www.bsc.es
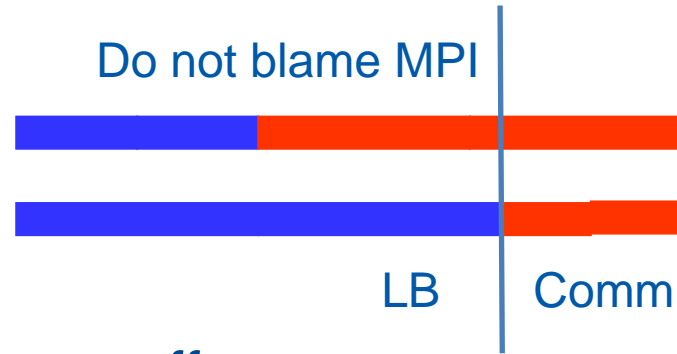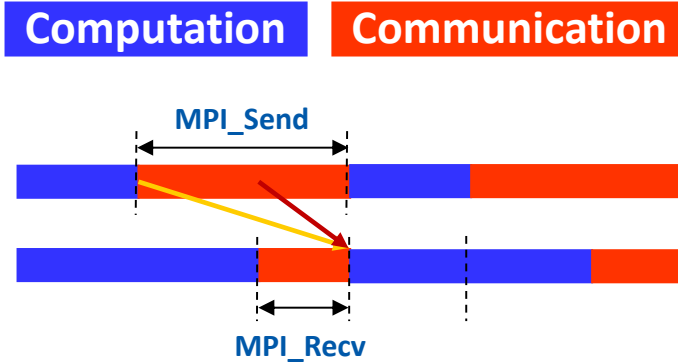
Barcelona Supercomputing Center
Centro Nacional de Supercomputación
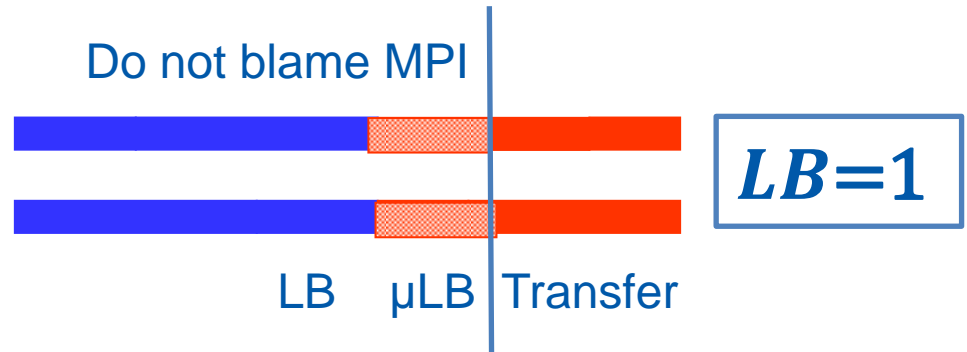
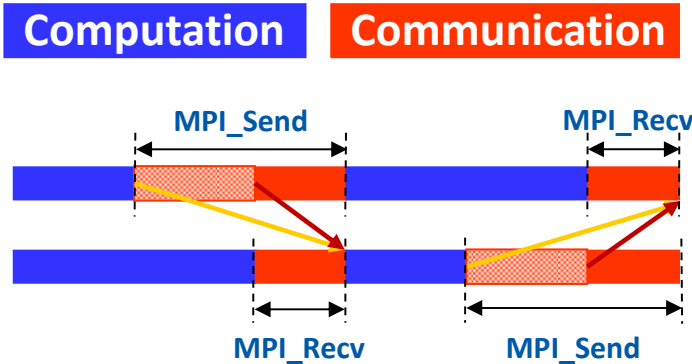# Is it me, or is it the machine?

Judit Gimenez (judit@bsc.es)

Scalable Tools Workshop (Tahoe)
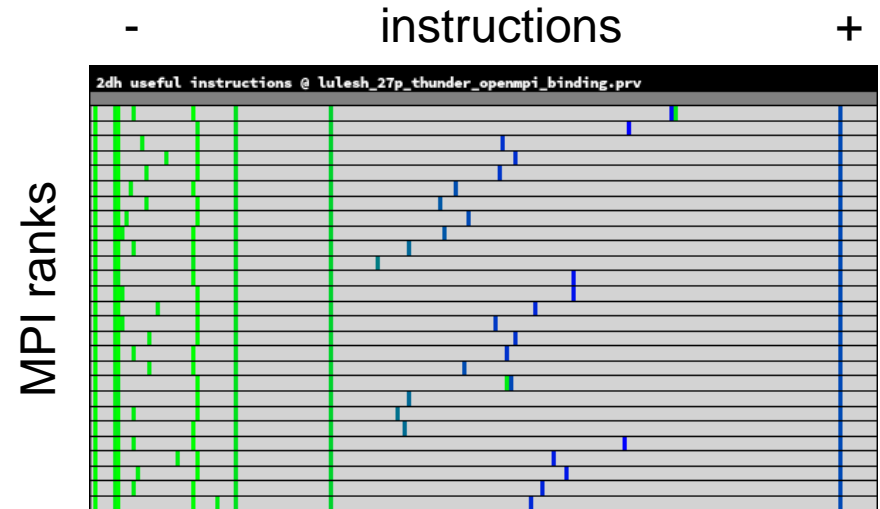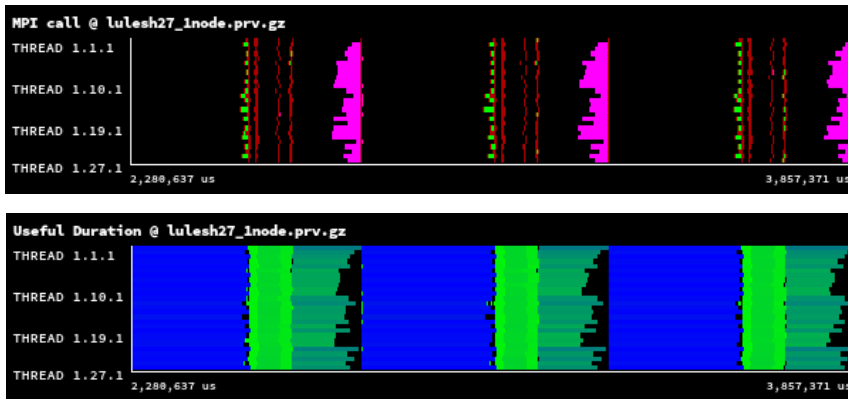August 2017

Parallel efficiency model

« Parallel efficiency = LB eff * Comm eff

« Parallel efficiency = LB eff * Serialization eff * Transfer eff

My favorite default app for trainings: Lulesh 2.0

《 Easy to install and does not require large input files

《 Iterative behaviour, well balanced except one region due to instructions unbalance

-                     instructions   +

MPI ranks

《 Requires a cube number of MPI ranks

→ my target = 27 ranks; no nodes/sockets sized 27 :)

– No OpenMP

《 Expected problem: some extra unbalance due to the unbalanced mapping

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Same code, different behavior

Code	Parallel efficiency	Communication efficiency	Load Balance efficiency
lulesh@machine1	90.55	**99.22**	91.26
lulesh@machine2	**69.15**	99.12	**69.76**
lulesh@machine3	70.55	96.56	73.06
lulesh@machine4	83.68	95.48	87.64
lulesh@machine5	**90.92**	98.59	**92.20**
lulesh@machine6	73.96	97.56	75.81
lulesh@machine7	75.48	**88.84**	84.06
lulesh@machine8	77.28	92.33	83.70
lulesh@machine9	88.20	98.45	89.57
lulesh@machine10	81.26	91.58	88.73

Warning::: Higher parallel efficiency does not mean faster!

Huge variability and worse than expected. Can I explain why?

Same code, same machine… still different behaviors

Playing with MPI @ A MPI	Parallel efficiency	Communication efficiency	Load Balance efficiency
IMPI	**85.65**	95.09	**90.07**
MPT	70.55	**96.56**	73.06

Playing with binding @ B configuration	Parallel efficiency	Communication efficiency	Load Balance efficiency
default	**81.26**	91.58	**88.73**
binding	75.10	**97.44**	77.07

Playing with both @ C MPI / configuration	Parallel efficiency	Communication efficiency	Load Balance efficiency
BullMPI / default	84.00	93.41	**89.35**
OpenMPI / default	79.45	**98.35**	80.73
OpenMPI / binding	82.10	95.08	86.35
BullMPI / binding	**85.15**	96.59	88.18

« Balance between nodes and across sockets

Useful Duration @ the_expected.prv.gz
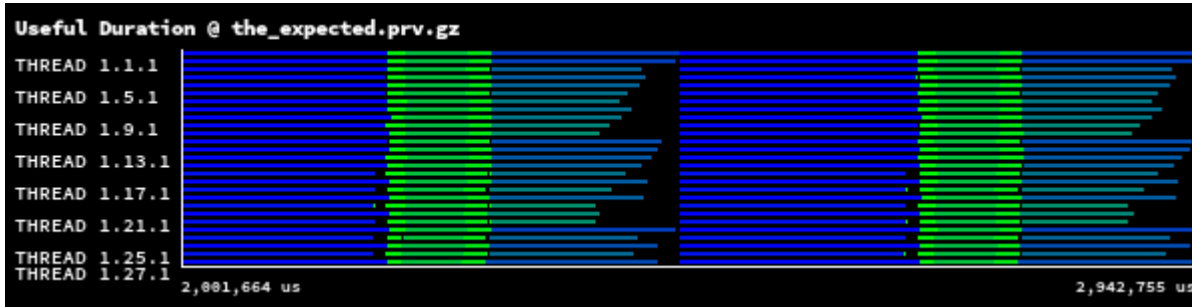
- Using 2 nodes x 2 sockets
- 3 sockets with 7 ranks, 1 socket with 6 ranks → small time unbalance

2DH useful duration correlated with @ the_expected.prv.gz
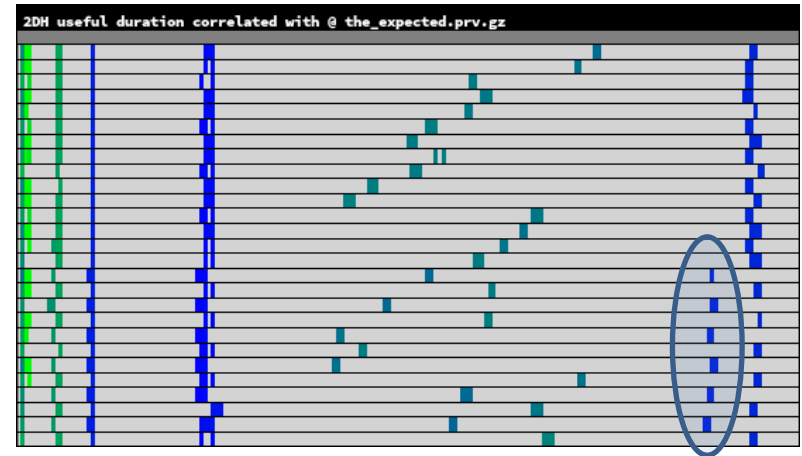
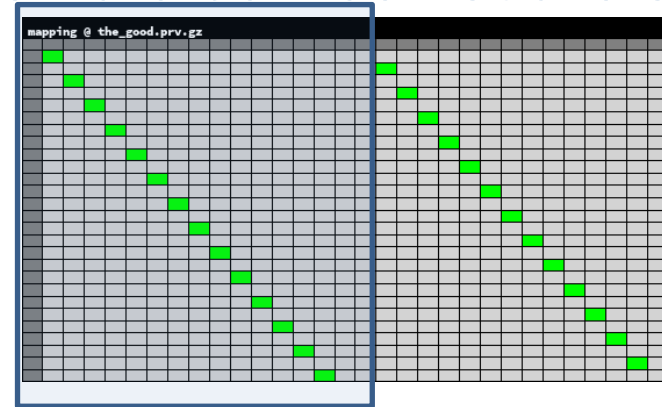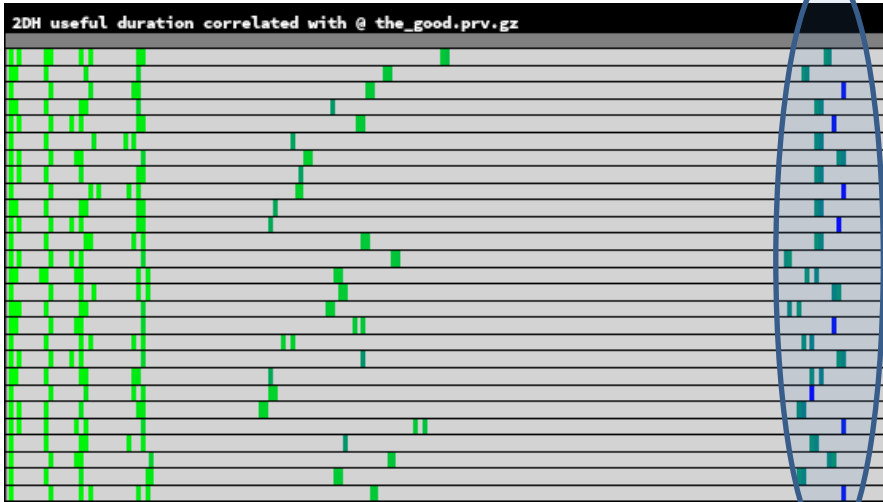**Less frequent than expected!**

6 guys
with more
resources

6

The good

Parallel eff. 90.92%
Comm 98.59%
LB   92.20%
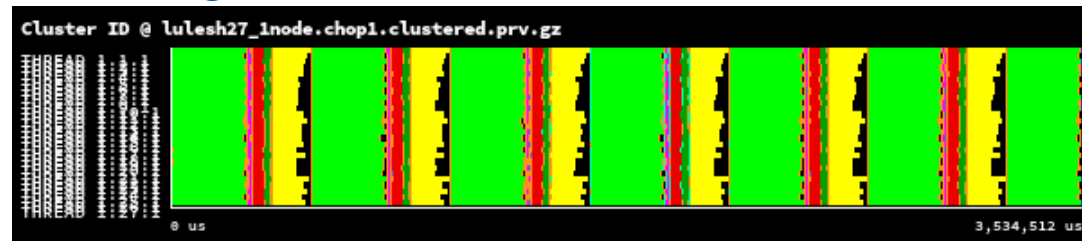
《 27 fits in a node

Alternate between sockets

first socket

Small IPC variability in the 2 main regions

2DH useful duration correlated with @ the_good.prv.gz

mapping @ the_good.prv.gz

Cluster Analysis Results of trace 'lulesh27₁node.chop1.prv'
DBSCAN (Eps=0.09, MinPoints=10)

Noise   +
Cluster 1   ×
Cluster 2   ✳
Cluster 3   □
Cluster 4   ■
Cluster 5   ⊙
Cluster 6   ●
Cluster 7   △
Cluster 8   ▲
Cluster 9   ▽

Instructions Completed
1×10^8
1×10^7

0.2   0.4   0.6   0.8   1
IPC

Cluster ID @ lulesh27_1node.chop1.clustered.prv.gz
@ us   3,534,512 us

Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación

The ugly

Same code! but now two trends (one per node)

Parallel eff. 69.15%
Comm 99.12%
LB   69.76%
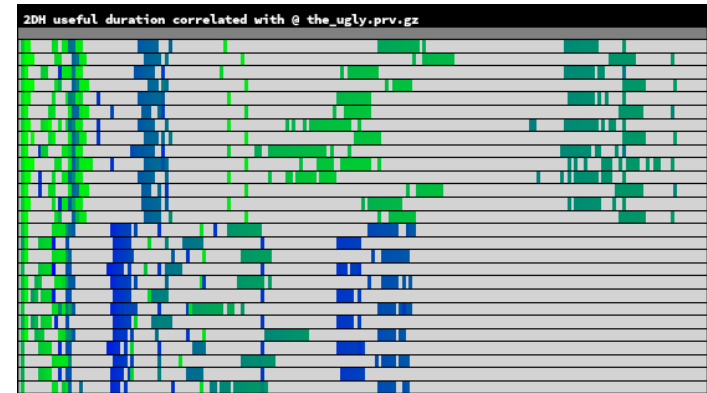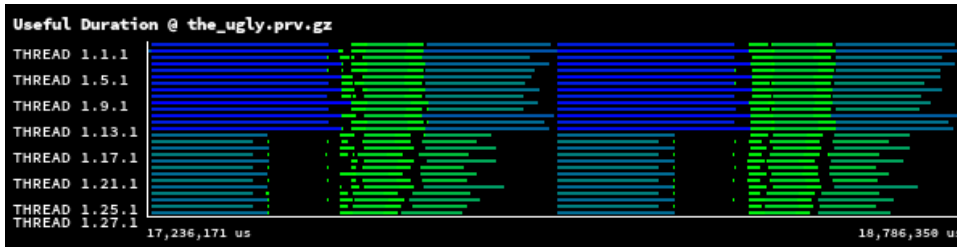
Slow node → significant lower IPC for almost all the regions
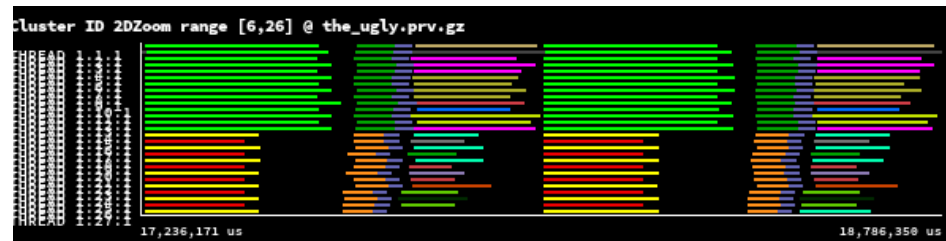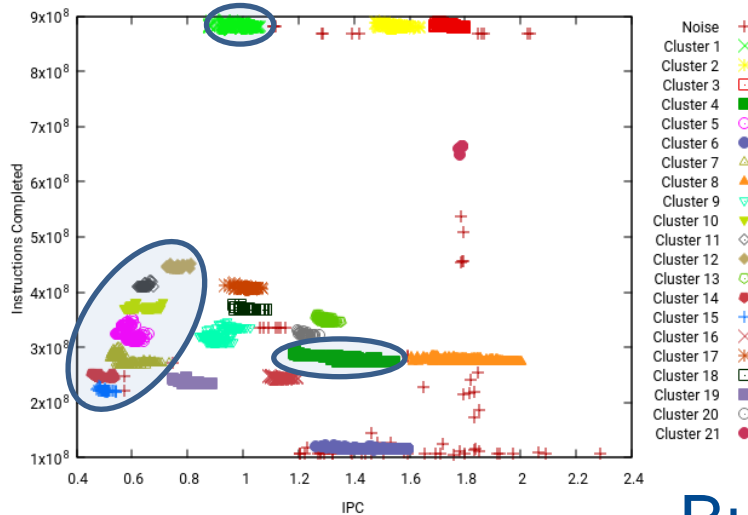
But all nodes are equal!

The ugly

Clock frequency sanity check
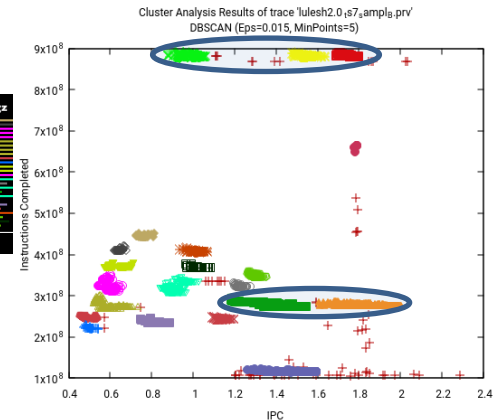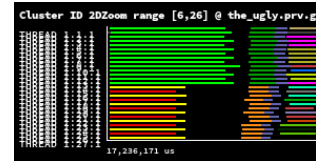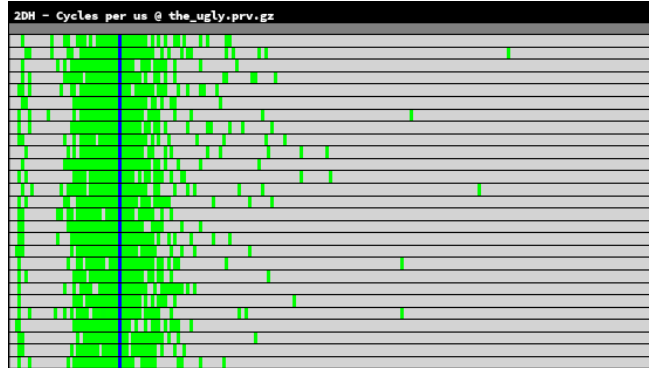
Insight checking hardware counters differences

Cluster Name	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 8
Avg. Duration	347038882	218366724	191517484	76087437	57184253
PAPI_L1_DCM	*4253914*	*6369746*	*6360328*	*2195915*	*1540034*
PAPI_L2_DCM	*1647293*	*1984225*	*1864857*	*448603*	*258565*
PAPI_L3_TCM	*945265*	*1399329*	*1390967*	*160690*	*111273*
PAPI_TOT_INS	881368852	880939731	881303626	275104547	274379604
PAPI_TOT_CYC	900031546	566347103	496687512	197363737	148307597
RESOURCE_STALLS:SB	**257407726**	**154854806**	**106684134**	**74990020**	**38463538**
RESOURCE_STALLS:ROB	3191720	3170735	2722761	2684869	562054
RESOURCE_STALLS:RS	43484959	63717139	63017342	38768779	32699838

Guess: Memory problem – confirmed by sysadmin tests!

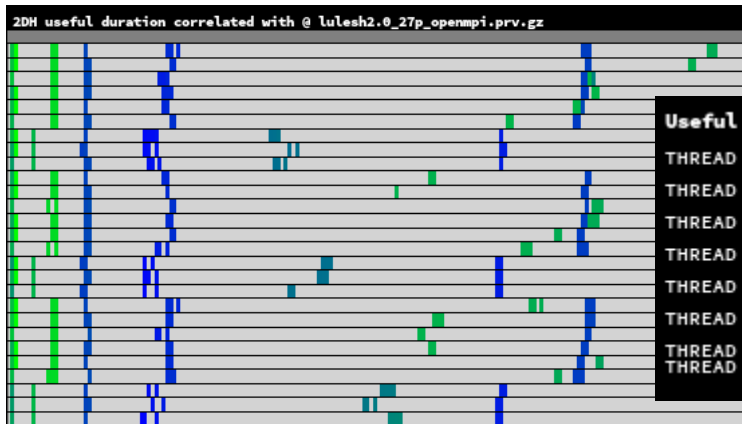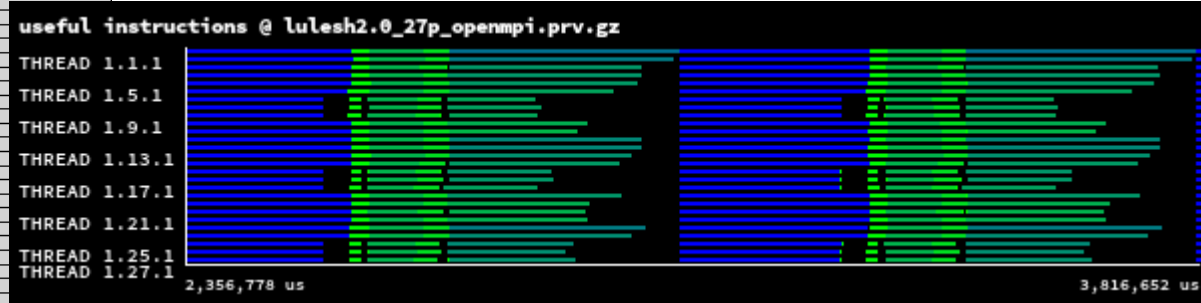# The unbalanced

« **Balance between nodes not between sockets**

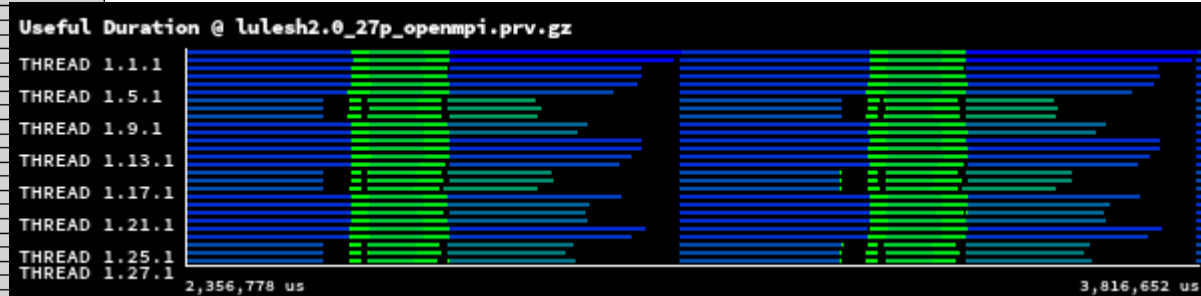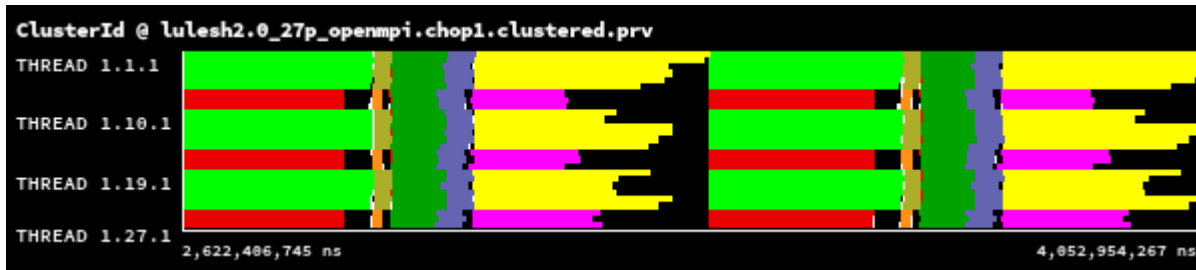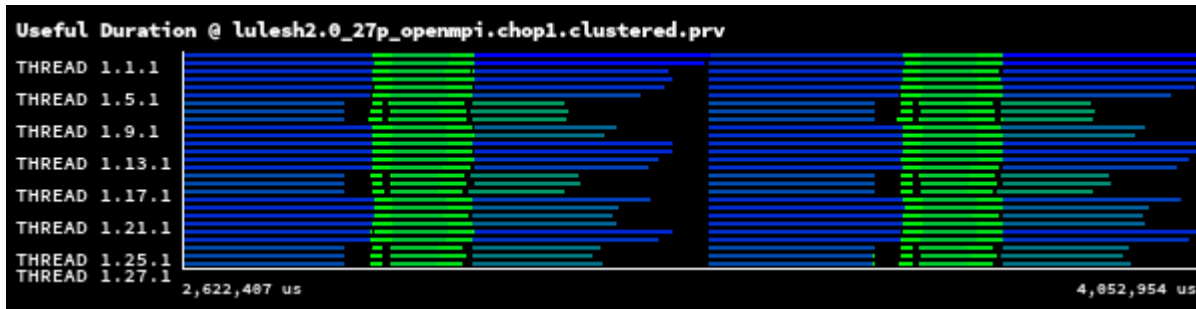Parallel eff. 79.45%
Comm 98.35%
LB 80.73%
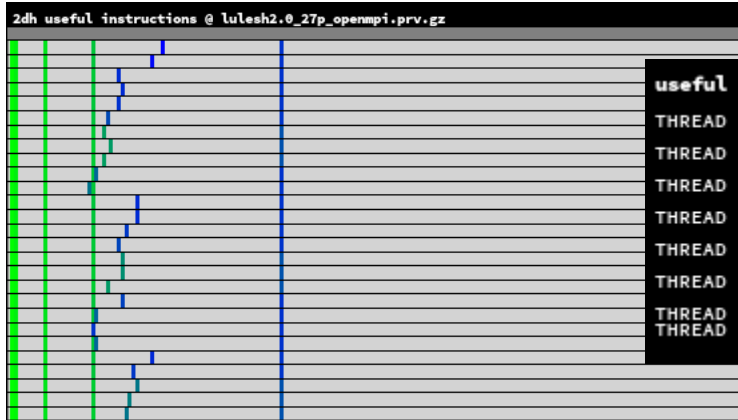
What lulesh does

What the machine does running lulesh

Balance between nodes → 9 per node

Fill first a socket → 6 + 3

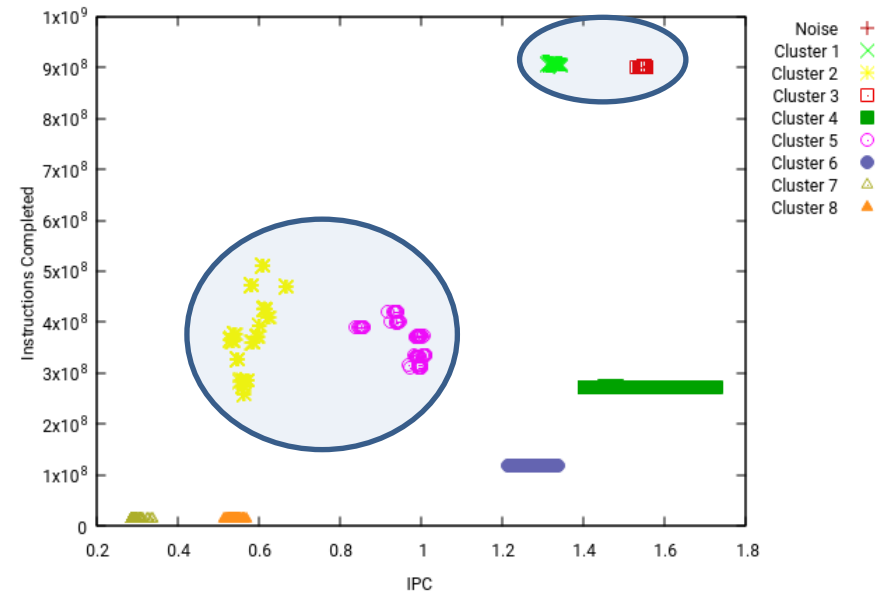# The unbalanced

Useful Duration @ lulesh2.0_27p_openmpi.chop1.clustered.prv

ClusterId @ lulesh2.0_27p_openmpi.chop1.clustered.prv

Two main regions suffer the penalty of the different socket occupancy

Most frequent behaviour!

Cluster Analysis Results of trace 'lulesh2.0 $27p_o$penmpi.chop1.prv'
DBSCAN (Eps=0.09, MinPoints=5)
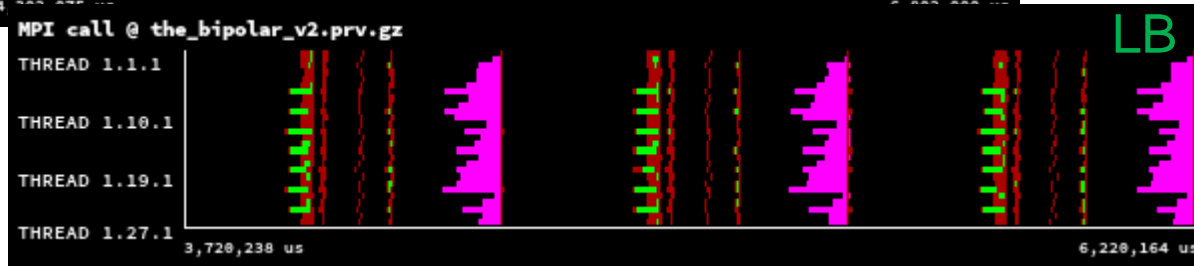
The bipolar

« MPI modify the computing phases behavior!

Parallel eff. 70.55%
  Comm 96.56%
  LB   73.06%

Parallel eff. 85.65%
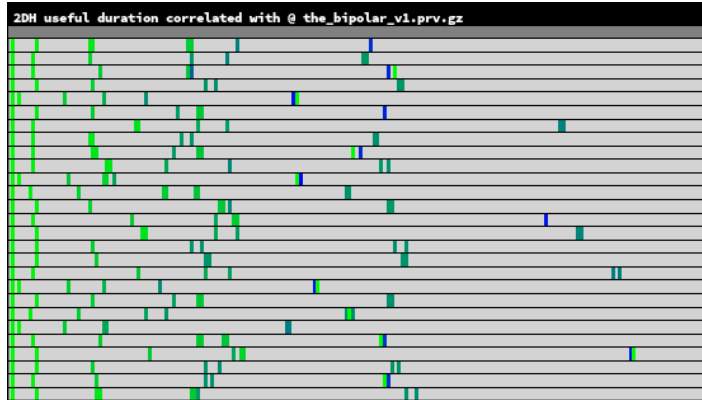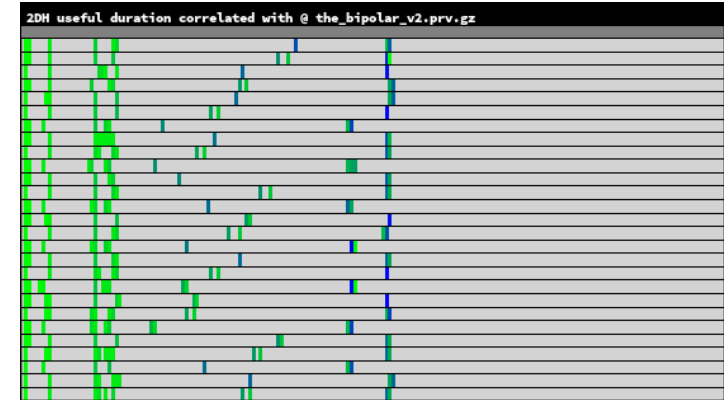 Comm 95.09%
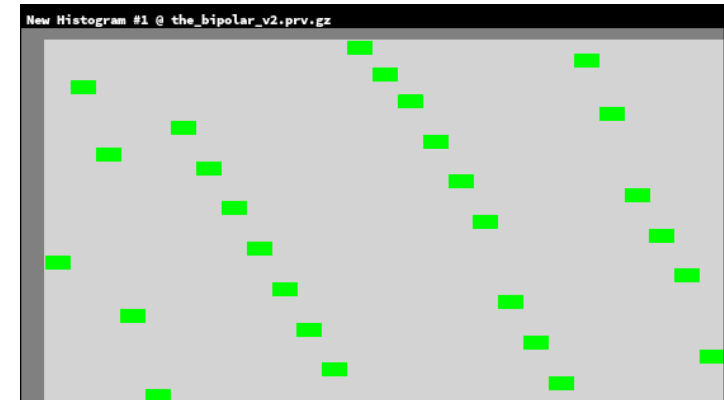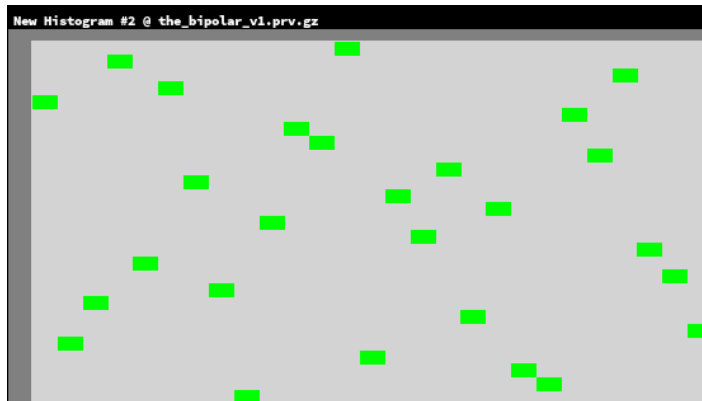 LB   90.07%

MPT
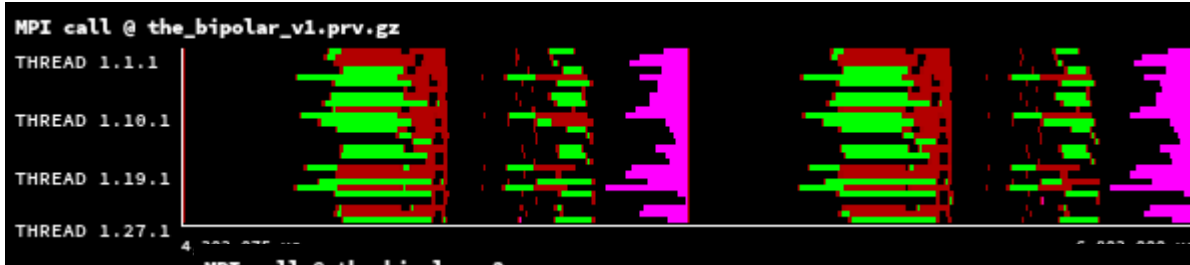
IMPI
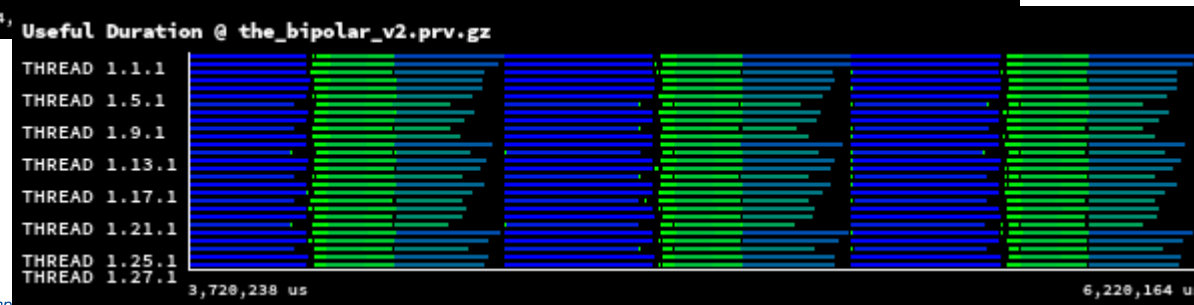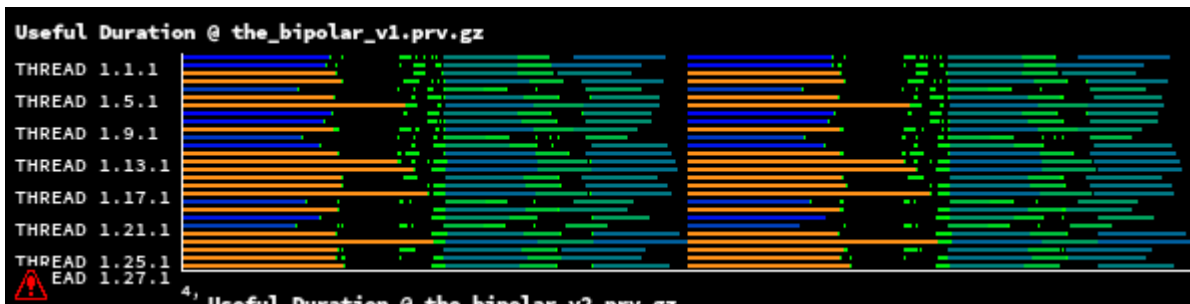
The bipolar

MPT   IMPI

Histogram of useful duration

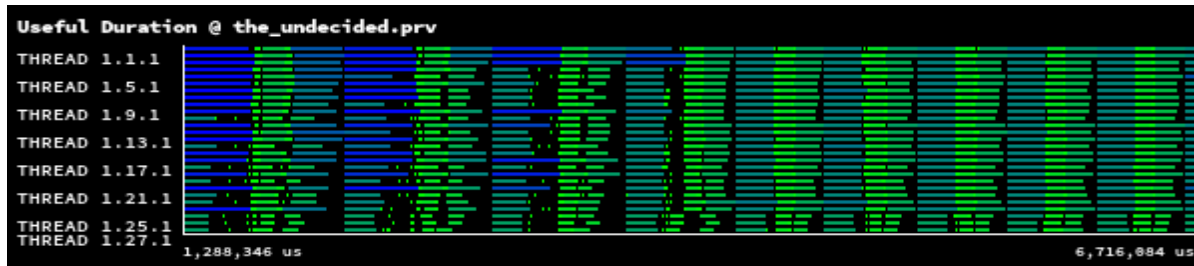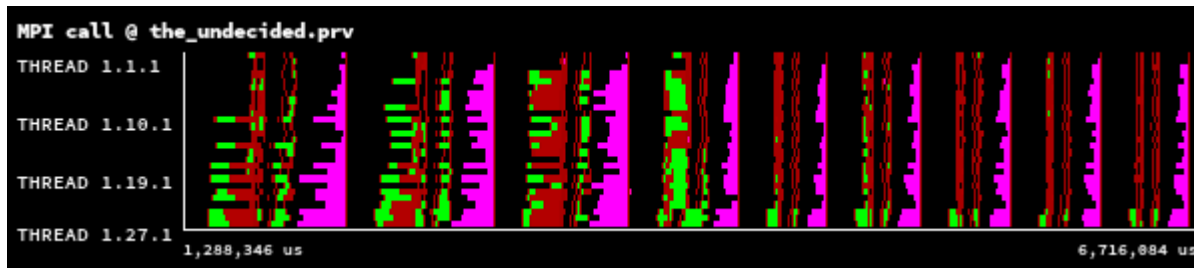**Process mapping "photo"**

… because it can select a different mapping / binding

« Same number of instructions per iteration showing a noisy behavior w.r.t time

Parallel eff. 77.28%
Comm 92.33%
LB   83.70%

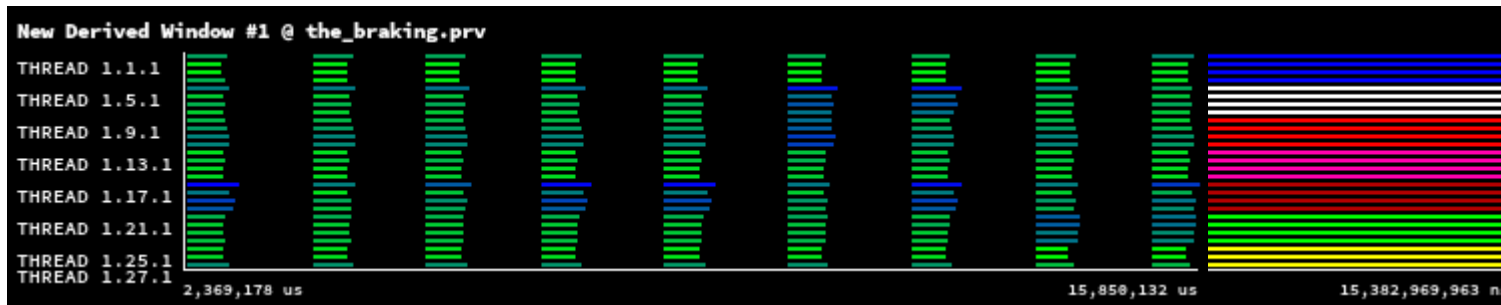… use to correspond to a system that does unnecessary process migrations
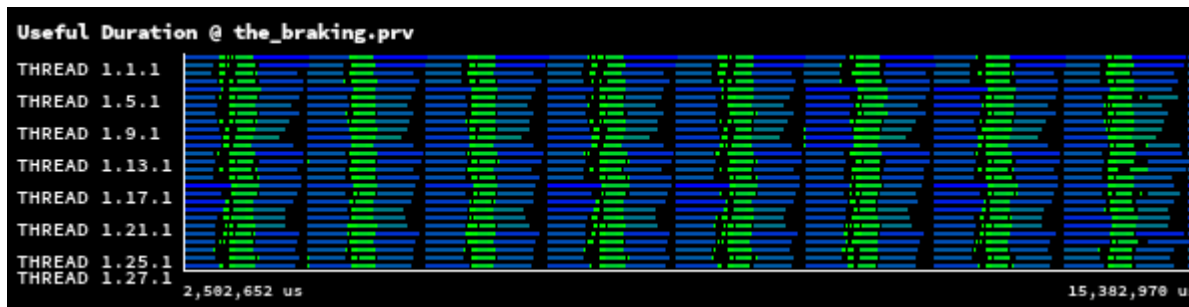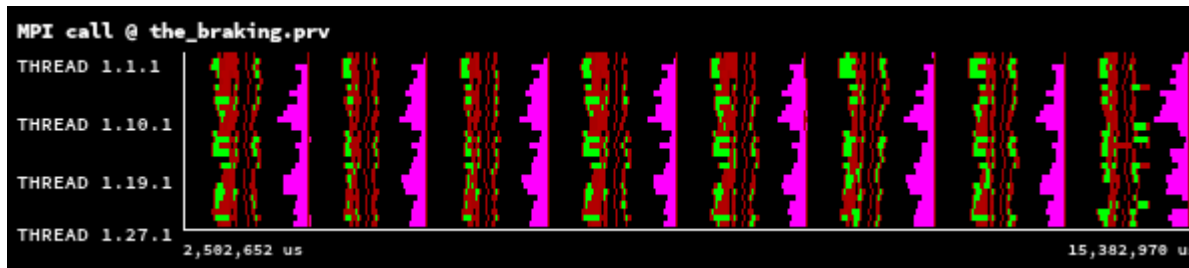
The braking

Parallel eff. 75.48%
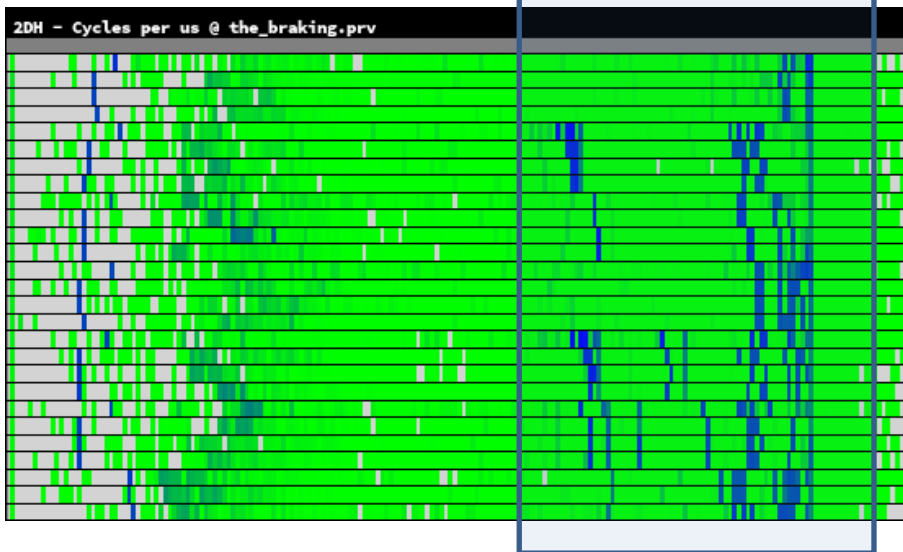**Comm 88.84%**
LB   84.06%

❞ But not always noise is caused by migrations

node id

The braking

« Clue: the noise affects to all processes within a node

The OS of each node is reducing for a while the clock frequency asynchronously!

Conclusions

‟ As code developer, better not to assume machines will do a good job running your code because you did a good job programming your application

‟ As performance analyst, do not assume where are the bottlenecks, be open minded and equipped with flexible tools (like Paraver ;)

As Bruce Lee said "Be water my friend!"