Science & Technology Facilities Council

Performance Studies of a Co-Array Fortran Applications Versus MPI

Mike Ashworth

Associate Director

Computational Science & Engineering Department

STFC Daresbury Laboratory

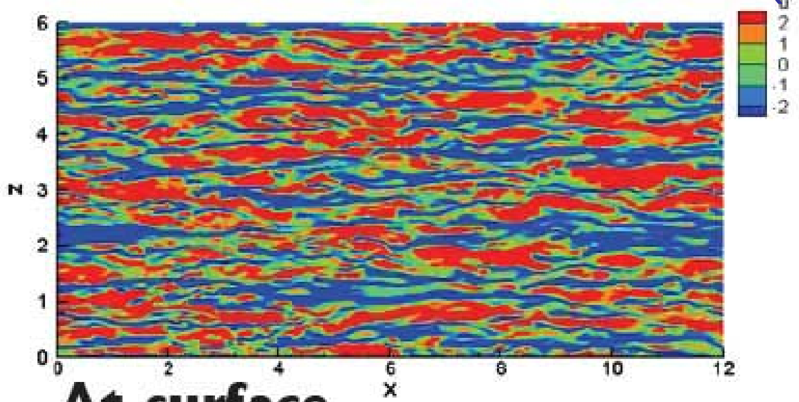
mike.ashworth@stfc.ac.uk

Outline

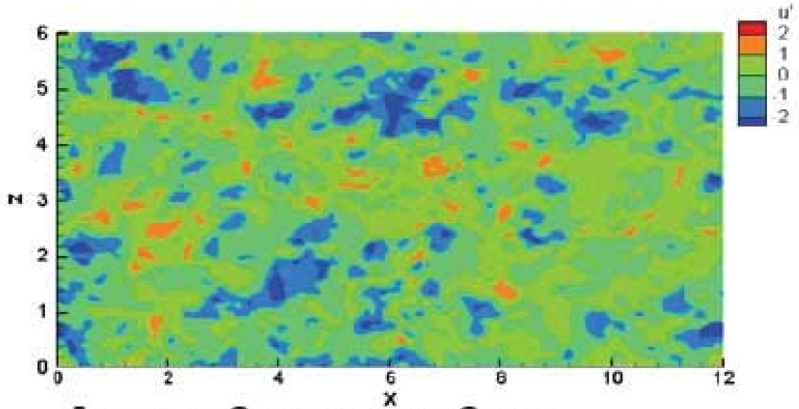
- The SBLI Scalable DNS Application
- The HECToR Cray XE6
- The Co-Array Implementation
- Performance Using Co-Arrays
- Conclusions

Shock Boundary-Layer Interaction (SBLI)

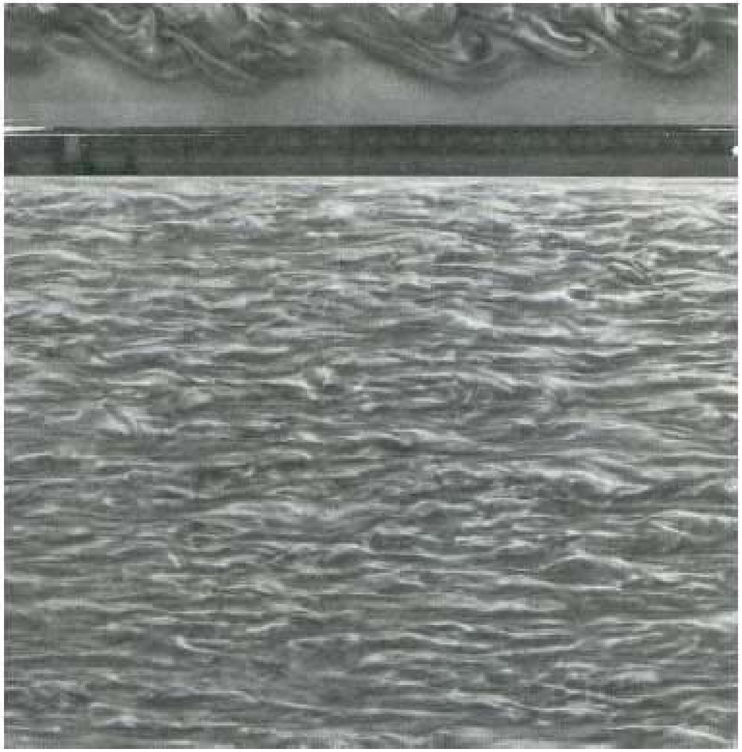
Direct Numerical Simulation (DNS) of near-wall turbulent flow



At surface



Away from surface



Experiment

ND Sandham, G Coleman et al, University of Southampton, UK

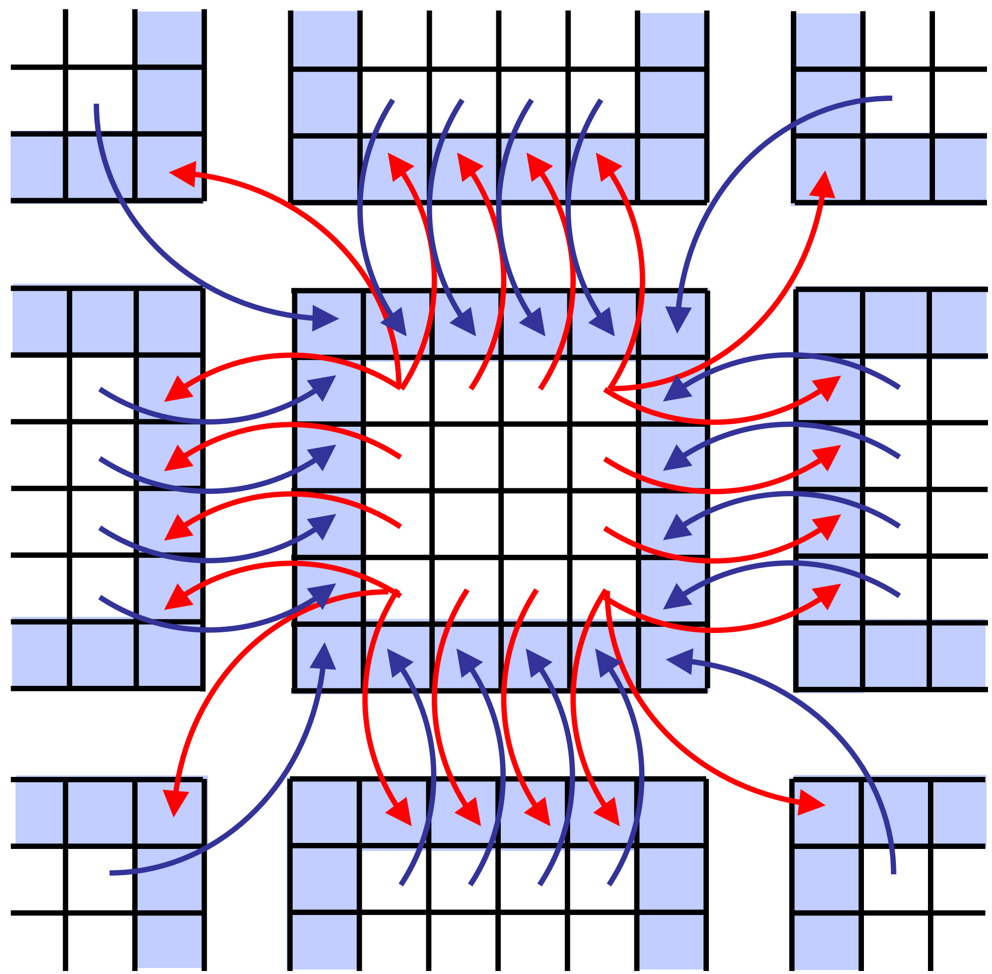
SBLI 3D grid partitioning with halo cells

Partitioned 3D grid with halo exchange

calculation cost:
scales as n^3

communication cost:
scales as n^2

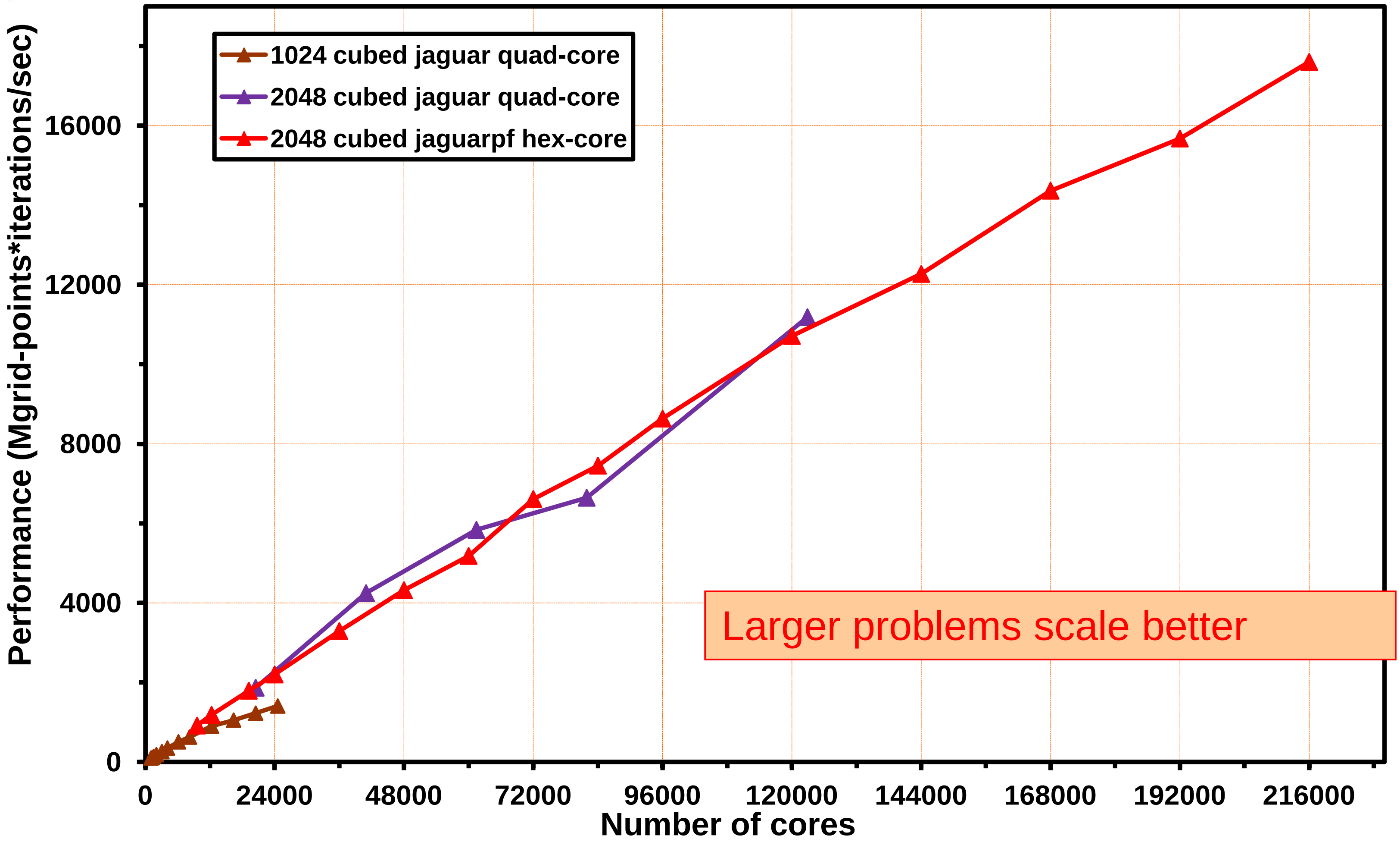
strong scaling:
increasing P
decreasing n
comms will dominate



Science & Technology Facilities Council

SBLI on Jaguar & Jaguarpf

Turbulent channel flow benchmark



Larger problems scale better

Science & Technology Facilities Council

The HECToR UK National Supercomputer

Cray XE6

30 cabinets

2816 compute nodes

90,112 cores

Each node has two 16-core AMD 2.3GHz Interlagos processors 32 GB memory

Cray Gemini routers

cce 7.4.4 –hcaf –O3

Co-Array version of SBLI
Ashby & Reid (2008)

CUG 2008

Science & Technology Facilities Council
Rutherford Appleton Laboratory

Migrating A Scientific Application from MPI to Coarrays

John Ashby and John Reid
HPCx Consortium
Rutherford Appleton Laboratory
STFC
UK

CUG 2008
Crossing the Boundaries

Allocatable arrays

Co-Arrays themselves cannot be allocatable as the address needs to be the same on all images

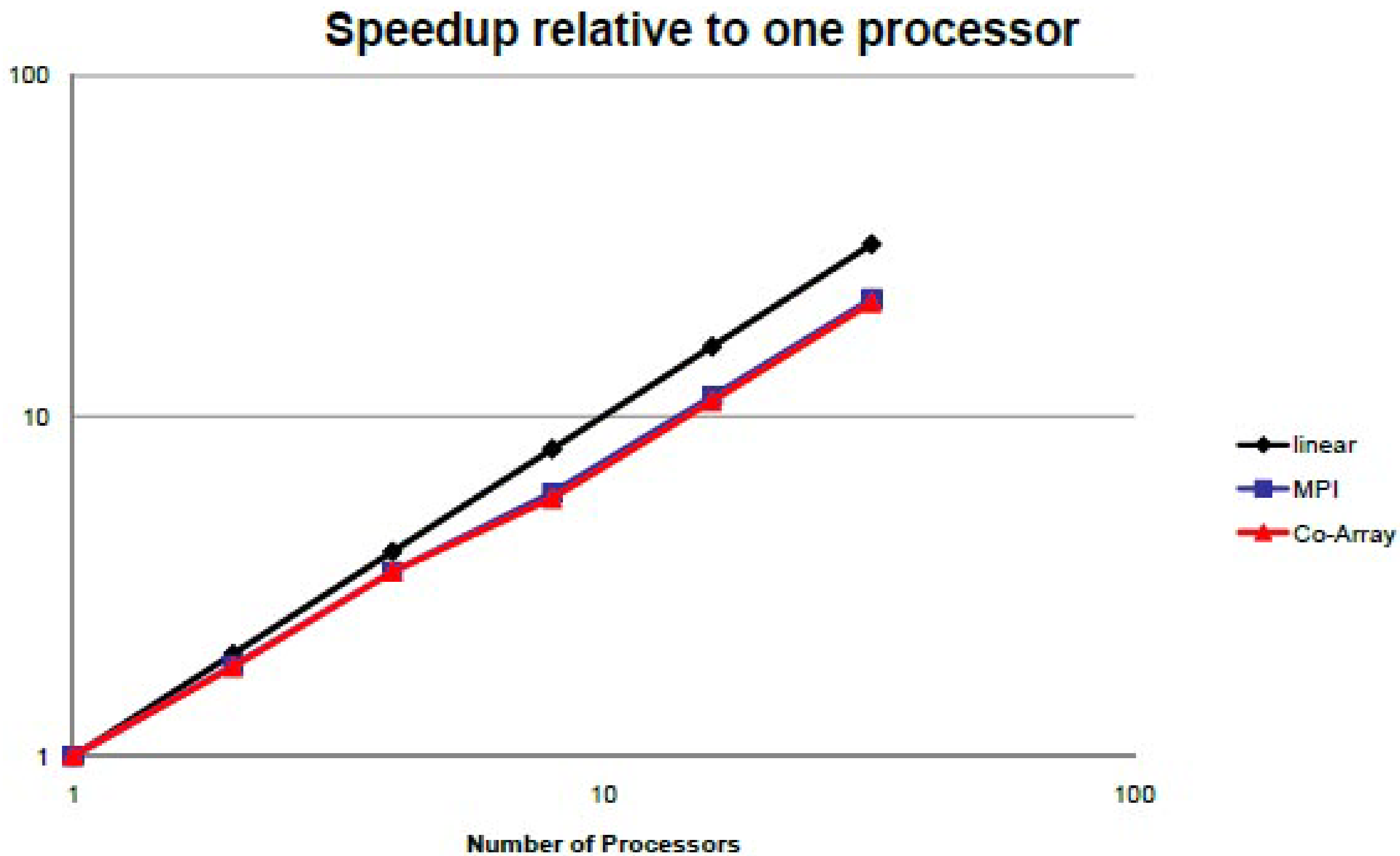
In order to maintain dynamic allocations Ashby & Reid use allocatable components:

```
type co_double_4
  double precision, allocatable, dimension(:,:,:,:):: array
end type co_double_4

type(co_double_4), save, dimension[*] :: q

q%array(i,j,k,n) = ...
```

Ashby & Reid (2008) Performance on Cray X1



Using the Ashby & Reid code out of the box did not immediately yield the same performance as MPI on the XE6

Science & Technology Facilities Council

Co-array options which were tested

Original: halo transfer using co-array assignment

Packed: halo data packed into contiguous array, transfer using co-array assignment, unpacked

F90: co-array assignment using F90 array syntax

F77: co-array assignment using F77 DO loops

Push: co-array assignment pushes data to remote processor (put); co-array on LHS

Pull: co-array assignment pulls data from remote processor (get); co-array on RHS

Science & Technology Facilities Council

Original code for a single halo transfer

```
call sync_all()
a[dimxm(1),dimxm(2),dimxm(3)]%array(nxp[procxm]+1:
   &               nxp[procxm]+xhalo,:,:) =
   &               a%array(1:xhalo,:,:)
call sync_all()
```

PUSH ⇧

PULL ⇩

Repeated six times for
north, south
east, west
up, down

```
call sync_all()
a%array(1-xhalo:0,:,:) =
   &  a[dimxm(1),dimxm(2),dimxm(3)]
   &   %array(nxp[procxm]-xhalo+1:nxp[procxm],:,:)
call sync_all()
```

Code with packing for a single halo transfer

```
xhsize = xhalo*(nyd+2*yhalo)*(nzd+2*zhalo)
allocate (ahixm%array(xhsize),stat=ialloc)
allocate (ahoxm%array(xhsize),stat=ialloc)
l=0
do k=1-zhalo,nzp+zhalo
   do j=1-yhalo,nyp+yhalo
      do i=1,xhalo
         l=l+1
         ahoxm%array(l) = a%array(i,j,k)
       enddo
   enddo
 enddo
call sync_all()
ahixm[procxm]%array(1:xhsize) = ahoxm%array(1:xhsize)
call sync_all()
l=0
do k=1-zhalo,nzp+zhalo
   do j=1-yhalo,nyp+yhalo
       do i=1,xhalo
         l=l+1
         a%array(i-xhalo,j,k) = ahixm%array(l)
       enddo
    enddo
enddo
deallocate(ahoxm%array,ahixm%array,ahoxp%array,ahixp%array)
```

Repeated six times for
north, south
east, west
up, down

Notes on sync_all

In halo exchange sync_all() could be replaced with sync_images(team), where team is the set of nearest neighbours

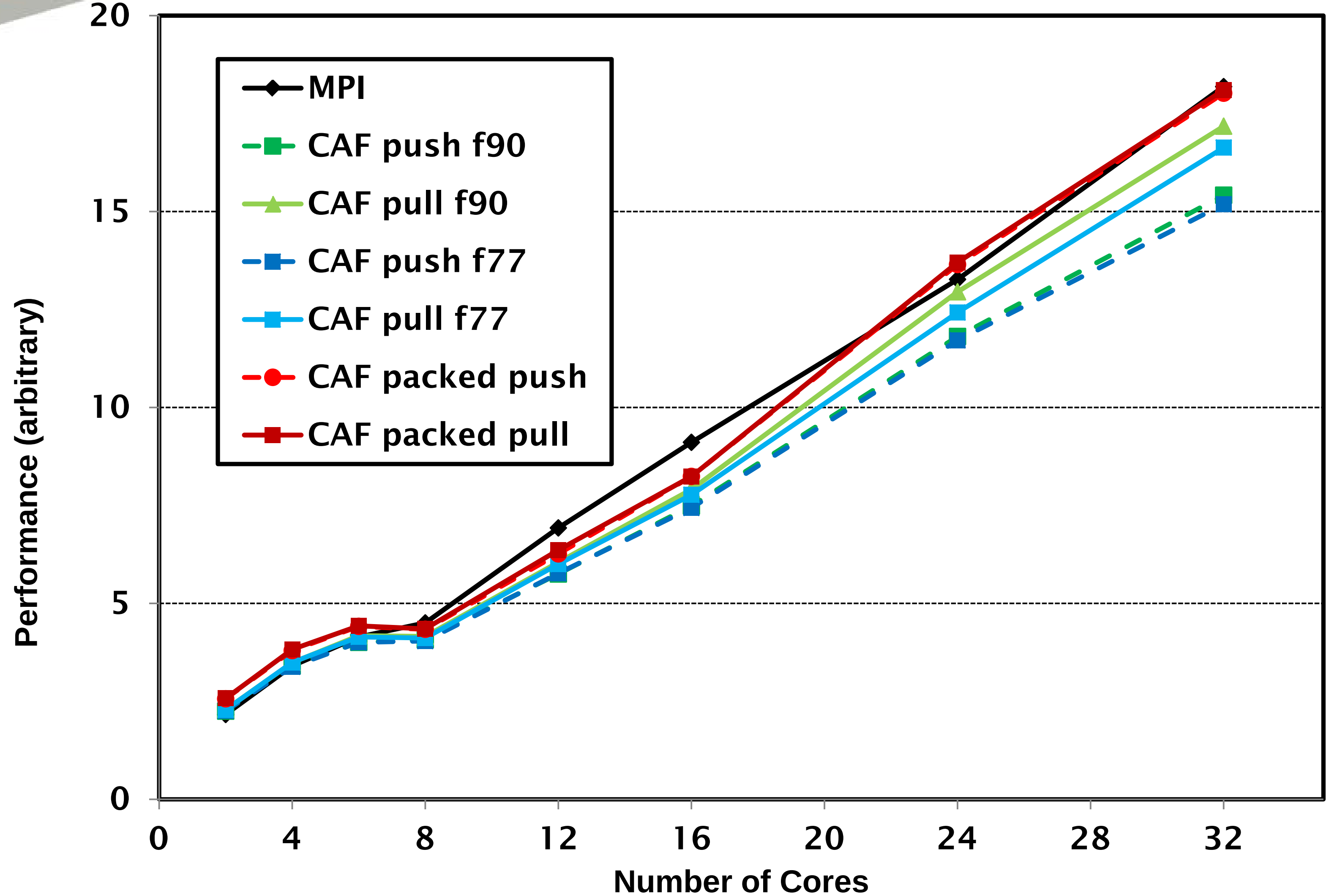
However sync_images is only quicker than sync_all on large core counts - see Henty (2012) at this conference

Sync_all is quicker than MPI_Barrier, but **IT NEEDS TO BE**

Sync_all is needed frequently, before and after every halo put/get

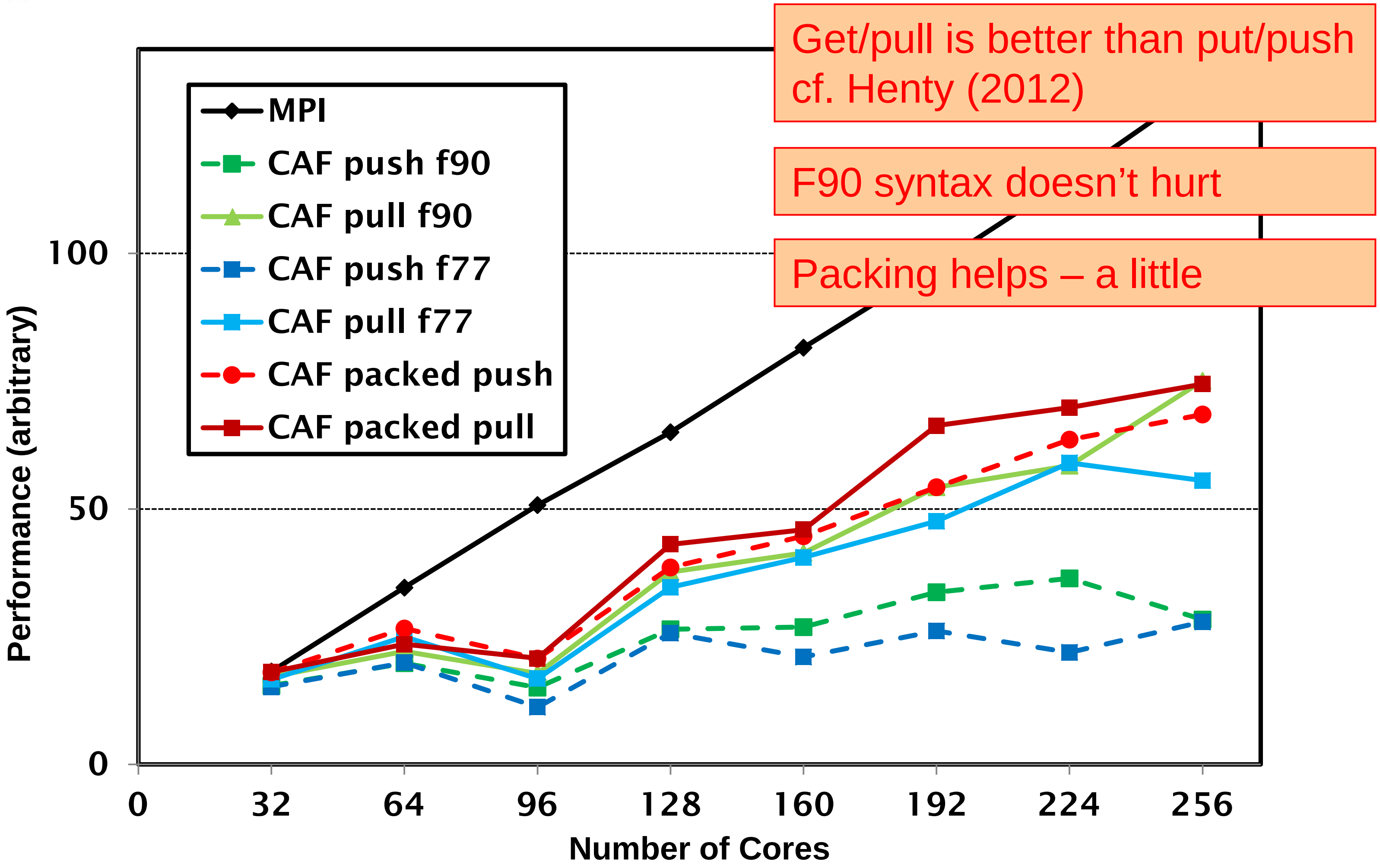
A good MPI code doesn't need MPI_Barrier, it self-synchronises through MPI_Send/MPI_Recv

Performance of Co-Arrays on the Cray XE6 – ON NODE



Science & Technology Facilities Council

Performance of Co-Arrays on the Cray XE6 – OFF NODE



Does CrayPAT help?

CAF push f90
256 cores

```
 Samp%  |   Samp  |  Imb.  |  Imb.  |Group
        |         |  Samp  | Samp%  | Function
        |         |        |        |  PE=HIDE

 100.0% |   2345 |    -- |    -- |Total
|--------------------------------------------------
|  80.9% |   1897 |    -- |    -- |USER
||-------------------------------------------------
||  18.6% | 436.594 | 280.41 |  39.3% |pdns3d_
||  15.9% | 372.070 | 138.93 |  27.3% |swap_x_4_
||  13.1% | 306.207 | 212.79 |  41.2% |period_xmb_
||   8.3% | 195.078 | 296.92 |  60.6% |period_z_4_
||   7.6% | 177.176 |  24.82 |  12.3% |rhs_
||   3.9% |  91.125 |  94.88 |  51.2% |bparam_
||   2.5% |  58.531 |  11.47 |  16.4% |pow
||   2.1% |  49.410 |  16.59 |  25.2% |ent_euler_
||   1.5% |  35.062 |  14.94 |  30.0% |d1xi_2$deriv_
||   1.4% |  33.066 |  17.93 |  35.3% |d1eta_2$deriv_
||   1.2% |  27.891 | 136.11 |  83.3% |period_x_
||   1.2% |  27.277 |  21.72 |  44.5% |d1z_2$deriv_
||=================================================
|  19.1% | 447.383 |    -- |    -- |ETC
||-------------------------------------------------
||  16.0% | 374.820 | 109.18 |  22.6% |syscall
||   2.9% |  68.617 |  31.38 |  31.5% |__strcmp
|==================================================
```

Ashby & Reid (2008) Conclusions on the Cray X1 & X1E

+ Simple assignment statements replace MPI calls
+ No need to pack and unpack data (scope for programming errors)
+ Simpler, shorter, more maintainable code
+ Performance comparable with MPI
- Added indirection through allocatable components

Revised (2012) Conclusions on the Cray XE6

? Simple assignment statements replace MPI calls

? No need to pack and unpack data (scope for programming errors)

? Simpler, shorter, more maintainable code

- Performance not comparable with MPI

- Added indirection through allocatable components

? depends on whether we need packing

Thank you for your attention

Mike Ashworth

mike.ashworth@stfc.ac.uk