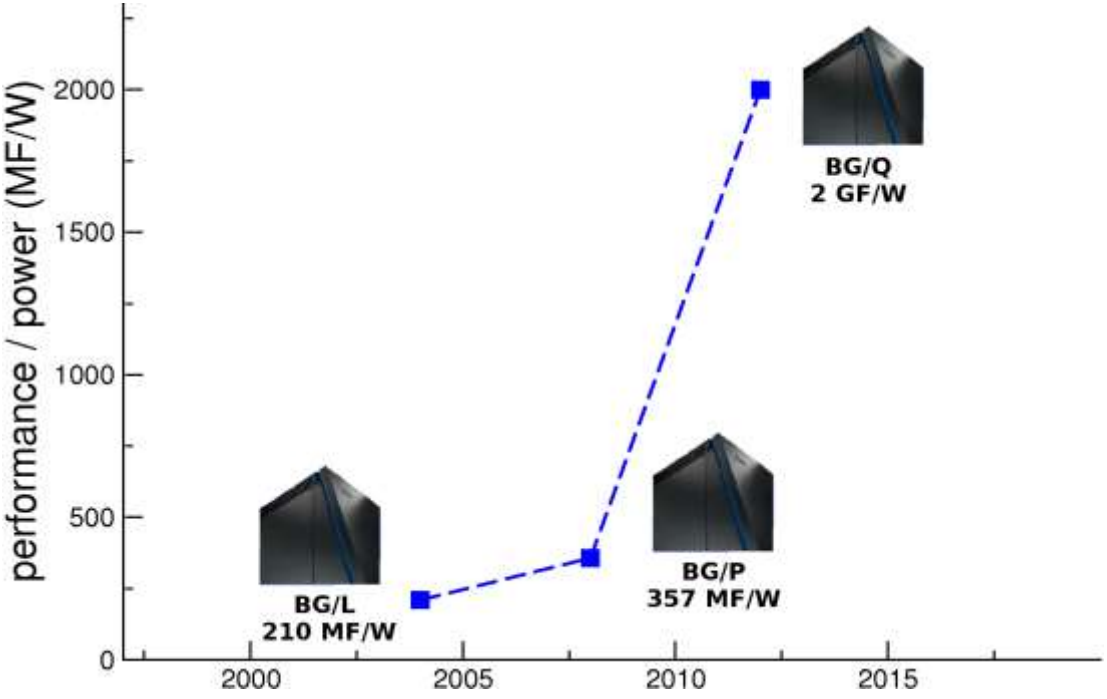
Running MD on Tier-0 HPC architectures

Giovanni Chillemi g.chillemi@cineca.it
HPC department, CINECA

- BG is a massively parallel supercomputer
- It holds different types of nodes (and networks)
- It is designed to have high energy-efficiency (performance/power)



BLUE GENE EVOLUTION

	Total		Biggest Config	Per rack		
	Performance [PF]	Efficiency [MF/W]	Max # of racks	Performance [TF]	Efficiency	# of cores
BG/L	0.596	210	104	5.7	2.02	2048
BG/P	1	357	72	13.9	4.96	4096
BG/Q	20	2000	96	209	20.83	16384

Towards higher and higher:

- Performance
- Efficiency
- Density of cores per rack

PARTNERSHIP FOR ADVANCED COMPUTING IN EUROPE

FERMI @ CINECA
PRACE Tier-0 System

Architecture: 10 BG/Q Racks
Model: IBM-BG/Q
Processor Type: IBM PowerA2, 1.6 GHz
Computing Cores: 163840
Computing Nodes: 10240
RAM: 1 GByte / core
Internal Network: 5D Torus
Disk Space: 2 PByte of scratch space
Peak Performance: 2 PFlop/s
Power Consumption: 1 MWatt

Content of a BGQ Rack

- 2 **midplanes** form a rack
- Each midplane has 16 **Node Cards**
- Each Node Card has 32 **Compute Nodes**
- Each compute node has 16 **cores**
  (16384 cores per rack)

- Flexible I/O nodes – node cards ratio
  (at least 512 cores per I/O node)
  - In our configuration: 2 rack con 1024 core per I/O node
    8 rack con 2048 core per I/O node

Note that..

The number of I/O nodes per rack constraints:

- I/O bandwidth to/from compute racks
  (each I/O node has 2 links (4GB/s in 4GB/s out))

- The minimum partition allocatable on a BG/Q system ("small block" jobs)

For FERMI:
  bg_size=64 (jobs running on R11 and R31) → 1024 cores
  bg_size=128 (jobs running on the other racks) → 2048 cores

BGQ Networks

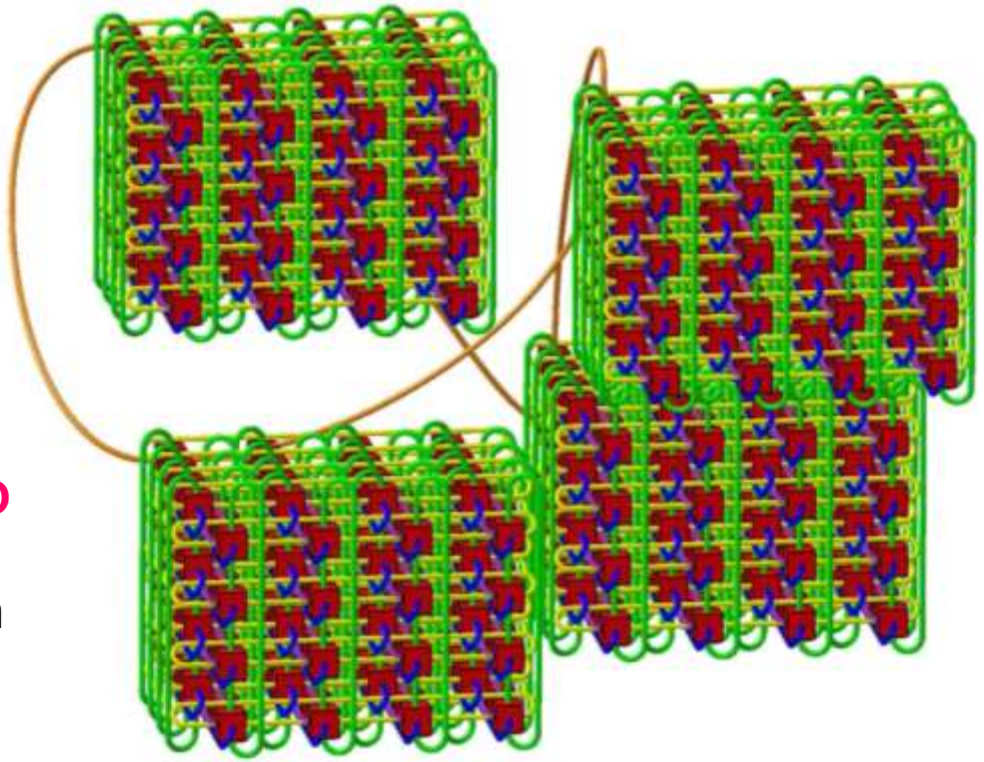
5D Torus Network

5D topology for point-to-point communication

- 2 GB/s bidirectional bandwidth on all (10+1) links, 5D nearest neighbour exchange at 1.75 GB/s per link
- Collective and barrier networks embedded in 5-D torus network.

External, independent and dynamic I/O system

- I/O nodes in separate drawers/rack with private interconnections
- I/O network to/from Compute rack: 2 links (4 GB/s in 4 GB/s out)



Job script: general structure

```
#!/bin/bash
# @ job_name  = bgsize.$(jobid)
# @ output  = z.$(jobid).out
# @ error  = z.$(jobid).err
# @ shell = /bin/bash
# @ job_type  = bluegene
# @ wall_clock_limit  = 02:00:00
# @ notification = never
# @ bg_size  = 256
# @ class = keyproject
# @ account_no  = cinstaff
# @ restart = no
# @ queue


cd /gpfs/scratch/userinternal/cin0753a/mydir

runjob –ranks-per-node  64 ./program.exe
```

LL keywords

Application block

runjob

Applications always need to be executed with **runjob** command to run on the compute nodes

- Syntax:
  - runjob [options]
  - runjob [options] binary [arg1 arg2 ... argn]
- Parameters can be set
  - by command-line options (higher priority!)
  - environment variables
- runjob –h for a complete list

Fermi Production environment

```
$>module av

------------ /cineca/prod/modulefiles/base/applications --------
abinit/6.12.3        crystal09/1.01        qe/5.0bgq
amber/12             dl_poly/4.03           siesta/3.1
bigdft/1.6.0         gromacs/4.5.5         vasp/5.2.12
cp2k/2.3             lammps/20120816       vasp/5.3.2
namd/2.9             cpmd/v3.15.3
```

To load a specific module, i.e. set specific env vars
$> module load <module_name>

Show the variables set by a module
$> module show <module_name>

Retrieve informations of a module
$> module help <module_name>

Available MD packages on Fermi

Some of the currently installed MD codes on Fermi and available for use:

```
$> module av

------------ /cineca/prod/modulefiles/base/applications ------------

abinit/6.12.3                crystal09/2.0.1(default)   octopus/4.1(default)
amber/12(default)            dl_poly/4.03               openfoam/2.1.1
bigdft/1.6.0                 dl_poly/4.05(default)      qe/5.0.3b(default)
cp2k/2.3                     gromacs/4.5.5(default)     qe/5.0bgq
cp2k/2.4(default)            gromacs/4.6.1              siesta/3.1
cpmd/3.15.3_rev2606          lammps/20120816            siesta/3.1-TS
cpmd/3.17.1(default)         namd/2.9(default)          vasp/5.2.12
namd/2.10                    enzo/2.2                   gromacs/4.5.5
crystal09/1.01               nwchem/6.3                 vasp/5.3.3(default)
```

NAMD template script for Fermi

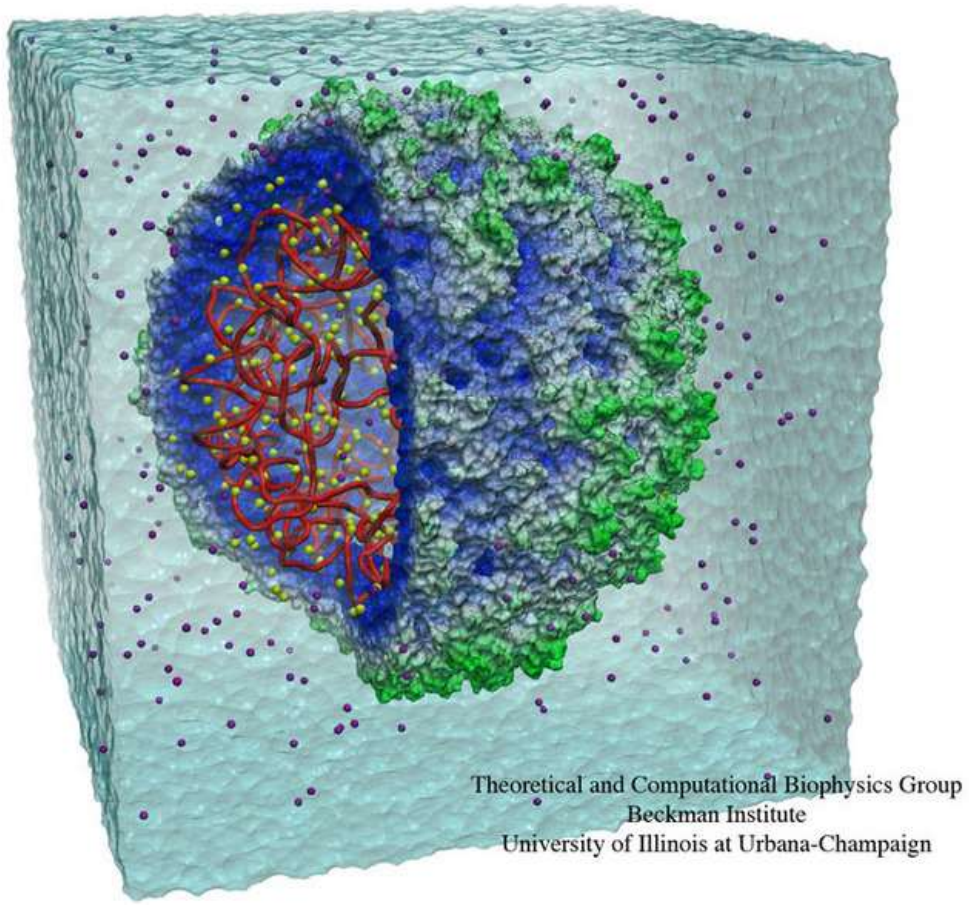
```
#!/bin/bash
# @ job_name  = namd.$(jobid)
# @ output  = $(job_name).out
# @ error = $(job_name).err
# @ shell = /bin/bash
# @ job_type  = bluegene
# @ wall_clock_limit  = 01:00:00
# @ bg_size  = 64
# @ account_no  = your_account_name
# @ queue

module  load namd/2.9

#launch with 256 processes (grouping 4 processes per node) using multi-thread (16 threads per process)
runjob  --rank-per-node  4 : $NAMD_HOME/namd2  +ppn16  input.namd  > output.log
```

64*4 = 256 MPI tasks

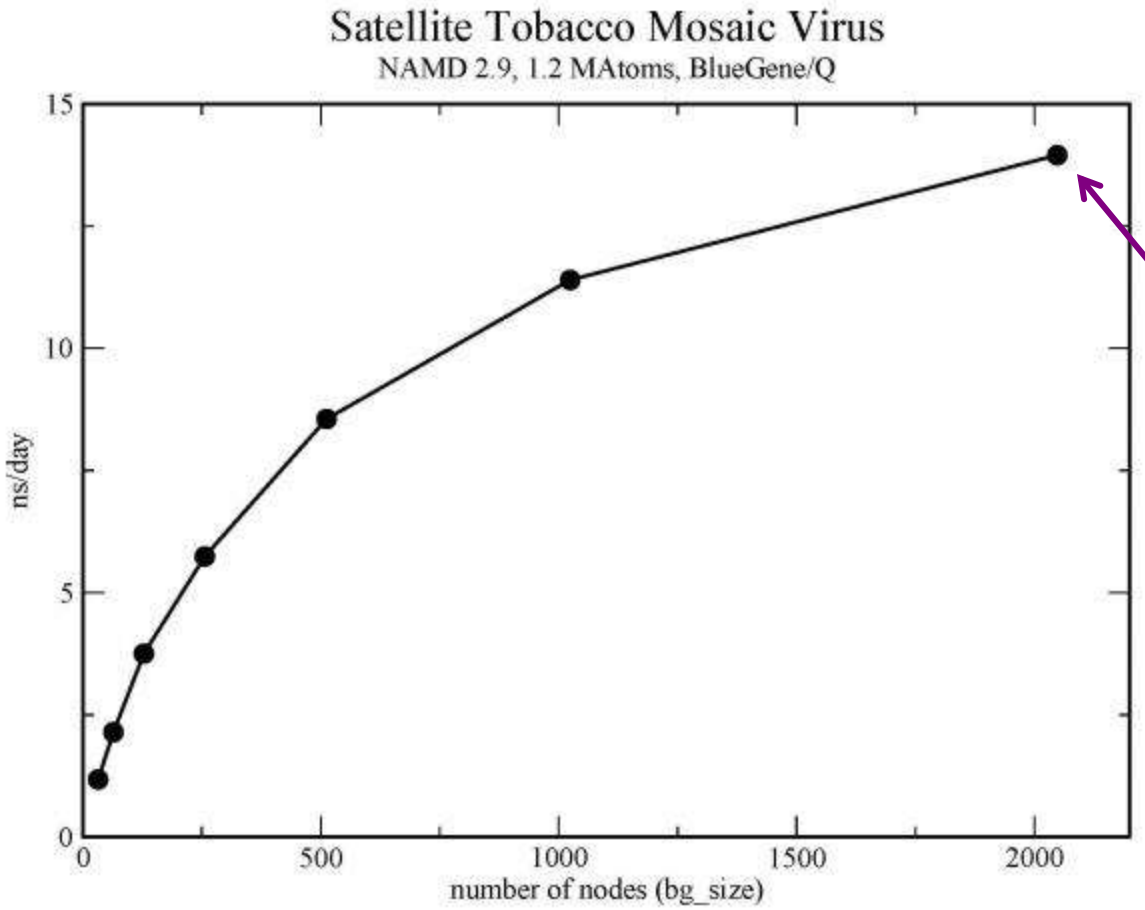
*16 = 4096 threads

Optimized by IBM namd version: adopts a mixed MPI/OpenMP thread approach for the parallel computation.
The number of MPI process per node are selected with the --ranks-per-node option of LoadLeveler, while the number of OpenMP threads per MPI process with the +ppn flag of namd.

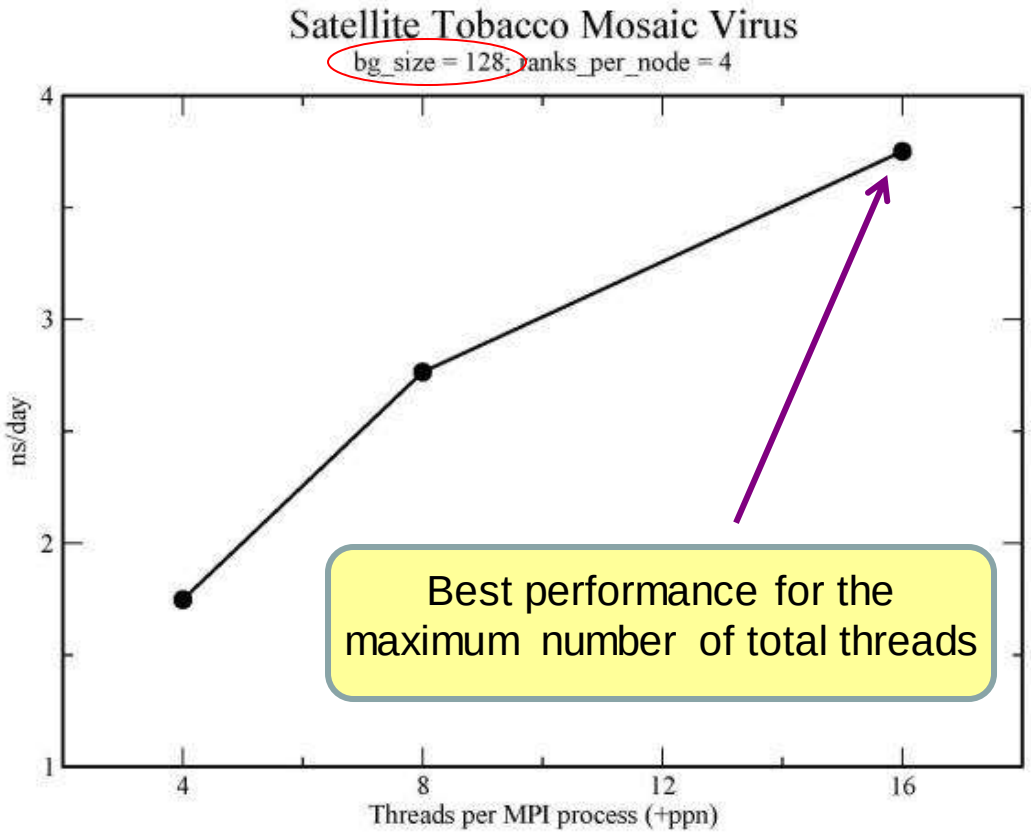
Satellite Tobacco Mosaic virus



STMV (virus) benchmark (1,066,628 atoms, periodic, PME)

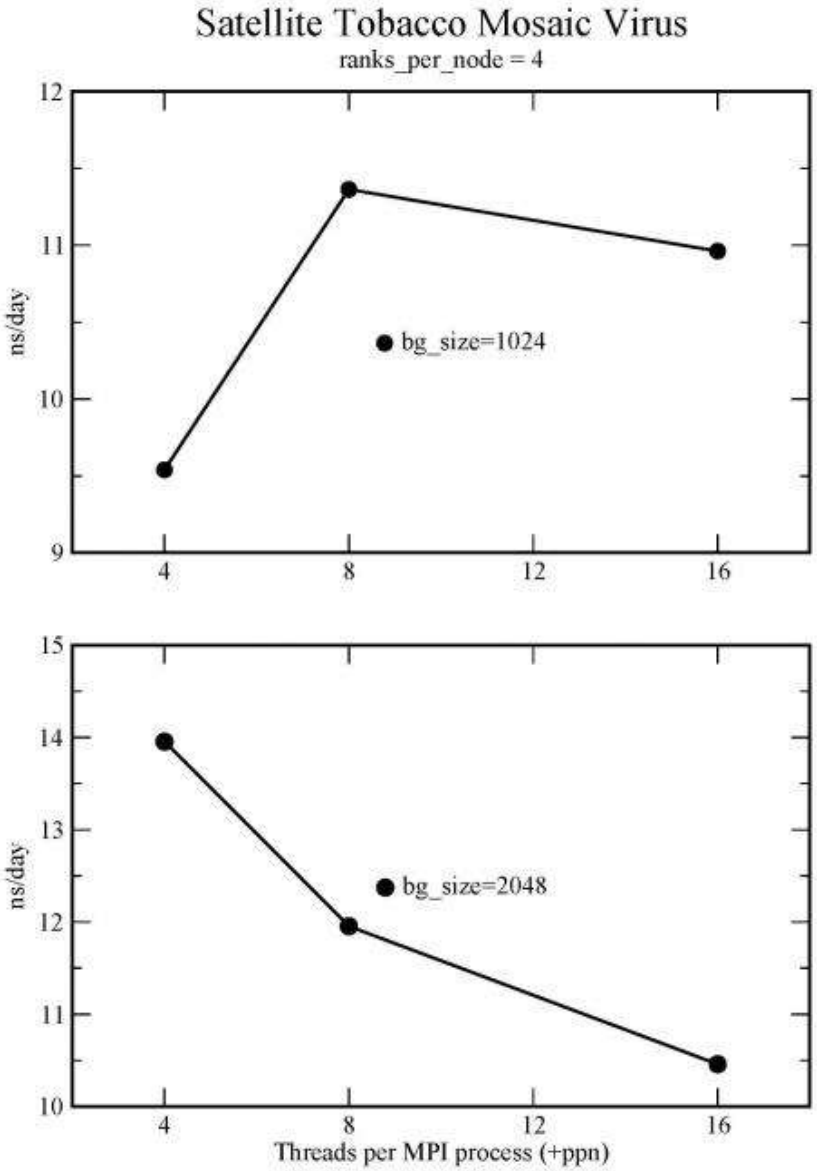


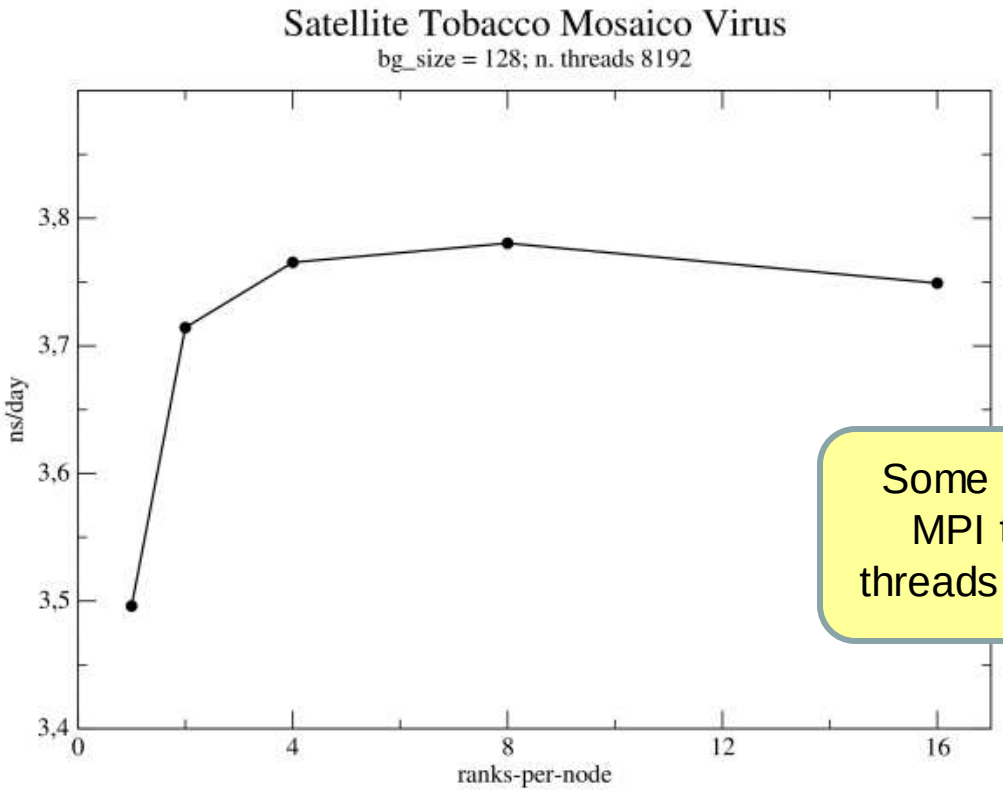
Increase of performance up to 32,768 cores! (bg_size=2048




```
runjob --np 512 --ranks-per-node 4 : $NAMD_HOME/namd2 stmv_ori.namd +ppn4 > stmv.out
runjob --np 512 --ranks-per-node 4 : $NAMD_HOME/namd2 stmv_ori.namd +ppn8 > stmv.out
runjob --np 512 --ranks-per-node 4 : $NAMD_HOME/namd2 stmv_ori.namd +ppn16 > stmv.out
```

That is not true anymore when increasing bg_size: Check the performance on your system!





Some combinations of MPI tasks/OpenMP threads are less efficient

```
runjob --np 2048 --ranks-per-node 16 : $NAMD_HOME/namd2 stmv_ori.namd +ppn4 > stmv.out
runjob --np 1024 --ranks-per-node  8 : $NAMD_HOME/namd2 stmv_ori.namd +ppn8 > stmv.out
runjob --np  512 --ranks-per-node  4 : $NAMD_HOME/namd2 stmv_ori.namd +ppn16 > stmv.out
runjob --np  256 --ranks-per-node  2 : $NAMD_HOME/namd2 stmv_ori.namd +ppn32 > stmv.out
runjob --np  128 --ranks-per-node  1 : $NAMD_HOME/namd2 stmv_ori.namd +ppn64 > stmv.out
```

Amber template script for Fermi

```
#!/bin/bash
# @ job_name  = amber.$(jobid)
# @ output  = z.$(jobid).out
# @ error = z.$(jobid).err
# @ shell = /bin/bash
# @ job_type  = bluegene
# @ wall_clock_limit  = 01:00:00
# @ notification = always
# @ bg_size  = 64
# @ account_no = my_account_no
# @ queue

module  load amber/12

# do any amber pre-processing on the front-end nodes (antechamber, tleaps, etc...

# get the path of the sander executable for the backend nodes
sander=$(which  sander.MPI)

# add any  sander options
exe="$sander  -i mdin  -o mdout  -p prmtop  -c inpcrd -r restrt -ref refc -mtmd  mtmd  -x mdcrd"

#launch sander on all the allocated back-end nodes (note the : which must be present with  this syntax)
# if you have  memory problems reduce the --ranks-per-node option
runjob  --ranks-per-node  16 --env-all : $exe
```

GROMACS template script for Fermi

```
#!/bin/bash
# @ job_name  = gromacs.$(jobid)
# @ output  = $(jobid).out
# @ error = $(jobid).err
# @ shell = /bin/bash
# @ job_type  = bluegene
# @ wall_clock_limit  = 01:00:00
# @ notification = always
# @ bg_size  = 64
# @ account_no = <my_account_no>
# @ queue

module  load gromacs/4.6.5

# get the path of the mdrun  executable  for the backend  nodes
mdrun=$(which  mdrun_bgq)

# add any mdrun  options.
# Here we are using 4 OpenMP  threads  for MPI task.
exe="$mdrun  -v -s topol.tpr  -ntomp  4"

#launch  single  precision mdrun  on all the back-end  nodes
runjob  --ranks-per-node  16 --env-all : $exe
```

64*16 = 1024 MPI tasks
*4 = 4096 threads

mixed MPI/OpenMP thread version

PRACE

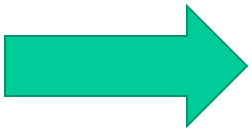
CINECA

Roadmap to Exascale
(architectural trends)

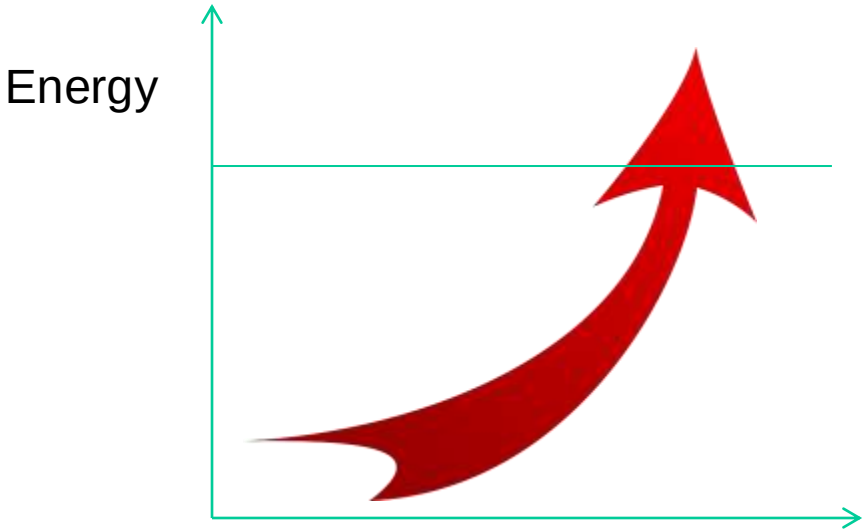
Systems	2009	2011	2015	2018
System Peak Flops/'s	2 Peta	20 Peta	100-200 Peta	1 Exa
System Memory	0.3 PB	1 PB	5 PB	10 PB
Node Performance	125 GF	200 GF	400 GF	1-10 TF
Node Memory BW	25 GB/s	40 GB/s	100 GB/s	200-400 GB/s
Node Concurrency	12	32	O(100)	O(1000)
Interconnect BW	1.5 GB/s	10 GB/s	25 GB/s	50 GB/s
System Size (Nodes)	18,700	100,000	500,000	O(Million)
Total Concurrency	225,000	3 Million	50 Million	O(Billion)
Storage	15 PB	30 PB	150 PB	300 PB
I/O	0.2 TB/s	2 TB/s	10 TB/s	20 TB/s
MTTI	Days	Days	Days	O(1Day)
Power	6 MW	~10 MW	~10 MW	~20 MW

Energy trends

"traditional" RISK and CISC chips are designed for maximum performance for all possible workloads

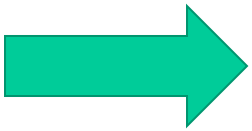


A lot of silicon to maximize single thread performace



Change of paradigm

New chips designed for maximum performance in a small set of workloads



Simple functional units, poor single thread performance, but maximum throughput