eCSE05-05: Open source exascale multi-scale framework for the UK solid mechanics community

A. Shterenlikht[1], L. Margetts[2], L. Cebamanos[3]

[1] Mech Eng Dept, The University of Bristol, Bristol BS8 1TR, UK, mexas@bris.ac.uk

[2] School of Mechanical, Aero and Civil Engineering, The University of Manchester, Manchester M13 9PL, UK, Lee.Margetts@manchester.ac.uk

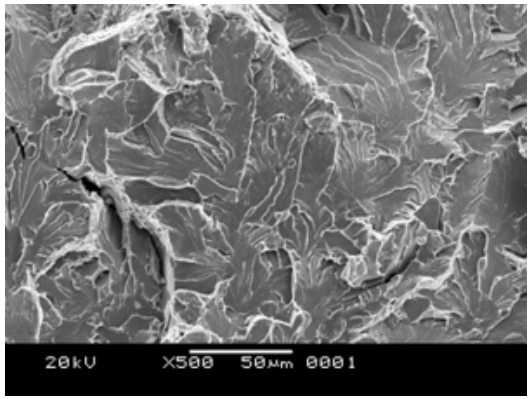
[3] Edinburgh Parallel Computing Centre (EPCC), The University of Edinburgh, King's Buildings, Edinburgh EH9 3FD, UK, l.cebamanos@epcc.ed.ac.uk

29-MAR-2017

Table of Contents

Table of Contents

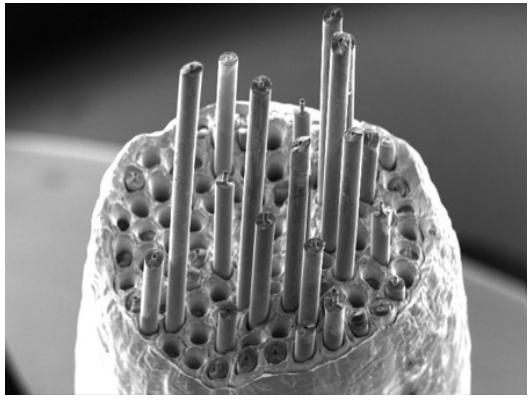
Fracture in heterogeneous materials



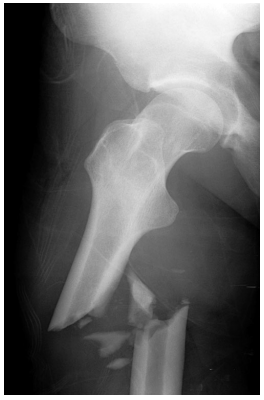
polycrystal cleavage

reinforced concrete

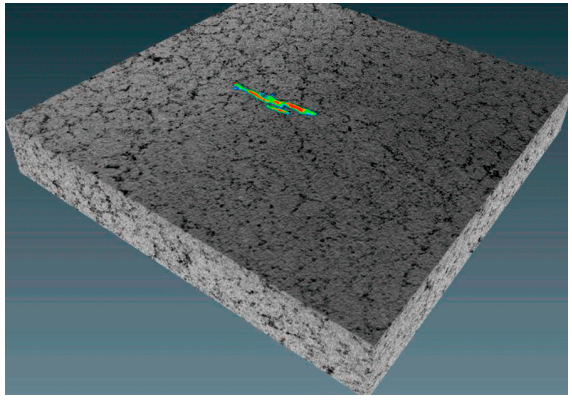
CFRP



metal matrix



bone



graphite

- All real materials are heterogeneous
- Multiple fracture and damage processes happen at different time and length scales $\rightarrow$ need multi-scale framework

Fracture: CA + FE = CAFE multi-scale model ▸ CGPACK

- Structured grids - cellular automata (CA), unstructured grids - finite elements (FE)
- CA (microstructure) + FE (continuum mechanics) = CAFE
- Transgranular cleavage - fracture stress or strain criteria
- FE $\rightarrow$ CA (localisation) - stress, strain fields
- CA $\rightarrow$ FE (homogenisation) - damage variables

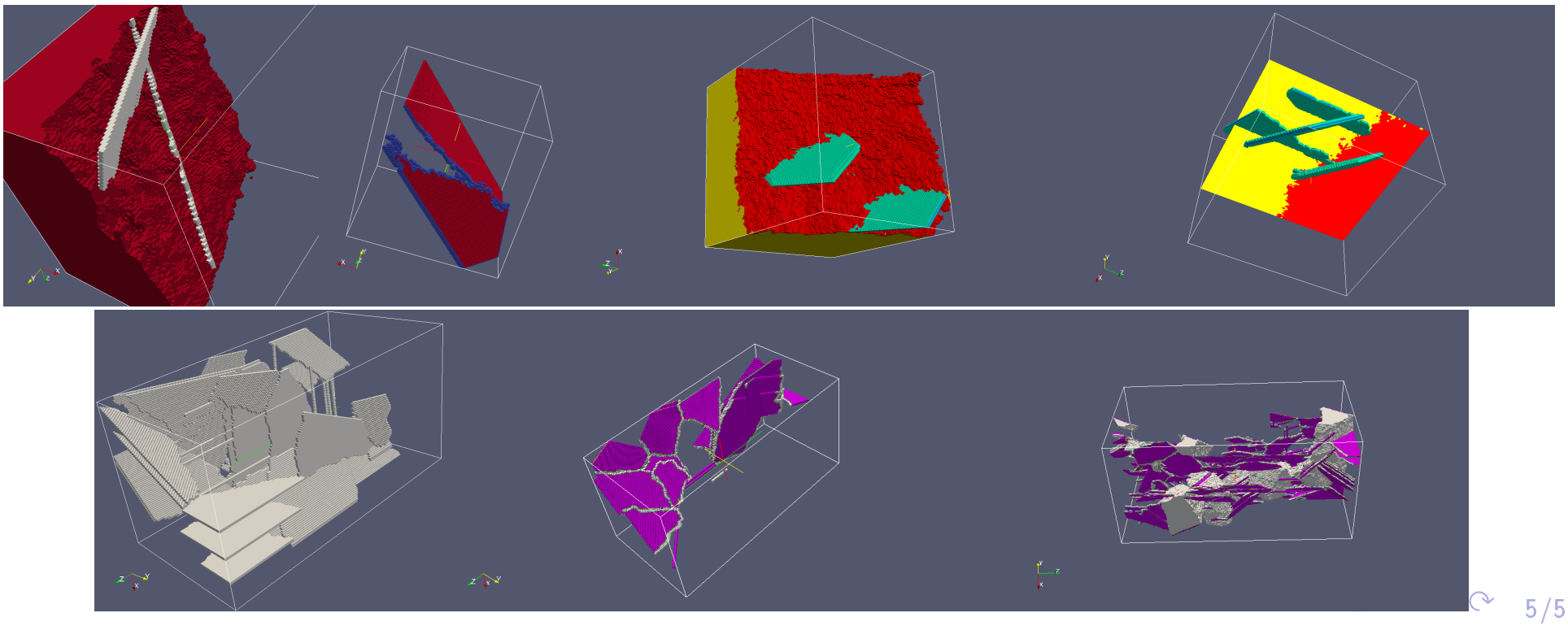
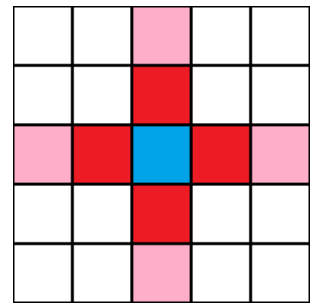
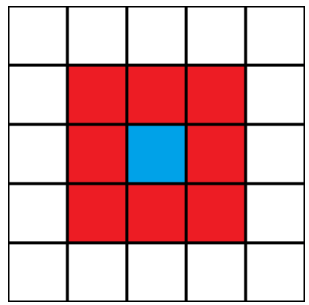


Table of Contents

Cellular automata (CA) basics

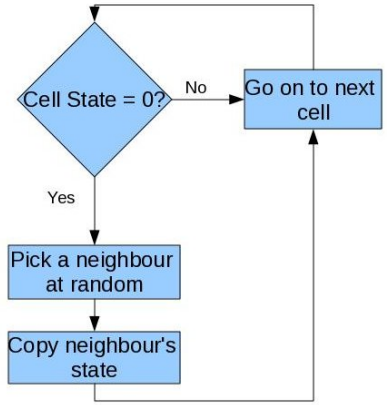
- discrete space, discrete time, discrete states - fully digital framework, structured grids
- finite or infinite space
- finite space: fixed or self-similar boundaries
- cell neighbourhood, e.g. von Neumann's:  ,

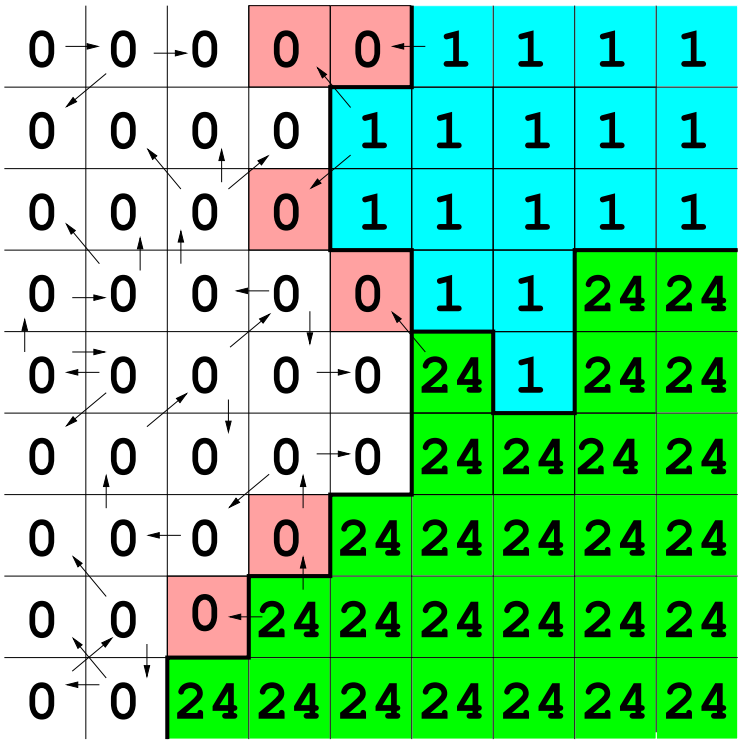
  or Moore's: 
- iterative process
- state of a cells at next iteration is a function of the state of this cells and of the states of its neighbourhood cells at the current iteration

Wikipedia

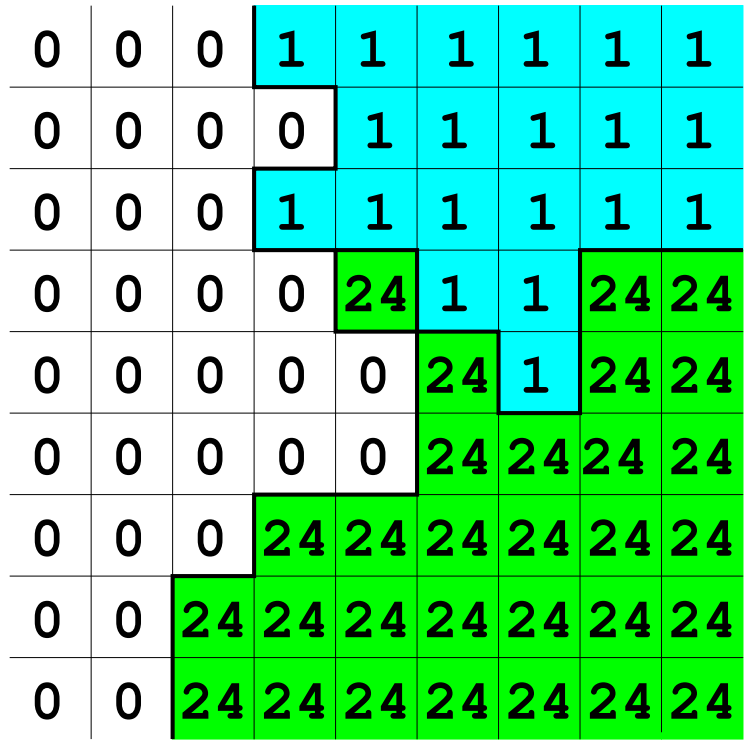
Primitive 3D solidification - probabilistic CA

- States: liquid $= 0$, crystals > 0.
- Cell state uniquely encodes crystal orientation tensor, i.e. a look-up table.
- Each iteration a liquid cell acquires a state of a randomly chosen neighbour (3D Moore's neighbourhood - 26 cells).



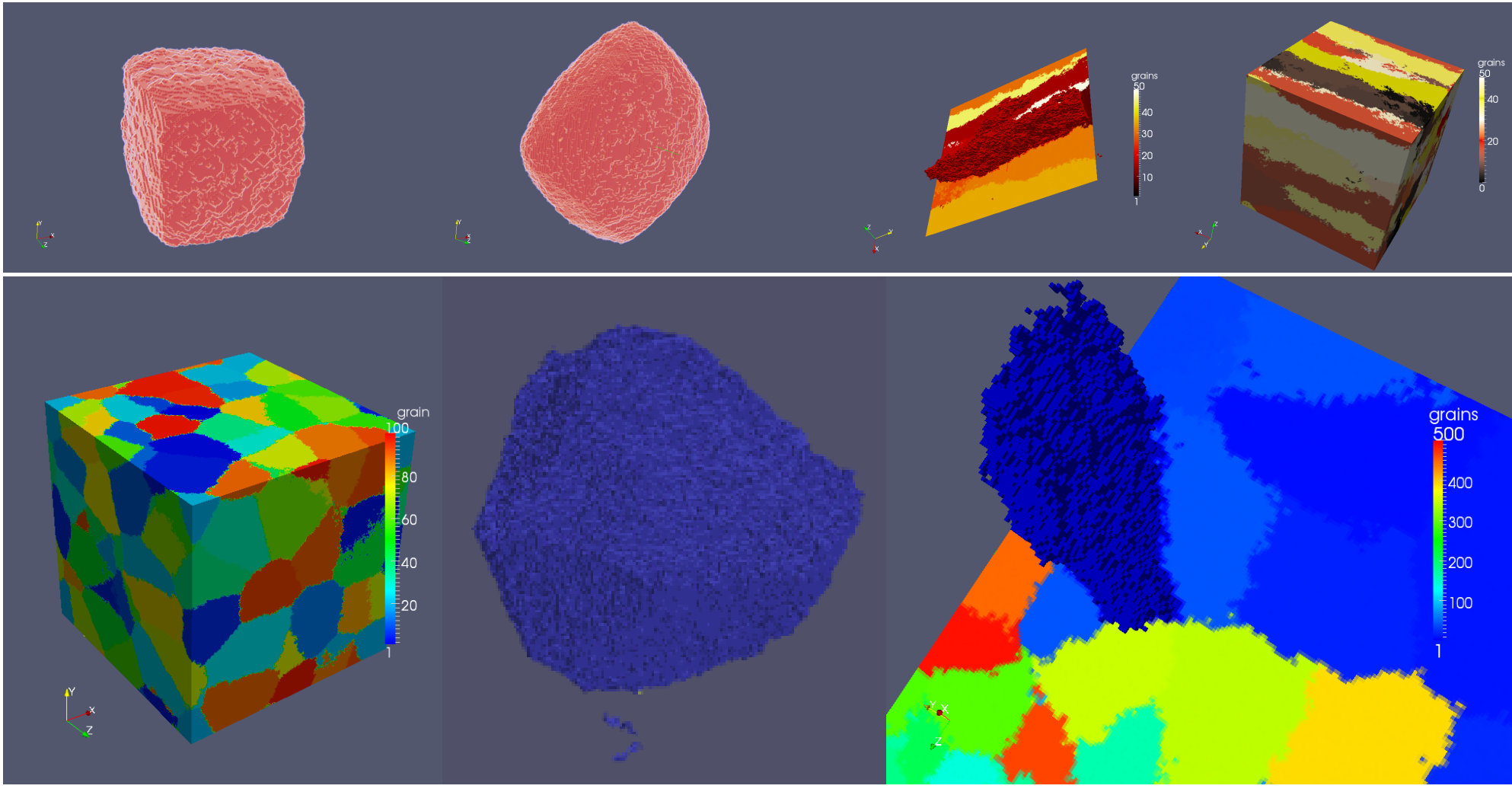


i



$i + 1$

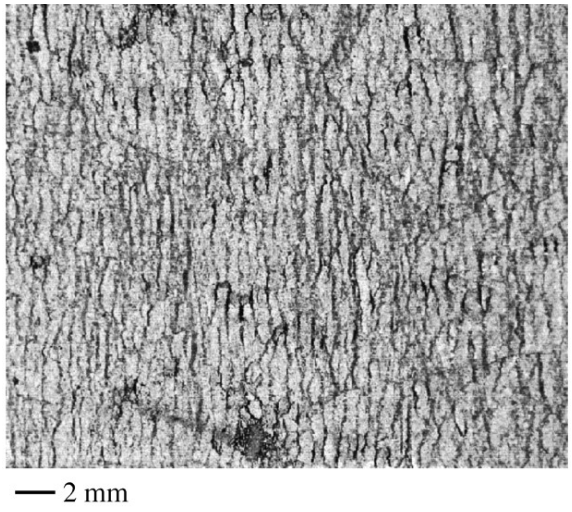
Primitive probabilistic 3D solidification - results

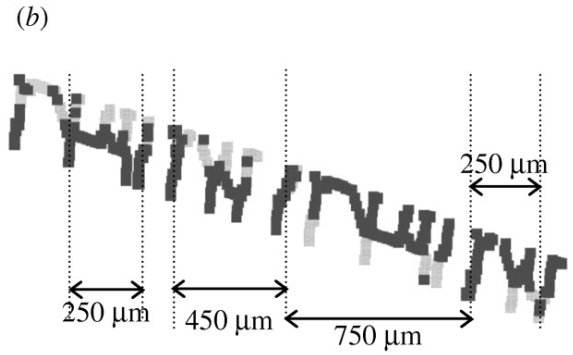


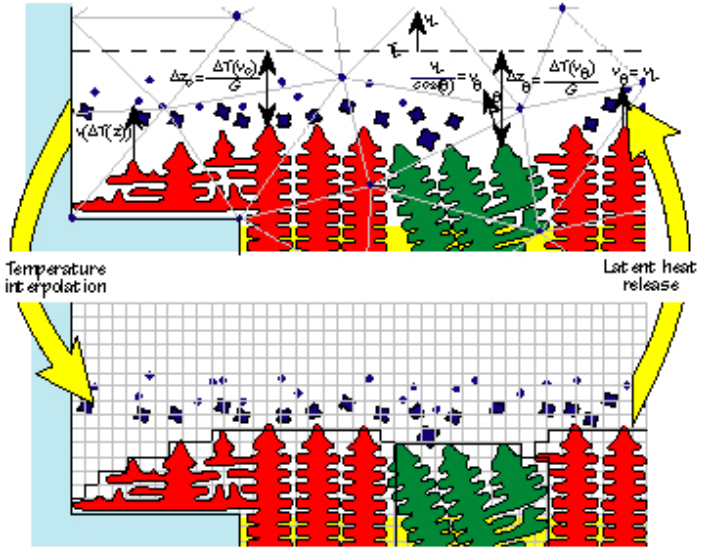
For more results ▸ CGPACK

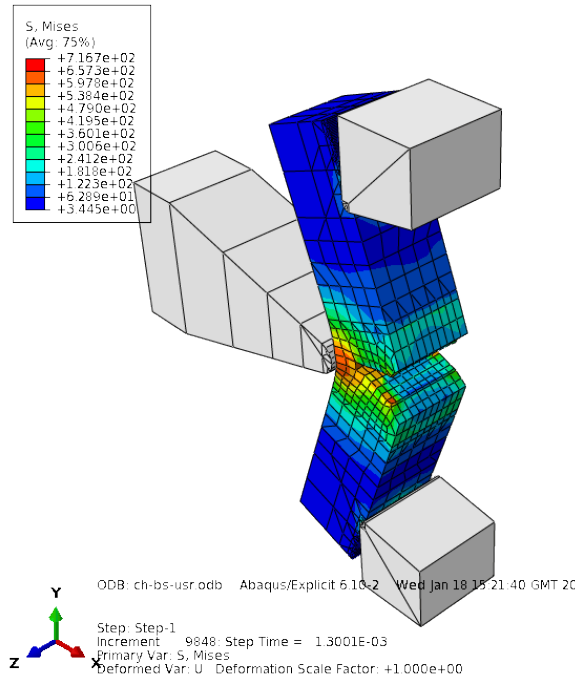
CA with fields

- Used for solidification [1], recrystallisation [2] and fracture [3, 4].
- FE - continuum mechanics - stress, strain, etc.
- CA - crystals, crystal boundaries, cleavage, grain boundary fracture
- FE $\rightarrow$ CA - stress, strain
- CA $\rightarrow$ FE - damage variables

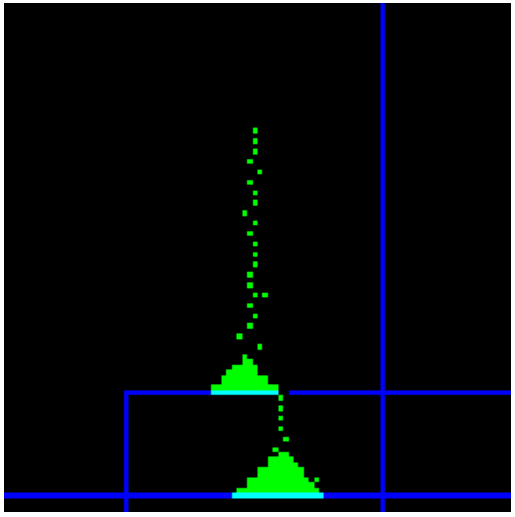




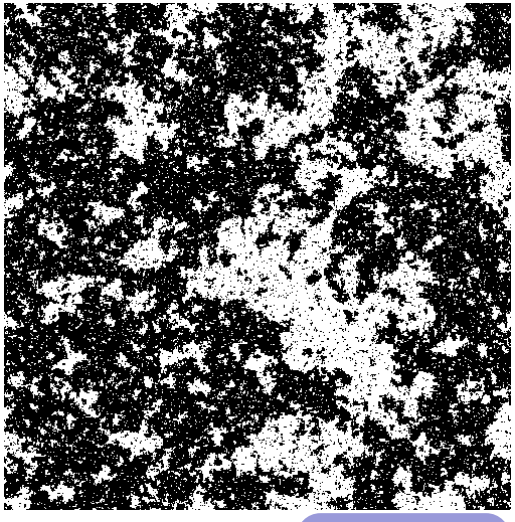




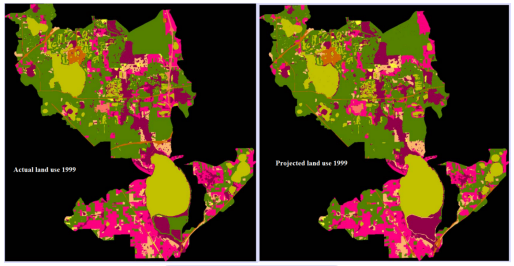
Other CA examples



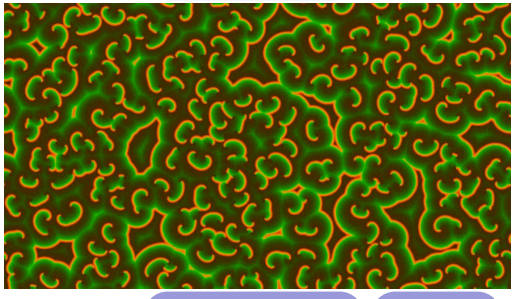
Sand pile formation



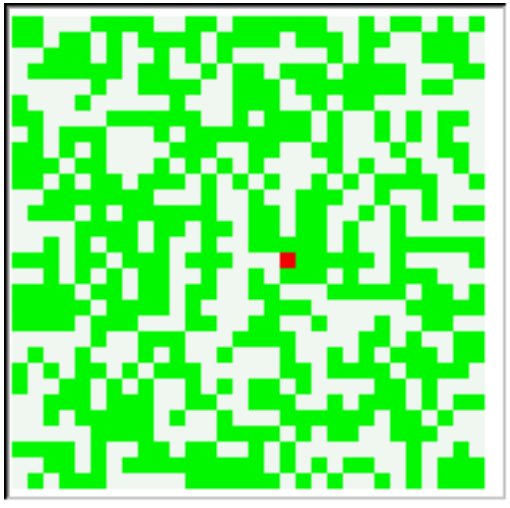
Ising magnetisation ▸ more info



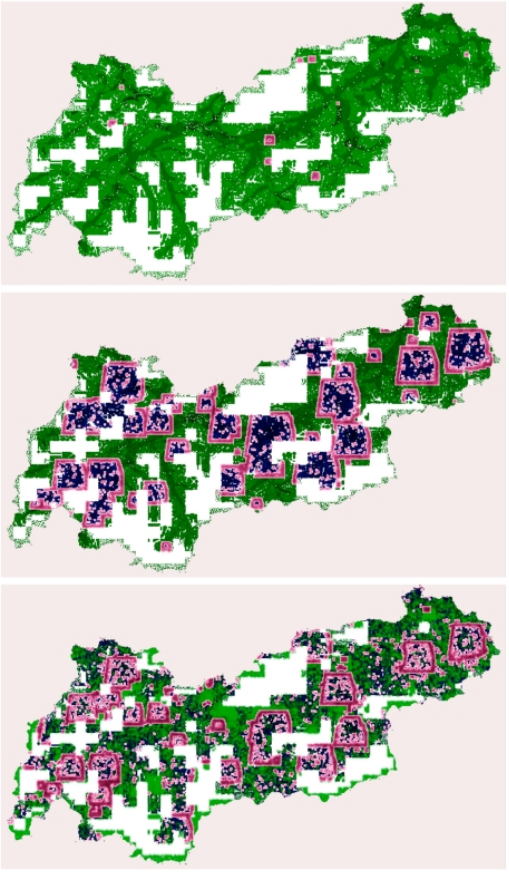
Land use ▸ more info



Diffusion ▸ animation ▸ info



Fire ▸ more info



Epidemics, from The Open Med. Inform. J. 2(1):70-81, 2008.

▸ PDF

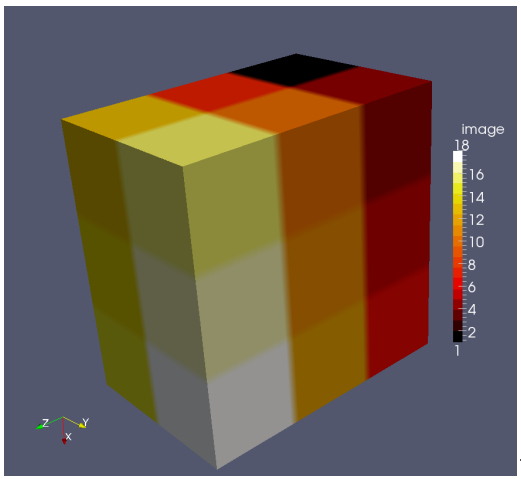
Table of Contents

Fortran coarrays for CA

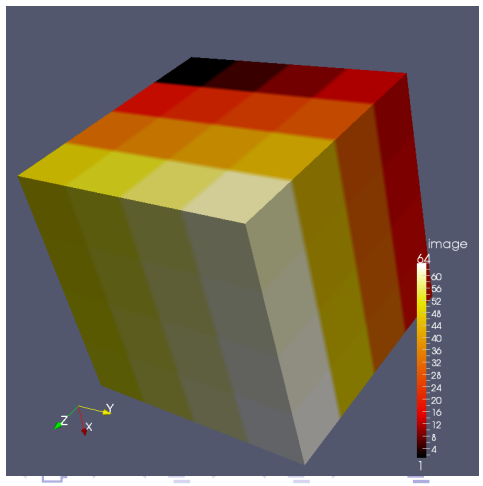
- Fortran native SPMD parallel programming feature
- Fortran standard since 2008. More features in 2015.
- Cray, Intel, OpenCoarrays/GCC support
- CGPACK - cellular automata microstructure simulation library: cgpack.sf.net. See also [5, 6, 7].
- Easy halo exchange
- CA space coarray - 4D array, 3 codimensions:

```
integer, allocatable :: space(:,:,:,:)[:,:,:]
```

- Ideal for structured grids:

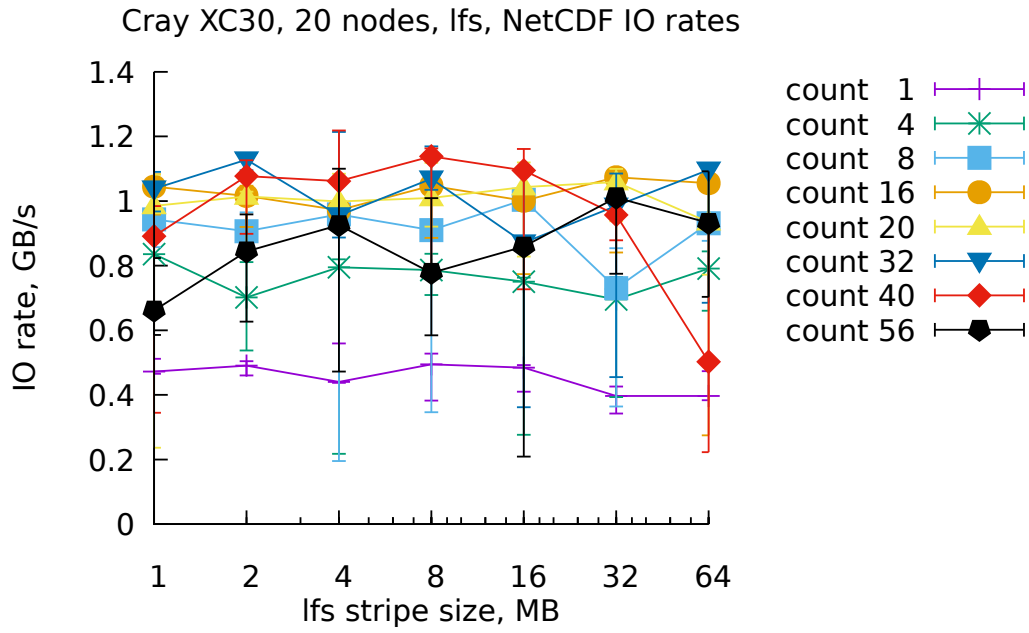


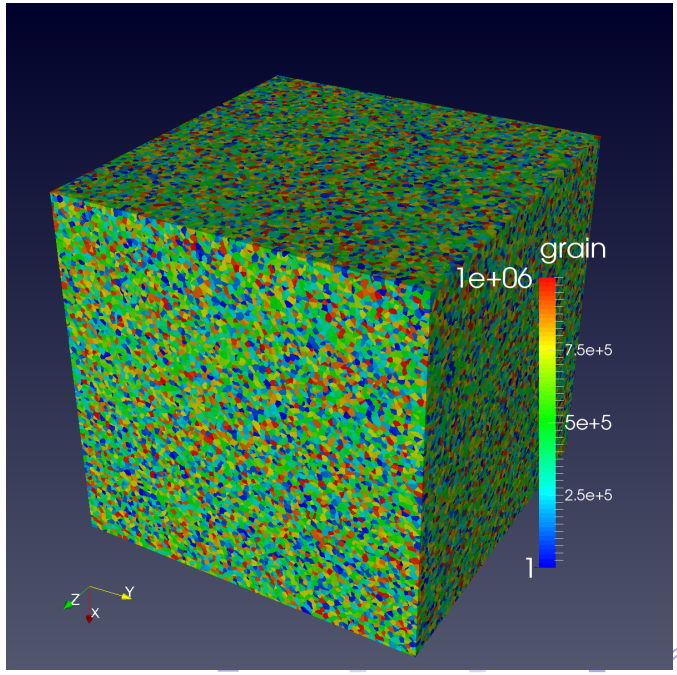
← 18 imgs; 64 imgs →



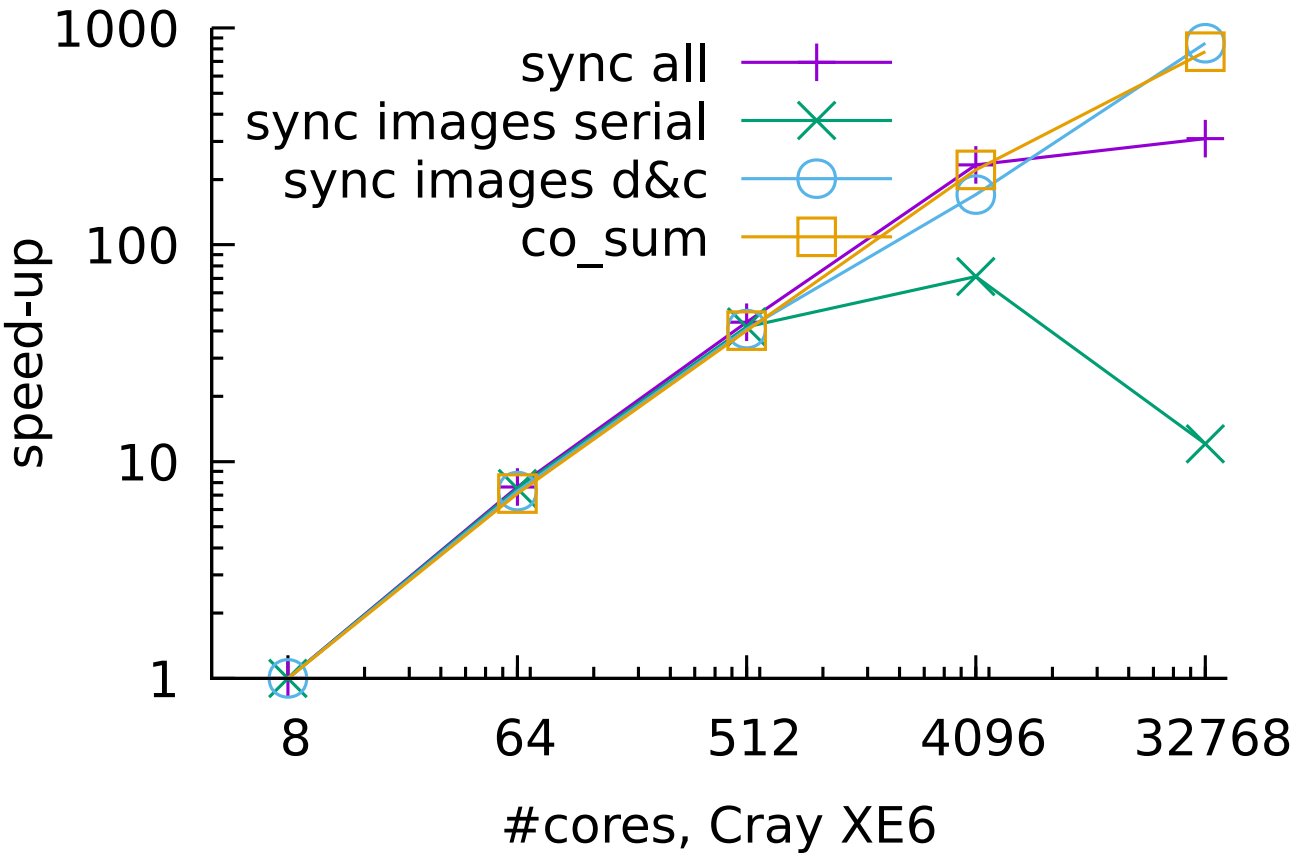
Coarray IO - no native Fortran parallel IO

- MPI/IO up to 2.3GB/s on Cray XE6 ▸ BCS talk
- MPI/IO up to 8GB/s on Cray XC30 (can reach 14GB/s [8])
- NetCDF 4.3, HDF5 1.8.14 - only up to 1.2GB/s on Cray XC30.
- lfs stripe count, size, number of images, file size, Cray hugepages...
- 0.5 - 1TB datasets





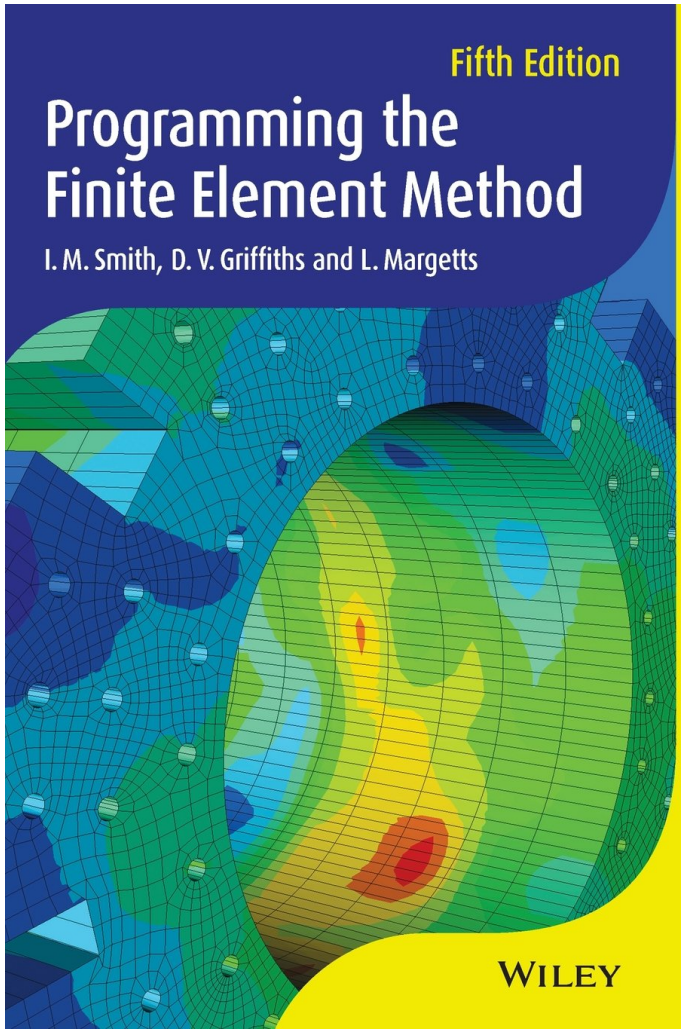
CGPACK solidification scaling



Scaling varies for different programs built with CGPACK, depending on which routines are called, in what order and requirements for synchronisation.

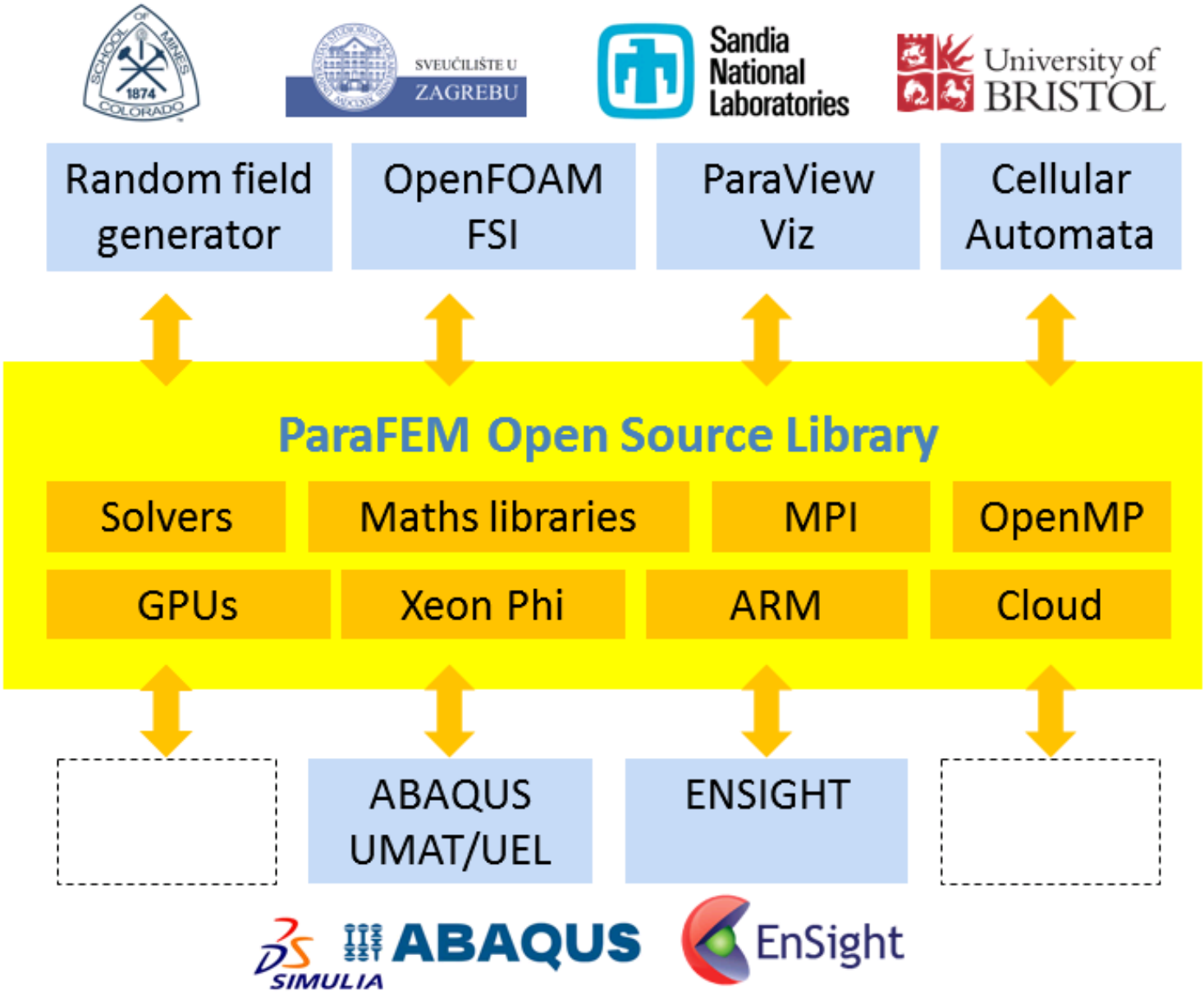
Table of Contents

ParaFEM - scalable MPI finite element library



- parafem.org.uk
- Open source library + ∼70 mini Apps
- Fortran 90 MPI
- Proven scaling up to ∼64,000 cores
- >1 billion degrees of freedom
- Used for teaching and research
- 1000+ registered users on website
- ∼1400 citations of text book [9]
- Highly portable
- BSD license

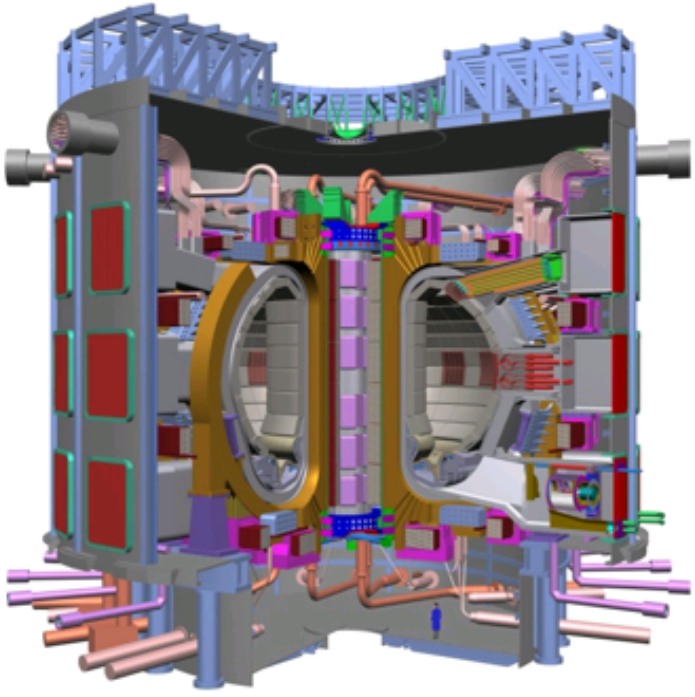
ParaFEM library interfaces



ParaFEM examples - nuclear fusion

simpleware

CCFE
CULHAM CENTRE FOR FUSION ENERGY



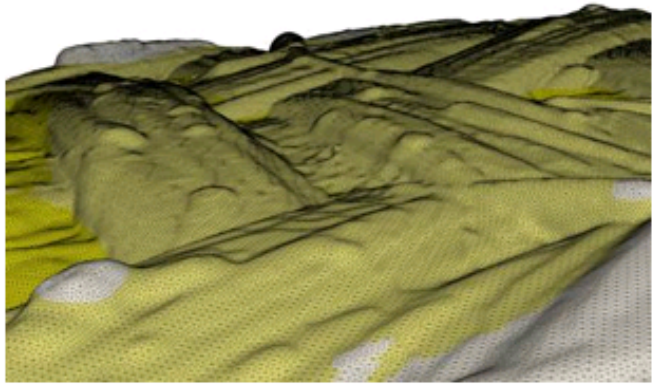
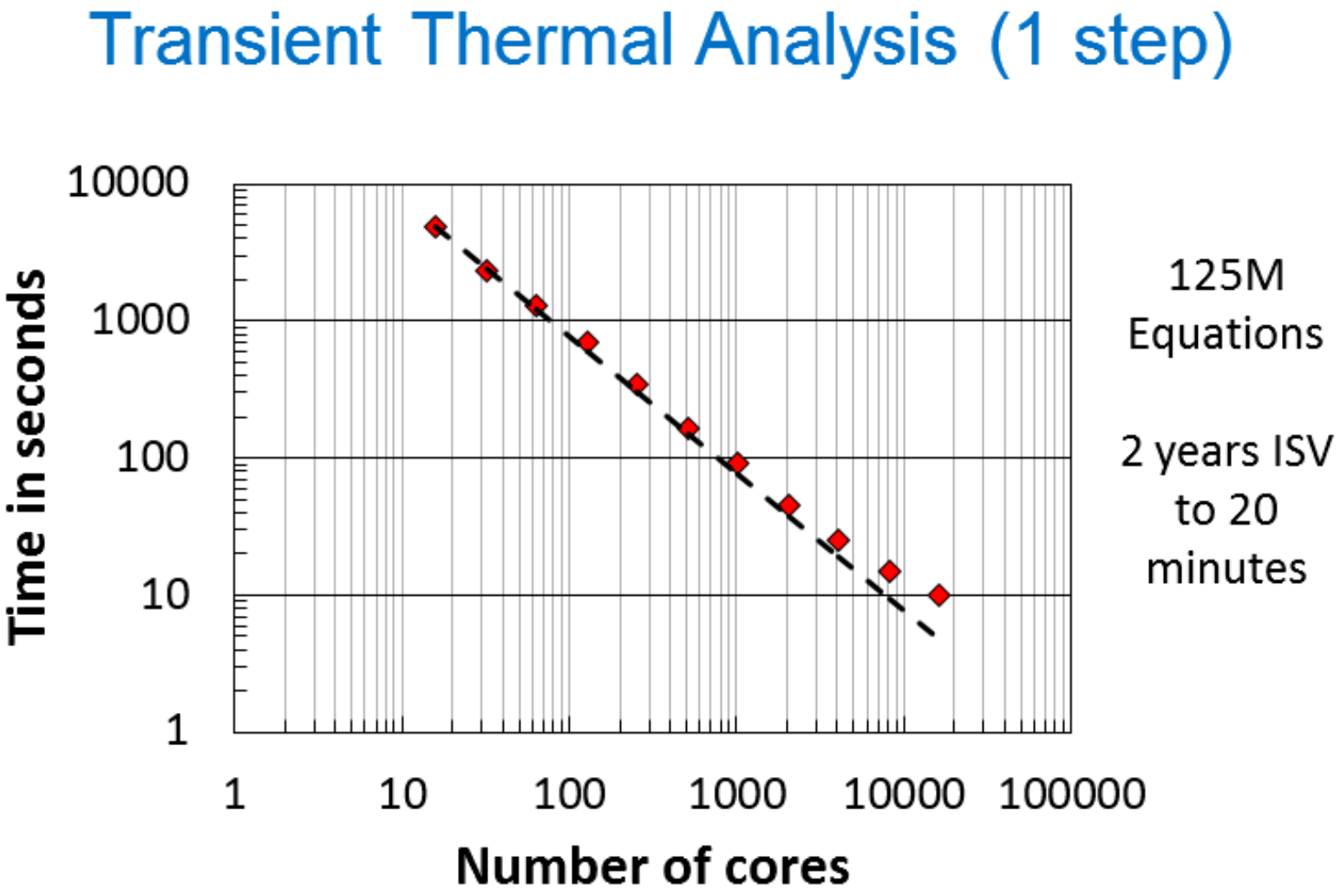
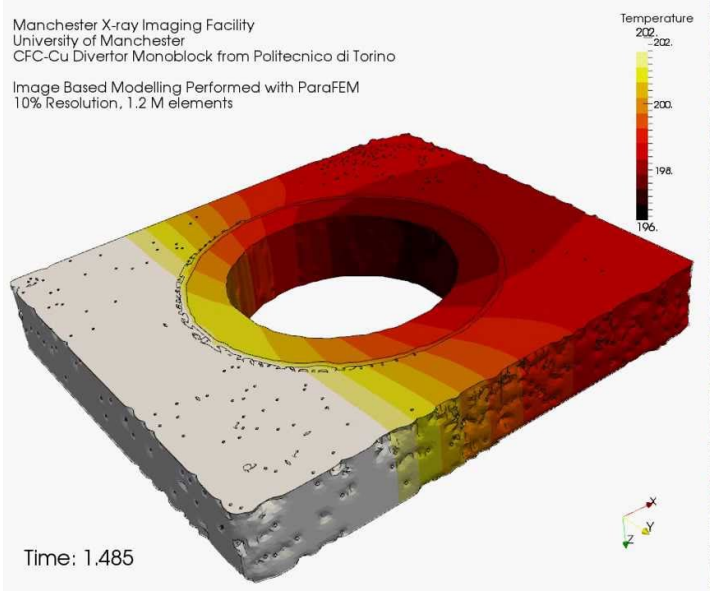


Image-based transient thermal analysis

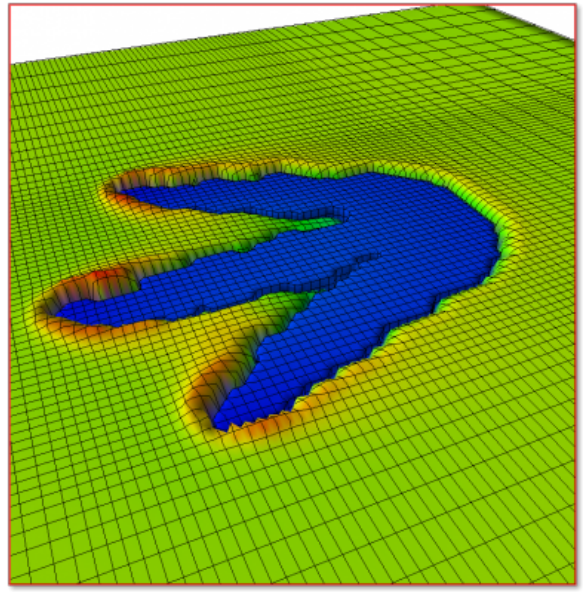
ParaFEM scaling



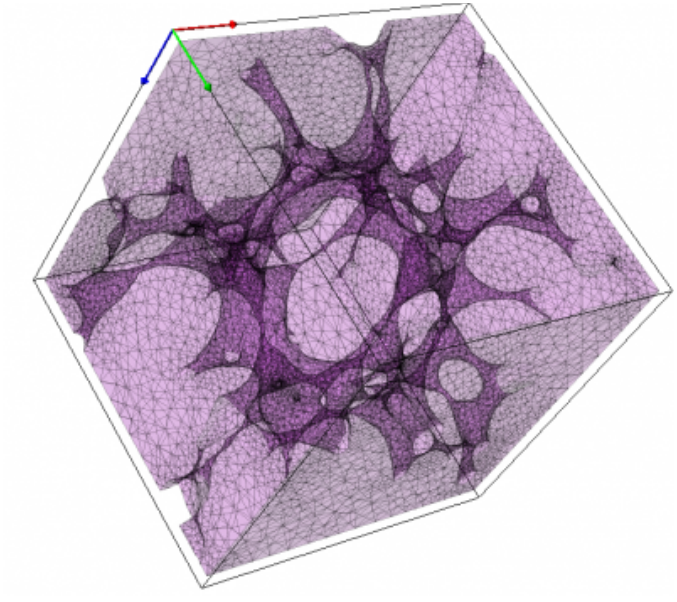
ParaFEM application areas



power generation



paleontology

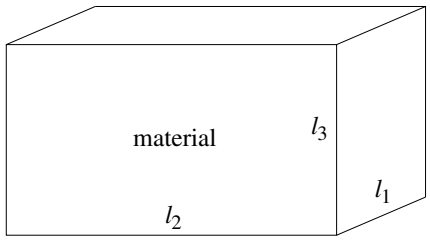


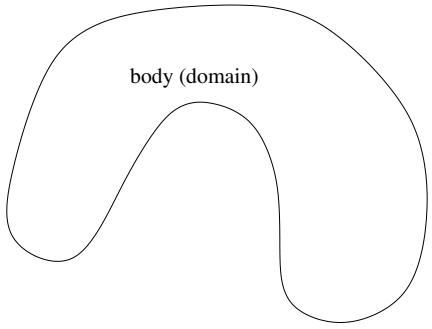
micro-mechanics

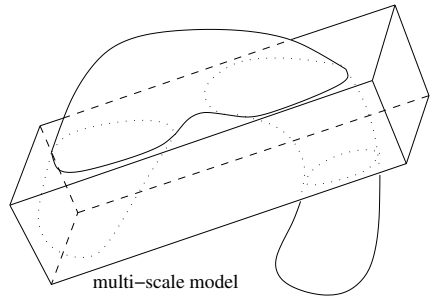
bio-mechanics

Table of Contents

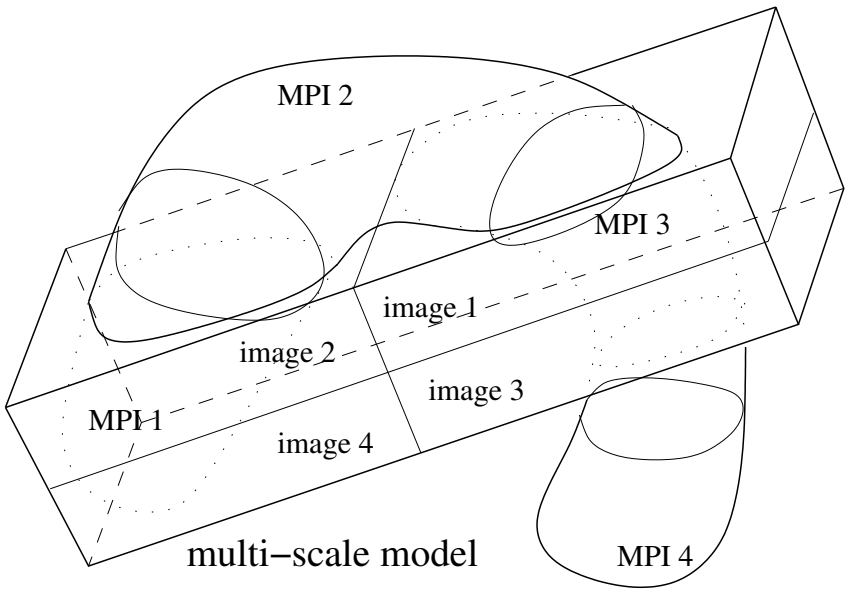
CAFE design: structured CA grid + unstructured FE grid



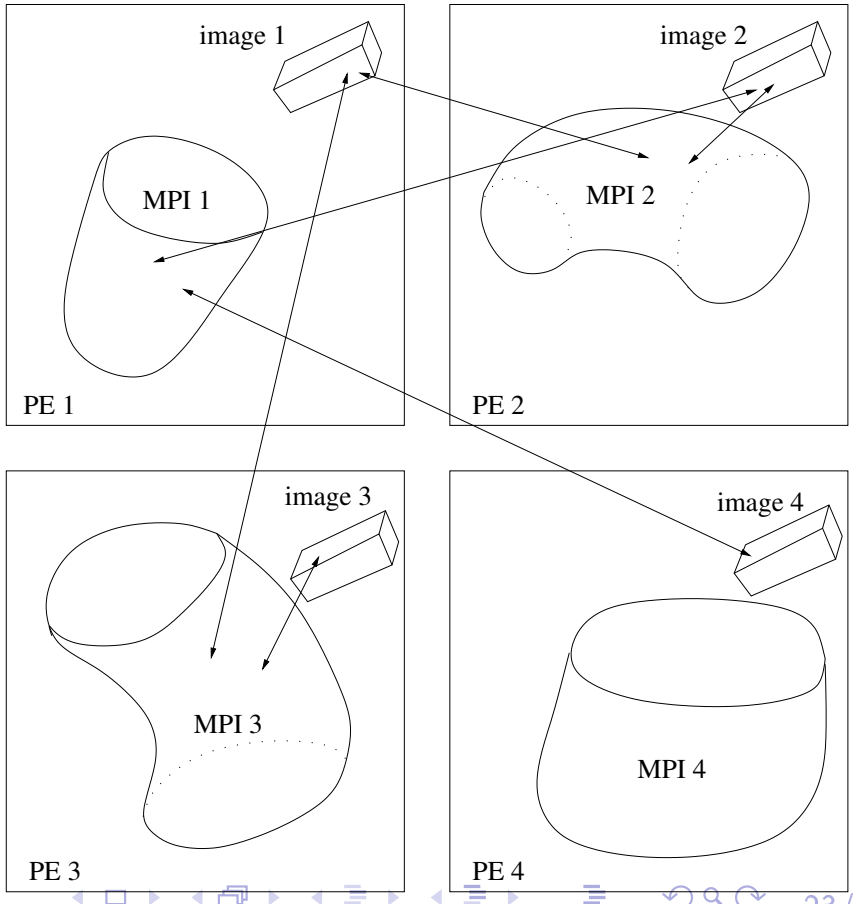




CA + FE = CAFE



Example with 4 PE (4 MPI processes, 4 coarray images). Arrows are FE $\leftrightarrow$ CA comms.



FE $\rightarrow$ CA mapping via a private allocatable array of derived type:

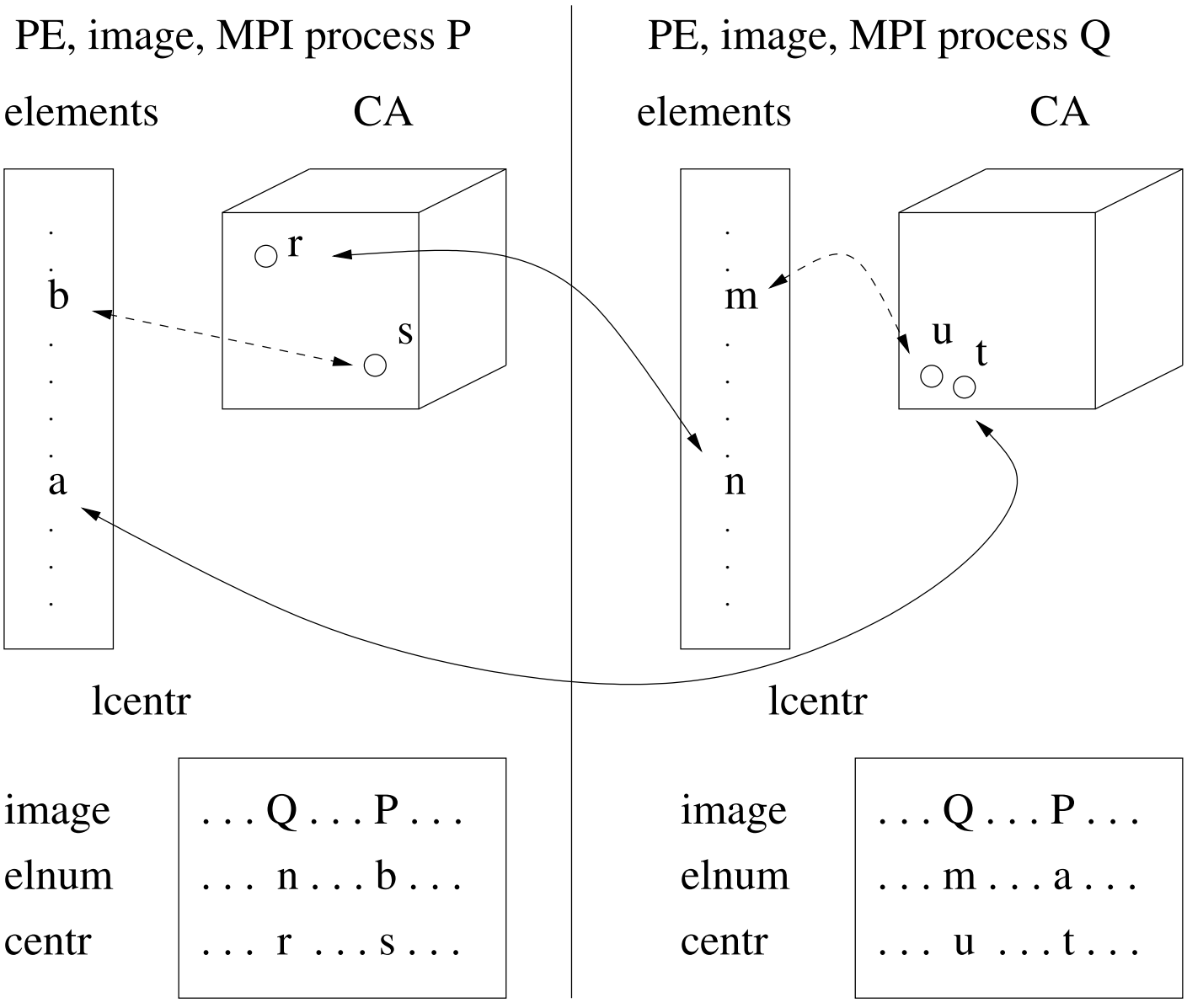
```
type mcen
  integer :: image, elnum
  real :: centr(3)
end type mcen
type( mcen ), allocatable :: lcentr(:)
```

based on coordinates of FE centroids calculated by each MPI process and stored in centroid_tmp coarray:

```
type rca
  real, allocatable :: r(:,:)
end type rca
type( rca ) :: centroid_tmp[*]
 :
allocate( centroid_tmp%r(3, nels_pp) )
```

where nels_pp is the number of FE stored on this PE.

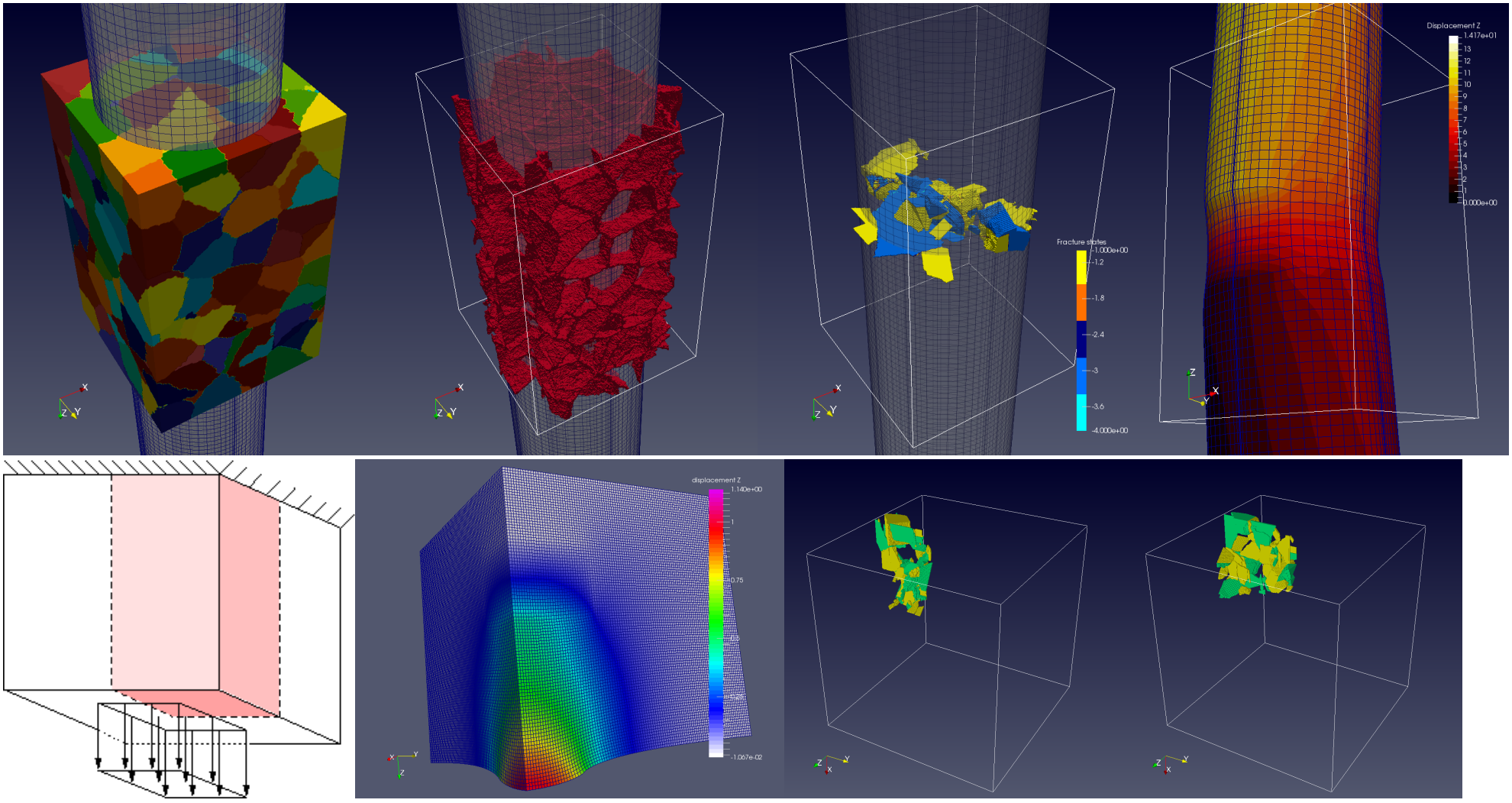
lcentr arrays on images P and Q



Fracture modelling

- Diverse CAFE fracture models can be constructed from CGPACK + ParaFEM libraries.
- Simple case: isotropic linear elastic FE (E, ν) + cleavage (fully brittle transgranular fracture mode) CA.
- FE stress tensor $\mathbf{t}$ passed to CA, resolved on normal stresses on $\{100\}$ and $\{110\}$ crystal planes - t_{100}, t_{110} [5, 10].
- 2 parameters - fracture stress, σ_F, linked to the free surface energy, γ, and a characteristic length, L.
- If $t_{100} \geq \sigma_F$ or $t_{110} \geq \sigma_F$ then a CA crack extends by L per unit of time.
- Crack morphology is reduced to a single damage variable, d. $d = 1$ initially (no damage). $d = 0$ - integration point has failed, no load bearing capacity.

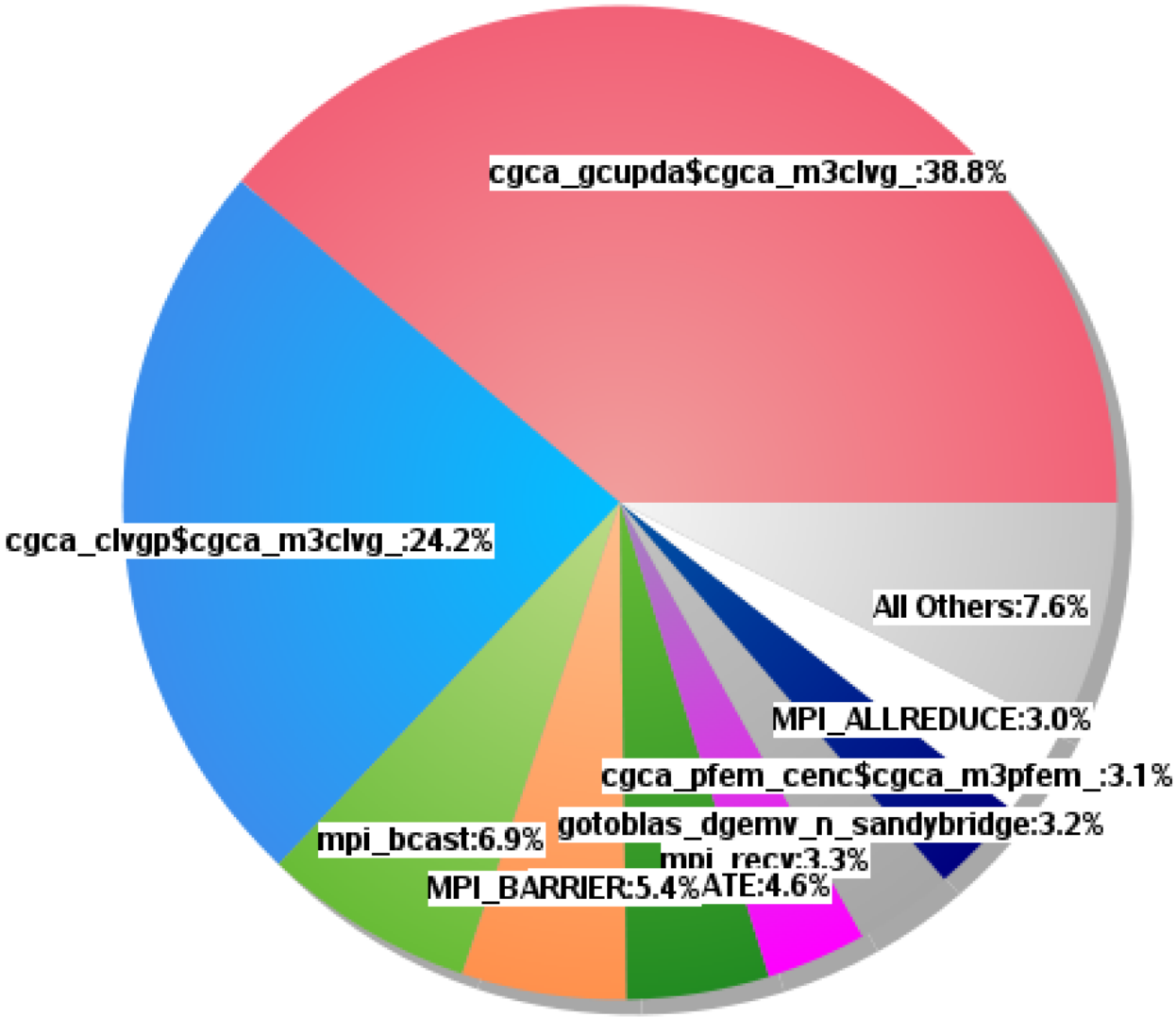
CAFE fracture results



Watch animations at ▸ CGPACK site

Table of Contents

CrayPAT profiling 1



Profiling function distribution for ParaFEM/CGPACK MPI/coarray miniapp with all-to-all routine cgca_gcupda at 7200 cores.

CrayPAT profiling 2

```
100.0% | 20,520.4 |    -- |    -- |Total
|-----------------------------------------------------------------
|  71.4% | 14,649.9 |    -- |    -- |USER
||----------------------------------------------------------------
||  38.7% |  7,950.6 | 913.4 | 10.3% |cgca_gcupda$cgca_m3clvg_
||  24.1% |  4,951.2 | 940.8 | 16.0% |cgca_clvgp$cgca_m3clvg_
||   3.1% |    638.0 |  70.0 |  9.9% |cgca_pfem_cenc$cgca_m3pfem_
||   1.8% |    367.5 | 578.5 | 61.2% |cgca_hxi$cgca_m2hx_
||   1.7% |    346.0 | 196.0 | 36.2% |cgca_clvgn$cgca_m3clvg_
||================================================================
|  19.8% |  4,061.4 |    -- |    -- |MPI
||----------------------------------------------------------------
||   6.9% |  1,413.5 | 356.5 | 20.1% |mpi_bcast
||   5.4% |  1,098.3 | 419.7 | 27.7% |MPI_BARRIER
||   3.3% |    670.0 | 322.0 | 32.5% |mpi_recv
||   3.0% |    615.3 |  61.7 |  9.1% |MPI_ALLREDUCE
||================================================================
|   8.8% |  1,797.2 |    -- |    -- |ETC
||----------------------------------------------------------------
||   4.6% |    950.5 |   5.5 |  0.6% |__DEALLOCATE
||   3.2% |    654.2 | 110.8 | 14.5% |gotoblas_dgemv_n_sandybridge
|================================================================
```

Raw profiling data for ParaFEM/CGPACK MPI/coarray miniapp with all-to-all routine cgca_gcupda at 7200 cores.

cgca_gcupda - all-to-all

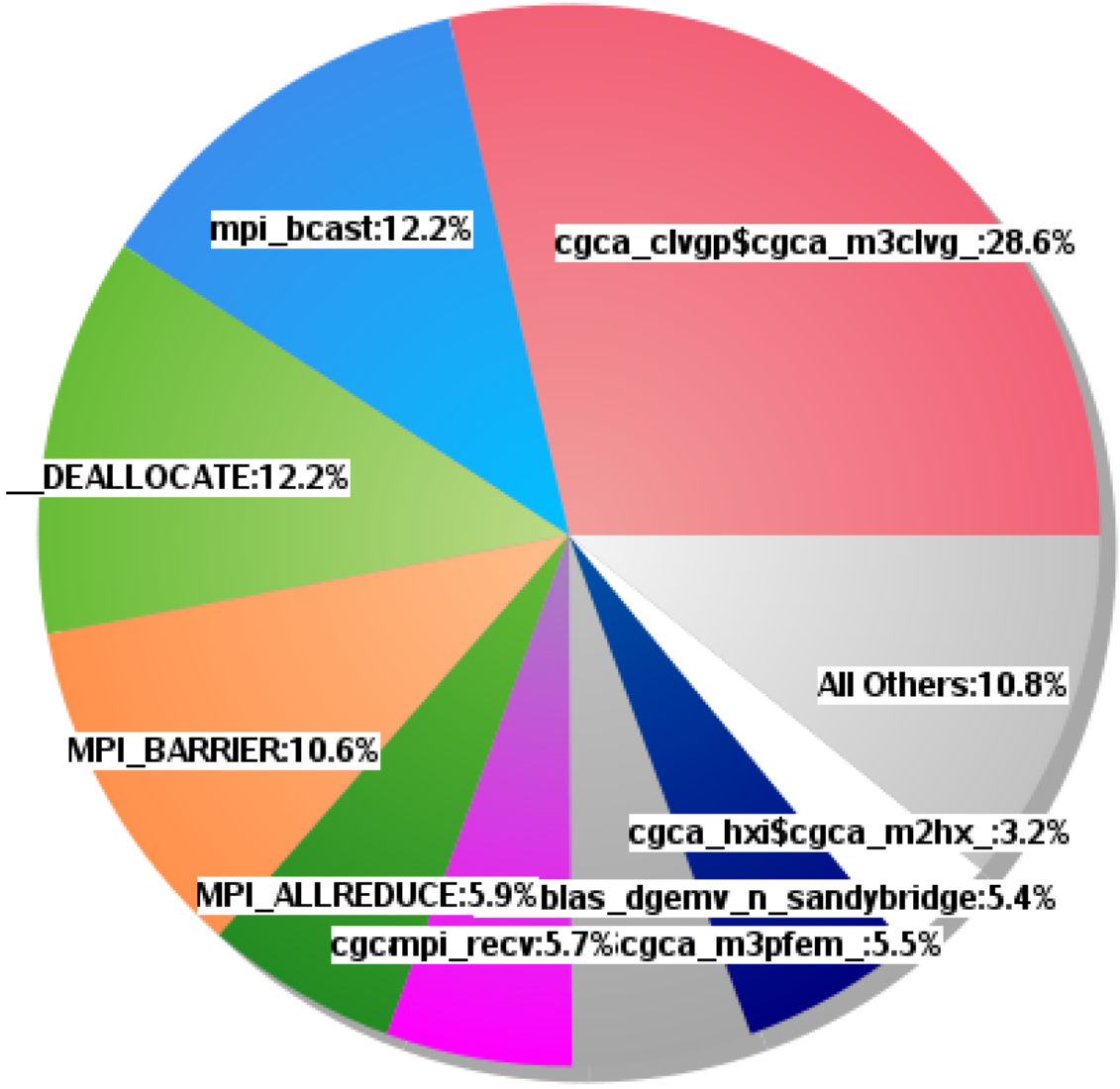
```
integer :: gcupd(100,3)[*], rndint, j, &
           img, gcupd_local(100,3)
real :: rnd
:
call random_number( rnd )
rndint = int( rnd*num_images() ) + 1
do j = rndint, rndint + num_images() - 1
 img = j
 if (img .gt. num_images()) &
       img = img - num_images()
  if (img .eq. this_image()) cycle
  :
  gcupd_local(:,:) = gcupd(:,:)[img]
  :
end do
```

cgca_gcupdn - nearest neighbour

```
do i = -1 , 1
do j = -1 , 1
do k = -1 , 1
 ! Get the coindex set of the neighbour
 ncod = mycod + (/ i , j , k /)
 :
 gcupd_local(:,:) = &
     gcupd(:,:)[ncod(1),ncod(2),ncod(3)]
 :
end do
end do
end do
```

Note: the nearest neighbour must be called *multiple times* to propagate changes from every image to all other images.

CrayPAT profiling cgca_gcupdn



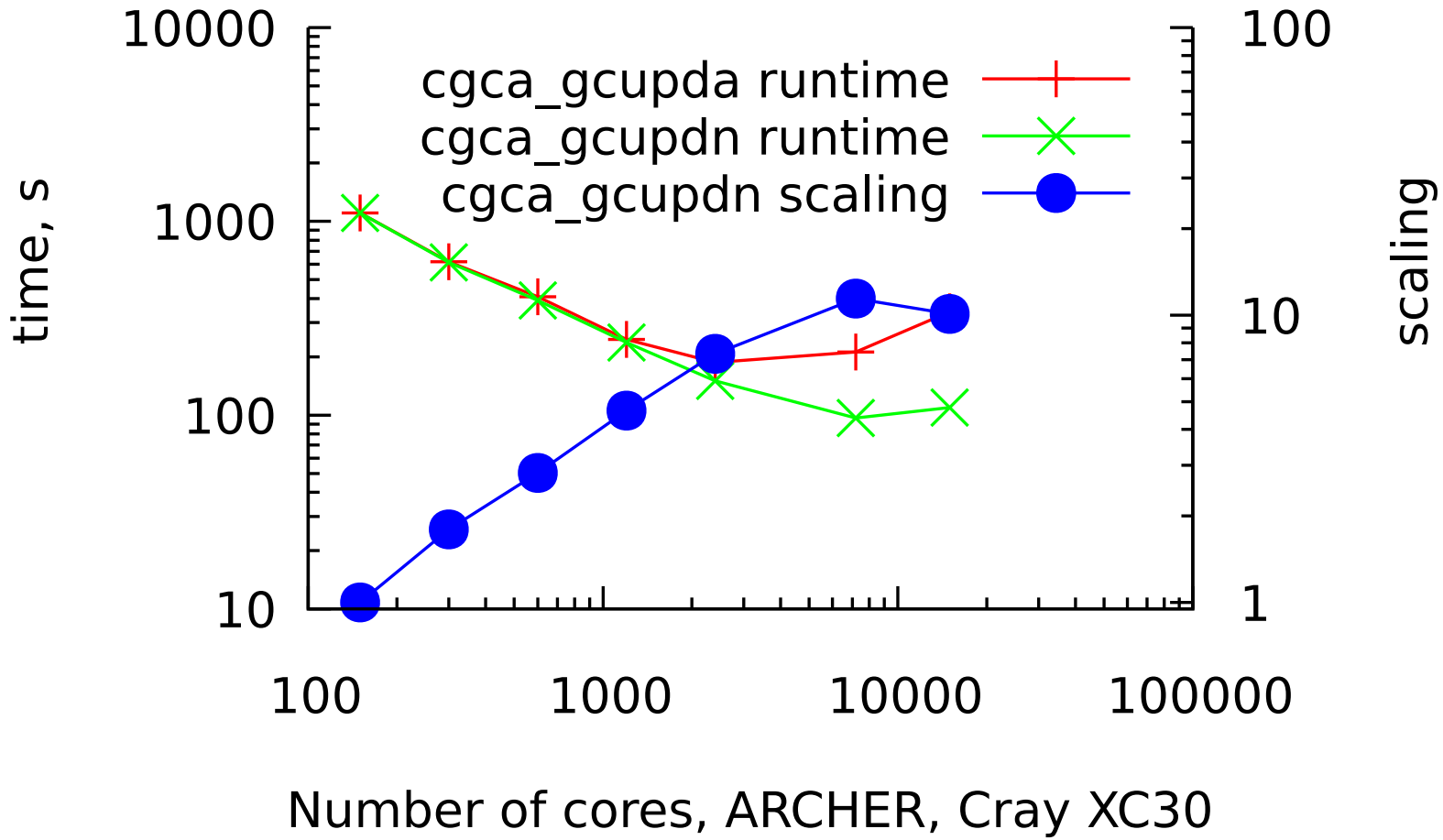
Profiling function distribution for ParaFEM/CGPACK MPI/coarray miniapp with the nearest neighbour routine cgca_gcupdn at 7200 cores.

CrayPAT profiling cgca_gcupdn

```
100.0% | 12,199.5 |    -- |    -- |Total
|-----------------------------------------------------------
|  44.8% |  5,459.7 |    -- |    -- |USER
||----------------------------------------------------------
||  28.6% |  3,484.0 | 582.0 | 14.3% |cgca_clvgp$cgca_m3clvg_
||   5.5% |    666.1 |  93.9 | 12.4% |cgca_pfem_cenc$cgca_m3pfem_
||   3.2% |    393.1 | 752.9 | 65.7% |cgca_hxi$cgca_m2hx_
||   2.8% |    346.0 | 176.0 | 33.7% |cgca_clvgn$cgca_m3clvg_
||   1.4% |    165.2 |  37.8 | 18.6% |cgca_sld$cgca_m3sld_
||   1.0% |    126.0 |  82.0 | 39.4% |xx14_
||==========================================================
|  36.7% |  4,472.1 |    -- |    -- |MPI
||----------------------------------------------------------
||  12.2% |  1,484.4 | 380.6 | 20.4% |mpi_bcast
||  10.6% |  1,287.9 | 389.1 | 23.2% |MPI_BARRIER
||   5.9% |    714.9 |  90.1 | 11.2% |MPI_ALLREDUCE
||   5.7% |    689.4 | 338.6 | 32.9% |mpi_recv
||   1.5% |    179.1 | 417.9 | 70.0% |MPI_REDUCE
||==========================================================
|  18.5% |  2,256.1 |    -- |    -- |ETC
||----------------------------------------------------------
||  12.1% |  1,480.9 |   4.1 |  0.3% |__DEALLOCATE
||   5.4% |    653.8 |  95.2 | 12.7% |gotoblas_dgemv_n_sandybridge
|===========================================================
```

Raw profiling data for ParaFEM/CGPACK MPI/coarray miniapp with the nearest neighbour routine cgca_gcupdn at 7200 cores.

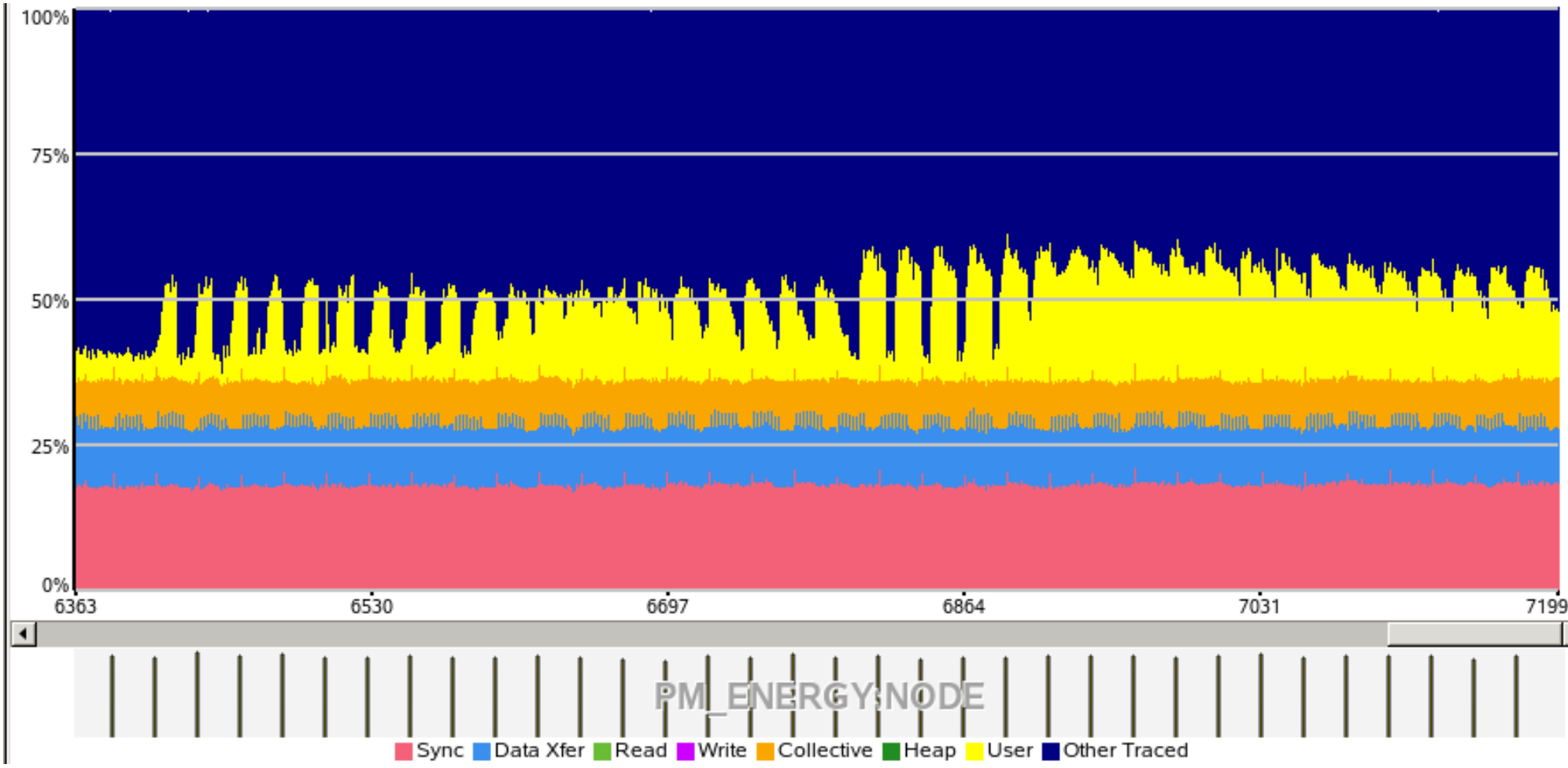
Scaling improvement with cgca_gcupdn over cgca_gcupda



Runtimes and scaling for ParaFEM/CGPACK MPI/coarray miniapp with the nearest neighbour, cgca_gcupdn, and all-to-all, cgca_gcupda, algorithms.

Scaling limit increased from 2k to 7k cores.

CrayPAT - load imbalance on 7k cores on Cray XC30



Whole program activity, shown in % total time per process.
Processes 6363 to 7199 are shown.

Issues with CrayPAT

```
|  71.4% | 14,649.9 |    -- |    -- |USER
||-----------------------------------------------------------------
||  38.7% |  7,950.6 | 913.4 | 10.3% |cgca_gcupda$cgca_m3clvg_
||  24.1% |  4,951.2 | 940.8 | 16.0% |cgca_clvgp$cgca_m3clvg_
||   3.1% |    638.0 |  70.0 |  9.9% |cgca_pfem_cenc$cgca_m3pfem_
||   1.8% |    367.5 | 578.5 | 61.2% |cgca_hxi$cgca_m2hx_
||   1.7% |    346.0 | 196.0 | 36.2% |cgca_clvgn$cgca_m3clvg_
||=================================================================
```

cgca_gcupda is top in sampling results, but is absent from tracing. It is called the same number of times as cgca_hxi.

```
|  29.7% |  99.743118 |       -- |    -- |  5,226,813.1 |USER
||-----------------------------------------------------------------------------
||  17.4% |  58.326659 | 36.082315 | 38.2% |          5.0 |cgca_clvgp$cgca_m3clvg_
||   5.6% |  18.876152 |  5.062089 | 21.1% |          1.0 |cgca_pfem_cenc$cgca_m3pfem_
||   3.3% |  11.145318 | 15.328335 | 57.9% |          1.0 |xx14_
||   1.7% |   5.705317 |  8.788733 | 60.6% |  5,224,771.1 |cgca_clvgn$cgca_m3clvg_
||   1.7% |   5.689672 |  1.910819 | 25.1% |      2,035.0 |cgca_hxi$cgca_m2hx_
||=============================================================================
```

Issues with CrayPAT

All profiling was done with single thread.

```
CrayPat/X:  Version 6.2.2 Revision 13378 (xf 13240)  11/20/14 14:32:58
Number of PEs (MPI ranks):   480

Numbers of PEs per Node:      24  PEs on each of  20  Nodes

Numbers of Threads per PE:     3

Number of Cores per Socket:   12
Execution start time:  Thu Mar  3 13:40:17 2016
System name and speed:  tdsmom 2701 MHz
```

Incorrect number of threads identified by CrayPAT in a tracing experiment of ParaFEM/CGPACK MPI/coarray miniapp with cgca_gcupda.

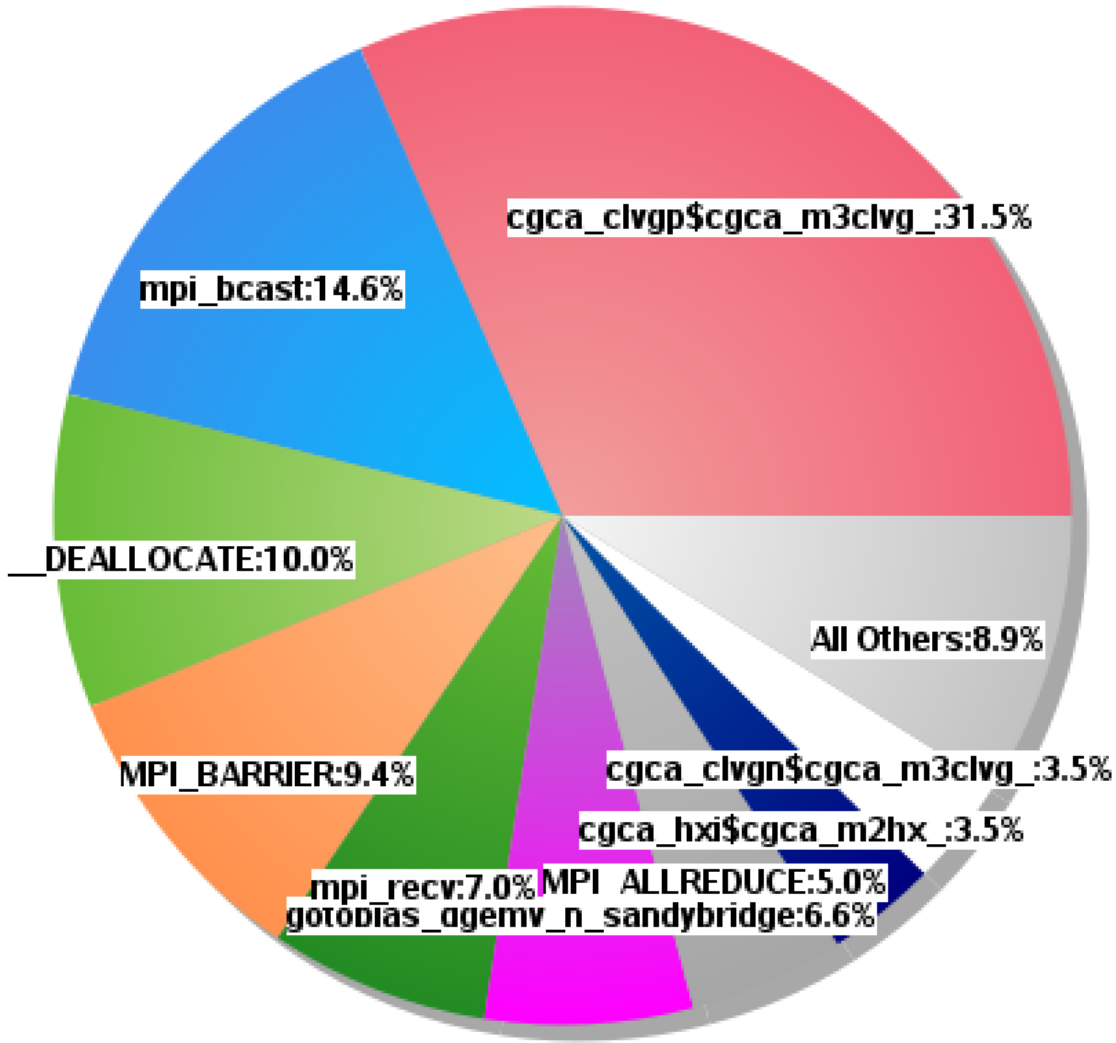
Establishing lcentr array

- Array lcentr is established by subroutine cgca_pfem_cenc which uses all-to-all comms.
- Subroutine cgca_pfem_map uses large temporary arrays and coarray collectives co_sum and co_max, parts of Fortran 2015 standard [11, 12].

Comms structure in cgca_pfem_map

```
integer :: maxfe, pos_start, pos_end, ctmpsize
real, allocatable :: tmp(:,:)
! Calculate the max number of FEs on any image
maxfe = size( centroid_tmp%r, dim=2 )
ctmpsize = maxfe
call co_max( maxfe )
allocate(tmp(maxfe*num_images(),5), source=0.0)
! Writes my data in a unique portion of tmp
pos_start = (this_image() - 1)*maxfe + 1
pos_end = pos_start + ctmpsize - 1
tmp( pos_start : pos_end, 1 ) = &
 real( this_image(), kind=4 )
! Write element number *as real*
tmp( pos_start : pos_end, 2 ) = &
 real((/ (j, j = 1, ctmpsize) /), kind=4)
! Write centroid coord
tmp( pos_start : pos_end, 3:5 ) = &
 transpose( centroid_tmp%r(:,:) )
call co_sum( tmp )
```

Profiling with cgca_pfem_map



Profiling function distribution for ParaFEM/CGPACK MPI/coarray miniapp with cgca_gcupdn and cgca_pfem_map at 7200 cores.

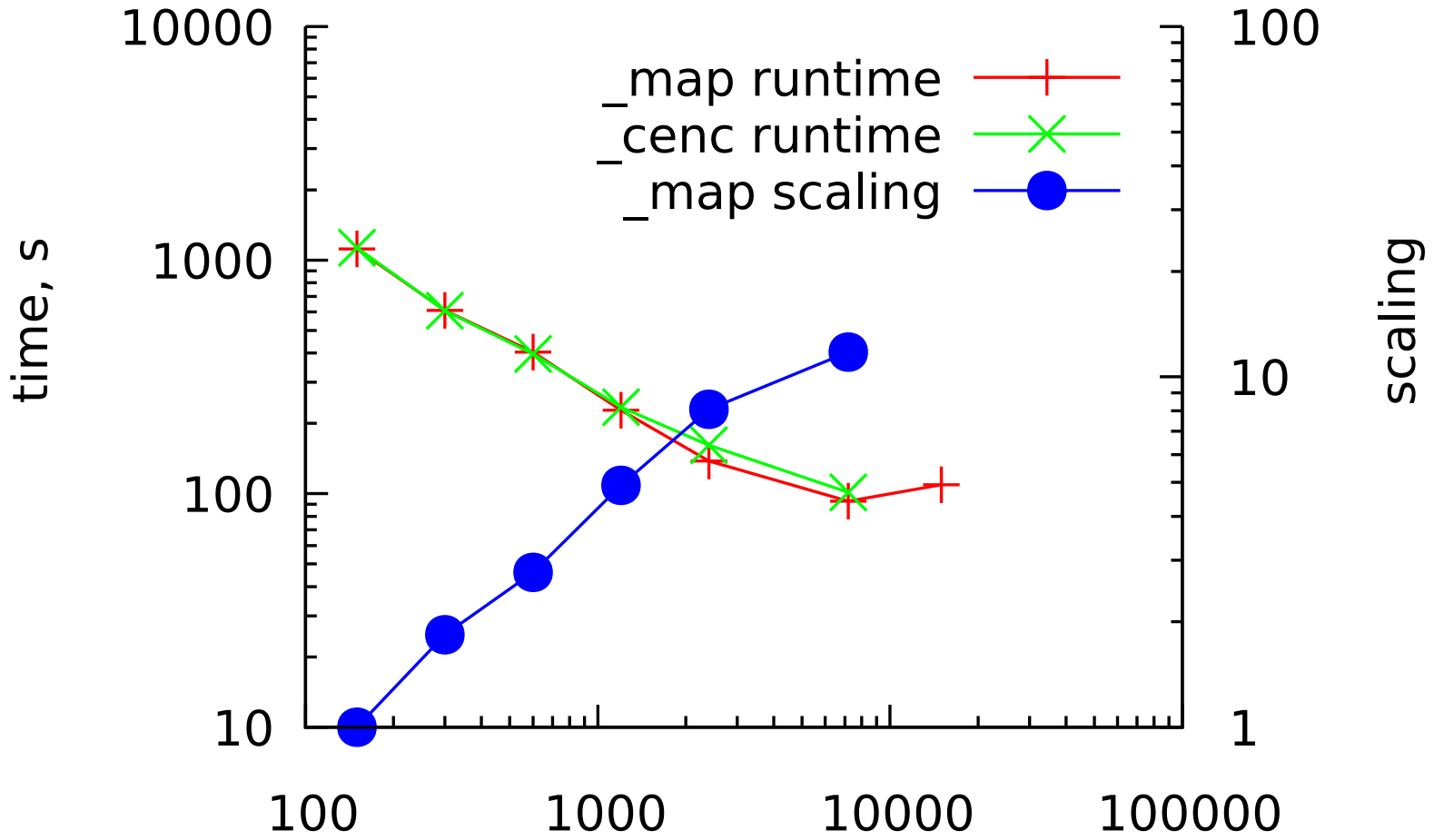
Profiling with cgca_pfem_map

Table 1: Profile by Function

Samp%	Samp	Imb. Samp	Imb. Samp%	Group / Function / PE=HIDE / Thread=HIDE
100.0%	9,903.4	--	--	Total
43.6%	4,321.6	--	--	USER
31.4%	3,110.7	589.3	15.9%	cgca_clvgp$cgca_m3clvg_
3.5%	346.0	513.0	59.7%	cgca_hxi$cgca_m2hx_
3.5%	342.0	175.0	33.8%	cgca_clvgn$cgca_m3clvg_
1.2%	116.3	4.7	3.9%	cgca_pfem_map$cgca_m3pfem_
1.1%	106.8	1,537.2	93.5%	cgca_clvgsd$cgca_m3clvg_
1.0%	99.9	24.1	19.5%	cgca_sld$cgca_m3sld_
38.4%	3,803.6	--	--	MPI
14.6%	1,446.6	350.4	19.5%	mpi_bcast
9.4%	932.4	473.6	33.7%	MPI_BARRIER
7.0%	689.5	371.5	35.0%	mpi_recv
4.9%	489.3	76.7	13.6%	MPI_ALLREDUCE
1.5%	145.4	314.6	68.4%	MPI_REDUCE
17.8%	1,766.8	--	--	ETC
9.9%	983.9	8.1	0.8%	__DEALLOCATE
6.6%	652.3	93.7	12.6%	gotoblas_dgemv_n_sandybridge

Raw profiling data for ParaFEM/CGPACK MPI/coarray miniapp with cgca_gcupdn and cgca_pfem_map at 7200 cores.

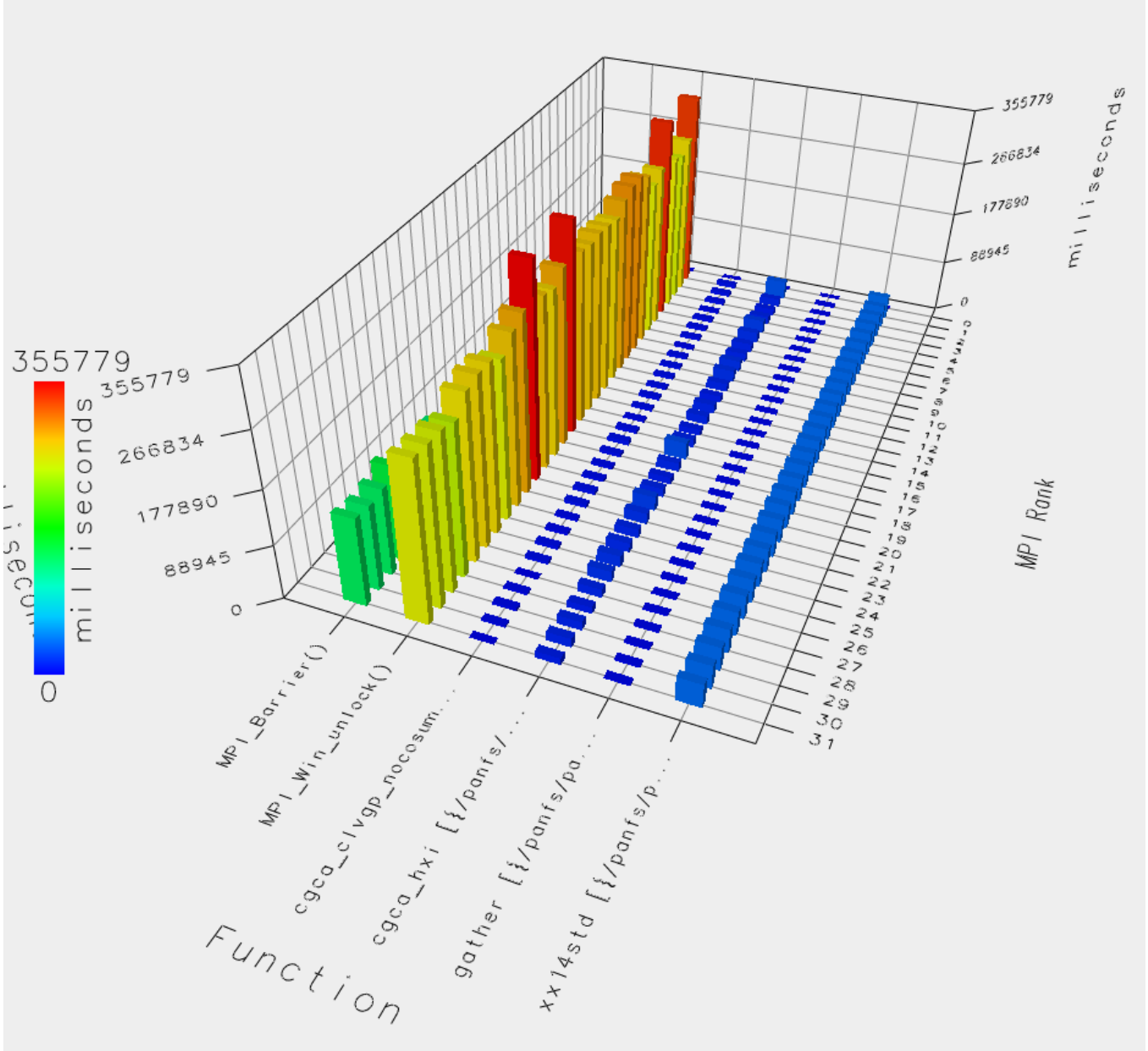
Profiling with cgca_pfem_map



Runtimes and scaling for ParaFEM/CGPACK MPI/coarray miniapp with cgca_pfem_map and cgca_pfem_cenc.
cgca_pfem_map or cgca_pfem_cenc are called only once during the execution of the miniapp. Hence only a minor improvement is obtained, only from about 1000 cores.

Table of Contents

TAU profiling: Intel 16 coarray implementation - MPI RMA



2x 16-core nodes, 32 images. Poor optimisation?

Table of Contents

CA - coarray (over)synchronisation?

```
call cgca_nr( space )  ! sync all inside
call cgca_rt( grt )    ! sync all inside
call cgca_sld( space ) ! sync all inside
call cgca_igb( space )
sync all
call cgca_hxi( space )
sync all
call cgca_gbs( space )
sync all
call cgca_hxi( space )
sync all
call cgca_gcu( space ) ! local routine no sync
```

- All images sync with their 26 neighbours.
- Some routines have sync inside.
- Other sync responsibility is left to end user.

Fortran 2015 events: more flexible than SYNC IMAGES [12]

```
use, intrinsic iso_fortran_env, only:event_type
type(event_type) :: var[:,:,:]
integer, allocatable :: space(:,:,:,:)[:,:,:]
integer :: errstat, myrank(3)
! allocate var, space
myrank = this_image( space )
! do some work, then notify neighbours
event post( var[ myrank(1)-1, myrank(2),      &
                 myrank(3) ], stat=errstat)
! 25 more posts
:
event wait(var, until_count=26, stat=errstat)
! when all 26 neighbours posted, continue work
:
```

Future: thread level parallelism: OpenMP, DO CONCURRENT

```
main: do iter = 1,N
 do x3 = lbr(3), ubr(3)
 do x2 = lbr(2), ubr(2)
 do x1 = lbr(1), ubr(1)
  live: if ...
   call cgca_clvgn( clvgflag )
   if ( clvgflag ) call sub( space )
  end if live
 end do
 end do
 end do
 call co_sum( clvgglob )
 sync all
 call cgca_hxi( space )
 sync all
 call cgca_dacf( space )
```

Conclusions

- Fortran coarrays are an ideal match for cellular automata
- Hybrid coarray+MPI multi-scale fracture framework is feasible
- Scaling up to 7k cores currently, work ongoing
- Profiling/tracing tools: CrayPAT, TAU, Score-P, Scalasca - coarray support is improving
- Coarray synchronisation - major issue: data integrity & performance

Acknowledgements

- This work was funded under the embedded CSE programme of the ARCHER UK National Supercomputing Service (http://www.archer.ac.uk), archer.ac.uk
- This work was carried out using the computational facilities of the Advanced Computing Research Centre, University of Bristol - https://www.acrc.bris.ac.uk, www.acrc.bris.ac.uk

[1] C. A. Gandin and M. Rappaz. A coupled finite element-cellular automaton model for the prediction of dendritic grain structures in solidification processes. *Acta Met. and Mat.*, 42(7):2233–2246, 1994.

[2] C. Zheng and D. Raabe. Interaction between recrystallization and phase transformation during intercritical annealing in a cold-rolled dual-phase steel: A cellular automaton model. *Acta Materialia*, 61:5504–5517, 2013.

[3] A. Shterenlikht and I. C. Howard. The CAFE model of fracture – application to a TMCR steel. *Fatigue Fract. Eng. Mater. Struct.*, 29:770–787, 2006.

[4] S. Das, A. Shterenlikht, I. C. Howard, and E. J. Palmiere. A general method for coupling microstructural response with structural performance. *Proc. Roy. Soc. A*, 462:2085–2096, 2006.

[5] A. Shterenlikht and L. Margetts. Three-dimensional cellular automata modelling of cleavage propagation across crystal boundaries in polycrystalline microstructures. *Proc. Roy. Soc. A*, 471:20150039, 2015.

[6] A. Shterenlikht, L. Margetts, L. Cebamanos, and D. Henty. Fortran 2008 coarrays. *ACM Fortran Forum*, 34:10–30, 2015.

[7] A. Shterenlikht. Fortran coarray library for 3D cellular automata microstructure simulation. In M. Weiland, A. Jackson, and N. Johnson, editors, *Proc. 7th PGAS Conf., 3-4 October 2013, Edinburgh, Scotland, UK*, pages 16–24. The University of Edinburgh, 2014.

[8] D. Henty, A. Jackson, C. Moulinec, and V. Szeremi. Performance of Parallel IO on ARCHER, version 1.1. ARCHER White Papers, 2015.

[9] I. M. Smith, D. V. Griffiths, and L. Margetts. *Programming the Finite Element Method*. Wiley, 5ed, 2014.

[10] A. Shterenlikht, S. Margetts, L. McDonald, and N. K. Bourne. Towards mechanism-based simulation of impact damage using exascale computing. In

Proc. 19th APS Conf. Shock Compression Condensed Matter SCCM-2015, JUN-2015, Tampa, Florida, USA, 2015.

[11] ISO/IEC JTC1/SC22/WG5 N2074. *TS 18508 Additional Parallel Features in Fortran*. 2015.

[12] B. Long. Additional parallel features in Fortran. *ACM Fortran Forum*, 35:16–23, 2016.