LS-DYNA Memory Management

A review of memory management in LS-DYNA.

Klas Engstrand, DYNAmore Nordic
Anders Jonsson, DYNAmore Nordic
Webinar 2022-05-19

Agenda

- General
  - Memory size specification
  - Convert words to bytes
- Memory in explicit LS-DYNA
  - memory and memory2
  - Rough estimation of `memory=…`
  - Rough estimation of `memory2=…`
  - LS-DYNA RAM memory usage
- Memory in implicit LS-DYNA
  - Static vs. dynamic, R10.2 vs. R11.1
  - Memory required for different models – empirical investigation
  - Memory settings
  - Out-of-core capabilities
  - Competing analyses

Memory size specification

- The memory size is given on the execution line in words
  - ls-dyna i=run.key memory=120m memory2=20m
- The memory size can also be specified on *KEYWORD
  - *KEYWORD 120m memory2=20m
- Memory specified on the execution line overrides memory specified on *KEYWORD
- If the memory size requested is larger than is available on the computer node an error message will be printed

```
*** Error 70023 (OTH+23) (processor # 0)
    LS-DYNA failed to allocate 10000000000 words of memory.
    Please use the memory option on the execute line.
```

Convert words to bytes

Mwords-To-Mbytes:

- Singel Precision: Mwords*4*1000*1000/(1024*1024) -> 1.0 Megawords equal 3.81 Megabytes
- Double Precision: Mwords*8*1000*1000/(1024*1024) -> 1.0 Megawords equal 7.63 Megabytes

https://deviceanalytics.com/words-to-bytes-converter/

MEMORY IN EXPLICIT LS-DYNA

memory and memory2

- At initialization (in MPP) the memory of core 0 is set according to `memory=…`
  After initialization this memory is reset to `memory2=…`, i.e. the same static memory as all other cores.
  - For SMP, the `memory=…` is the total for the simulation

- The static memory required to start, and complete, the simulation can be found in d3hsp.

```
Memory required to begin solution (memory=    272M memory2=     18M)
         Minimum   8824K on processor    17
         Maximum     18M on processor    23
         Average     13M

Memory required to complete solution (memory=    272M memory2=     18M)
         Minimum   8824K on processor    17
         Maximum     18M on processor    23
         Average     13M
```

Note: memory2 is only for MPP

- The memory required to begin and complete the simulation (see d3hsp) are most often the same.

- The default values for `memory=…` and `memory2=…` is 20Mwords.
  (corresponds to 76 MB in Single Precision, 152 MB in Double Precision. Same default values for SMP and MPP)

memory and memory2

- If only the `memory=…` argument is given, then memory2=memory.
- Note that `memory=…` is independent from the number of used cores, for both SMP and MPP LS-DYNA.
- There is no gain to give larger `memory=…/memory2=…` execution line arguments than what LS-DYNA requires (see d3hsp). I.e. the performance and/or memory usage will not be improved.
- Due to numerical limitations 2147 Mwords (8 GB) is the maximum memory that can be requested in single precisions to avoid overflow. If that is not enough, which can happen for very large models ~>50M elements, double precision must be used at least for the initialization.

```
*** Error 10904 (KEY+904)
    Integer in *KEYWORD or command line overflows the
    machine limits; 2148
    Please try the double precision version
```

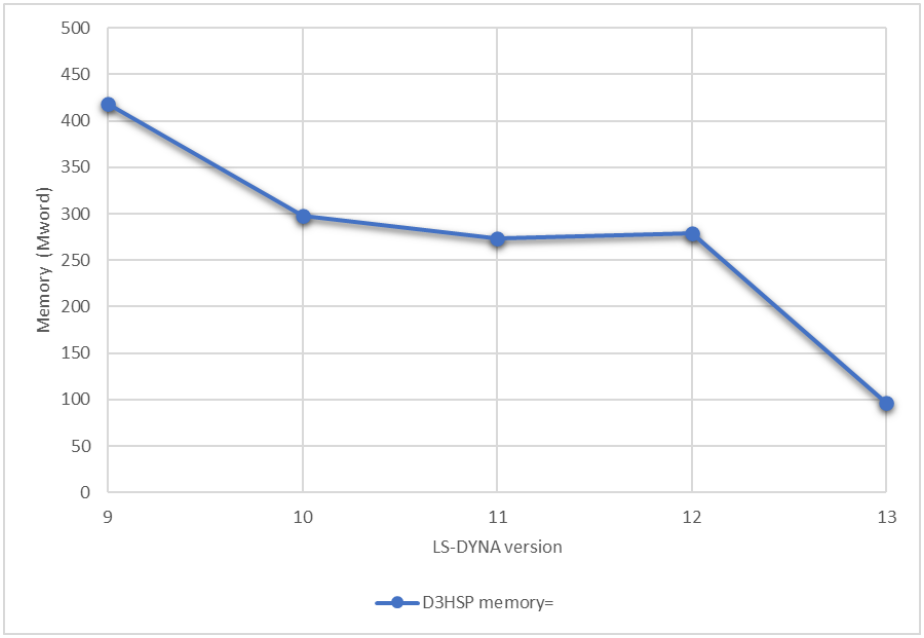
memory and memory2

- As from R9.3.1 and R11.1 (R10-branch is excluded) LS-DYNA does allocate only the memory/memory2 that is needed, i.e. regardless of the memory/memory2 execution line arguments. This functionality is implemented for both SMP and MPP LS-DYNA. Note, however, that the given execution line memory/memory2 arguments must still be large enough for LS-DYNA to begin the simulation and less than the memory available on the computer node.

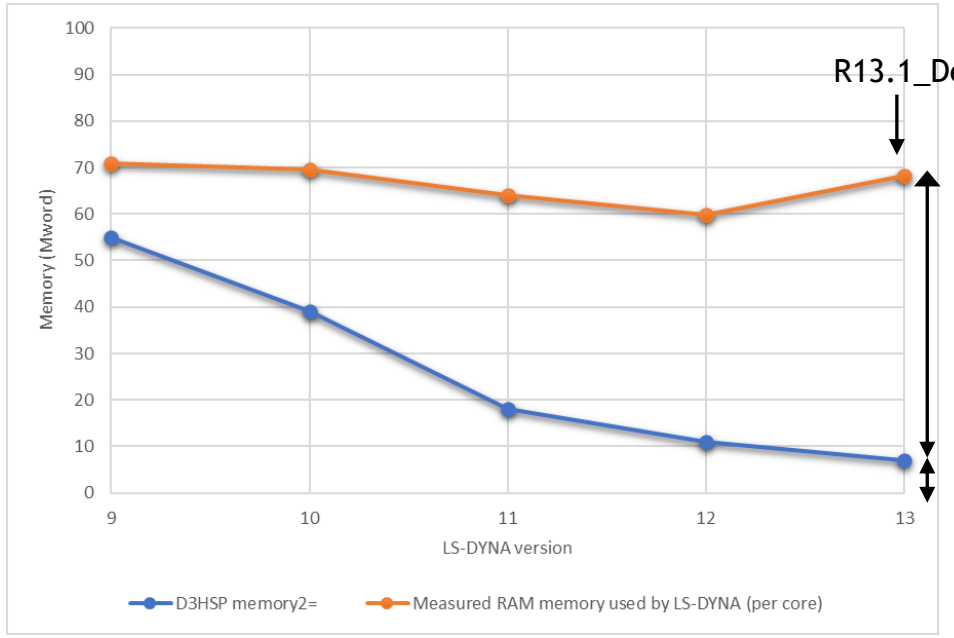
- As from R11.1 the above is also true for implicit.

memory and memory2

- The `memory=…` and `memory2=…` as required by LS-DYNA (see d3hsp) has gradually decreased with new LS-DYNA versions. This means that more features has been moved from the statically allocated memory to the dynamically allocated memory.



Memory=



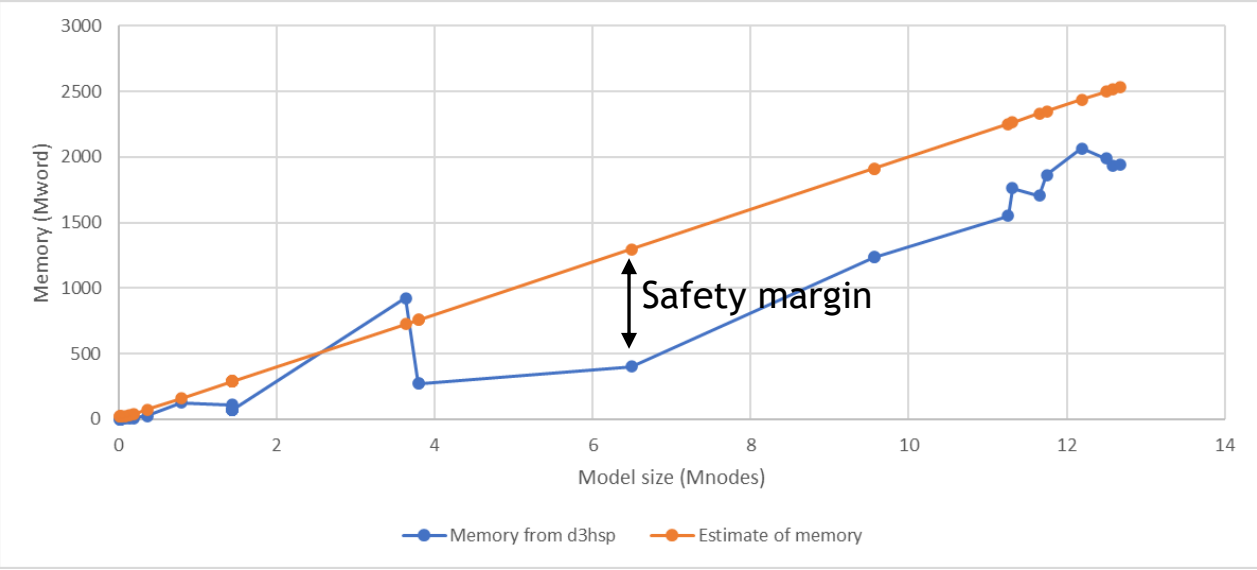
Memory2=

Note: Memory needed for the LS-DYNA binary is not included in the orange curve

Rough estimation of `memory=…`

- A reasonable, and most often conservative, `memory=…` argument can be estimated, based on the model size:
  - memory = 200 Mwords/Mnodes (MPP).
  - memory = 400 Mwords/Mnodes (SMP)

A lower limit of 20 Mwords is recommended.



Note: data collected from MPP-simulations

A safety margin is anticipated by design.

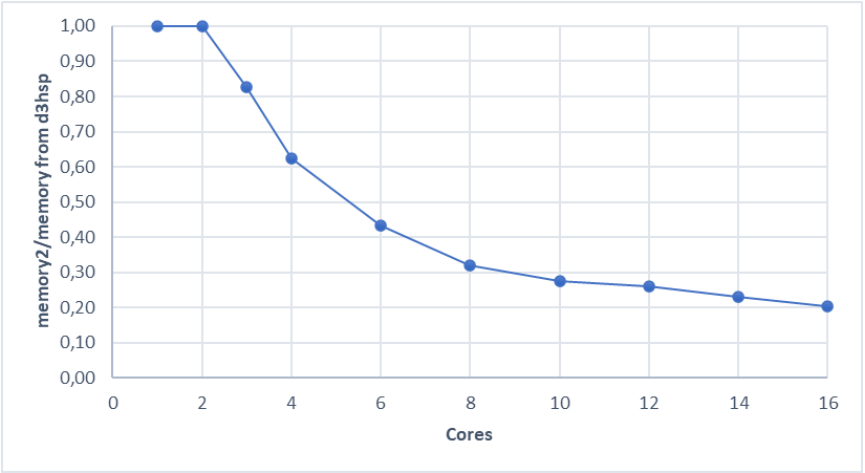
This estimation is used in LSRUN if AUTO=ON.

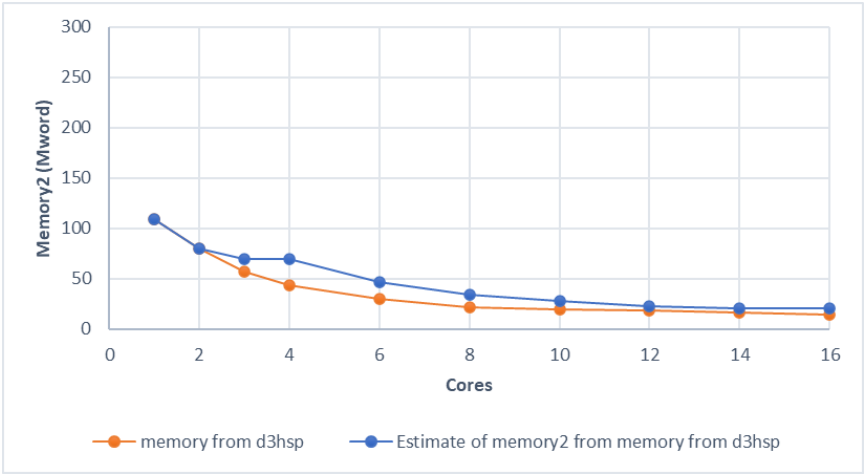
Rough estimation of the `memory2=…`

- `Memory2=…`, as required by LS-DYNA (see d3hsp), decreases with the number of cores that is used.
- A reasonable `memory2=...` argument can be estimated, based on the number of cores and the `memory=…` argument as follows:

  memory2 = MAX(MIN(4/cores;1.0);0,1)*memory

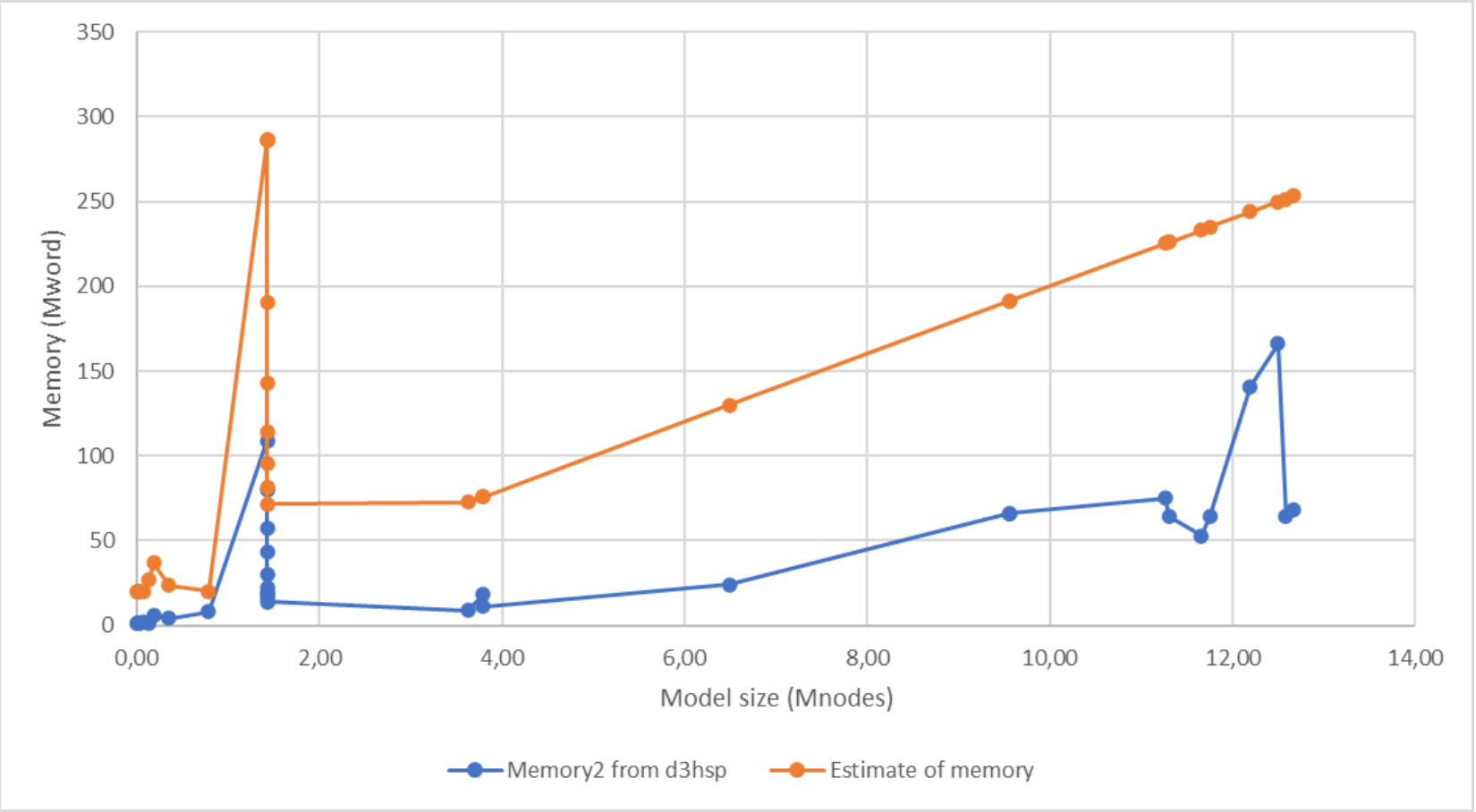
  A lower limit of 20Mwords is recommended.





Note: a safety margin is anticipated through the estimated ”memory=”- argument.

Rough estimation of the `memory2=...`

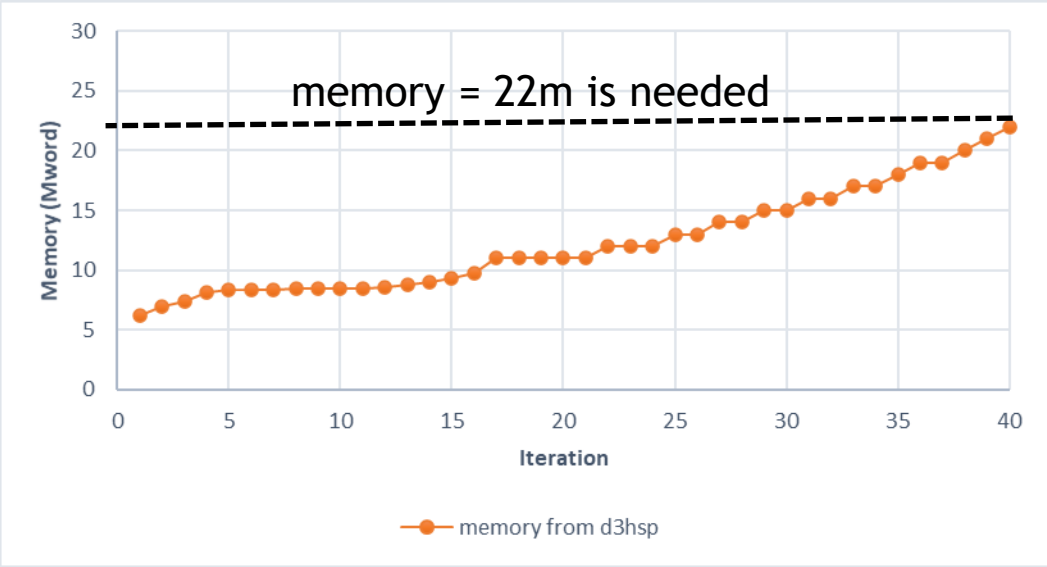


memory and memory2

- In practice, `memory=…` and `memory2=` is often set “once-and-for-all” based on experience on the site at hand so that it never become too low.

  E.g. memory=2000M and memory2=200M for very large simulations on many cores. If all this memory was to be used by LS-DYNA it would be equivalent to 37 GB RAM memory (on a 40 cores machine) for the initialization and, after that, 30 GB RAM for the analysis (Single Precision), plus RAM memory required for the LS-DYNA binary.

- Worth mentioning is that when running a simulation with adaptive re-meshing the given `memory=…` and `memory2=…` arguments must take into consideration the iterative increase of model size that will take place.



LS-DYNA memory usage

- memory/memory2 give little information about the RAM memory that is actually used by LS-DYNA during the simulation. That is, today most memory allocated by LS-DYNA is dynamically allocated rather than statically allocated, plus the additional memory needed for the LS-DYNA binary.

- Starting with LS-DYNA R10.2 and R11.1 the RAM memory available (per computer node) can be monitored through the simulation through glstat (binout format). See “Mem_avl_GB_XXX). By this feature one can calculate how much memory that is used by LS-DYNA and, more importantly, see how far the computer node is from running out of memory.

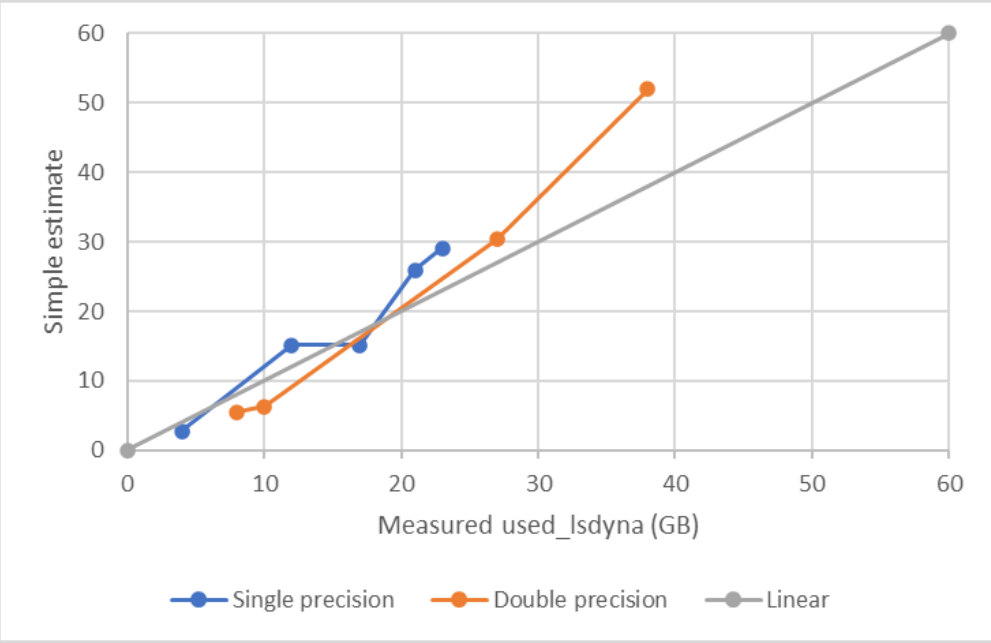
LS-DYNA memory usage

- Memory log on a Linux computer node: free -g -s 60 -t | grep --line-buffered Mem > times.txt
- In Windows 10 corresponding data is found through “Task Manager”–”Performance”-“Memory”. For a more detailed analysis use Performance Monitor.
- For pure explicit simulation the lack of RAM-memory is seldom a problem. I.e. most computer nodes have more than enough RAM memory installed. However, if implicit LS-DYNA and explicit LS-DYNA is performed on the same computer node then problem may arise.

LS-DYNA memory usage

- A rought estimate of the RAM memory needed by LS-DYNA, based on model size:
  - 4 GB RAM/Mnodes/Computer node (Single Precision)
  - 8 GB RAM/Mnodes/Computer node (Double Precision)

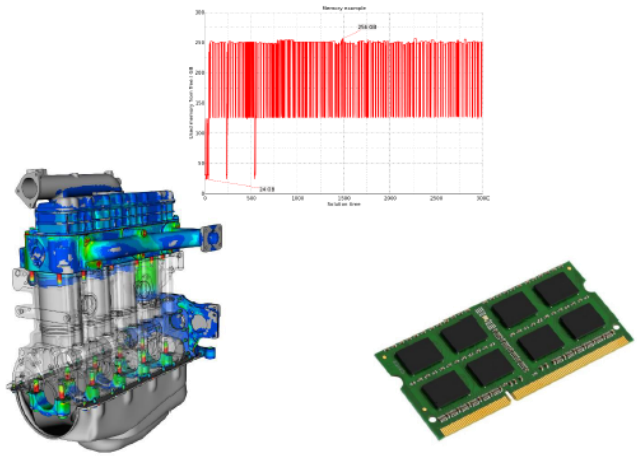
Note: the memory usage, per computer node, is problem dependent, and it also depends on the number of computer nodes used, which means that the estimate is rough.



MEMORY IN IMPLICIT LS-DYNA

Guideline for implicit memory available from files.dynamore.se > Client Area

Memory management in LS-DYNA implicit



2021-10-14 LS-DYNA / LS-PrePost

Client Area › 1_LS-DYNA_Guidelines › 2_LS-DYNA_Implicit_G...

Add notes, lists or links ...

Name
ImplicitPackage_220420.zip
LS-dyna_implicit_guideline_220420.pdf
Memory_management_in_LS-dyna_implicit_220516.pdf

Implicit vs. explicit

- An implicit analysis will in general require significantly more RAM memory than an equivalent explicit analysis
- This is due to the fundamental differences in solution scheme

Explicit	Implicit
Function of earlier configurations	Solve a non-linear equation system
$\mathbf{x}_{n+1} = \mathbf{f}(\mathbf{x}_n, \mathbf{x}_{n-1}, \ldots, t_{n+1}, t_n, \ldots)$	$\mathbf{0} = \mathbf{f}(\mathbf{x}_{n+1}, \mathbf{x}_n, \mathbf{x}_{n-1}, \ldots, t_{n+1}, t_n, \ldots)$
Equilibrium at time t	Equilibrium at time $t+\Delta t$
$\mathbf{M}\ddot{\mathbf{U}}_t = f_{e,t} - f_{i,t} - f_{d,t}$	$\mathbf{M}\ddot{\mathbf{U}}_{t+\Delta t} = f_{e,t+\Delta t} - f_{i,t+\Delta t} - f_{d,t+\Delta t}$
Conditionally stable: Δt_{crit}	Can be made unconditionally stable

Implicit vs. explicit

- In an implicit analysis, a system of non-linear equations must be solved in each time step
- This is done iteratively, by local linearization (Newton / BFGS)
  - Constraint matrix C
  - Compute and store global stiffness matrix: K
  - Factorize, K = LU
  - Factorization and storage of factors require additional memory
  - Back-substitution
  - To conclude: a lot of linear algebra requires a lot of memory for storage
  - Use of RAM is preferred for performance reasons

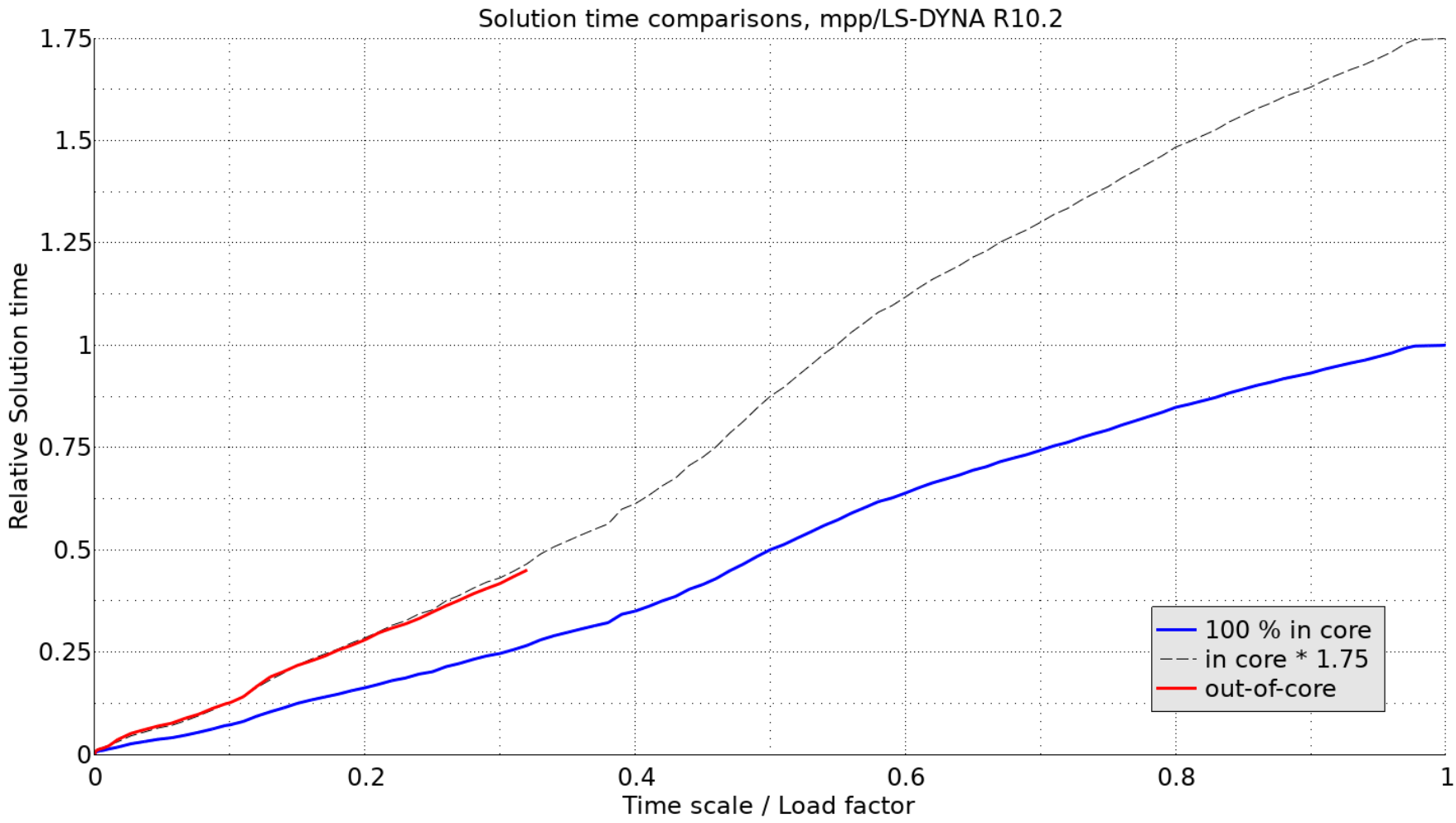
Memory vs. implicit performance

- It is possible to use disk storage instead of RAM memory
- LS-DYNA applies auto out-of-core
  - Put as much as possible in RAM
  - If extra storage is required, use disk
  - Solution can be partially out-of-core
  - Or only some MPI Threads go out-of-core
- Using modern SSD disks, going out-of-core may not be catastrophic for performance

- An all in-core solution is preferable for optimal performance
  - NOTE! If enough RAM memory is already available for an in-core solution, further increase of available memory will not increase performance.

Memory vs. implicit performance

- Using modern SSD disks, going out-of-core may not be catastrophic for performance
- An all in-core solution is preferable for optimal performance



Memory vs. implicit performance

- An all in-core solution is preferable for optimal performance
  - NOTE! If enough RAM memory is already available for an in-core solution, further increase of available memory will not increase performance.
  - For example: Say that an implicit analysis requires 100 GB of RAM, and the cluster has 192 GB available. Then, expanding the available memory to for example 512 GB of RAM will not increase performance for this particular analysis.
    - Increasing the memory makes it possible to run more analyses in parallel.
  - On the other hand, reducing the amount to for example 64 GB of RAM may severely reduce performance.

Static vs. dynamic memory in implicit, R10 and previous

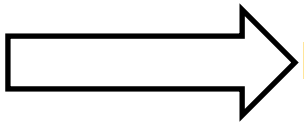
- In R10.X and previous versions, the linear solver uses mainly static memory
  - Some dynamic allocation will also take place
- This means that the user must tell LS-DYNA how much memory is available for an analysis
  - This is done by the `memory=...` command line option, for example

```
mppsub 16 mppdyna_d_... [..] i=run.key memory=201m
```

  - For mpp, `memory2=...` should *not* be used. The objective is to obtain an even memory distribution between the processes.
  - The total memory used will be nMPI × memory
  - 16 × 201 MW in the example above
- LS-DYNA will then occupy all memory that is given
  - Even if the given amount is higher than what is required

Static vs. dynamic memory in implicit, R10 and previous

- After the implicit solution iteration starts, LS-DYNA will print (in the `mes0*` files) an estimate of the required memory settings to obtain a 100 % in-core solution. One example from R10.2:



```
BEGIN implicit dynamics step      1 t= 1.0000E-04             05/05/20 09:41:12
 ============================================================
                time =  1.00000E-04
  current step size =  1.00000E-04
 expanding   memory to     9965937 d     9223239 implicit lsolvr allocation 1
 no. of global spc constraints loaded =        41352
      total no. of constraints loaded =        41352
      total no. of dofs               =        96127
      total no. of indep. dofs        =        54775

 Memory required for implicit mtx strg  :     13777558
 expanding   memory to    11693516 d     9228880 implicit matrix storage
 expanding   memory to    15053820 d     9228880 linear eqn. solver allocation 2d
 calling my_allreduce(),     locerr =   79, world  =   87
 expanding   memory to    15972543 d     9228880 linear eqn. solver allocation 2g
 min. memory for in-core stiffness factorization=          32.8 Mw on processor      0
 expanding   memory to    32804665 d     9228880 linear eqn. solver numerical phase 1
 expanding   memory to    51000000 d     9228880 linear eqn. solver numerical phase 2
```

Static vs. dynamic memory in implicit, R10 and previous

- After the implicit solution iteration starts, LS-DYNA will print (in the `mes0*` files) an estimate of the required memory settings to obtain a 100 % in-core solution.
  - All `mes0*` files must be checked!
  - The max value + abut 10 % can be used for specifying `memory=` on re-submits.
  - NOTE! An *initial* estimate is printed! Depending on how the solution evolves, more memory may be required later on, causing LS-DYNA to switch to partial out-of-core mode.
- Even though the solution may go out-of-core, some operations are still performed in-core. Up to about 90 % out-of-core may be possible.
  - It may occur that only some of the mpi threads work out of core, while other are 100 % in-core.
- For performance reasons, it is recommended to try to keep a 100 % in-core solution if possible.

Static vs. dynamic memory in implicit, R10 and previous

- In case an all in-core solution is not possible, LS-DYNA prints the Warning message `IMP+120` in the `d3hsp` and `mes0*` files, with recommended settings for `memory=`

```
*** Warning 60120 (IMP+120)
     The stiffness matrix for this job is being factorized
     in out-of-core mode (using disk files), which may severely
     decrease performance. For best performance, increase
     available memory using the command line option
     memory=nnnM, where for this job nnn is at least ...
     (adding an additional 10% is recommended).
```

  - If this occurs early in an analysis, and a fast solution is of interest, stop the analysis and re-submit with the recommended `memory=` settings from the Warning message.

- A-priori estimates of `memory=`... settings for R10
  - Very roughly: 150 GB of RAM / million FE nodes
  - Set `memory=` (number of FE nodes) `*` (18750 / ncpu)

Memory settings in R10 and previous

- Option 1:
  - Start out with the a-priori estimate, based on the number of nodes in the model.
  - Submit the job, look in all message files
    "`min. memory for in-core stiffness factorization= X Mw on processor K`"
  - If required, re-submit with adjusted `memory=` settings
  - Again, look in message and d3hsp for out-of-core warnings
  - Further memory increases may be required to stay in-core

- Option 2:
  - Assign as much memory as you can afford to LS-DYNA
  - For example, assume that 192 GB is available on a cluster node, with 24 cores. Save 15 % for system resources and other. Remains 163 GB ≈ 20 GW for LS-DYNA.
  - You want to be able to run 2 implicit jobs in parallel, each with 12 cores.
  - Then set memory = 20 GW / 24 ≈ 850 MW. Submit:

```
mppsub 12 mppdyna_d_... [..] i=run.key memory=850m
```

Static vs. dynamic memory in implicit, R11 and after

- From R11, the linear solver uses mainly dynamic memory allocation
- Static memory is required to store the keywordfile during initialization
  - On process 0 (main process) static storage corresponding to the complete keywordfile is required.
  - On processes > 0 only the partitioned parts of the model
  - Note that in case a `dynain` or `dynain.lsda` file is involved, this will require additional storage.

- From R11, both `memory=...` and `memory2=...` *can* be used,

```
mppsub 16 mppdyna_d_... [..] i=run.key memory=201m memory2=50m
```

- LS-DYNA will only use as much memory as required
  - Memory as specified by `memory=...` and `memory2=` will only be used if necessary
  - Memory that is not required will be released
  - No risk of over-occupying memory

Static vs. dynamic memory in implicit, R11 and after

- The only requirements for `memory=...` settings are
  - high enough to store input data (keywordfiles, dynain.lsda, ...)
  - and below the total available memory

- After the implicit solution begins, LS-DYNA will print (in the `mes0*` and `d3hsp` files) recommended memory settings.

```
==============================================
==            IMPLICIT USAGE ALERT          ==
==============================================
== Memory Management for Implicit has changed ==
== after R10.  Please use:                   ==
==       memory=      YM memory2=    ZM       ==
==============================================
```

  - This is just an information message and can in practice be ignored.

Static vs. dynamic memory in implicit, R11 and after

- The use of dynamic memory can be controlled by *RDCMEM* on *CONTROL_IMPLICIT_SOLVER.

From the keyword manual:

> [...] This factor caps the amount of dynamic memory requested for linear algebra applications to RDCMEM times the amount that the operating system declares available. 0.85 seems to work well for most systems. If you are using a workstation and starting up other applications while running LS-DYNA, you may need to use a number like 0.50.

- Limits amount of RAM memory used by a single LS-DYNA simulation
  - Possible to use for resource management,
  - but a bit complicated ...

Static vs. dynamic memory in implicit, R11 and after

- Using *RDCMEM* for resource management
  - For example, assume that one cluster node, with 24 cores, is available.
  - You want to be able to run 2 implicit jobs of equal size in parallel.
  - Save 15 % for system resources and other. Remains 85 %.
  - Set *RDCMEM* = 0.425 on `*CONTROL_IMPLICIT_SOLVER` in the keywordfile of the first submitted job
  - For the 2nd submitted job, the system will report (at lesat) 57.5 % memory as available.
  - Set *RDCMEM* = 0.425/0.575 ≈ 0.74 on `*CONTROL_IMPLICIT_SOLVER` in the keywordfile of the second submitted job
  - Note that the *RDCMEM* setting will depend on the order in which the jobs are submitted, since it is measured relative to the currently available memory reported by the Operating System

Static vs. dynamic memory in implicit, R13 and after

- From R13, the use of dynamic memory can be controlled also by *ABSMEM* on `*CONTROL_IMPLICIT_SOLVER`.

From the keyword manual:

> Absolute upper bound for the dynamic memory allocated for factorization. The allocated memory will be bounded above by the min(RDCMEM× NWORDS, ABSMEM) where NWORDS is the number of available words as determined by the operating system. If the predicted amount of required memory is less than this value, then less memory than this bound may be allocated

- Limits amount of RAM memory used by a single LS-DYNA simulation
  - For example, assume that one cluster node, with 192 GB (24 GW) is available.
  - Save 15 % for system resources and other. Remains 85 % for LS-DYNA.
  - You want to be able to run 2 implicit jobs in parallel.
  - Set *ABSMEM* = 0.425 * 24E9 ≈ 1.E10 on `*CONTROL_IMPLICIT_SOLVER` in the keywordfile of each simulation, respectively.

Static vs. dynamic memory in implicit, R11 and after

- LS-DYNA R11 and R12 will silently go out-of-core if available RAM is insufficient.
  - No Warning message printed
  - Look for the disk.* files in the run directory. If zero-size, then simulation is in-core

```
-rw-rw-r-- 1 andersebj andersebj        0 May 20 09:50 disk.LU.063.0008
-rw-rw-r-- 1 andersebj andersebj        0 May 20 09:50 disk.LU.063.0007
.
.
.
-rw-rw-r-- 1 andersebj andersebj        0 May 20 09:50 disk.LU.063.0001
-rw-rw-r-- 1 andersebj andersebj        0 May 20 09:50 disk.LU.063.0000
```

Static vs. dynamic memory in implicit, R11 and after

- LS-DYNA R11 and R12 will silently go out-of-core if available RAM is insufficient.
  - No Warning message printed
  - Look for the disk.* files in the run directory. If non-zero, then simulation is out-of-core

```
-rw-rw-r-- 1 andersebj andersebj          0 Oct 12 16:07 disk.RS.063.0008
.
.
-rw-rw-r-- 1 andersebj andersebj          0 Oct 12 16:07 disk.RS.063.0005
-rw-rw-r-- 1 andersebj andersebj 2192834560 Oct 12 16:07 disk.LU.063.0000
-rw-rw-r-- 1 andersebj andersebj 1759510528 Oct 12 16:07 disk.LU.063.0014
-rw-rw-r-- 1 andersebj andersebj 1744568320 Oct 12 16:07 disk.LU.063.0009
-rw-rw-r-- 1 andersebj andersebj 1441689600 Oct 12 16:07 disk.LU.063.0013
```

- LS-DYNA R13 (and later) will print the Warning message `IMP+120`:

```
*** Warning 60120 (IMP+120)
    The stiffness matrix for this job is being factorized
    in out-of-core mode (using disk files), which may severely
    decrease performance.
    Memory being used for storing the factorization is XXXM.
    For best performance, increase available memory using
    options RDCMEM and ABSMEM on *CONTROL_IMPLICIT_SOLVER if possible.
```

Memory settings in R11 and after

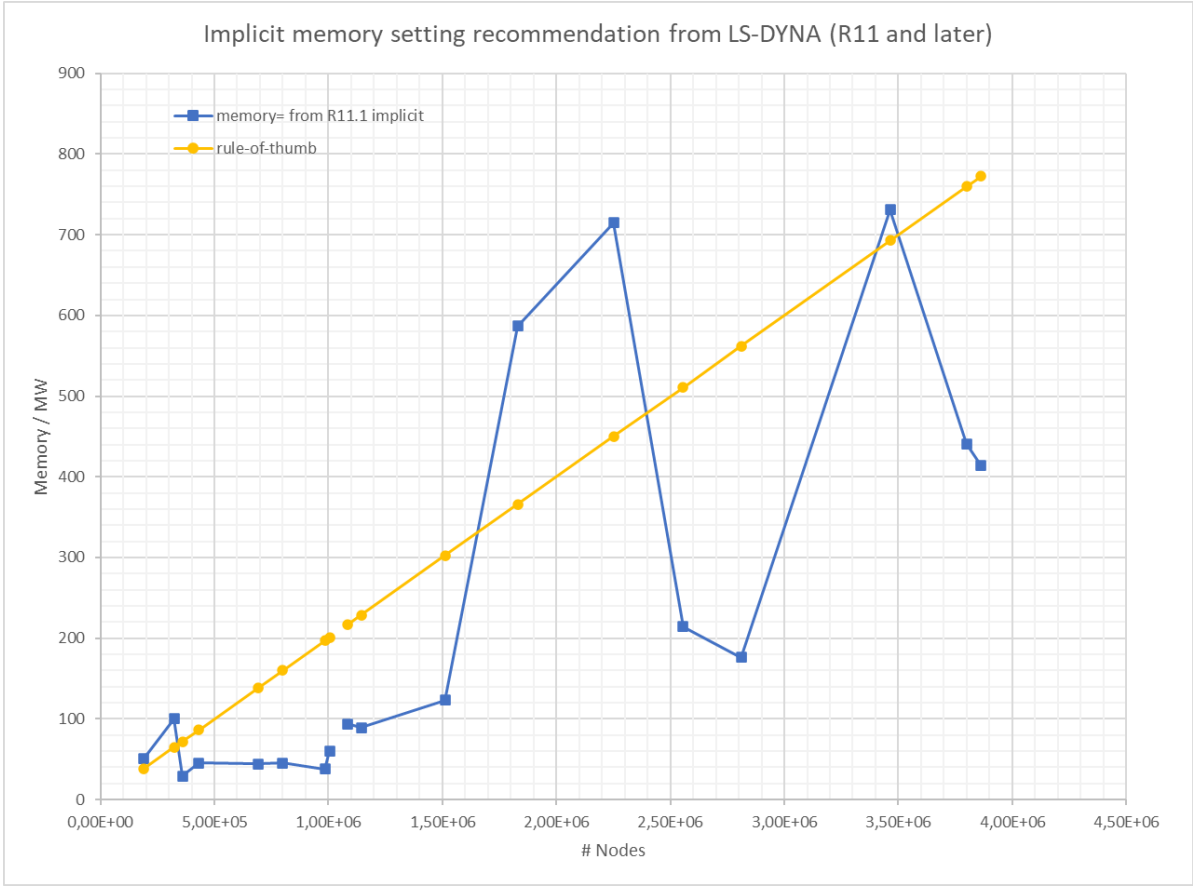
- Option 1:
  - Use the `memory=...` setting as a signal to the queueing system, with regards to resource allocation.
  - Similar to R10, start out with an a-priori estimate (roughly 105 GB of RAM per 1.E6 FE nodes)

- Option 2:
  - Set `memory=` [85 % of the available RAM].
  - By this, the simulation will “always” start

- **NOTE**: Once the simulation starts, the amount of memory used is limited by *RDCMEM* on `*CONTROL_IMPLICIT_SOLVER` (and from R13 also *ABSMEM*)

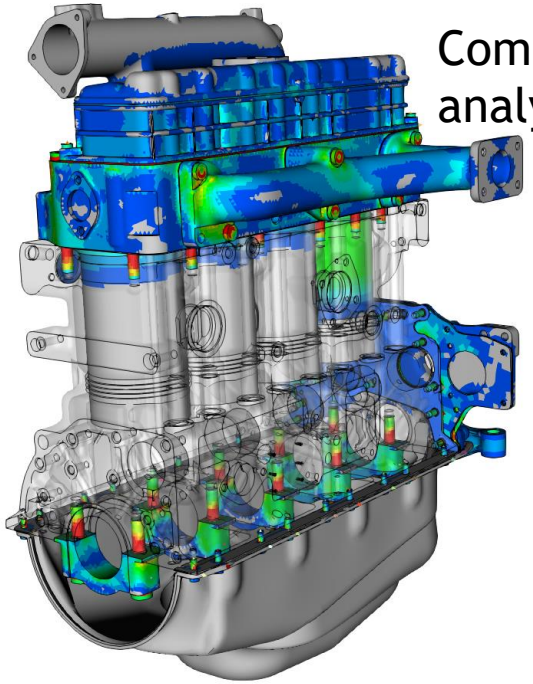
Memory settings in R11 and after

- Memory-setting must be high enough to perform initialization
- From R11, LS-DYNA prints a recommendation for `memory=...` and `memory2= ...` at the beginning of the implicit analysis
  - For the studied cases, this recommendations seem quite problem-dependent (blue line)
- The rule-of-thumb for explicit may be useful as a starting point
  - Yellow line in figure
  - Set `memory=`
    [200 * (number of FE nodes) / 1.E6] MW
  - `memory2=` MAX(MIN(4/nMPI;1.0); 0.1) * `memory`

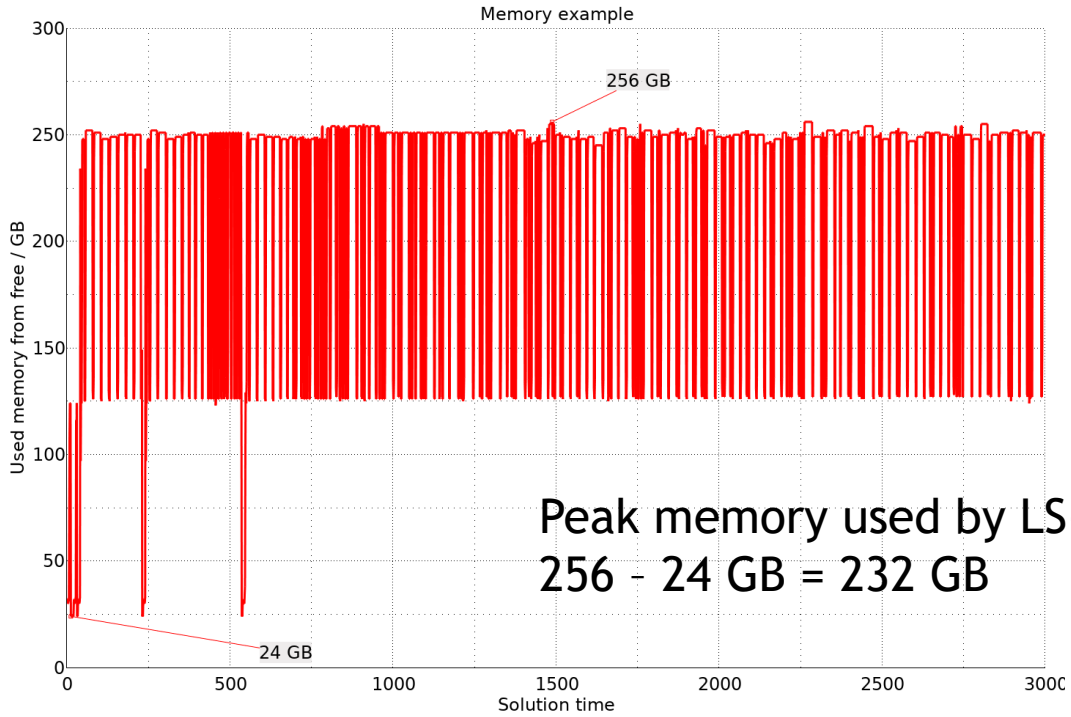


Memory required by different models

- Use system monitoring tools to obtain the amount of memory truly used by LS-DYNA
  - For example, the Performance Monitor in Windows, or the `free` - command in Linux



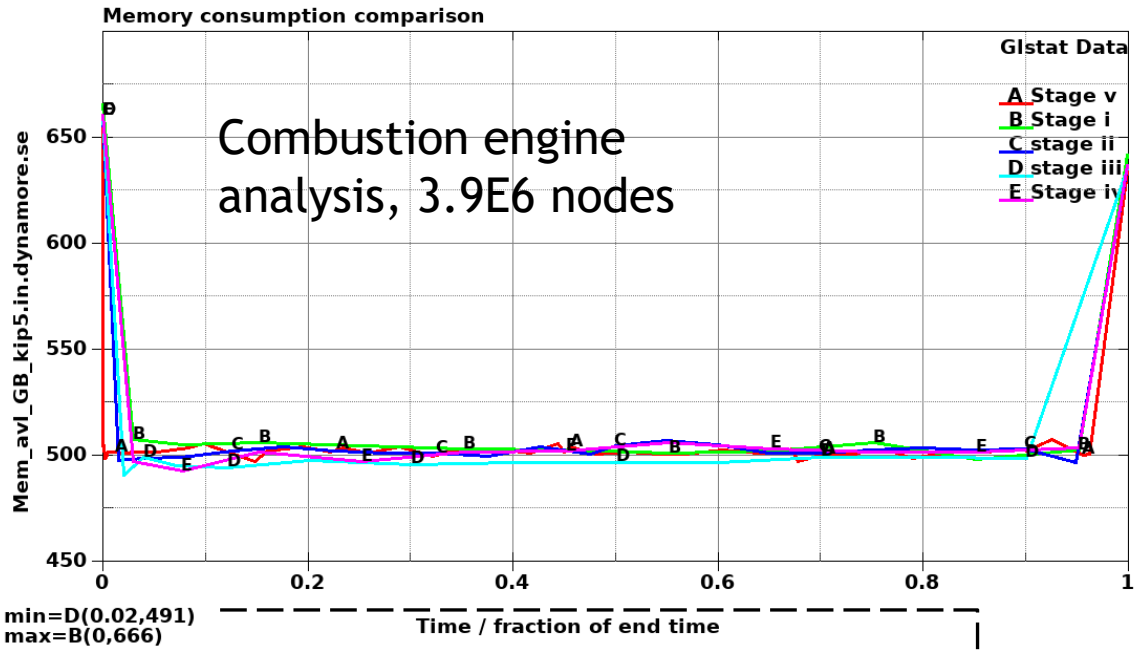
Combustion engine analysis, 3.9E6 nodes



Peak memory used by LS-DYNA = 256 – 24 GB = 232 GB

Memory required by different models

- Memory from Linux cluster by the `glstat` file
  - From R11, the `glstat` results in the `binout*` - files contain information regarding available memory: `Mem_avl_GB_`<hostname>
  - Can be plotted in LS-PrePost
  - The baseline available memory must be known (run `free` when cluster is “empty”)



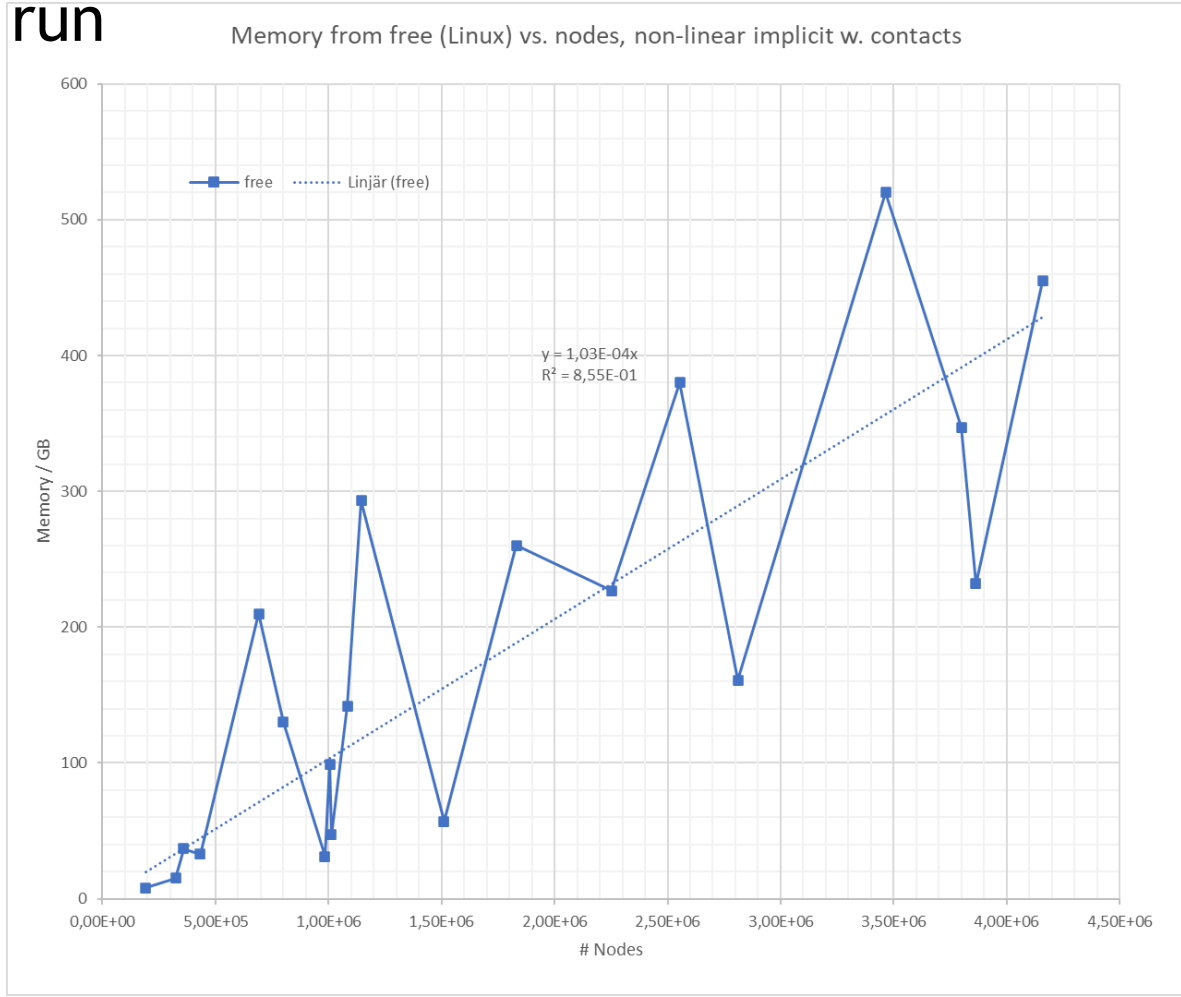
Peak memory used by LS-DYNA = 720 - 491 GB = 229 GB

min. value from glstat in binout

available when cluster is empty, from free

Memory required by different models

- Memory extracted from Linux cluster by the `free` command
- Models between 0.2E6 to 4E6 nodes were run
- Empirical investigation
  - Highly problem dependent memory requirement
  - Memory depends on material models, contact definitions, element types etc.
  - A very rough estimate is that about 105 GB of RAM per 1E6 nodes is required, for versions R11 and after



Conflicts

- When several users share a single cluster node for implicit jobs, some kind of agreement is required to avoid conflicts over memory
  - In R10, it's possibly to occupy (almost) all RAM by setting "too high" `memory=...`
  - In R11 and after, the default setting allows LS-DYNA implicit to allocate up to 85 % of available RAM for the linear algebra
- R10, use `memory=...` to limit amount of RAM
- R11 and after, use *`RDCMEM`*, and *`ABSMEM`* from R13
- Over-occupying RAM on a system may lead to severe problems, if not enough memory is left for the Operating System, or the MPI
- LS-DYNA cannot resolve the potential conflicts
  - Agreement between users, set rules for how much memory can be used
  - Resource allocation by submit script / queueing system, for example refuse start if `memory=` is too high, read through keyword file and check that `RDCMEM` is correct ...

Conflicts

- Assume that a single cluster node with for example 16 cores and 64 GB of RAM is available.
- An implicit simulation that would require about 57 GB is run on 8 cores.
- An explicit simulation is run at the same time on the remaining 8 cores.
- Situations like this may be very hard to manage automatically
- Different types of problems may arise
  - LS-DYNA runs out of memory, and the implicit job stops

```
*** Fatal Error - Implicit using F95 memory allocation
    could not allocate minimum amount of memory for
    this factorization.
    minimum amount (Mwords) =          287
    [...]
```

  - The MPI runs out of memory, and the job stops
- Reducing *RDCMEM* on *CONTORL_IMPLICIT_SOLVER (or from R13, using *ABSMEM*) may help

Conclusions

- For explicit analyses, available RAM memory is seldom a limiting factor
  - Very large models may encounter problems during initialization
- From R9.3.1 and R11.1, LS-DYNA will only allocate the static memory needed
  - No risk of over-allocating static memory
  - From R11.1, this is true also for implicit
- Implicit analyses will in general require much more memory than explicit
  - Try to keep the implicit solution in-core for best performance
  - In R10 (static memory allocation) the user must set a reasonable value for `memory=...`
  - From R11.1, basically the same static memory as in explicit. Additional memory will by allocated dynamically
- Avoid analyses competing for the same memory resources
  - Use *RDCMEM* in R11 (*ABSMEM* from R13) to limit the amount of RAM that can be occupied by LS-DYNA implicit

Thank you!

DYNA MORE

Your LS-DYNA distributor and more