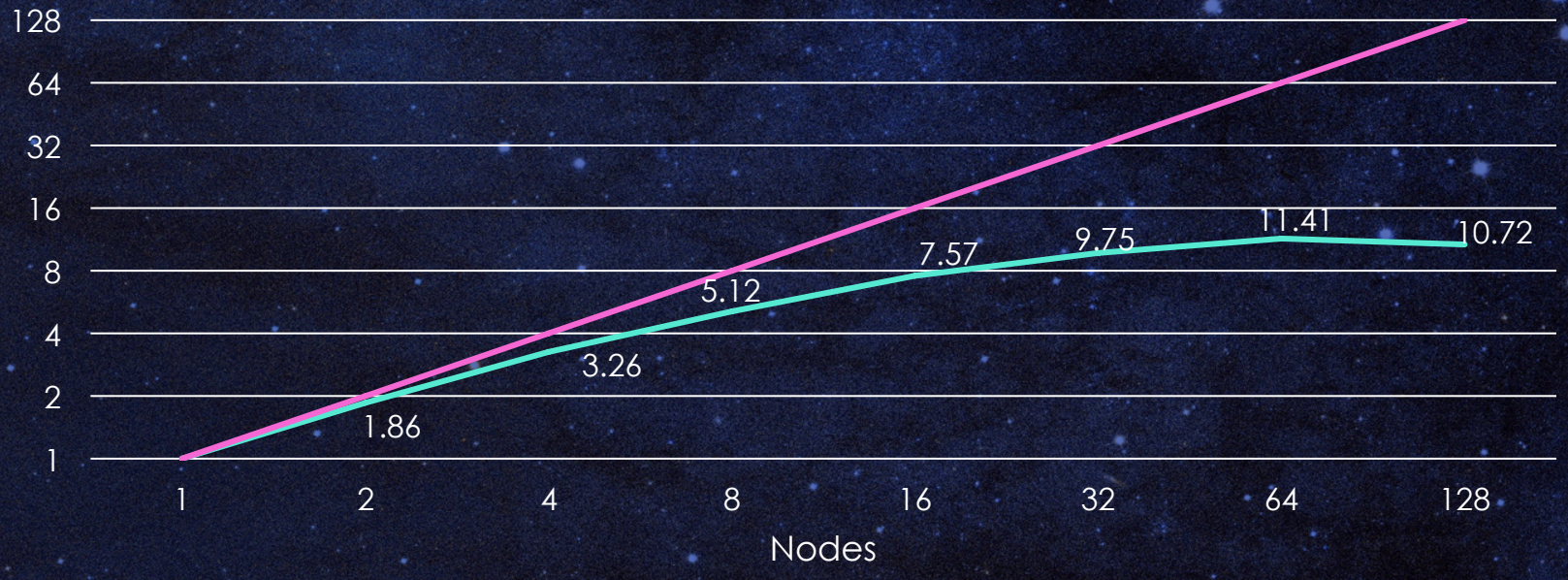
120k^2 Coalesced Elements
Stampede @TACC – Dual socket 8-core Haswell (E5-2680) @ 2.67 GHz

MULTIPLE NODE THROUGHPUT (GFLOPS)



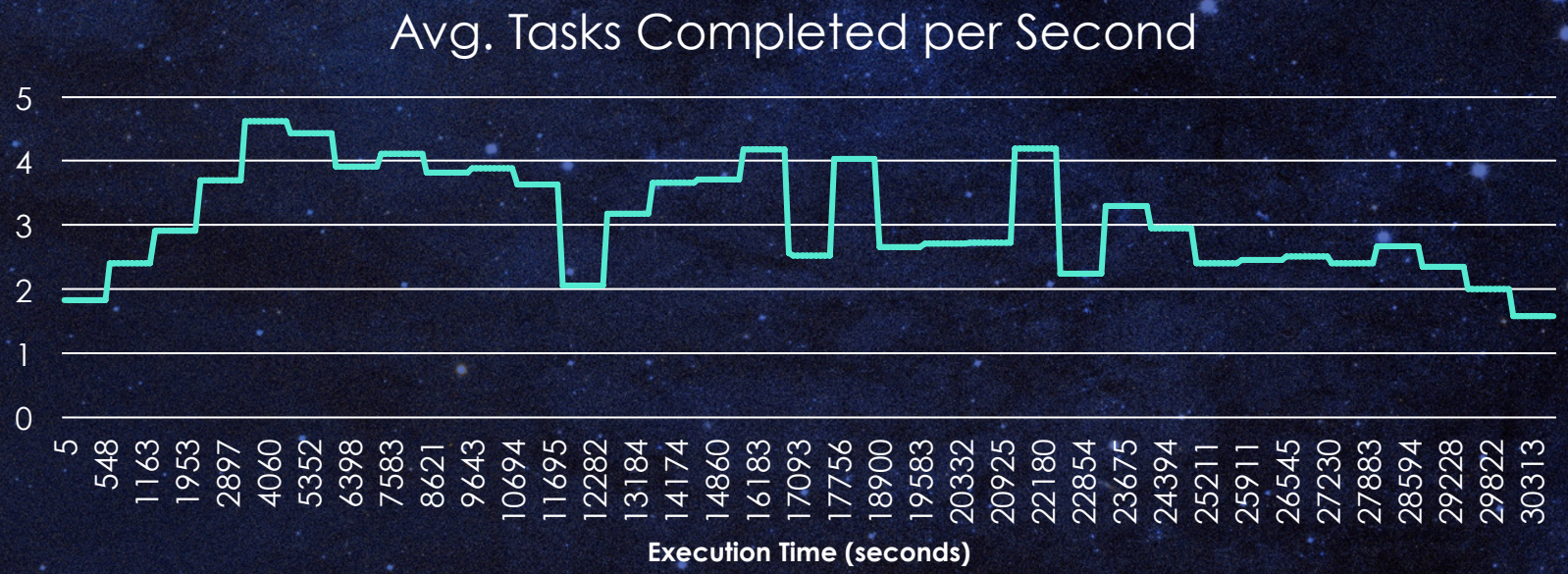
120k^2 Coalesced Elements
Stampede @TACC – Dual socket 8-core Haswell (E5-2680) @ 2.67 GHz

SCALED SPEEDUP

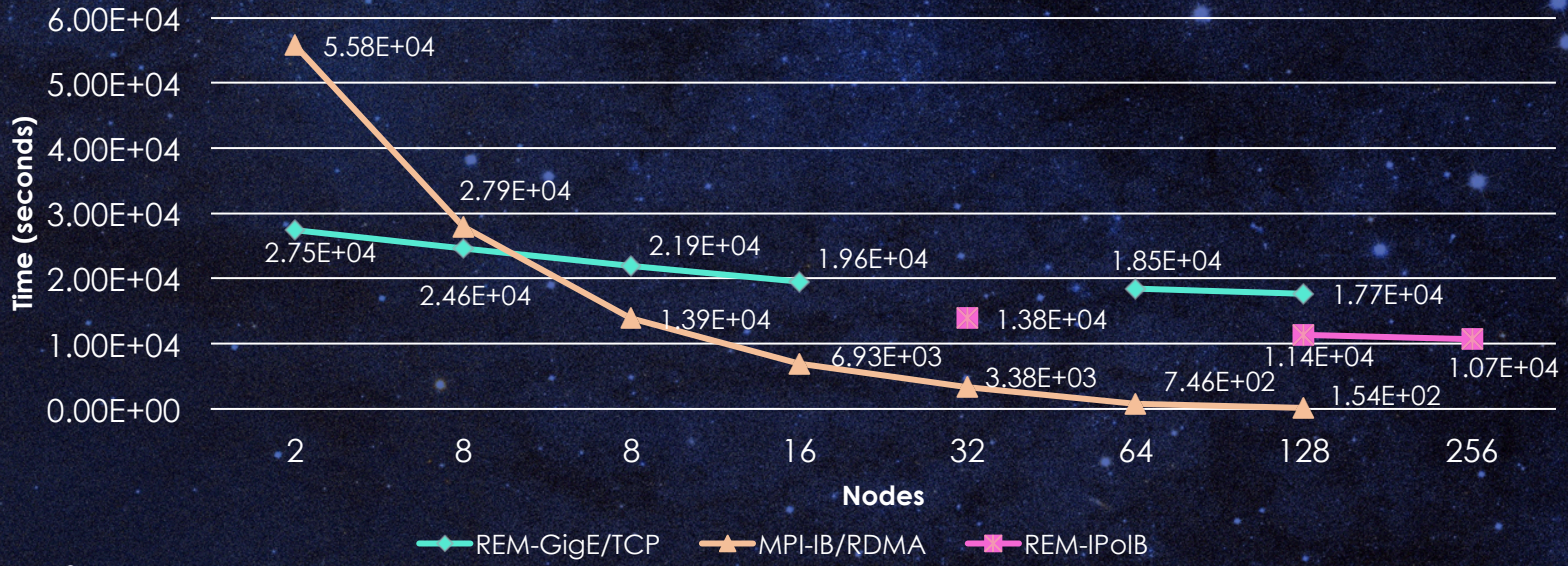


$16k^2$ Coalesced Elements – 16 processes (16x1) per node

ELASTICITY


Avg. Tasks Completed per Second
5
4
3
2
1
0
5
548
1163
1953
2897
4060
5352
6398
7583
8621
9643
10694
11695
12282
13184
14174
14860
16183
17093
17756
18900
19583
20332
20925
22180
22854
23675
24394
25211
25911
26545
27230
27883
28594
29228
29822
30313
Execution Time (seconds)

COMPARISON – REM & IMPI



120k^2 Coalesced Elements
2 processes per node (8 threads per process)
Stampede @TACC – Dual socket 8-core Haswell (E5-2680) @ 2.67 GHz

CONTRIBUTIONS

- Wilson and von Ronne, "A Task-uncoordinated Distributed Dataflow Model for Scalable High Performance Parallel Program Execution, " *Parallel Computing: Systems and Applications,* Elsevier, DOI: 10.1016/j.parco.2015.10.013
- Wilson and von Ronne, "StenSAL: A Single Assignment Language for Relentlessly Executing Stencil Algorithms," in proceedings of the Workshop on Optimizing Stencil Computations (WOSC), October 2014.
- Wilson and von Ronne, "A Distributed Dataflow Model for Task-uncoordinated Parallel Program Execution," in proceedings of the Seventh International Workshop on Parallel Programming Models and Systems Software for High End Computing (P2S2), September 2014.
- Wilson and Lockman, "The Relentless Computing Paradigm: A data-oriented programming model for distributed memory computation", IEEE International Conference on High Performance Computing, Networking, Storage, and Analysis (SC11), November 2011, Seattle, WA.
- Wilson and Lockman, "Relentless Computing: Enabling Fault-tolerant, Numerically Intensive Computation in Distributed Environments," in proceedings of The 2011 International Conference on Parallel and Distributed Processing Tools and Applications (PDPTA'11), July 2011.

FUTURE WORK

- Intelligent task scheduling
  - Better mapping of tasks to data storage location
  - Better performance for conditional loops and branches
- Distribute key-value pairs in some logical fashion
  - Spatial locality
- RDMA aware dictionary storage

WRAPPING IT UP

- Our work has shown that it is possible to efficiently execute parallel algorithms in a task-uncoordinated fashion
- REM allows elastic addition/removal of tasks from executing program
- StenSAL provides a simple means of expressing explicit stencil algorithms in a form which can be optimized into large coalesced blocks to be executed by the REM runtime
  - Vectorization and worksharing with OpenMP 4.0
- Closing the performance gap with MPI

THANK YOU

Come by the Doctoral Showcase Poster Session today @ 4PM in Ballroom E