LS-DYNA High Performance Computing

Presented by Jason Wang

Oct 26, 2022

Ansys

Outline

- Virtual testing
- New features
  - Code restructure and decomposition
  - Re-decomposition
  - OneMPI
  - Trouble shooting on cluster computing
- High Performance Computing on Cloud
- CPUs and Compilers updates
- Summary

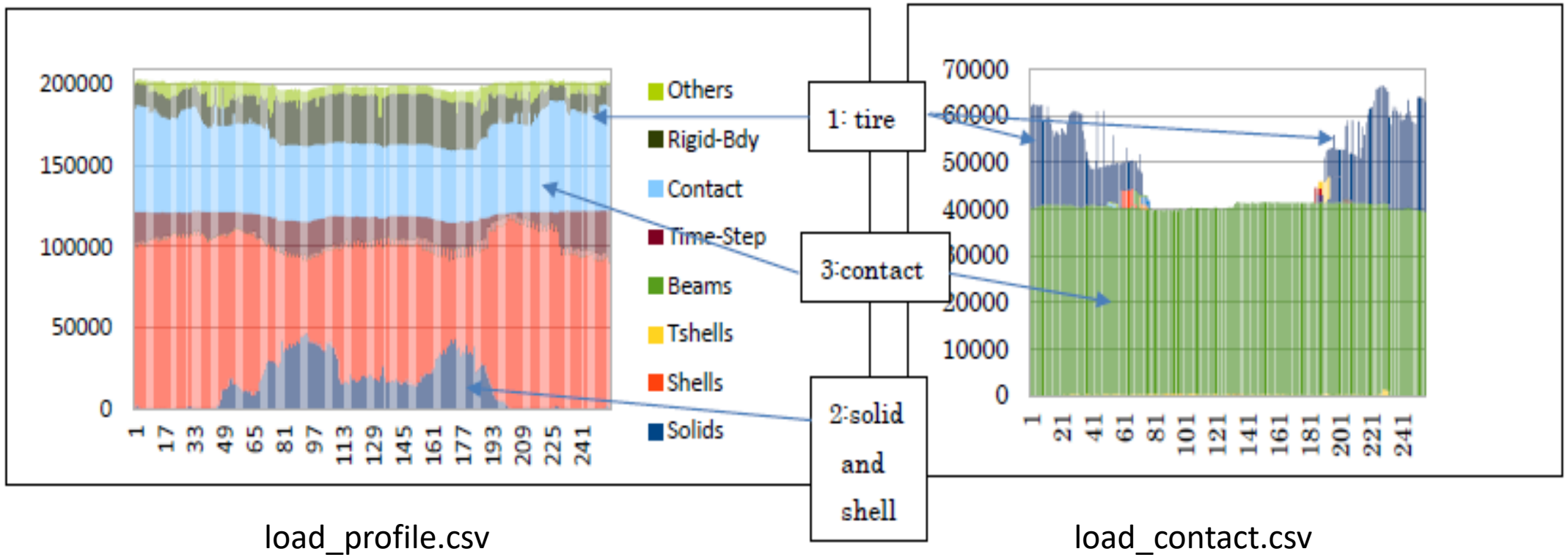
Virtual Testing

- Smaller element size (mixed element types and sizes)
- Eroding contacts – solid elements tetra 10, 13
- Sophisticated material models (3rd party user material)
- Connector capabilities (cluster of solids, HAZ, SPR, etc)
- Fine mesh barriers and dummies/Bio-dummies
- Multi-physics: multiple airbags, fuel tank, battery safety, etc
- More expensive element formulation (high order elements, IGA, etc.)
- Explicit/Implicit analysis

➢ Better prediction and better performance under same amount for computing resource
➢ Better scaling to large number of cores

Code restructure – make code faster

Baseline running with R7.1.2, 256 MPP ranks
solid: tet13
shell: type16
HG8
Soft=2, bucket sort frequency=100

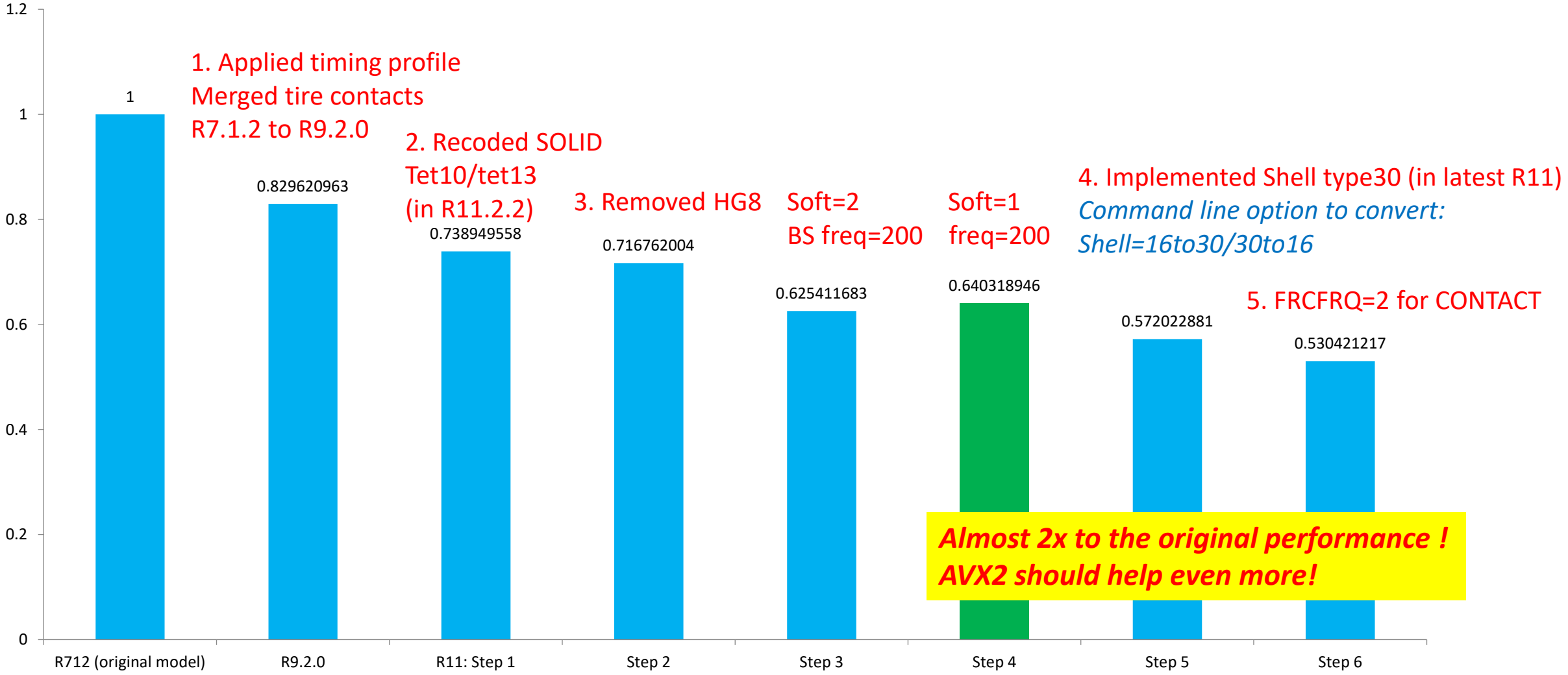


load_profile.csv

load_contact.csv

Team: Dave Benson, Jason Wang, Leslie Lum, Lee Bindeman, Dynapower, JSOL

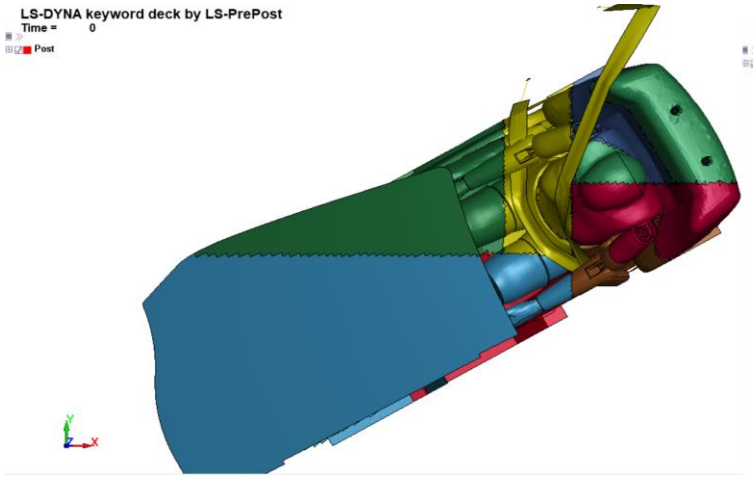
Code restructure – make code faster



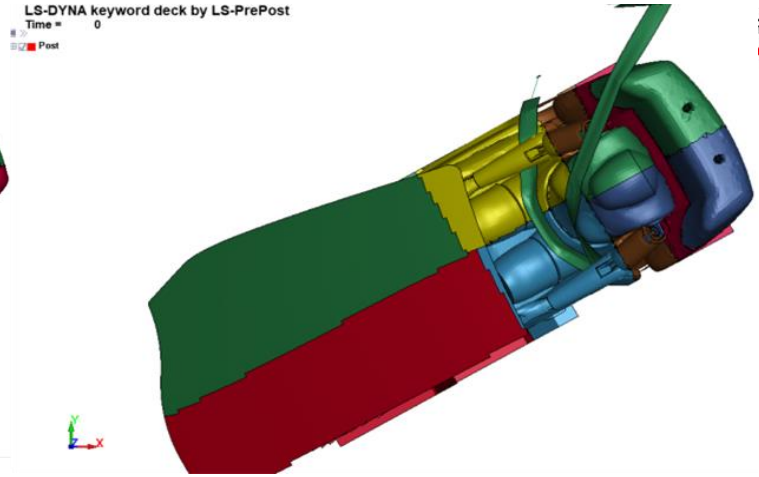
Team: Dave Benson, Jason Wang, Leslie Lum, Lee Bindeman, Dynapower, JSOL

Decomposition/transformation

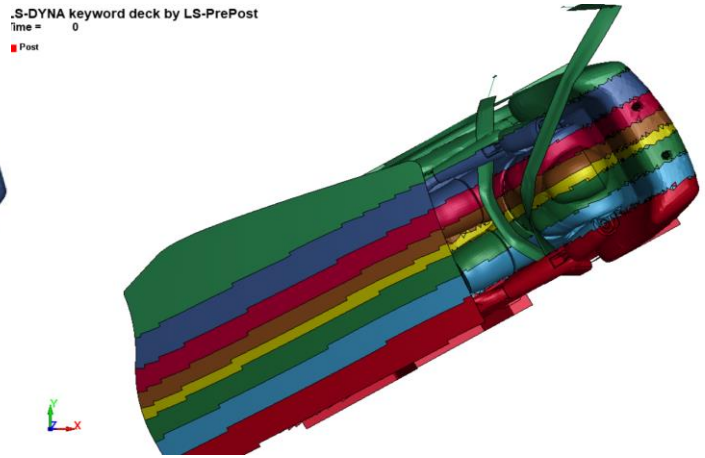
- *CONTROL_MPP_DECOMPOSITION_TRANSFORMATION: It is possible to combine various transformation.



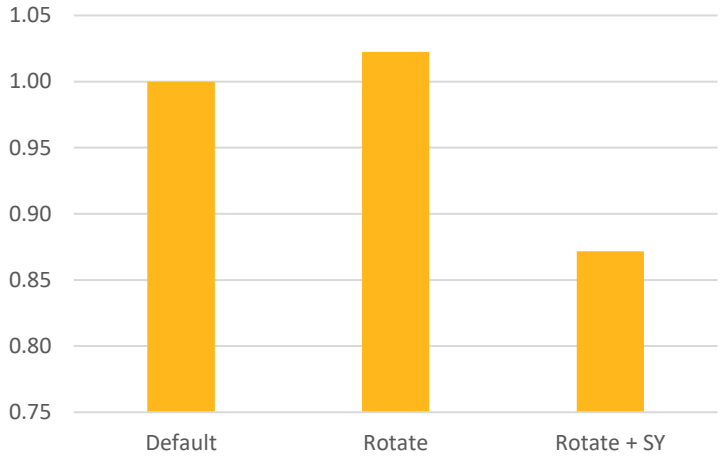
Default



Rotate



Rotate+SY




```
*CONTROL_MPP_DECOMPOSITION_TRANSFORMATION
$#     Type        val
         RZ        -30
$#     Type        val
         SY        100
```

Team: Julien Santini

Decomposition

Customer Goal

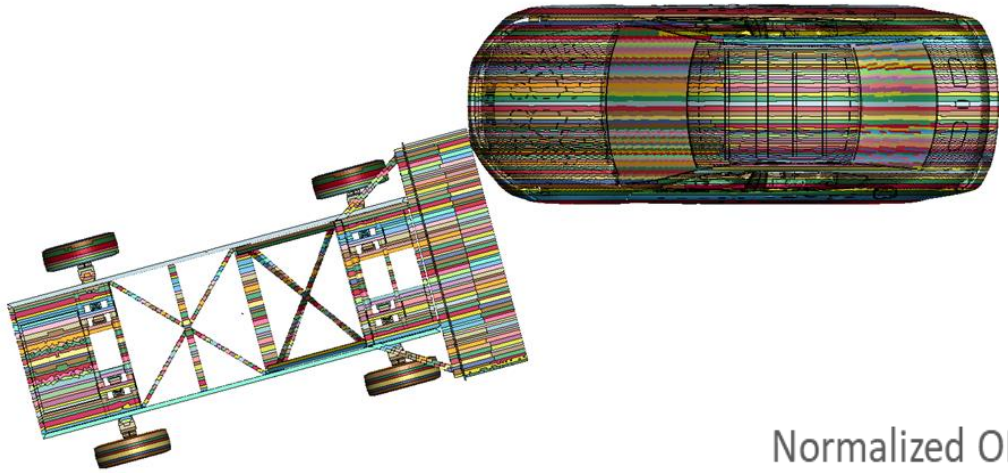
- Establish model setup procedure based on impact mode to insure efficient use of resources and reasonable runtime

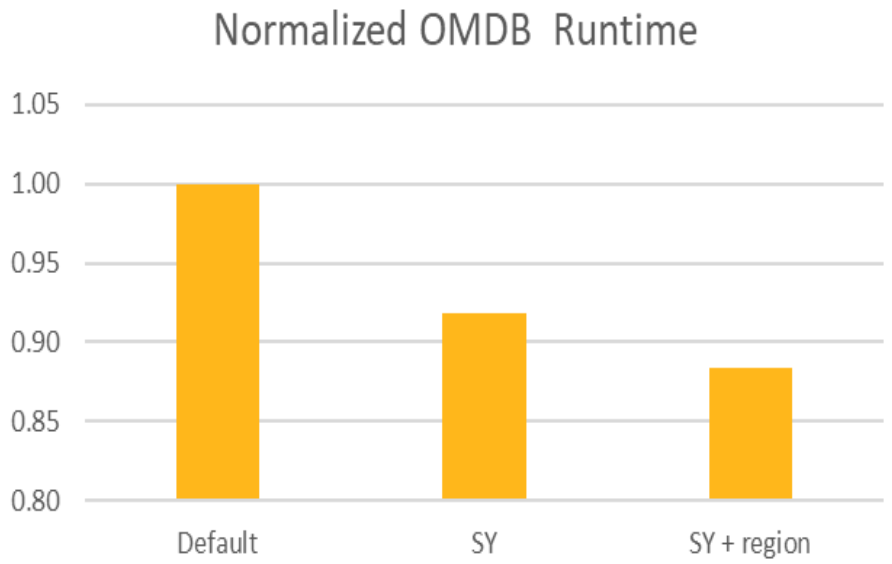
Evaluations

- Global decomposition based on impact mode
- Distribute airbags based on size
- Distribute some parts based on material
- Reduce number of contacts, use groupable

Success

- **Achieved reduction ranging from 15% to 65% in runtime depending on model content and impact mode**



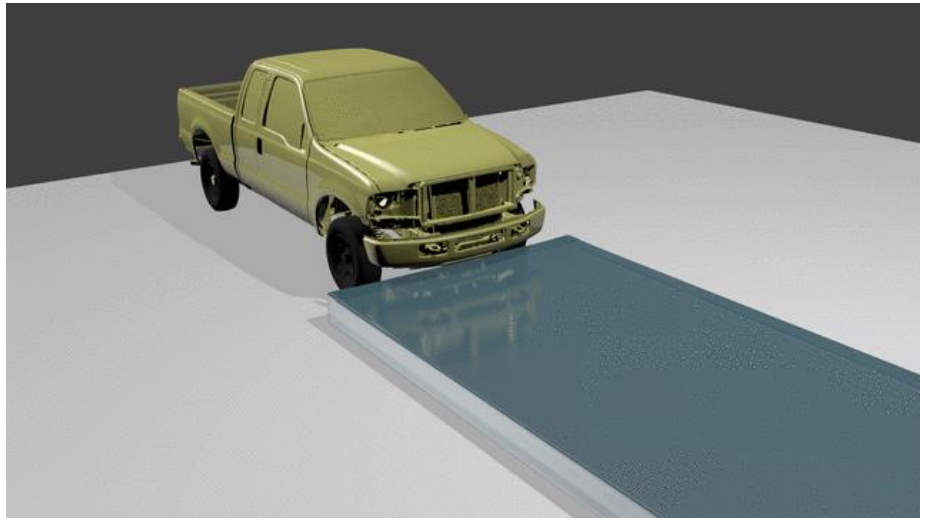


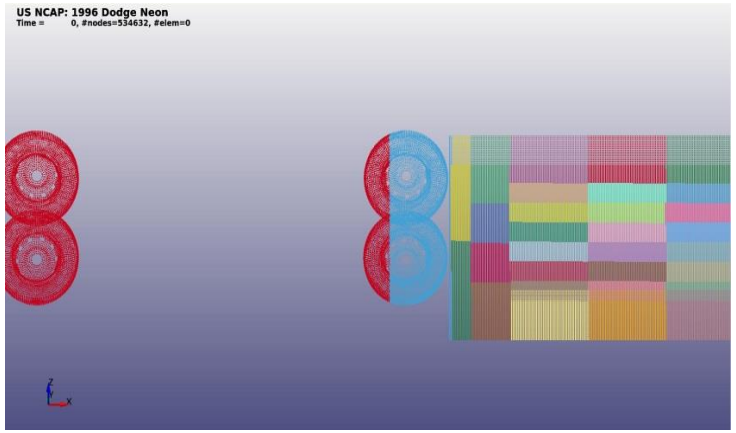
Team: Julien Santini

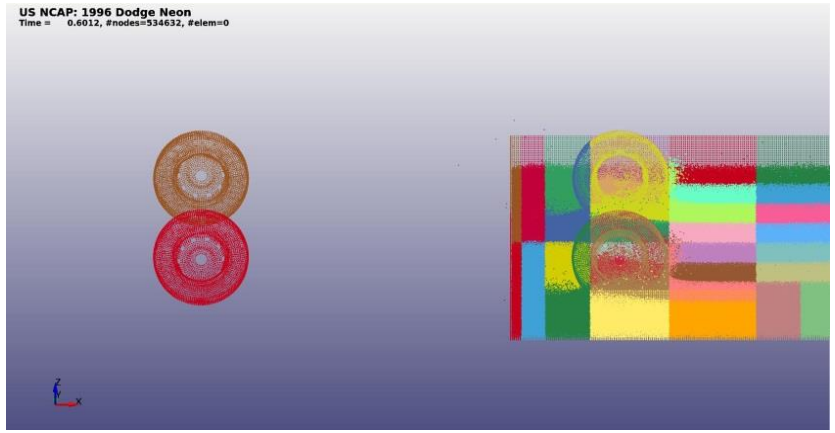
REDECOMP - model with large relative motion

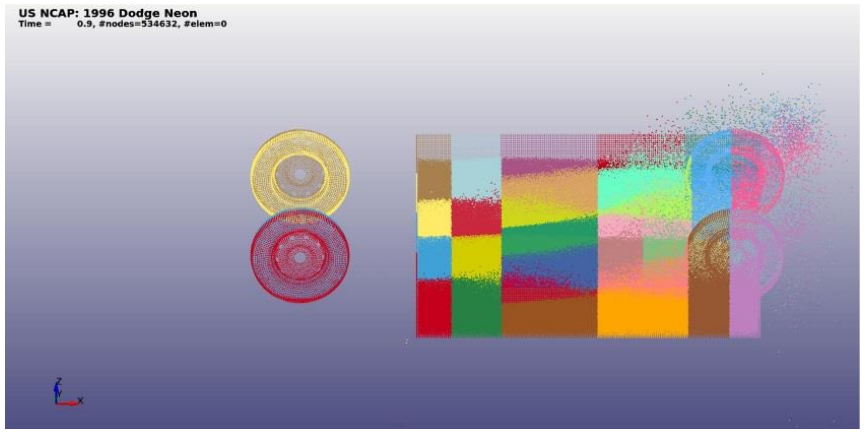
- Deformation/translation creates extra communication and causes load unbalancing.

- Re-decomposition rearranges partition
  1. reduces network traffic
  2. Keep neighbor elements in the same processor









Elements have the same color are in the same MPP domain

Decomposition/REDECOMP

Load balancing issue from unweighted features , i.e. user defined functions, user material, etc

Method 1:

Use following option for the initial run

decomp { timing_start ncycle_start timing_end ncycle_end }

Collect runtime cost of parts between the cycles and store cost profile to file – DECOMP_TIMINGS.OUT

In the subsequent runs, one uses the following pfile

decomp { timing_file DECOMP_TIMING.OUT }

Method 2:

***CONTROL_MPP_DECOMPOSITION_REDECOMPOSITION_{ONCE}**

{ONCE}: Only performance redecomposition once (usually use with **SAMPT** for better runtime.)

FREQ: REDECOMP frequency

DEFGEO: REDECOMP uses original/deformed geometry

WEIGHT: Scale factor for element in contact/plastic strain for next step

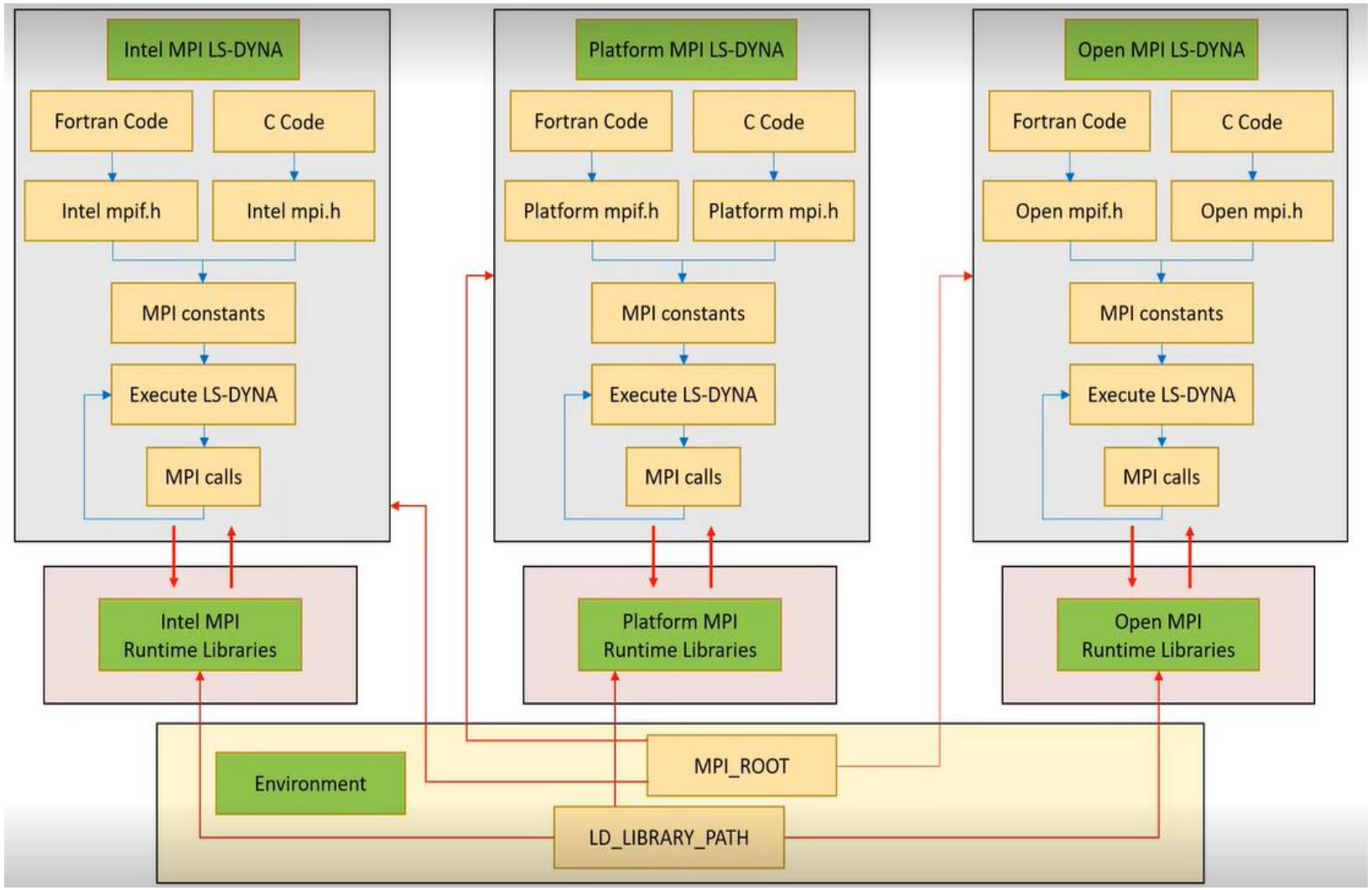
STIME: Starting time for REDECOMP

SAMPT: Runtime element cost for the next step

LS-DYNA OneMPI - R14.0

Currently under x86-64 Linux:

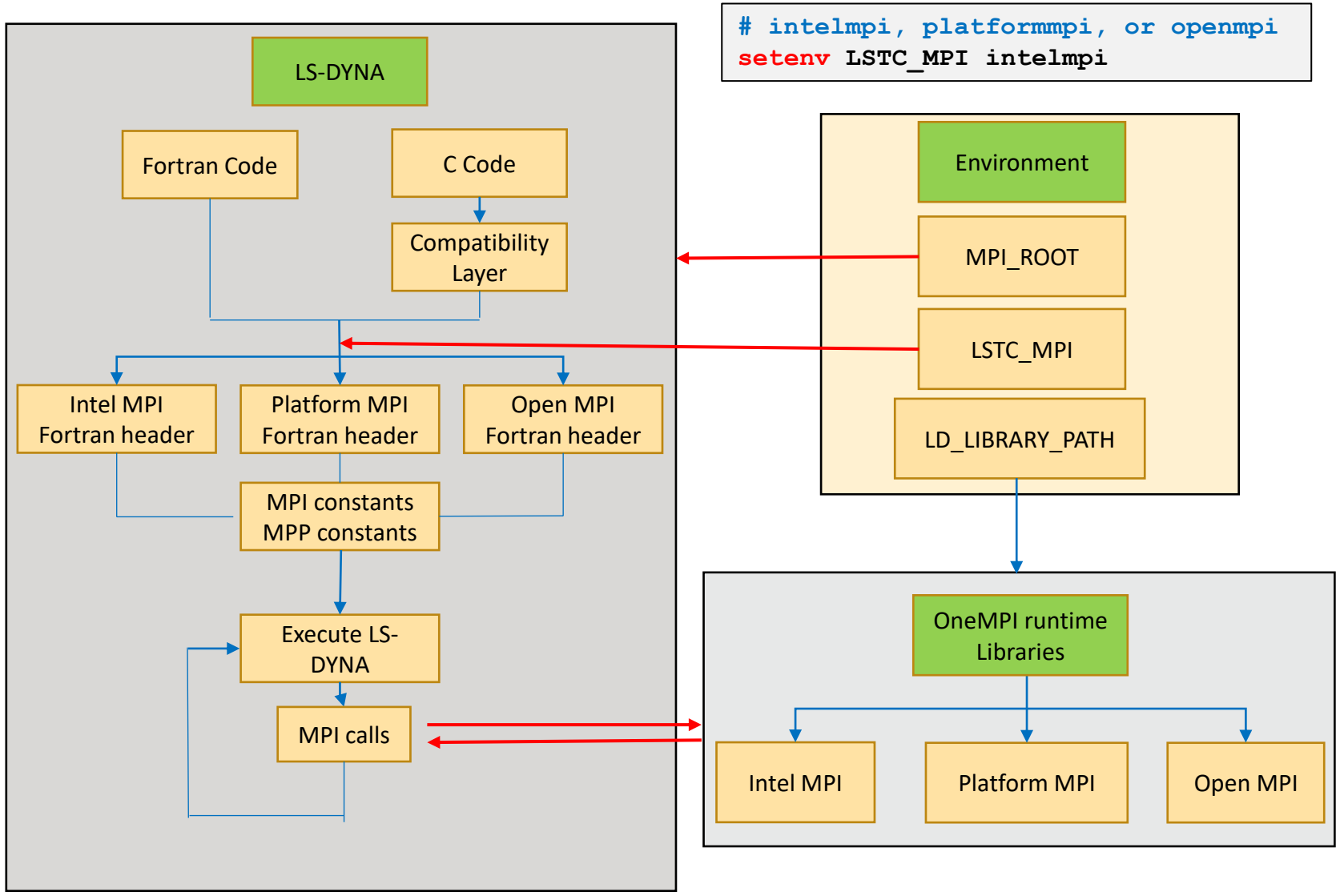
1. Three different binaries for Intel MPI, Platform MPI and OpenMPI
2. There is NO more update for Platform MPI (MPI 2.x)
3. Groups may select their own MPI for joint project. (they need use the same revision, same optimization, etc. to get the consistent results between groups.)



Team: Eric Day, François-Henry Rouet, Bob Lucas, Jason Wang

LS-DYNA OneMPI – R14.0

1. **Intel MPI binary** has the functionality of three MPP binaries in one on Linux (Intel MPI 2018, Open MPI 4.x, Platform MPI) and has Intel MPI 2018 and MSMPI v10 on Windows.
2. Identical solutions between MPI (LSTC_REDUCE on) to make easier between different groups
3. Help to convert from Platform (MPI 2.x) MPI to new default (MPI 3.x and above)
4. Accelerate LS-DYNA release process by reducing MPP QA three-fold
5. Help LS-DYNA align with other Ansys Products by limiting quantity of binaries



Team: Eric Day, François-Henry Rouet, Bob Lucas, Jason Wang

Trouble Shooting for stalling job or deadlock on clusters

- Several scripts will be created for each job with the following naming convention.

  Main script: $JOBID.exec-gdb.sh

  Script for each NODE: $JOBID.exec-gdb-$HOSTNAME01.sh

  $JOBID.exec-gdb-$HOSTNAME02.sh

- From the job current working directory (cwd) and execute the following commands to generate traceback information

  ```
  script debug_out
  ./$JOBID.exec-gdb.sh
  exit
  ```

- Please check the output information of debug_out or send this file back to us

LS-DYNA High Performance Computing on Cloud

✓Cloud Service Provider (CSP): AWS, Azure, Oracle, Google, etc

✓Hardware vendors: AMD, Intel, AWS(ARM), IBM(Power 10), etc

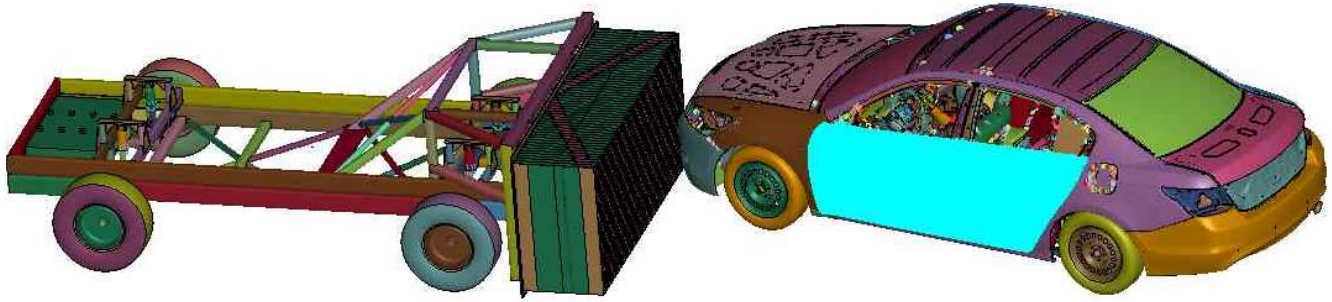
✓Ansys Cloud services

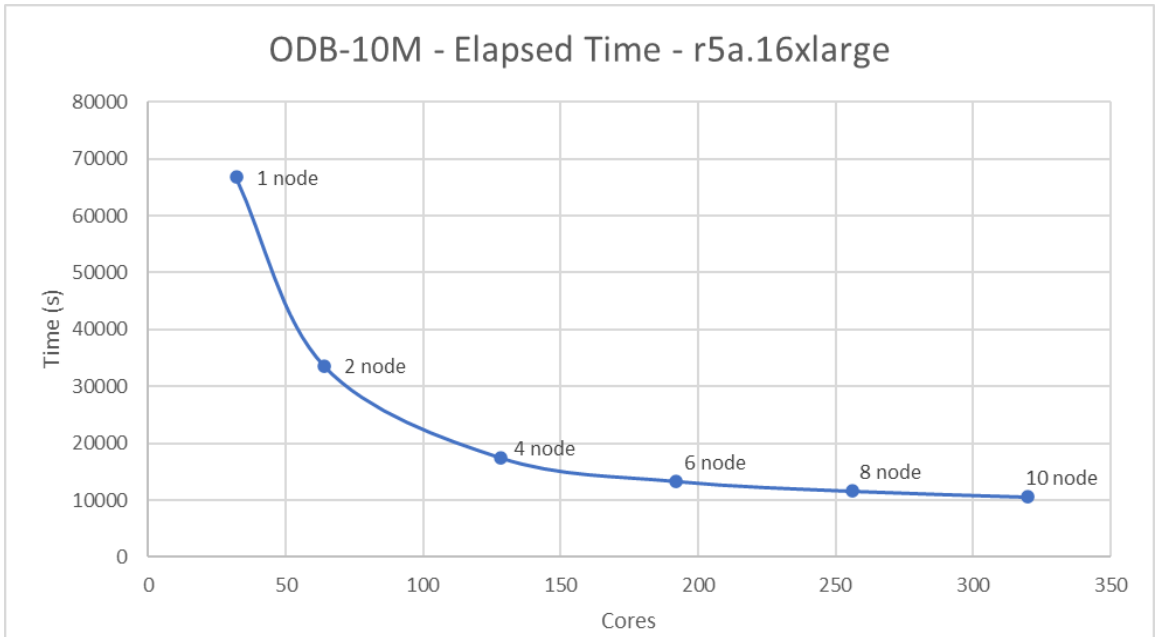
- Best practice: package container, system configuration, storage, …
- User friendliness: job ready script, application environment, …
- Best performance: affinity, connectivity, memory, …

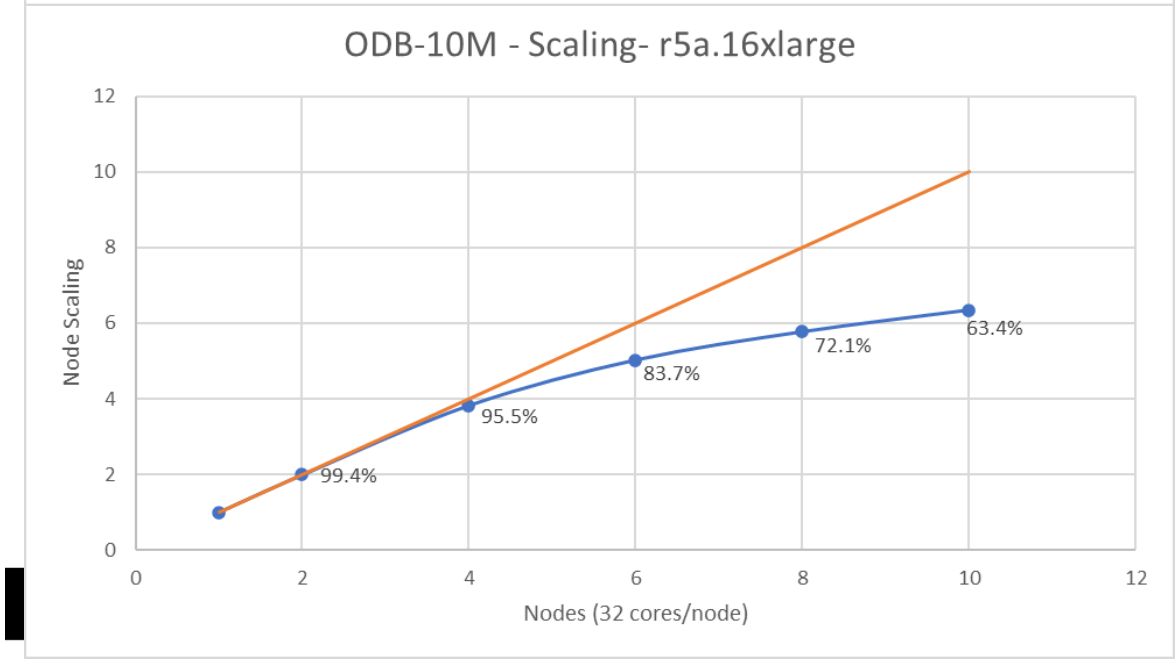
LS-DYNA AWS Cluster Benchmarking

- Benchmarking example: ODB-10M – 10 million elements
- AWS instance: r5a.16xlarge (32 core/node) - AMD EPYC 7571
- Good Performance with unideal (temporary) Cluster settings
  - EFA: **no** Fabric: **tcp** Hyperthreading: **on**
- Performance expected to improve once EFA is live on Ansys Gateway
  - EFA: **yes** Fabric: **ofi** Hyperthreading: **off**

nodes	cores	elapsed time	speedup	efficiency
1	32	18 hrs 33 min 23 sec	1	1
2	64	9 hours 20 min 12 sec	1.987474711	0.993737356
4	128	4 hrs 51 min 20 sec	3.821681922	0.955420481
6	192	3 hours 41 min 38 sec	5.023537374	0.837256229
8	256	3 hours 12 min 56 sec	5.770818936	0.721352367
10	320	2 hours 55 mins 31 sec	6.343462159	0.634346216



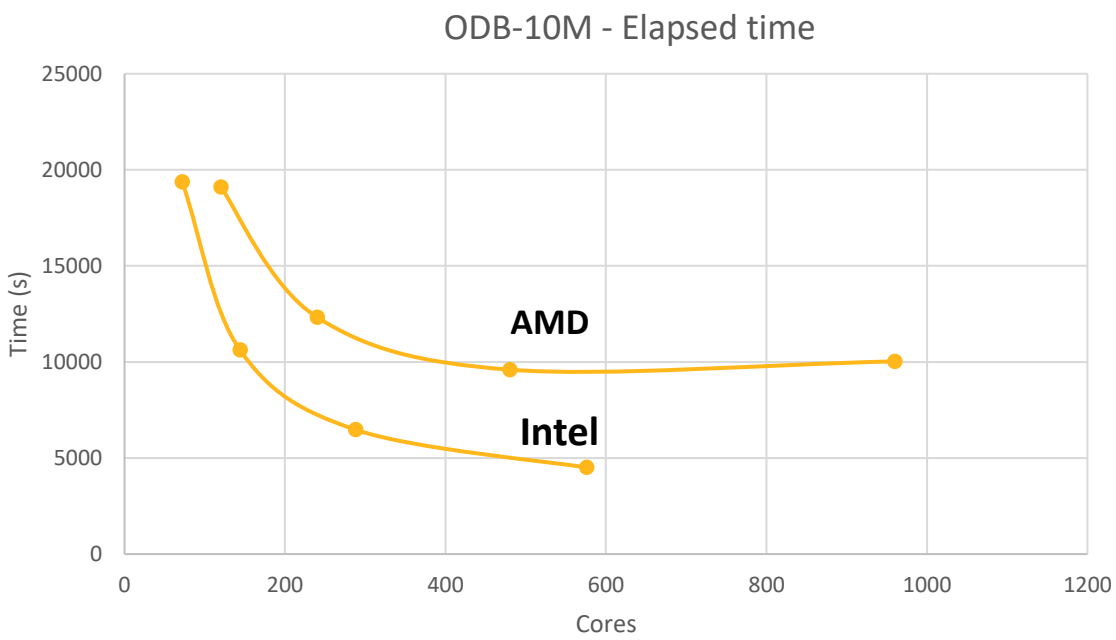


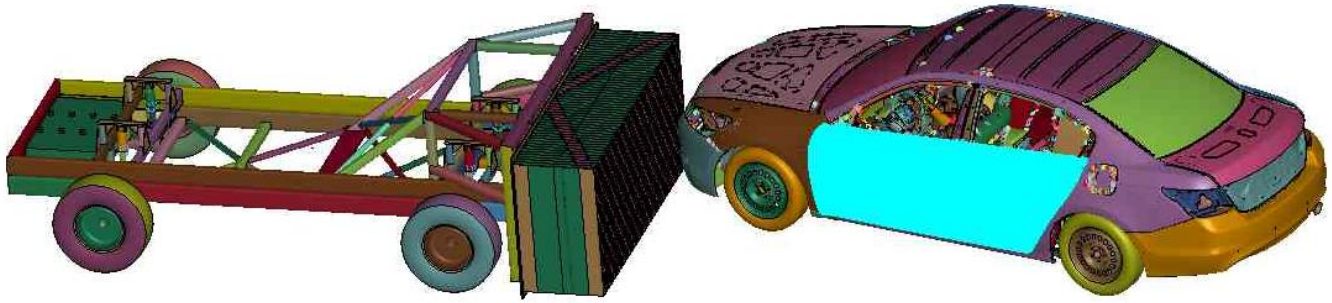


LS-DYNA Azure Cluster Benchmarking

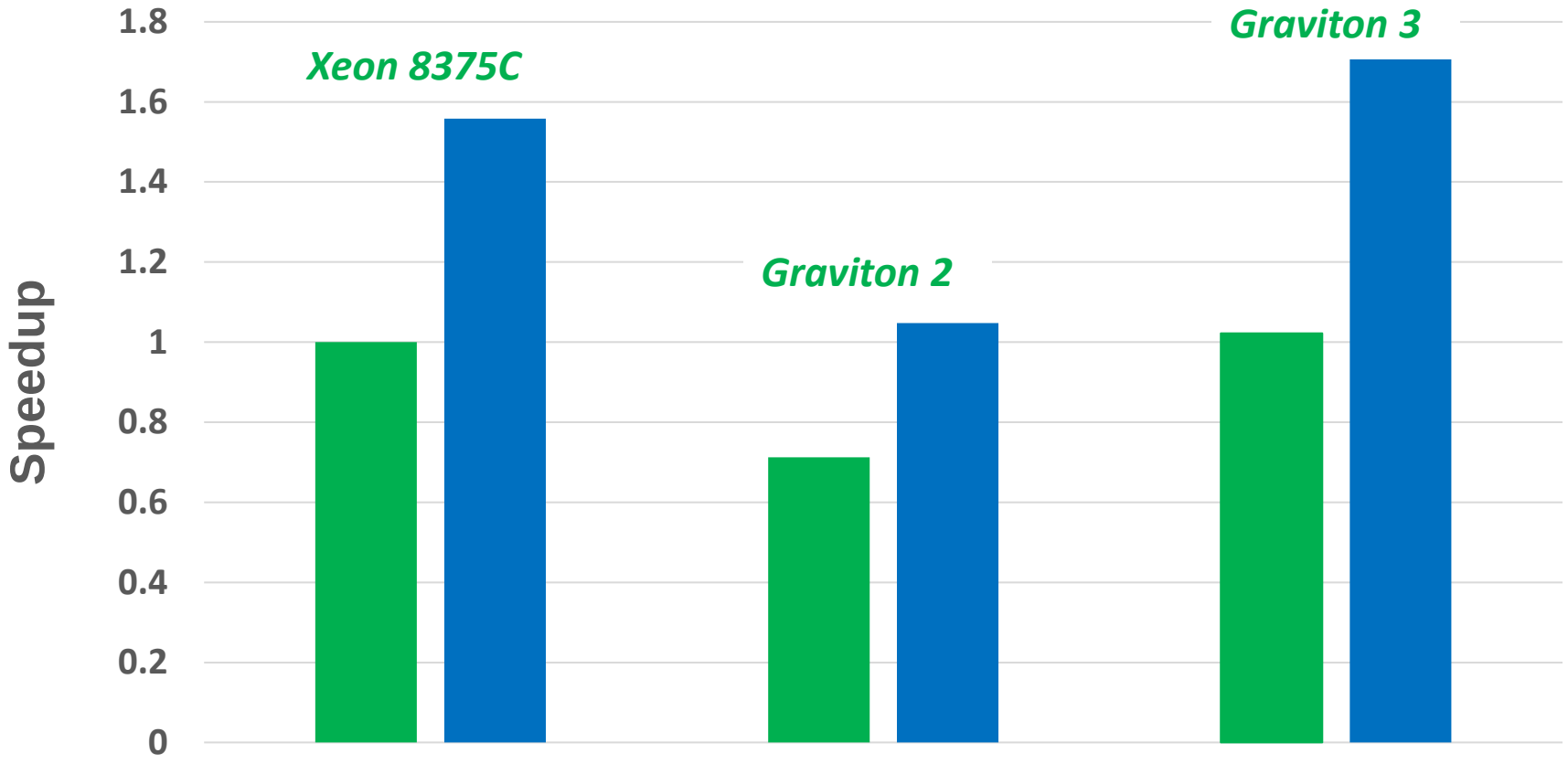
- Benchmarking example: ODB-10M – 10 million elements, 30ms
- Azure instance: Execute (24 core/node), Execute-AMD(120 core/node)
- Good Performance with unideal (temporary) Cluster settings
  - Fabric: **ofa** Hyperthreading: **off** VM, R12.1, AVX2, Intel MPI

nodes	cores	elapsed time	speedup
Intel			
3	72	19366	1
6	144	10618	1.823883971
12	288	6471	2.992736826
24	576	4505	4.298779134
AMD			
1	120	19113	1
2	240	12328	1.550373134
3	480	9599	1.991144911
4	960	10031	1.905393281



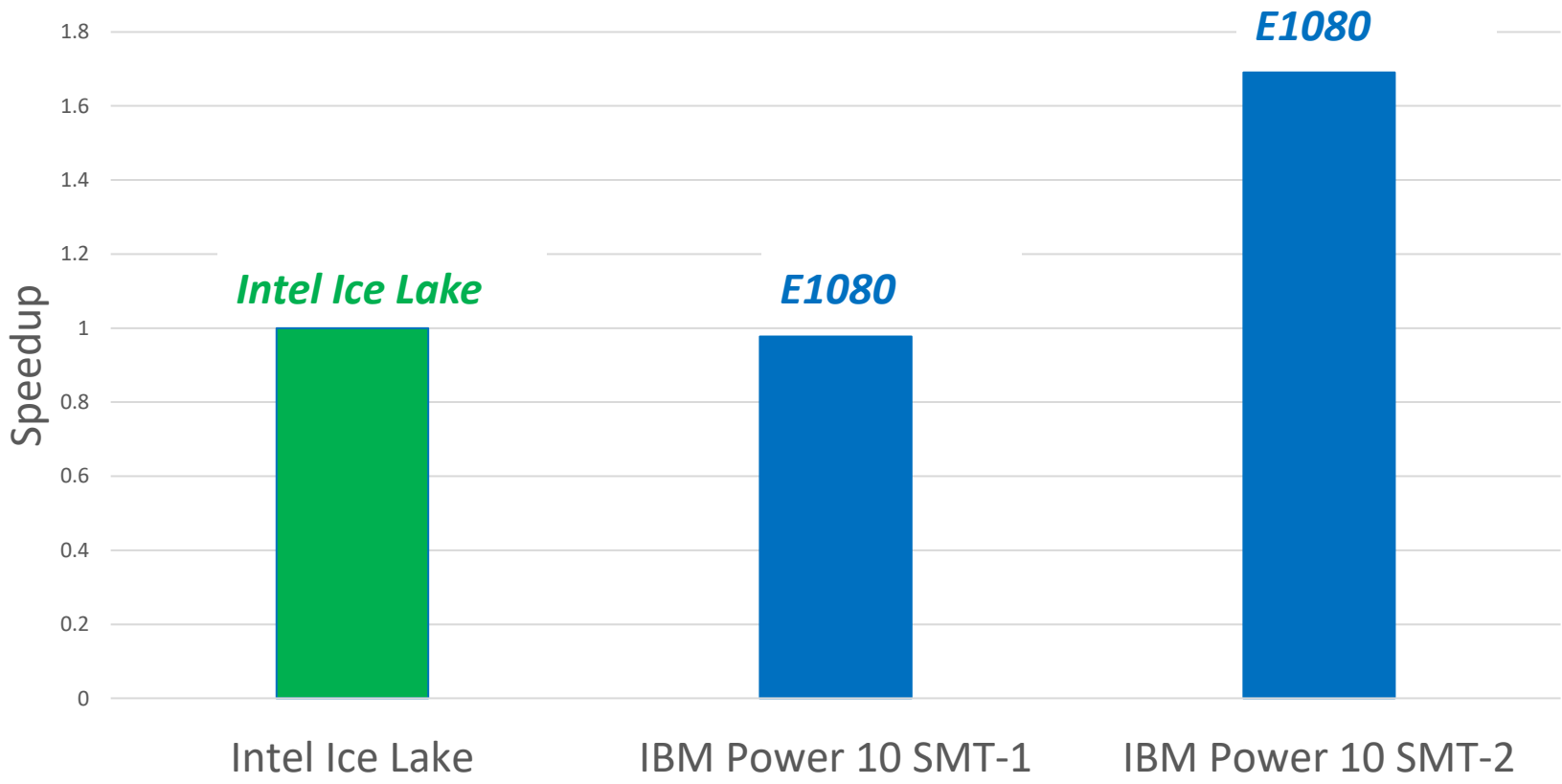


New CPU – ARM Graviton/Honda Accord (Higher is better)



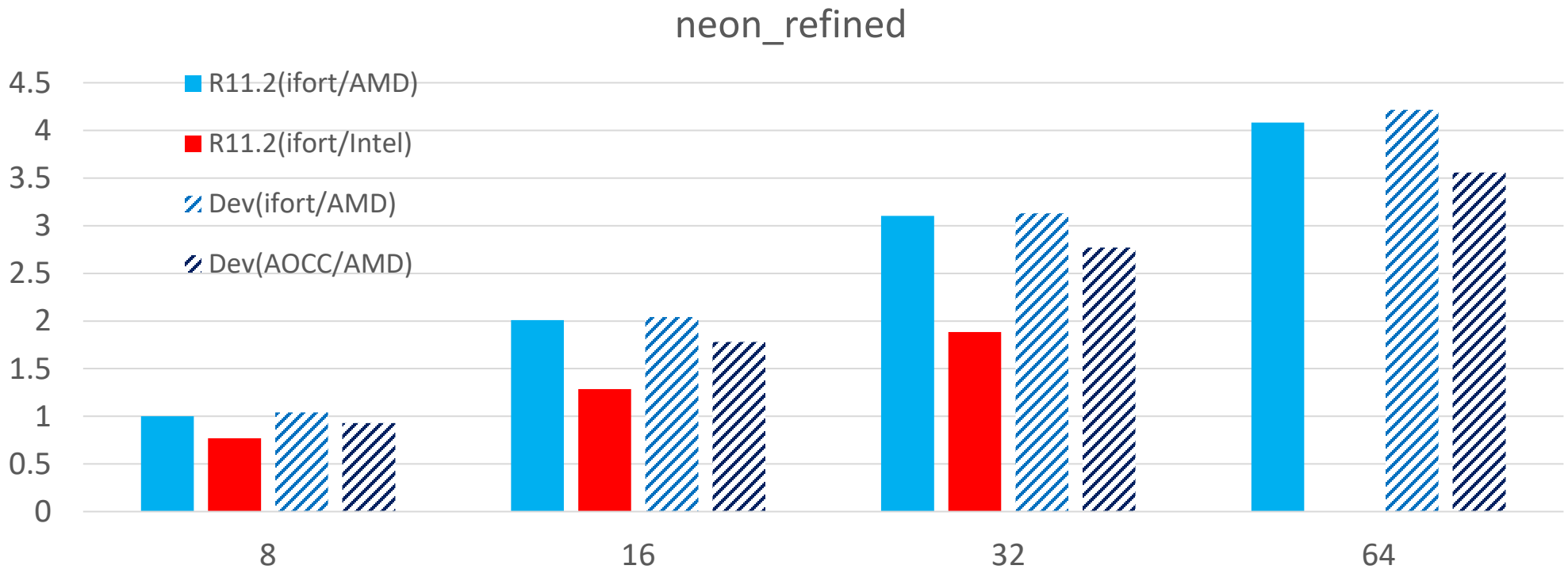
- Performance norm by 64 cores Xeon 8375C
- Single precision, 64 and 128 cores
- armclang/armflang, OpenMPI 4.0.0, released R12.
- The same binary can also execute on Ampere Altra Arm (performance comparable to Graviton 2)

New CPU – IBM Power E1080/Customer data (Higher is better)



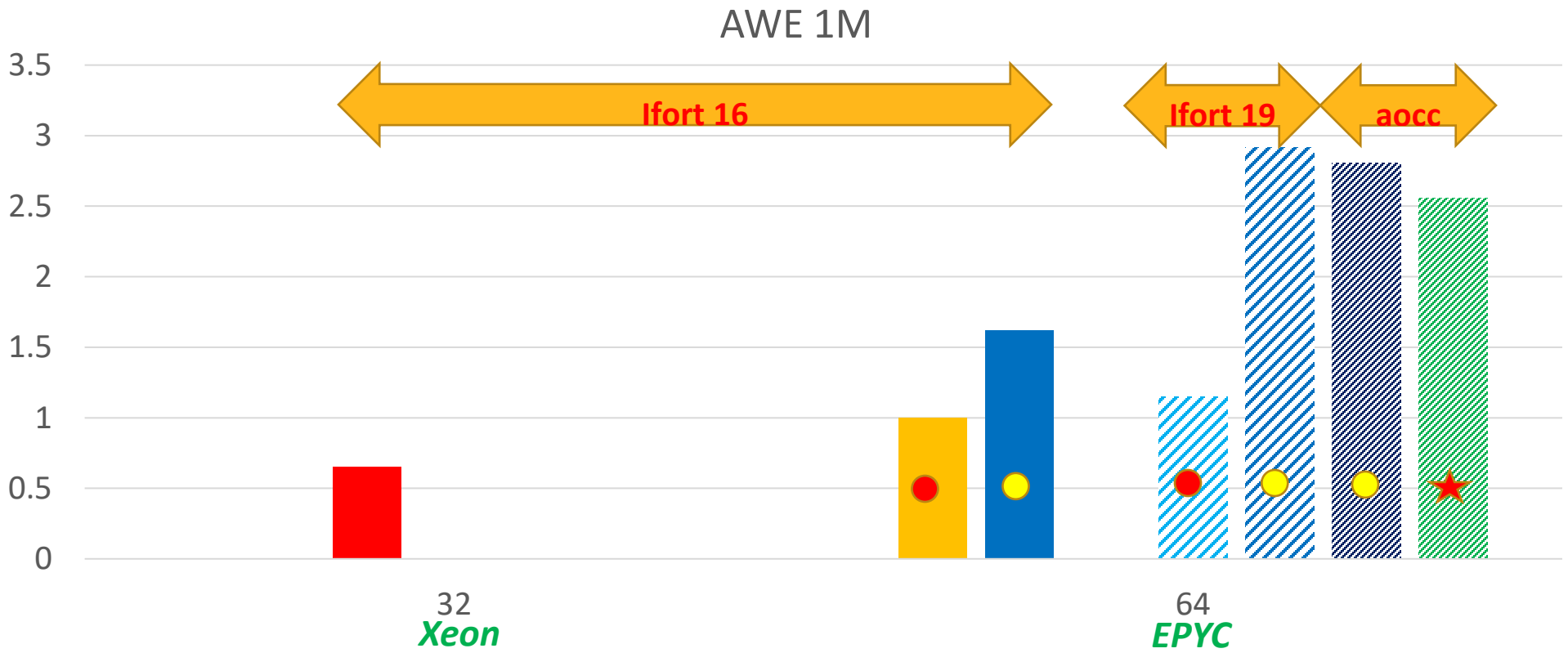
- Performance norm by 76 cores Intel Ice Lake. Results are from an initial test run on an IBM Power E1080 server compared to results from a comparably configured x86 Ice Lake server
- R12/Double precision
- Xlf 16.1.1/xlc 16.1.1: -qarch=pwr9 -qtune=pwr9 -qstrict=vectorprecision -qhot:level=2
- OpenMPI 4.0.0

New CPU – AMD Milan/Explicit (Higher is better)



- R11.2/ifort 16: same binary on different x86-64
- DEV/ifort 19, DEV/AOCC 3.1.0: different binaries on same x86-64
- No special MATH library
- Single precision, AVX2
- Intel - Intel(R) Xeon(R) Gold 6142 CPU @ 2.60GHz

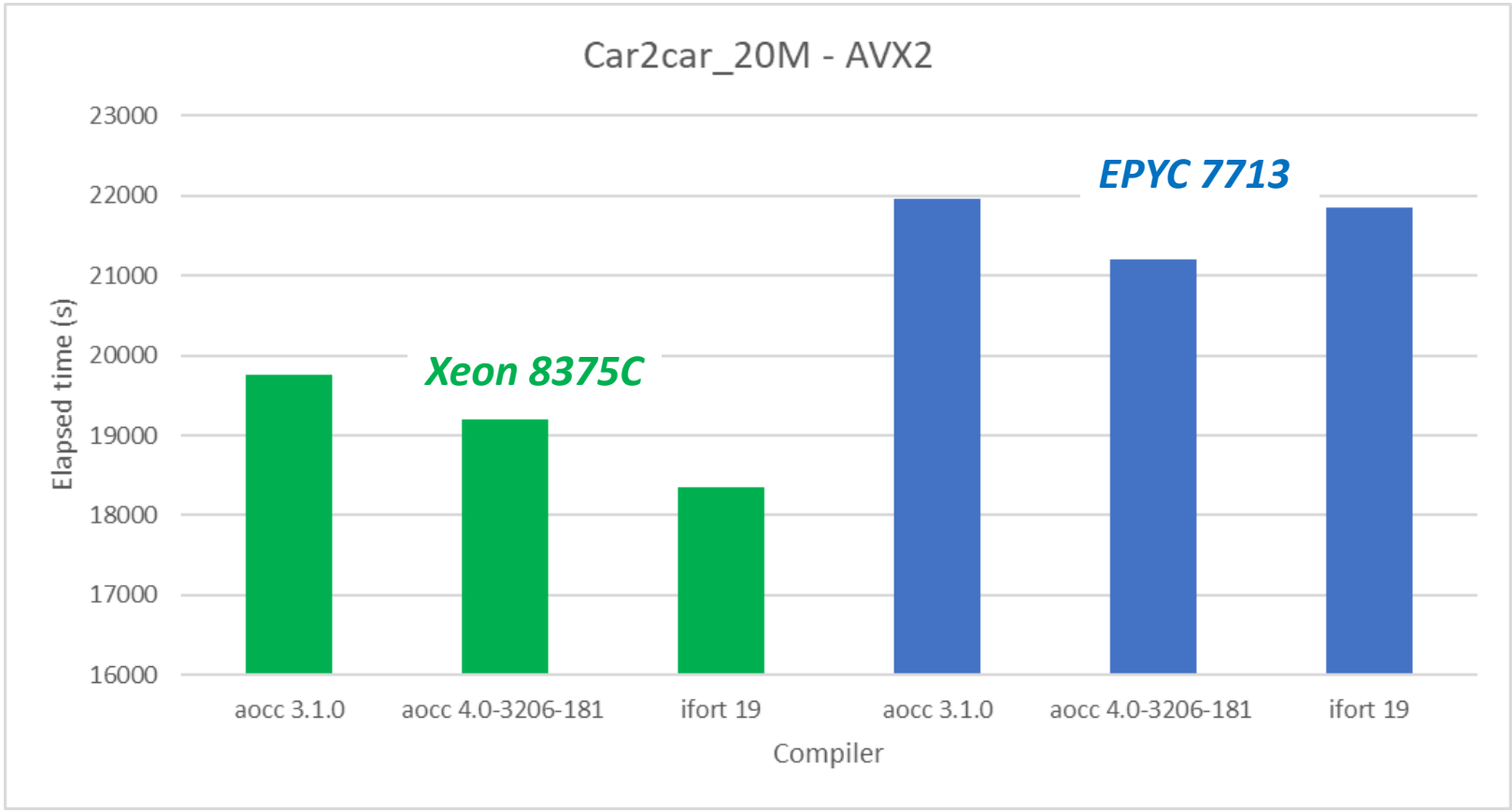
New CPU – AMD Milan/Implicit (Higher is better)



- R11.2: release version/double precision
- Dev: Current development version
- MKL: Linked with Intel MKL library
- MKL_DEBUG_CPU_TYPE=5: Special flag to use correct MATH library on AMD CPU
- AOCC: AMD AOCC 2.3 compiler
- AOCL: AMD MATH library

MKL
MKL + MKL_DEBUG_CPU_TYPE=5
AOCL

Compilers on x86-64 (Lower is better)



- R13/single precision/AVX2
- 64 MPP ranks

Summary

Arch	Release	Compiler	Instruction	MPI	Math libraries	Consistency
x86-64	< R12	ifort 16 Ifort 18	SSE2 AVX2 AVX512	Intel MPI OpenMPI 4 Platform MPI	Double: MKL,*MKL_CPU_DEBUG_TYPE=5*	Same results on (INTEL, AMD)
x86-64	> R13	ifort 19	SSE2 AVX2 AVX512	Intel MPI(OneMPI) OpenMPI 4 Platform MPI	Double: MKL,*MKL_CPU_DEBUG_TYPE=5*	Need verifications
x86-64	R13 (R14)	aocc 3.1.0 (aocc 4.0.0)	AVX2	Intel MPI(OneMPI) OpenMPI 4 Platform MPI	Double: MKL,*MKL_CPU_DEBUG_TYPE=5* AOCL	Same results on (INTEL, AMD)
ARM	R12	armflang	ARMv8.1 ARMv8.2-A SVE	OpenMPI 4 Fujitsu MPI	Double: armpl	
Power 10	R12	xlf, xlc		OpenMPI 4	Double: ESSL	

- No more update for Platform MPI (MPI 2.x). Need move forward to get active support and latest MPI features – OneMPI
- SSE2 to AVX2
- Working with CSP, hardware vendors, compiler groups, users for seamless transition to High performance computing on Cloud with optimum turnaround time
- Willing to explore new technologies and new programming scheme, vectorization, CPU+GPU, etc

Thank you!