Charm++ Workshop 2010

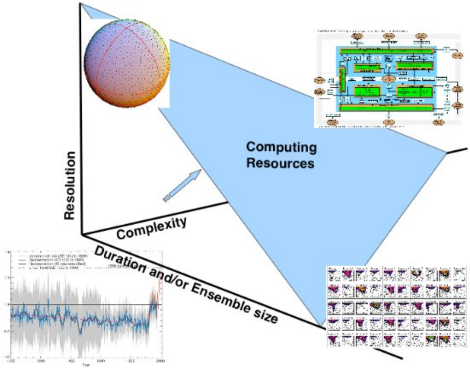
Processor Virtualization in Weather Models

Eduardo R. Rodrigues

Institute of Informatics
Federal University of Rio Grande do Sul - Brazil
(**visiting scholar at CS-UIUC**)
errodrigues@inf.ufrgs.br

Supported by Brazilian Ministry of Education - Capes, grant 1080-09-1

Limit of computing resources affecting Weather Model execution



James L. Kinter III and Michael Wehner,

Computing Issues for WCRP Weather and Climate Modeling, 2005.

Load imbalance

"Because atmospheric processes occur nonuniformly within the computational domain, e.g., active thunderstorms may occur within only a few sub-domains of the decomposed domain, the load imbalance across processors can be significant."

Xue, M.; Droegemeier, K.K.; Weber, D. *Numerical Prediction of High-Impact Local Weather: A Driver for Petascale Computing*. In: **Petascale Computing: Algorithms and Applications**. 2007.

animation

The Promise of Load Balancing the Parameterization of Moist Convection Using a Model Data Load Index

S. P. Muszala and D. A. Connors

Electrical and Computer Engineering, University of Colorado, Boulder, Colorado

J. J. Hack

Load-Balancing Algorithms for Climate Models*

Ian T. Foster and Brian R. Toonen

Mathematics and Computer Science Division

Argonne National Laboratory

[redacted]

LOAD BALANCING AND SCALABILITY OF A SUBGRID OROGRAPHY SCHEME IN A GLOBAL CLIMATE MODEL

Steven Ghan

1 Introduction

A subgrid orography scheme (Ghan et al., 2002) has been applied to the National Center for Atmospheric Research (NCAR) Community Atmosphere Model (CAM3) and Common Land Model (CLM3). CAM3 (Collins et al., 2004) is a global atmospheric circulation code designed to run on a variety of computational platforms, including single processor workstations, shared memory machines, distributed memory systems, symmetric multiple processor (SMP) systems, and most recently on distributed vector

"Most implementations of atmospheric prediction models do not perform dynamic load balancing, however, because of the complexity of the associated algorithms and because of the communication overhead associated with moving large blocks of data across processors."

Xue, M.; Droegemeier, K.K.; Weber, D. *Numerical Prediction of High-Impact Local Weather: A Driver for Petascale Computing*. In: **Petascale Computing: Algorithms and Applications**. 2007.

Adaptive MPI

- Since parallel weather models are typically implemented in MPI, can we use AMPI to reduce complexity of the associated algorithms?
- Can we deal with the communication overhead of this environment?

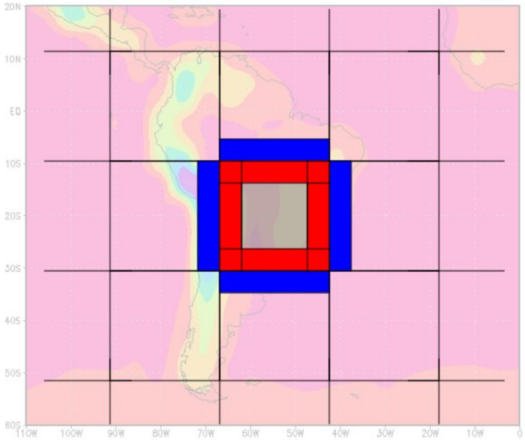
BRAMS

Brazilian developments on the Regional Atmospheric Modeling System

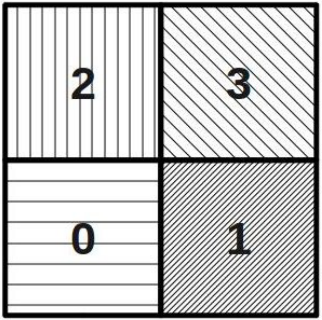
- It is a multipurpose regional numerical prediction model designed to simulate atmospheric circulations at many scales;
- It is used both for production and research world wide;
- It has its roots on RAMS, that solves the fully compressible non-hydrostatic equations;
- It is equipped with a multiple grid nesting scheme which allows the model equations to be solved simultaneously on any number of two-way interacting computational meshes of increasing spatial resolution;
- It has a set of state-of-the-art physical parameterizations appropriate to simulate important physical processes such as surface-air exchanges, turbulence, convection, radiation and cloud microphysics.

BRAMS

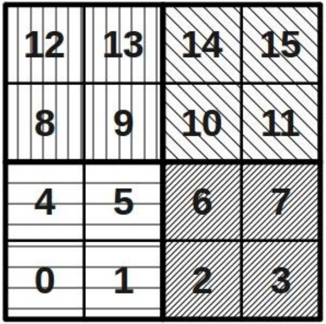
Domain decomposition



Virtualization with AMPI



4 processors



4 procs. - 16 virtual procs.

Benefits of Virtualization

- Adaptive overlapping of communication and computation;
- Automatic load balancing;
- Flexibility to run on arbitrary number of processors;
- Optimized communication library support;
- Better cache performance.

Chao Huang and Gengbin Zheng and Sameer Kumar and Laxmikant V. Kale, *Performance Evaluation of Adaptive MPI*, Proceedings of **ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming** 2006.

Benefits of Virtualization

- Adaptive overlapping of communication and computation;
- Automatic load balancing;
- Flexibility to run on arbitrary number of processors;
- Optimized communication library support;
- Better cache performance.

Chao Huang and Gengbin Zheng and Sameer Kumar and Laxmikant V. Kale, *Performance Evaluation of Adaptive MPI*, Proceedings of **ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming** 2006.

Benefits of Virtualization

- Adaptive overlapping of communication and computation;
- Automatic load balancing;
- Flexibility to run on arbitrary number of processors;
- Optimized communication library support;
- Better cache performance.

Chao Huang and Gengbin Zheng and Sameer Kumar and Laxmikant V. Kale, *Performance Evaluation of Adaptive MPI*, Proceedings of **ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming** 2006.

Global Variable Privatization

- Manual Change
- Automatic Globals Swapping (swapglobals)

Global Variable Privatization

- Manual Change

	global	static	commons
BRAMS	10205	519	32
WRF3	8661	550	70

- Automatic Globals Swapping (swapglobals)

Global Variable Privatization

- Manual Change

	global	static	commons
BRAMS	10205	519	32
WRF3	8661	550	70

- Automatic Globals Swapping (swapglobals)
  - It does not support static variables

Global Variable Privatization

- Manual Change

	global	static	commons
BRAMS	10205	519	32
WRF3	8661	550	70

- Automatic Globals Swapping (swapglobals)
  - It does not support static variables
  - We can transform static in globals and keep the same semantic

BRAMS: Performance with only virtualization

	initialization	parallel	total
4p - No Virtualization	3.94s	164.86s	168.80s
4p - 64vp	8.25s	223.15s	231.40s

On ABE - x86 cluster

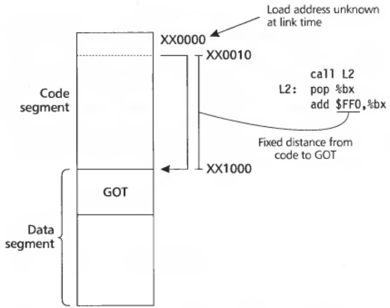
BRAMS: Performance with only virtualization

	initialization	parallel	total
4p - No Virtualization	3.94s	164.86s	168.80s
4p - 64vp	8.25s	223.15s	231.40s

On ABE - x86 cluster

Automatic Globals Swapping

- the code is compiled as a shared library (with PIC - Position Independent Code)



Levine, J.R. **Linker & Loaders**. 2000.

Global variables

```
extern int a;
a = 42;
movl a@GOT(%ebx), %eax
movl $42,(%eax)
```

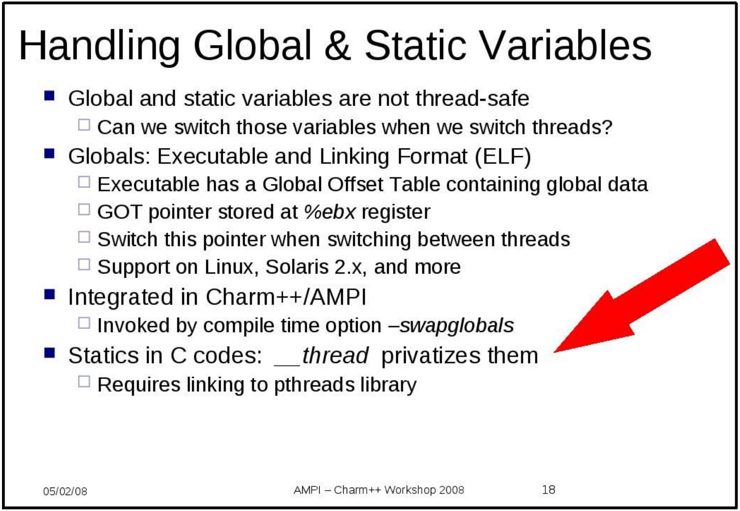
In a context switch, to change every entry in the GOT.
A drawback is that the GOT might be big.

Thread Local Storage (TLS)

Thread local storage is used by kernel threads to privatize data.

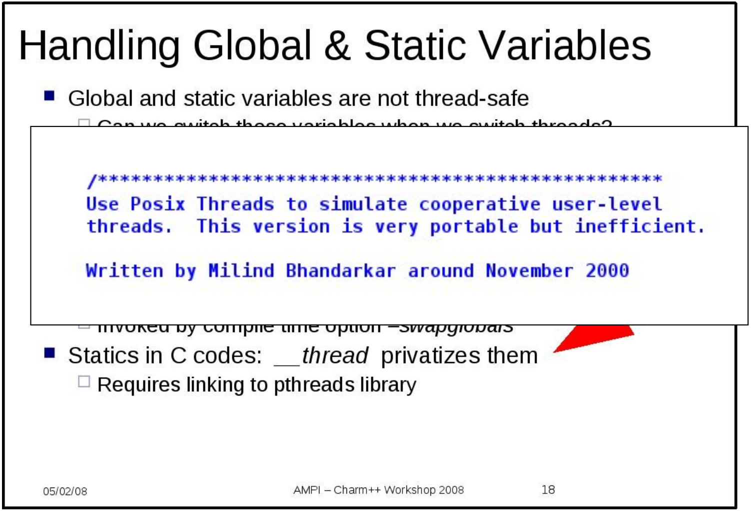
Thread Local Storage (TLS)

Thread local storage is used by kernel threads to privatize data.



Thread Local Storage (TLS)

Thread local storage is used by kernel threads to privatize data.

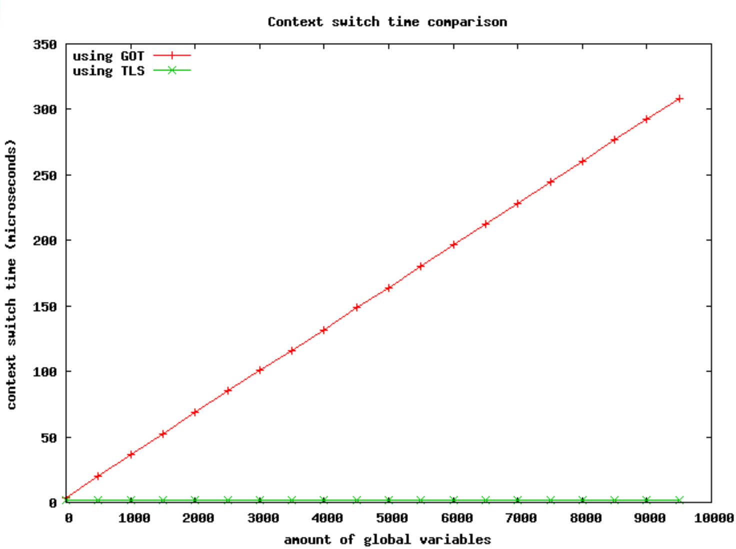


Our approach

1. **Use TLS to privatize data in user-level threads;**
2. **Employ this mechanism in AMPI (including thread migration);**
3. **Change the gfortran compiler to produce TLS code for every global and static data.**

RODRIGUES, E. R.; NAVAUX, P. O. A.; PANETTA, J.; MENDES, C. L. *A New Technique for Data Privatization in User-level Threads and its Use in Parallel Applications*. In: **ACM 25th Symposium On Applied Computing** , 2010.

Comparison between Swapglobals and TLS



BRAMS: Performance virtualization with TLS

	initialization	parallel	total
4p - No Virtualization	3.94s	164.86s	168.80s
4p - 64vp (swapglobals)	8.25s	223.15s	231.40s
TLS 4p - 64vp	7.94s	141.16s	149.10s

On ABE - x86 cluster

BRAMS: Performance virtualization with TLS

	initialization	parallel	total
4p - No Virtualization	3.94s	164.86s	168.80s
4p - 64vp (swapglobals)	8.25s	223.15s	231.40s
TLS 4p - 64vp	7.94s	141.16s	149.10s

On ABE - x86 cluster

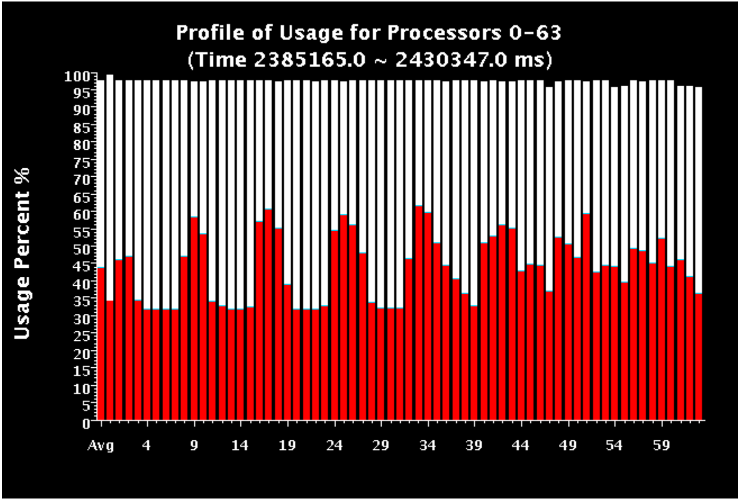
Benefits of Virtualization

- Evaluate the reasons for the improvement
- Run a bigger case: Brams on 64 processors and up to 1024 virtual processors (threads)
- We performed these experiments on Kraken - Cray XT5 at Oak Ridge

Benefits of Virtualization

Adaptive overlapping of communication and computation

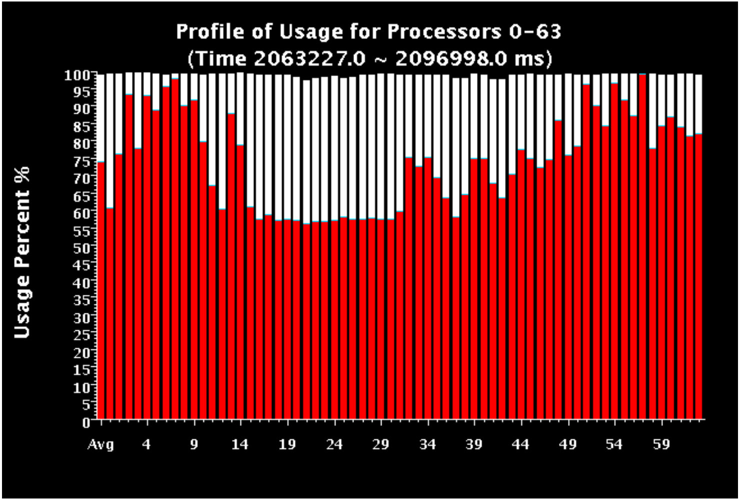
64 processors - No virtualization – Average usage 43.78%



Benefits of Virtualization

Adaptive overlapping of communication and computation

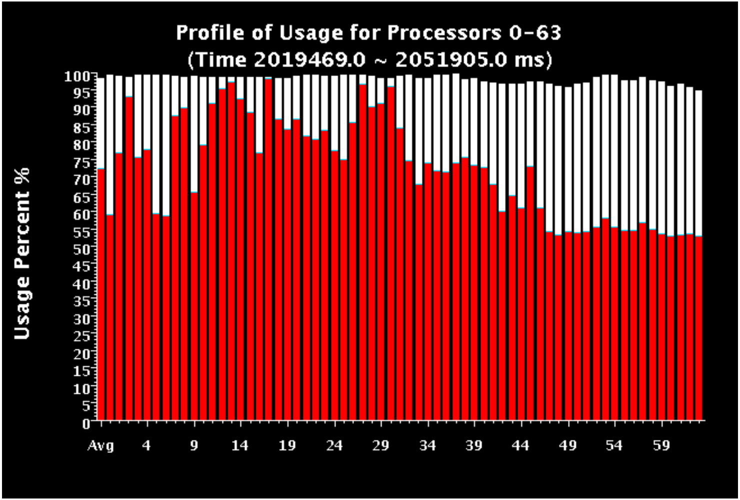
64 processors - 256 virtual processors – Average usage 73.52%



Benefits of Virtualization

Adaptive overlapping of communication and computation

64 processors - 1024 virtual processors – Average usage 73.02%



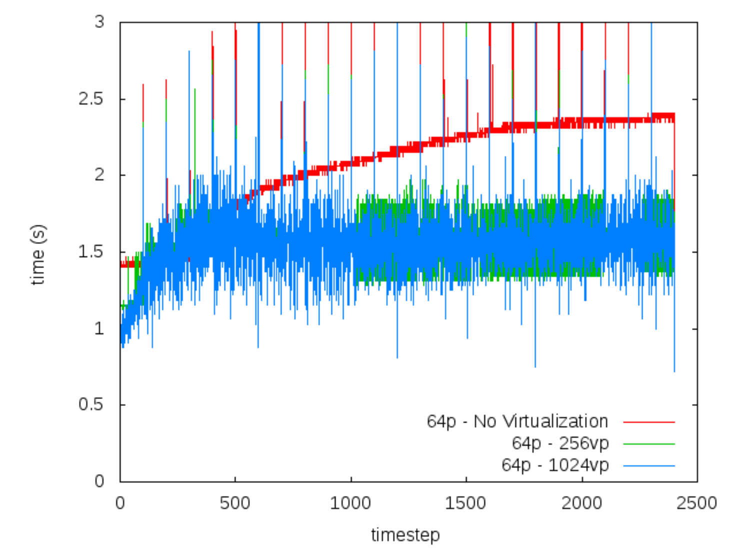
Benefits of Virtualization

Better cache performance

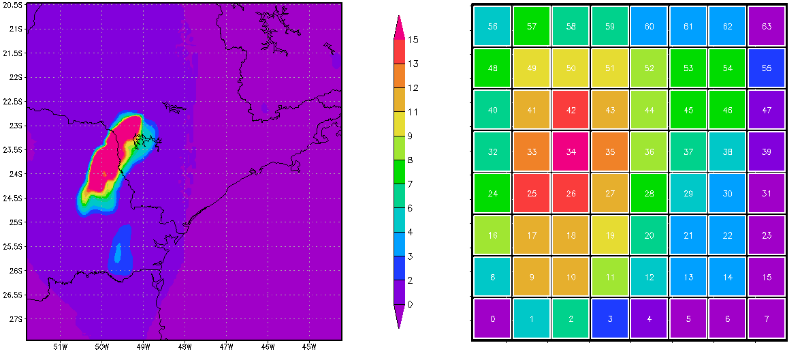
	L2 cache misses	L3 cache misses
64p - No Virtualization	194M	132M
64p - 256vp	165M	70M
64p - 1024vp	147M	61M

average per processor, 20 timesteps

Benefits of Virtualization



Brams Load Imbalance



Load Balancing

Since the application has a fixed communication pattern and the cost of migrating threads may be high (due to the large memory footprint), we decided to test the existing load balancer RefineCommLB.

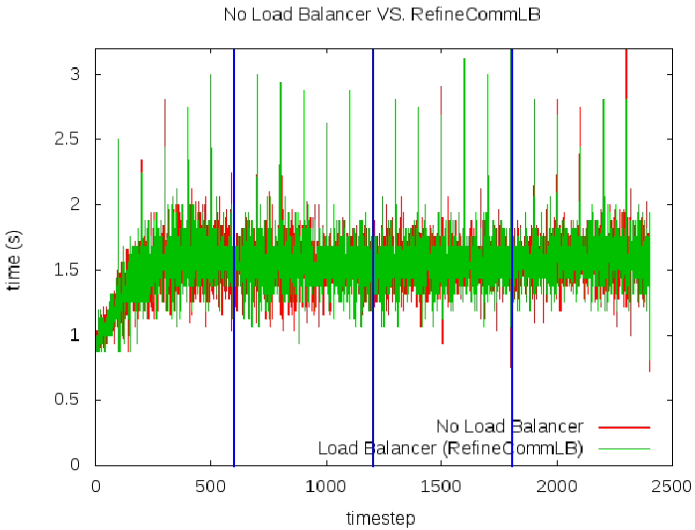
Load Balancing

Since the application has a fixed communication pattern and the cost of migrating threads may be high (due to the large memory footprint), we decided to test the existing load balancer RefineCommLB.

RefineCommLB is a Charm++ load balancer that improves the load balance by incrementally adjusting the existing thread distribution. It also takes into account the communication among threads.

BRAMS: Load Balancing

every 600 timesteps



New Load Balancer

- Keep neighbor threads close to each other;
- Assign contiguous threads in 2D space to the same processor;
- Possibly use application information to adjust rebalance.

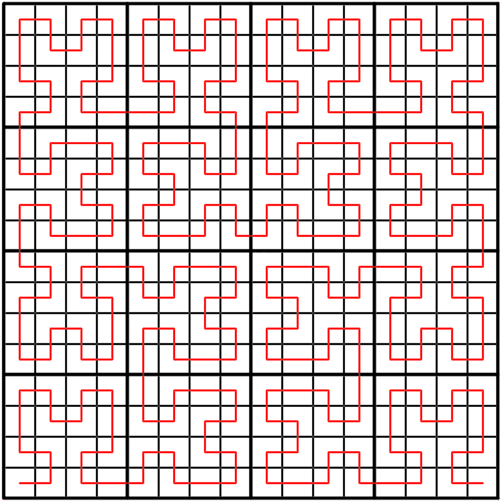
New Load Balancer

- Keep neighbor threads close to each other;
- Assign contiguous threads in 2D space to the same processor;
- Possibly use application information to adjust rebalance.

Implementing a Load balancer on Charm is straightforward (see G.Zheng's presentation at the 4th Charm++ Workshop)

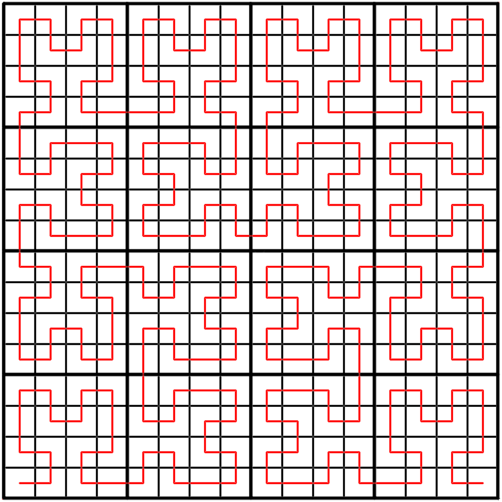
Hilbert Curve

maps a multidimensional space to a 1-D space



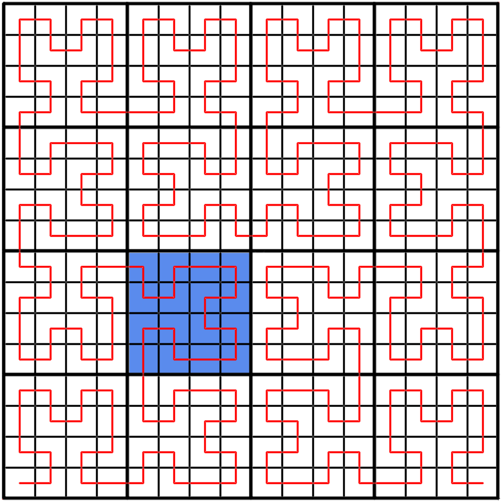
Hilbert Curve

Neighbor points on the curve are also close in the N-D space



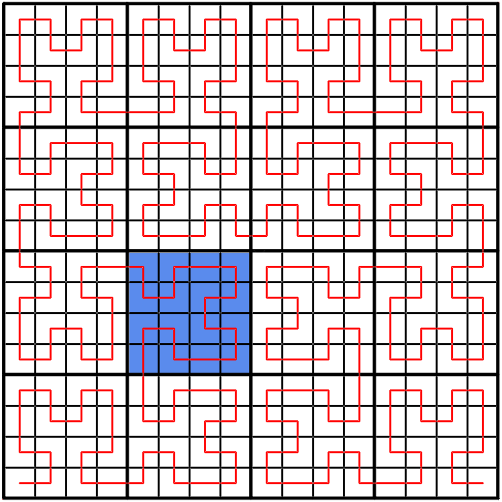
Hilbert Curve

In this figure, there are 256 threads and 16 processors



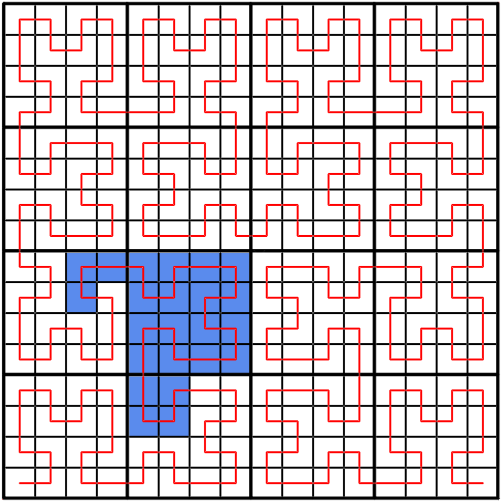
Hilbert Curve

We cut it so that each segment has approximately the same load



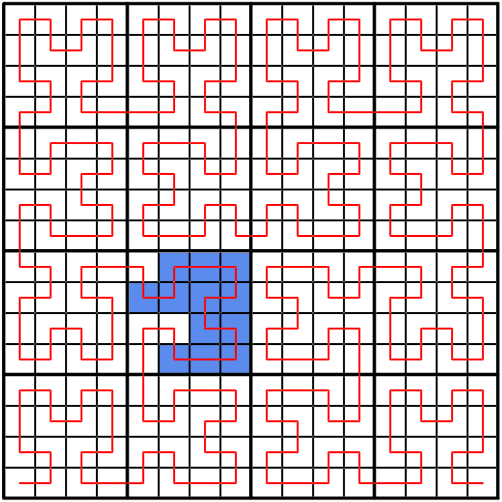
Hilbert Curve

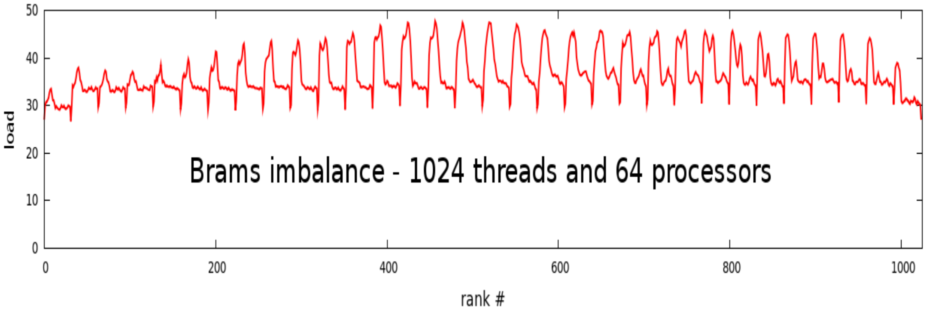
We may expand...

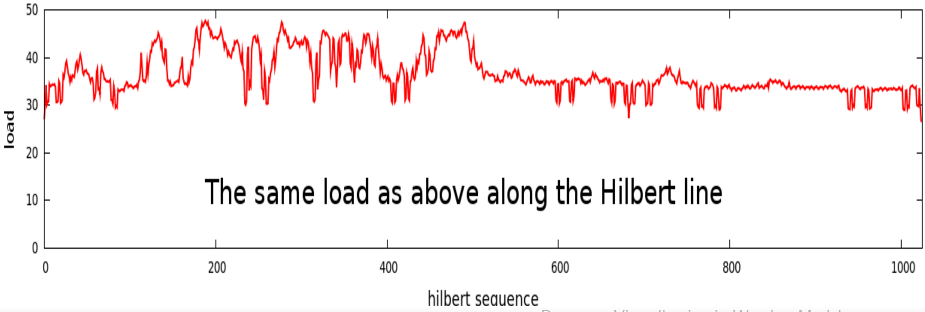


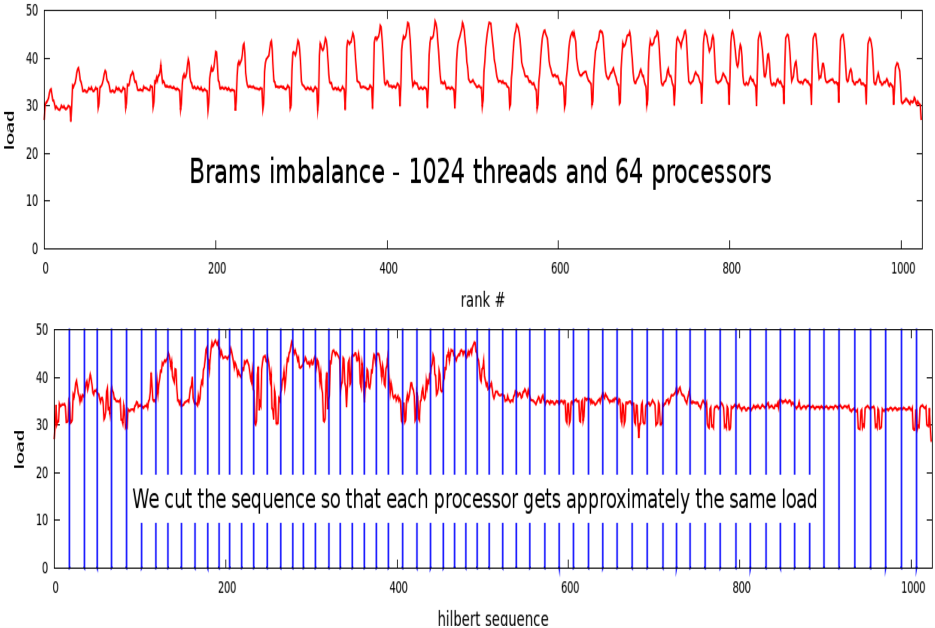
Hilbert Curve

or shrink each segment according to the measured loads



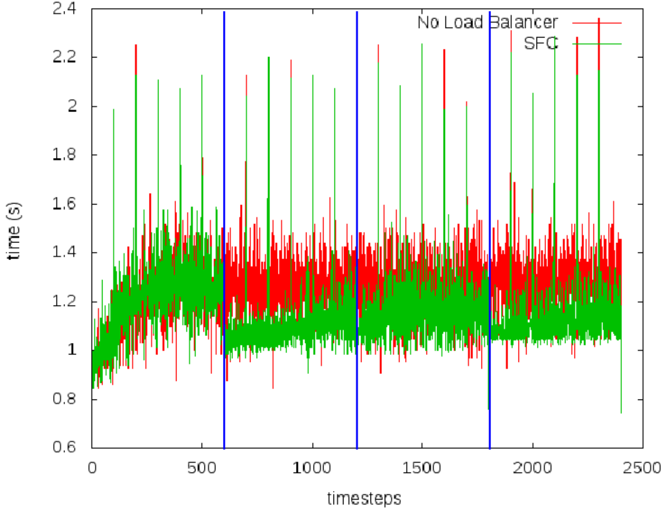

Brams imbalance - 1024 threads and 64 processors
load
rank #
0
10
20
30
40
50
0
200
400
600
800
1000


The same load as above along the Hilbert line
load
hilbert sequence
0
10
20
30
40
50
0
200
400
600
800
1000


Brams imbalance - 1024 threads and 64 processors
load
rank #
0
200
400
600
800
1000
0
10
20
30
40
50
We cut the sequence so that each processor gets approximately the same load
hilbert sequence

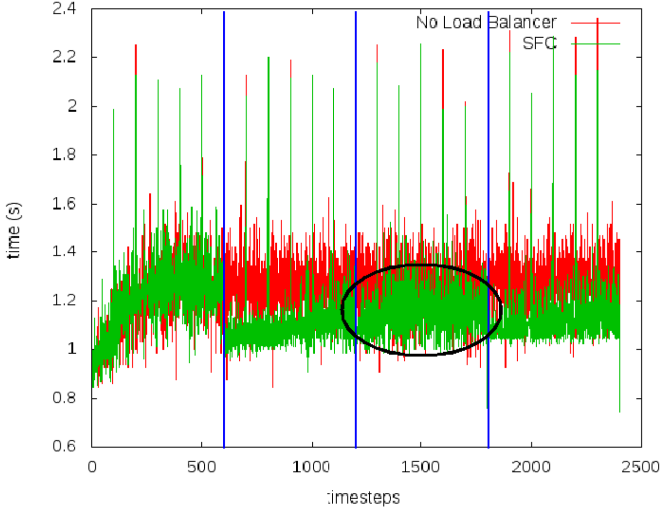
Results

New Load Balancer called every 600 timesteps



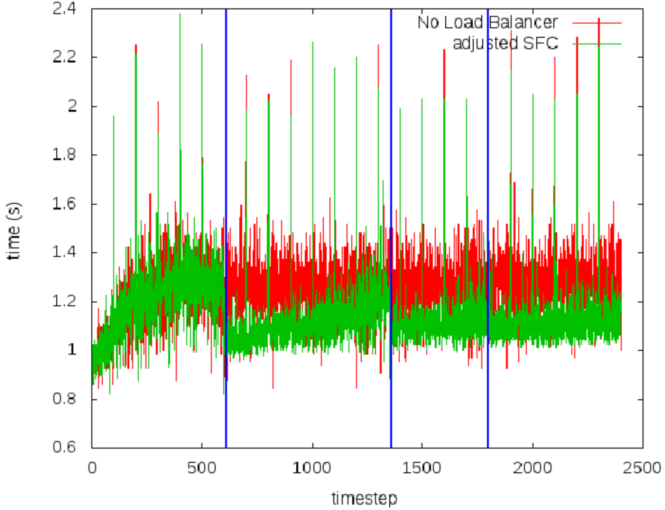
Results

New Load Balancer called every 600 timesteps



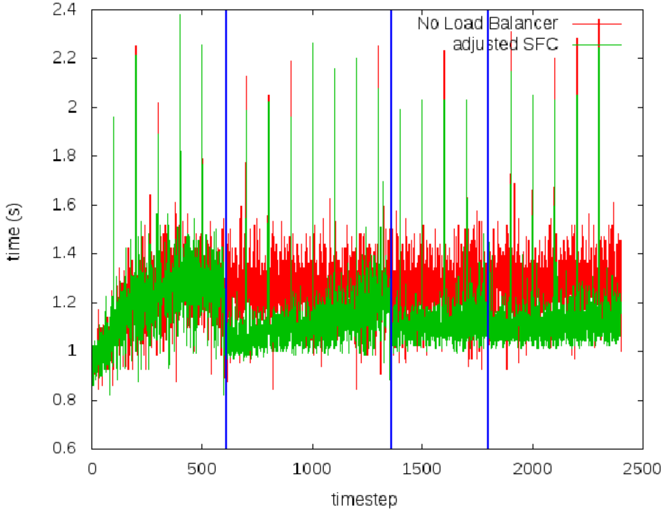
Results

delaying the second Load Balancer call 150 timesteps



Lesson

We may need an adaptive scheme to call the Load Balancer



Conclusions

- Weather models may suffer from load imbalance even with a regular domain decomposition due to nonuniform atmospheric processes;
- Virtualization itself improved performance;
- Execution time of the rebalanced run was reduced up to 10% in comparison to the purely virtualized execution;

Ongoing work

- Investigate adaptive schemes to call the Load Balancer;
- Possibly use application information to enhance the balancing schemes based solely on observed load;
- Evaluate other Load Balancers.

Questions

?