Exploring the Performance Potential of Chapel

Richard Barrett, Sadaf Alam, and Stephen Poole

Scientific Computing Group
National Center for Computational Sciences

Future Technologies Group
Computer Science and Math Division

Oak Ridge National Laboratory

Cray User Group 2008, Helsinki
May 7, 2008

OAK RIDGE
National Laboratory

Chapel Status

- Compiler version 0.7, released April 15.
    - running on my Mac; also Linux, SunOS, CygWin
    - Initial release December 15, 2006.
    - End of summer release planned.
- Spec version 0.775
- Development team "optimally" responsive.

Productivity

Programmability

Performance

Portability

Robustness

Programmability: Motivation for "expressiveness"

"By their training, the experts in iterative methods expect to collaborate with users. Indeed, the combination of user, numerical analyst, and iterative method can be incredibly effective. Of course, by the same token, inept use can make any iterative
method not only slow but prone to failure. Gaussian elimination, in contrast, is a classical black box algorithm demanding no cooperation from the user.

Surely the moral of the story is not that iterative methods are dead, but that too little attention has been paid to the user's current needs?"

"Progress in Numerical Analysis",
Beresford N. Parlett,
SIAM Review, 1978.

"Expressive" language constructs

Syntax and semantics that enable:

- algorithmic description ← Programmability
- provide **intent** to compiler & RTS ← Performance

Prospective for Adoption

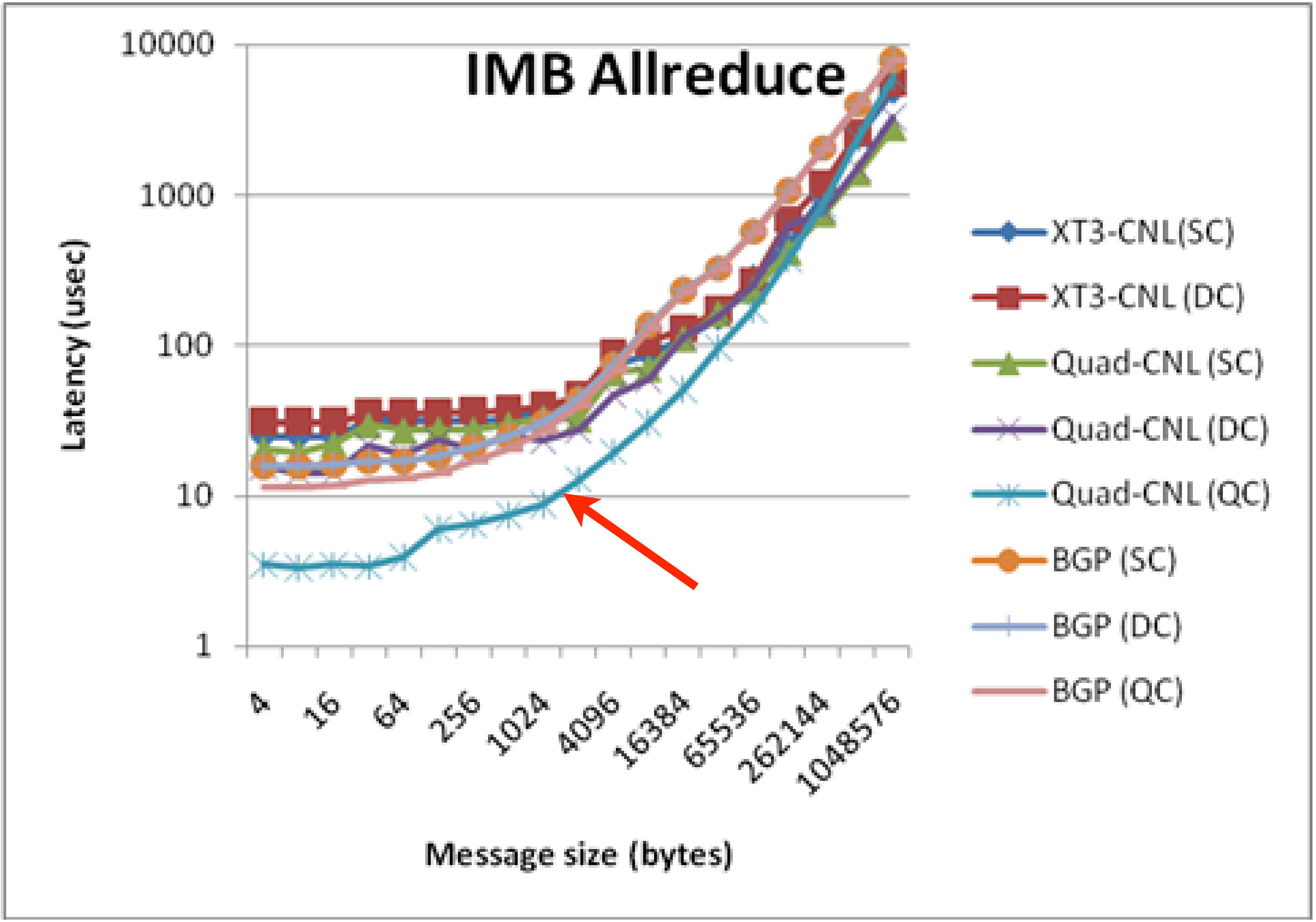
Must provide compelling reason

Performance

My view: Must exceed performance of MPI.

(Other communities may have different requirements.)

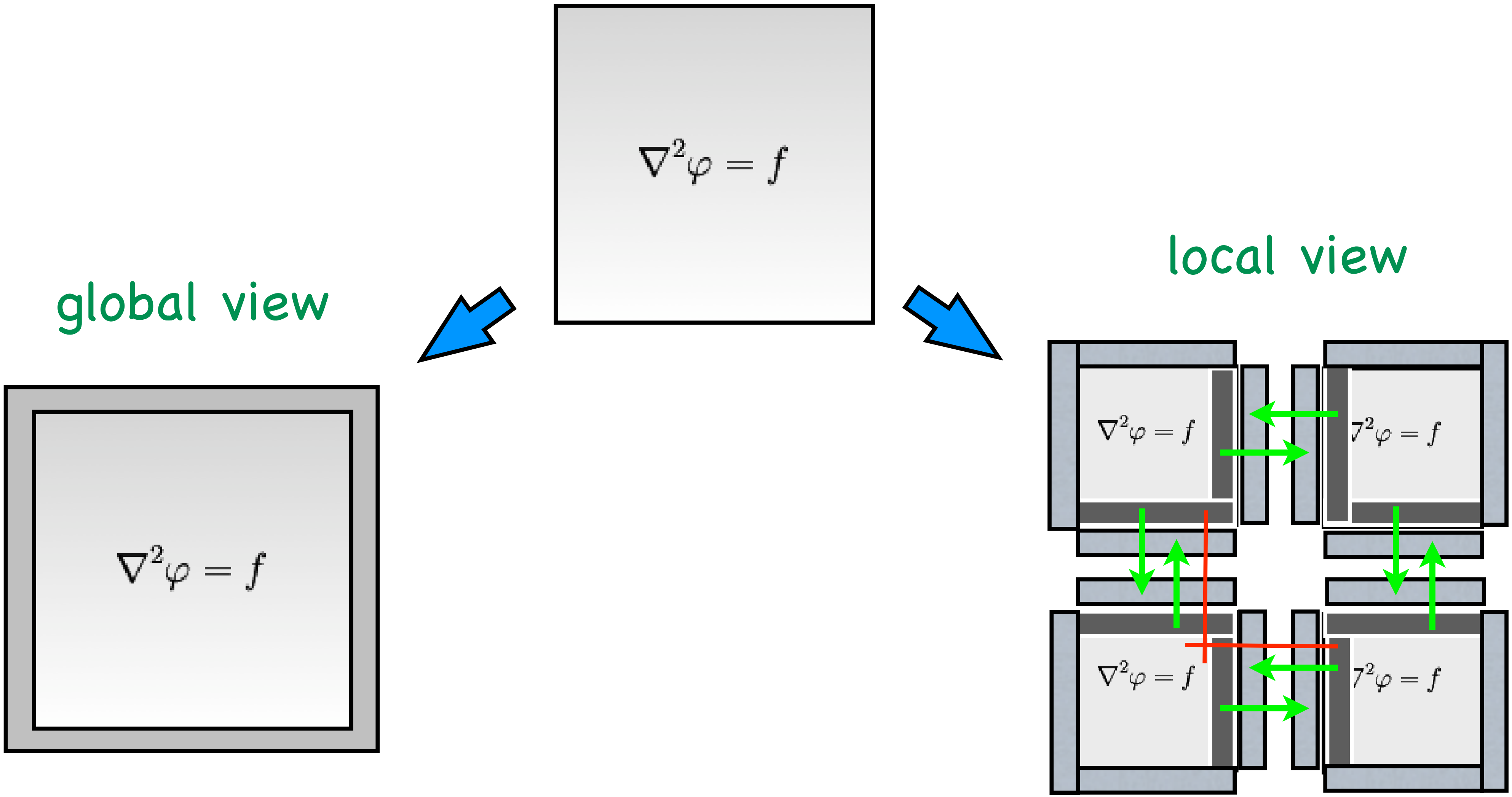
Rename "FORTRAN"


IMB Allreduce
Latency (usec)
10000
1000
100
10
1
4
16
64
256
1024
4096
16384
65536
262144
1048576
Message size (bytes)
XT3-CNL(SC)
XT3-CNL (DC)
Quad-CNL (SC)
Quad-CNL (DC)
Quad-CNL (QC)
BGP (SC)
BGP (DC)
BGP (QC)

The Chapel Memory Model

There ain't one.

Finite difference solution of Poisson's Eqn



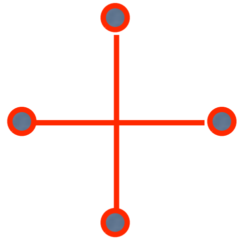
Solving Ax=b

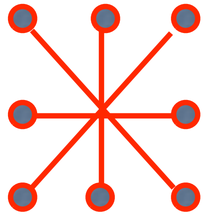
Method of Conjugate Gradients

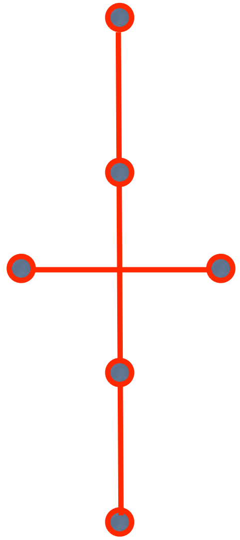
for i = 1, 2, ...
    solve $Mz^{(i-1)} = r^{(i-1)}$
    $\rho_{i-1} = r^{(i-1)T} z^{(i-1)}$
    if (i = 1)
        $p = z^{(0)}$
    else
        $\beta = \rho_{i-1} / \rho_{i-2}$
        $p = z^{(i-1)} + \beta p^{(i-1)}$
    end if
    $q = Ap$
    $\alpha = \rho_{i-1} / p^T q$
    $x = x^{(i-1)} + \alpha p$
    $r = r^{(i-1)} - \alpha q$
    check convergence; continue if necessary
end

"Linear Algebra", Strang
"Matrix Computations", Golub & Van Loan

Linear equations may often be defined as ``stencils" (Matvec, preconditioner)



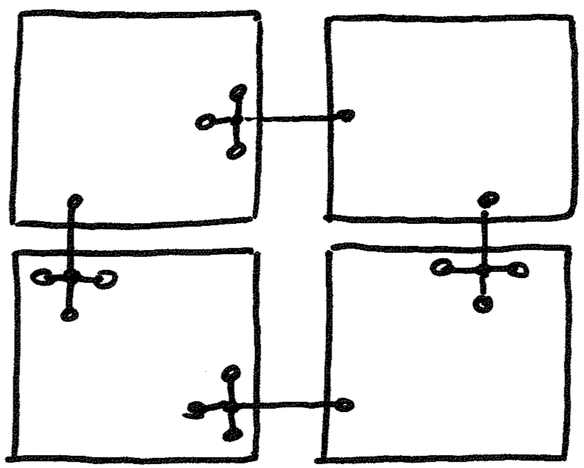




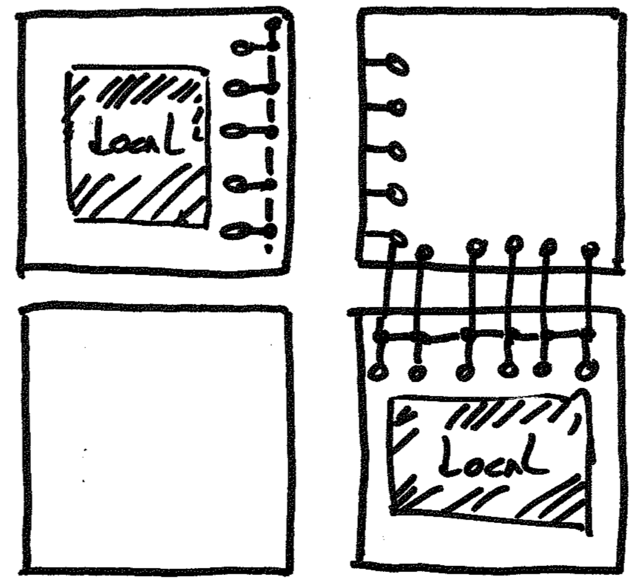
Fortran-MPI

```
CALL BOUNDARY_EXCHANGE ( ... )

DO J = 2, LCOLS+1
   DO I = 2, LROWS+1

      Y(I,J) =

        A(I-1,J-1) *X(I-1,J-1)  + A(I-1,J) *X(I-1,J) + A(I-1,J+1) X(I-1,J+1) +

        A (I,J-1)*X(I,J-1)      +  A(I,J)*X(I,J)      +   A (I,J+1) *X(I,J+1)  +

        A(I+1,J-1) X(I+1,J-1) + A(I+1,J)*X(I+1,J) + A(I+1,J+1)*X(I+1,J+1)

   END DO
END DO
```

Co-Array Fortran implementations



Load-it-when-you-need-it



Boundary sweep

```
IF ( NEIGHBORS(SOUTH) /= MY_IMAGE ) &
  GRID1( LROWS+2, 2:LCOLS+1 ) = GRID1( 2,2:LCOLS+1 )[NEIGHBORS(SOUTH)]
```

One-sided

Cray X1E

Heterogeneous, Multi-core

1024 Multi-streaming vector processors (MSP)

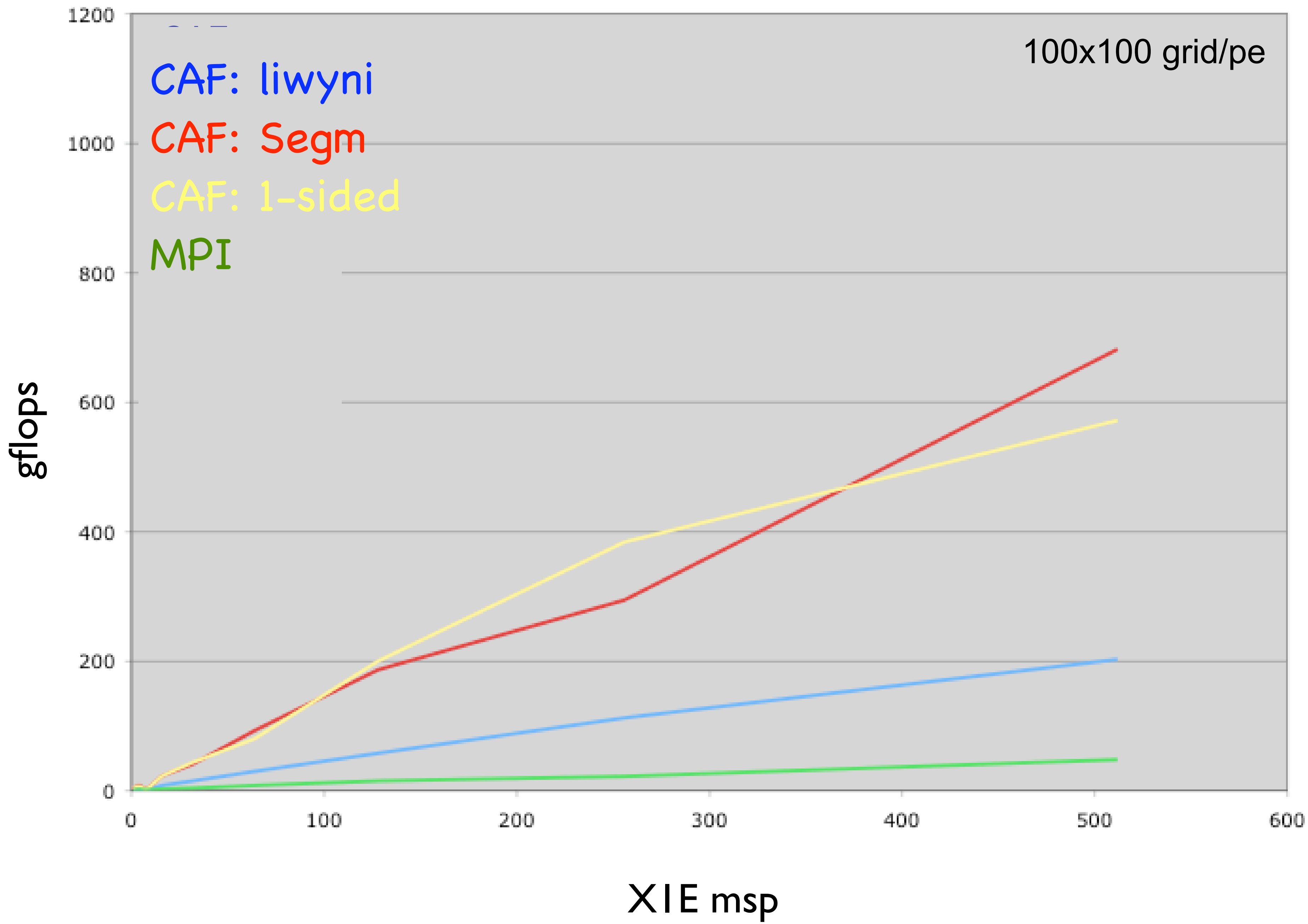
Each MSP

- 4 Single Streaming Processors (SSP)
- 4 scalar processors (400 MHz)
- Memory bw is roughly half cache bw.
- 2 MB cache
- 18+ GFLOP peak

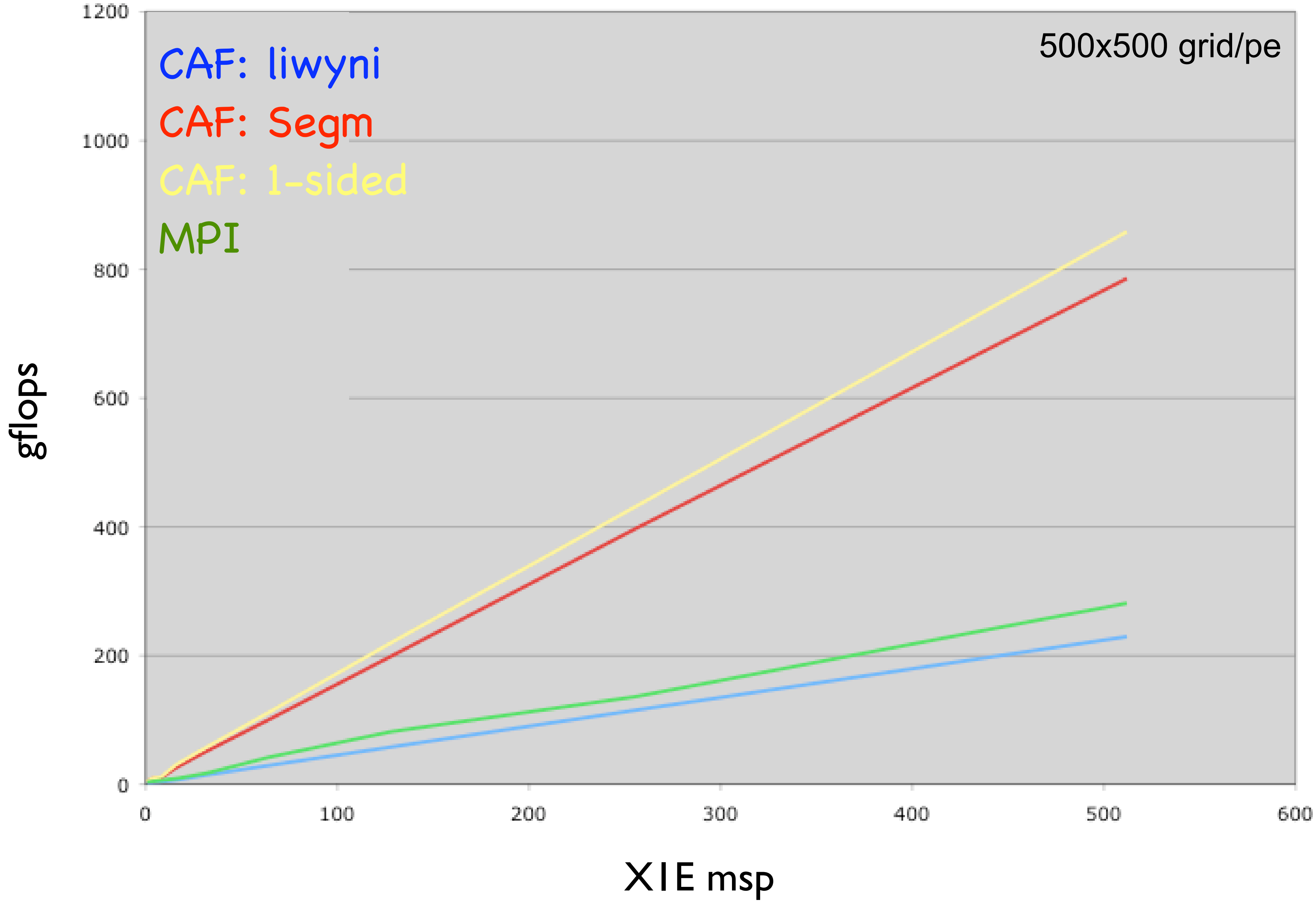
4 MSPs form a node

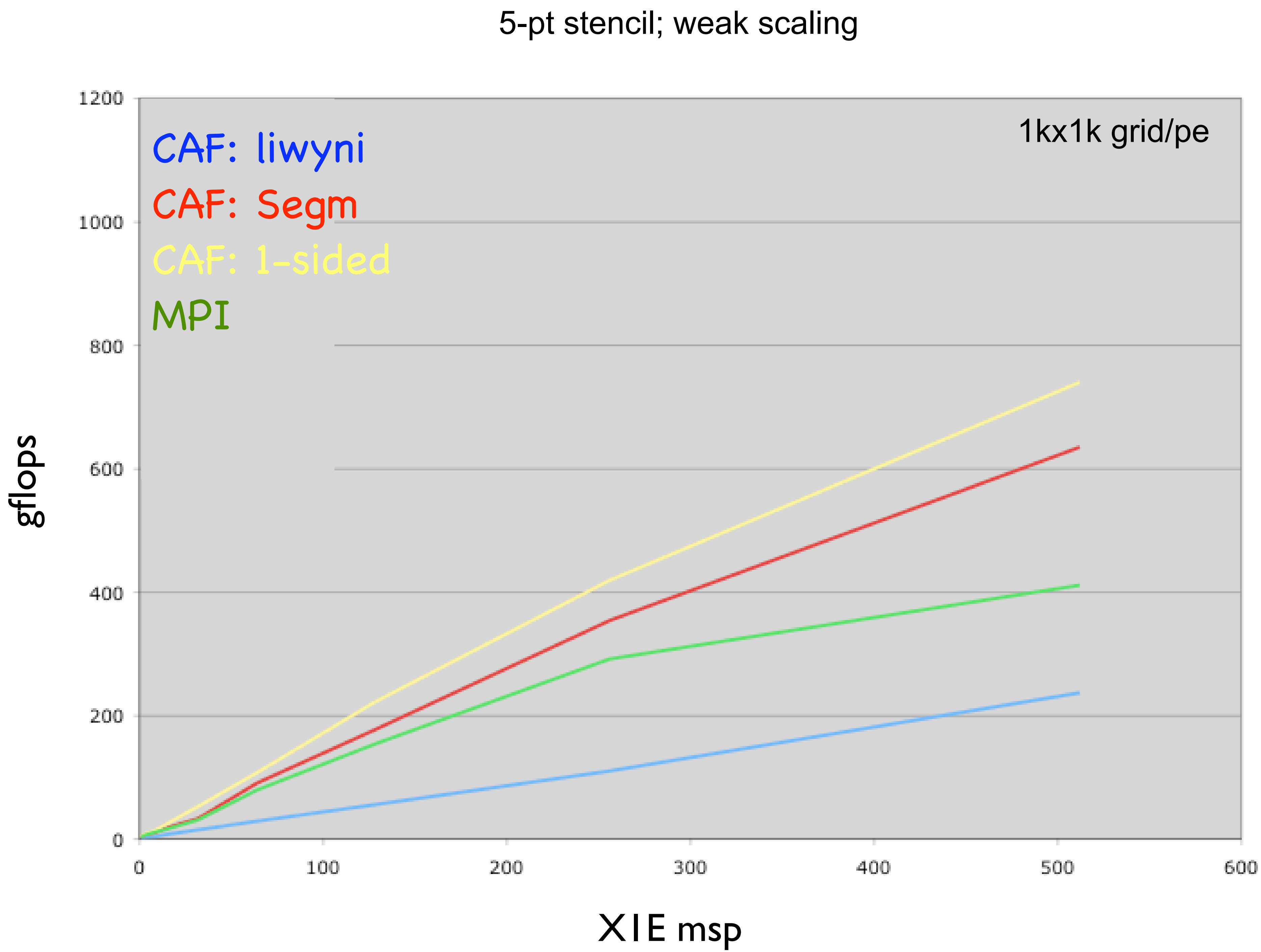
- 8 GB of shared memory.
- Inter-node load/store across network. 56 cabinets

5-pt stencil; weak scaling



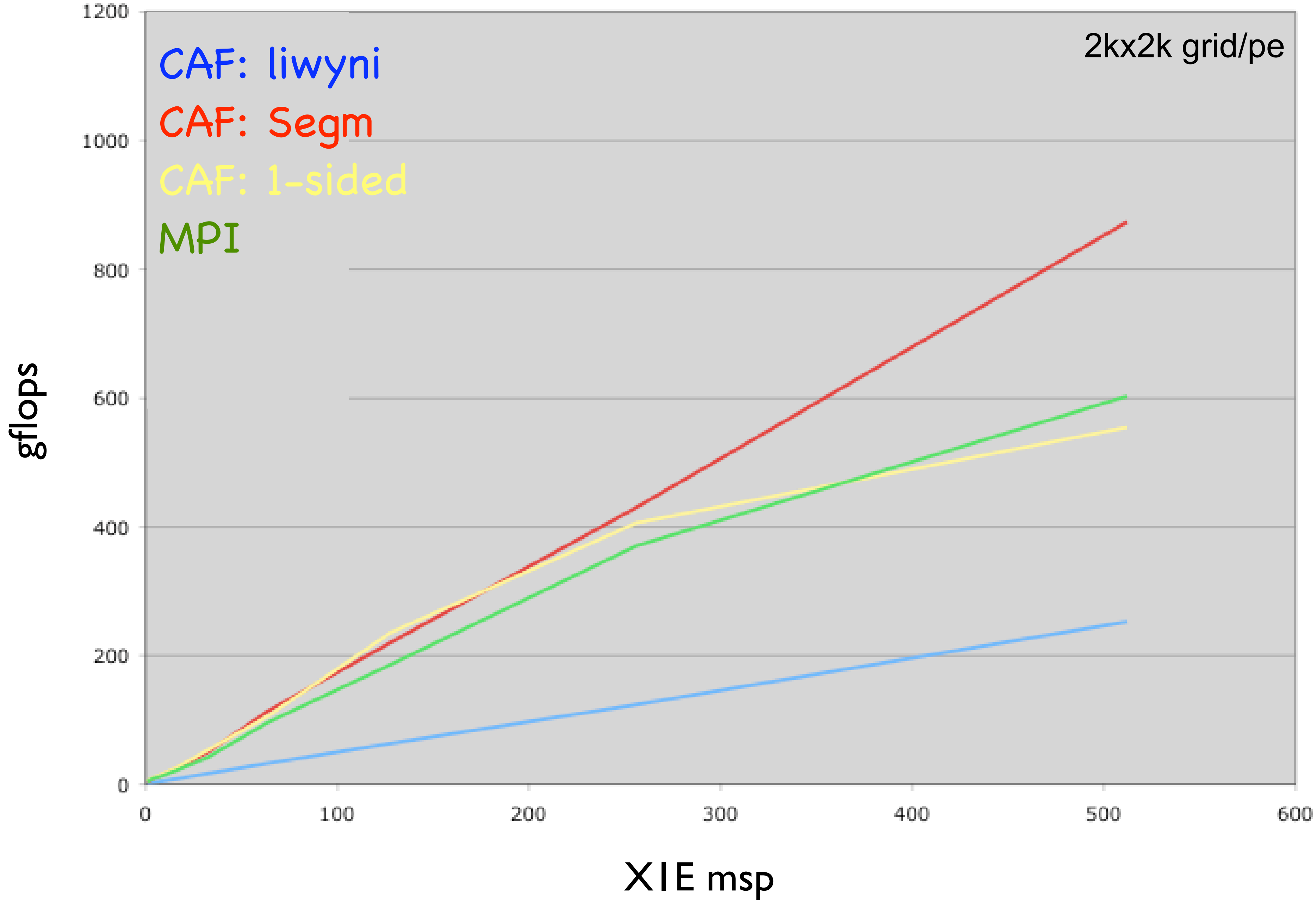
5-pt stencil; weak scaling



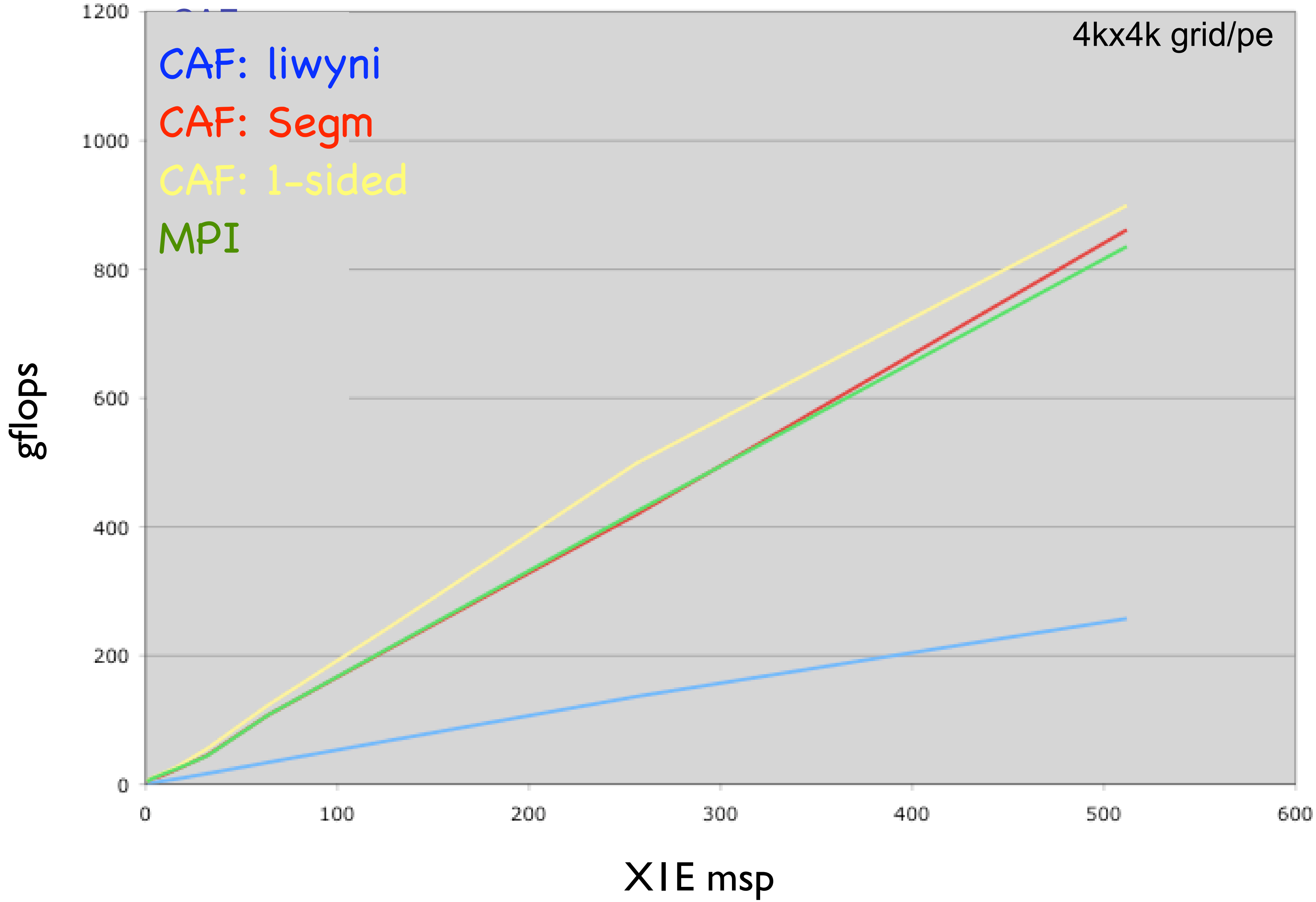


5-pt stencil; weak scaling
1kx1k grid/pe
CAF: liwyni
CAF: Segm
CAF: 1-sided
MPI
gflops
0
200
400
600
800
1000
1200
0
100
200
300
400
500
600
X1E msp

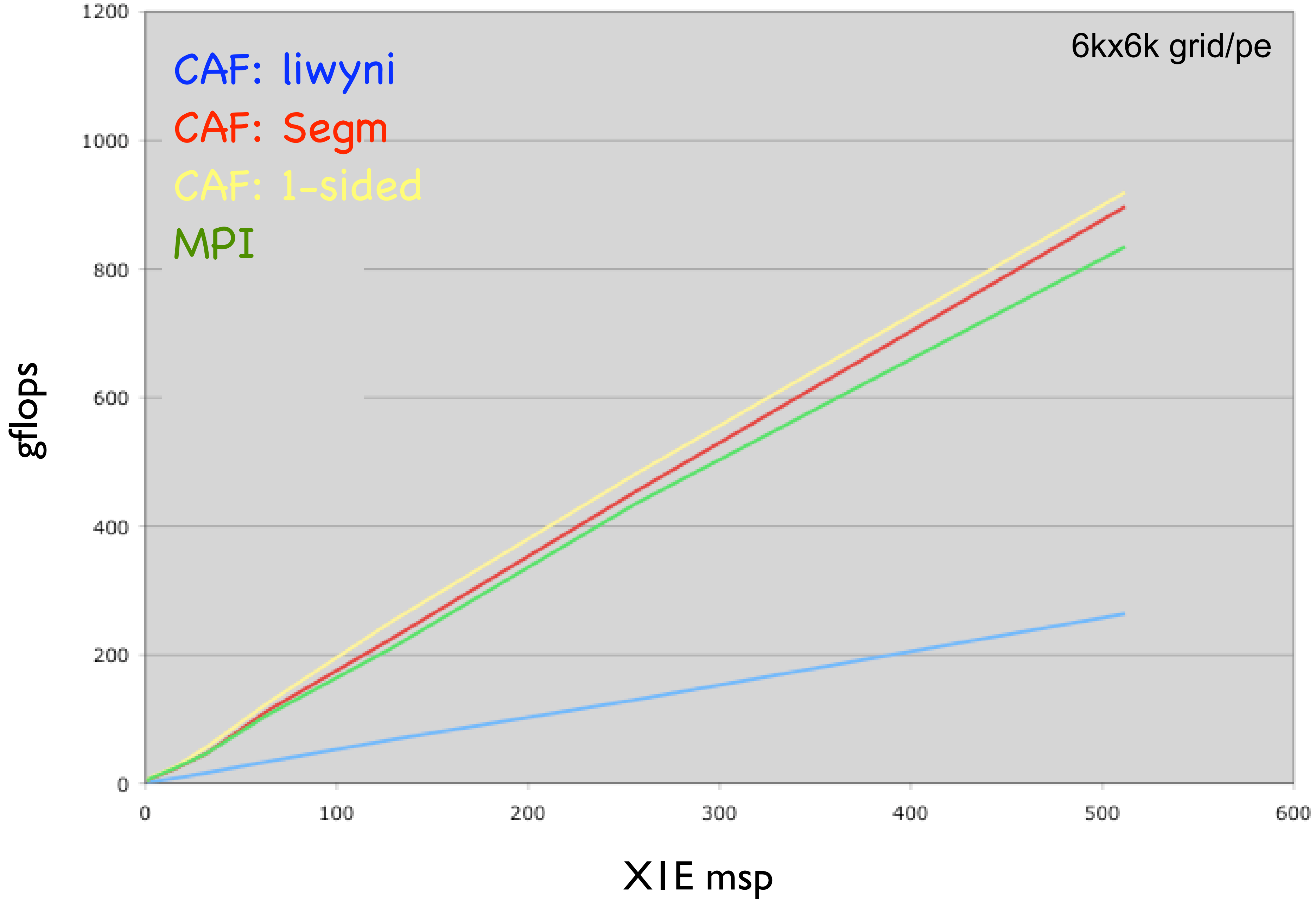
5-pt stencil; weak scaling

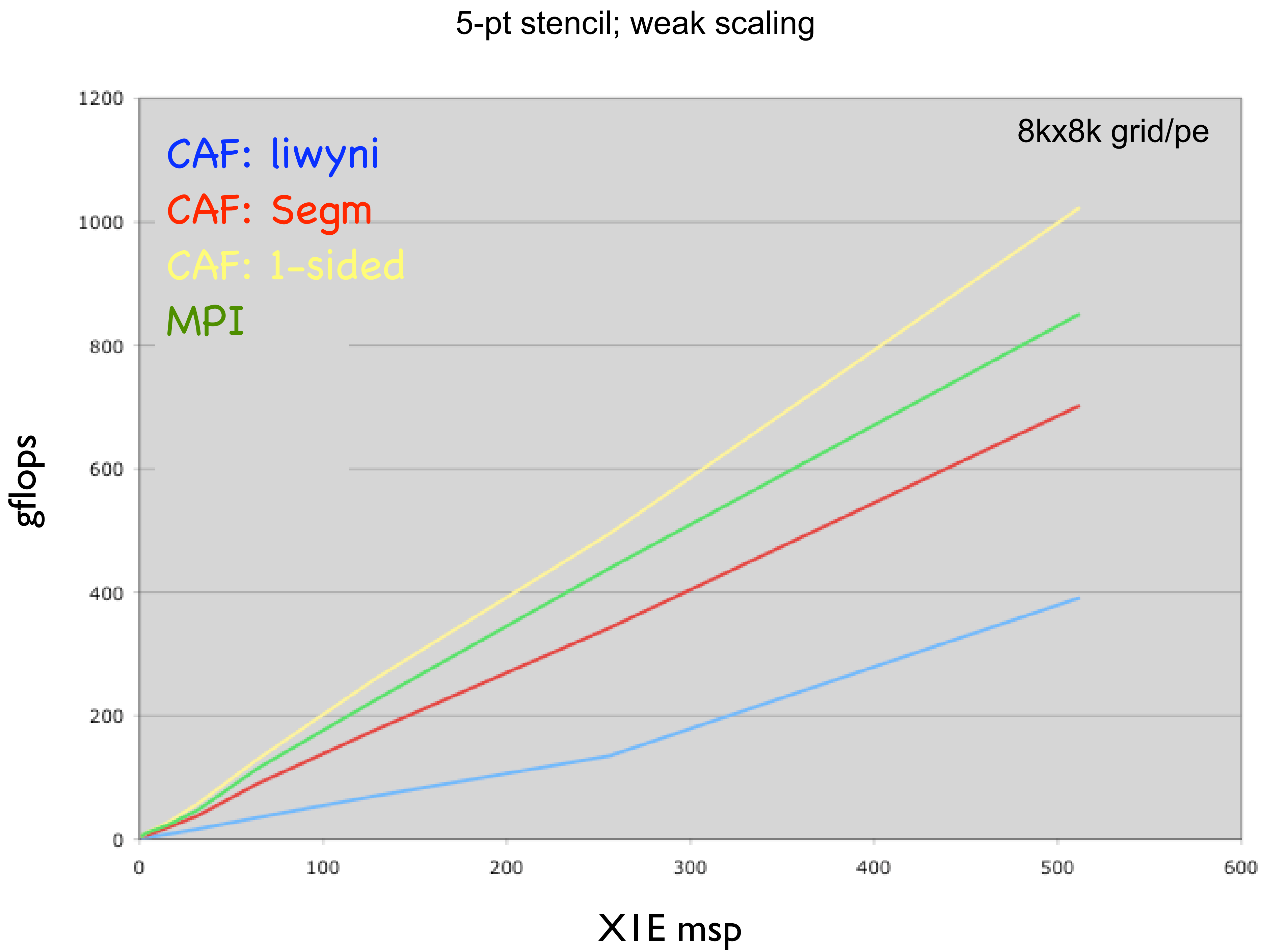


5-pt stencil; weak scaling



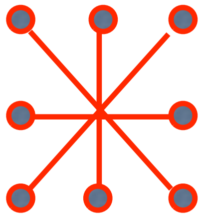
5-pt stencil; weak scaling



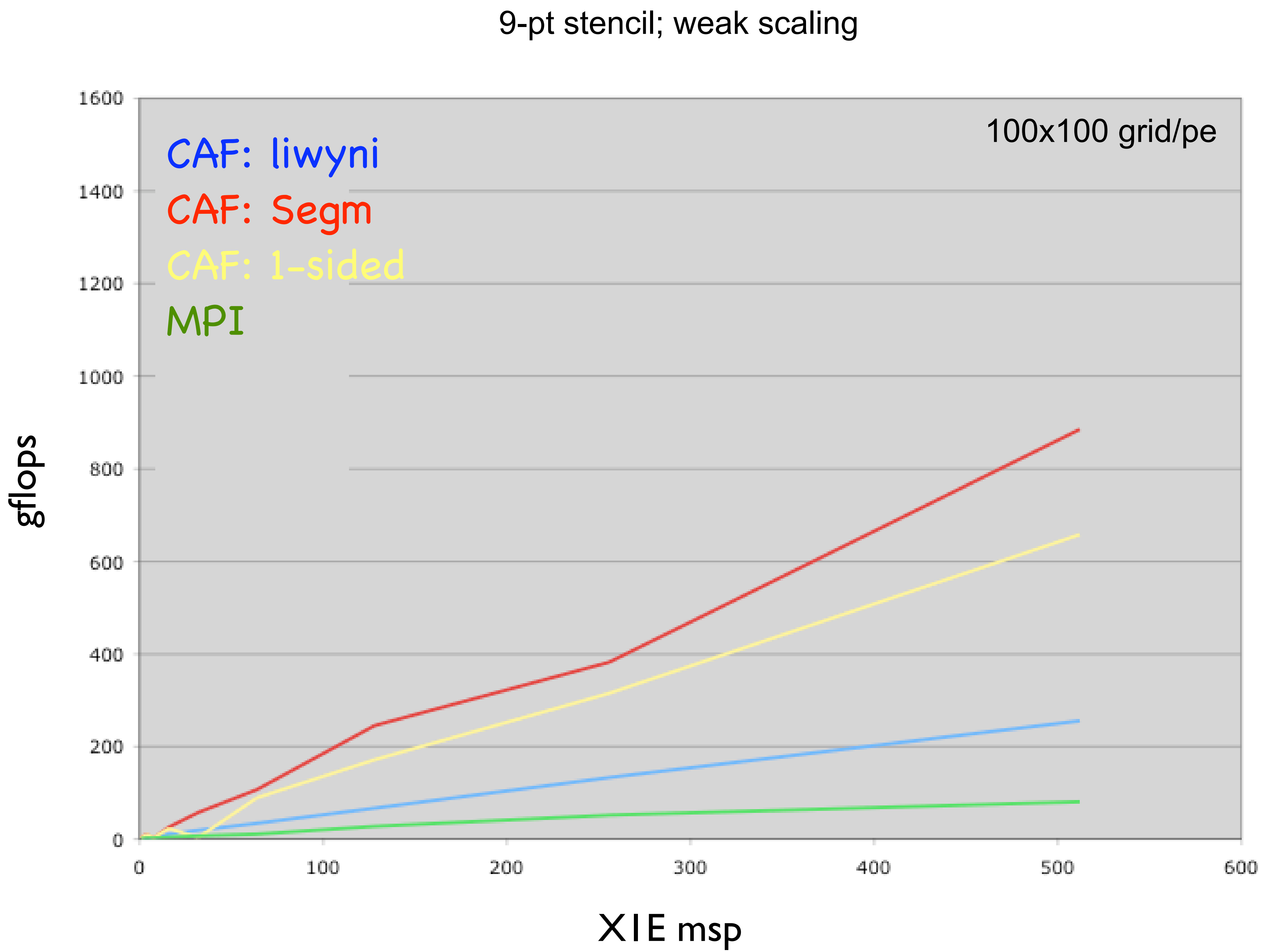

5-pt stencil; weak scaling
8kx8k grid/pe
CAF: liwyni
CAF: Segm
CAF: 1-sided
MPI
gflops
X1E msp
0
200
400
600
800
1000
1200
0
100
200
300
400
500
600

9-point stencil

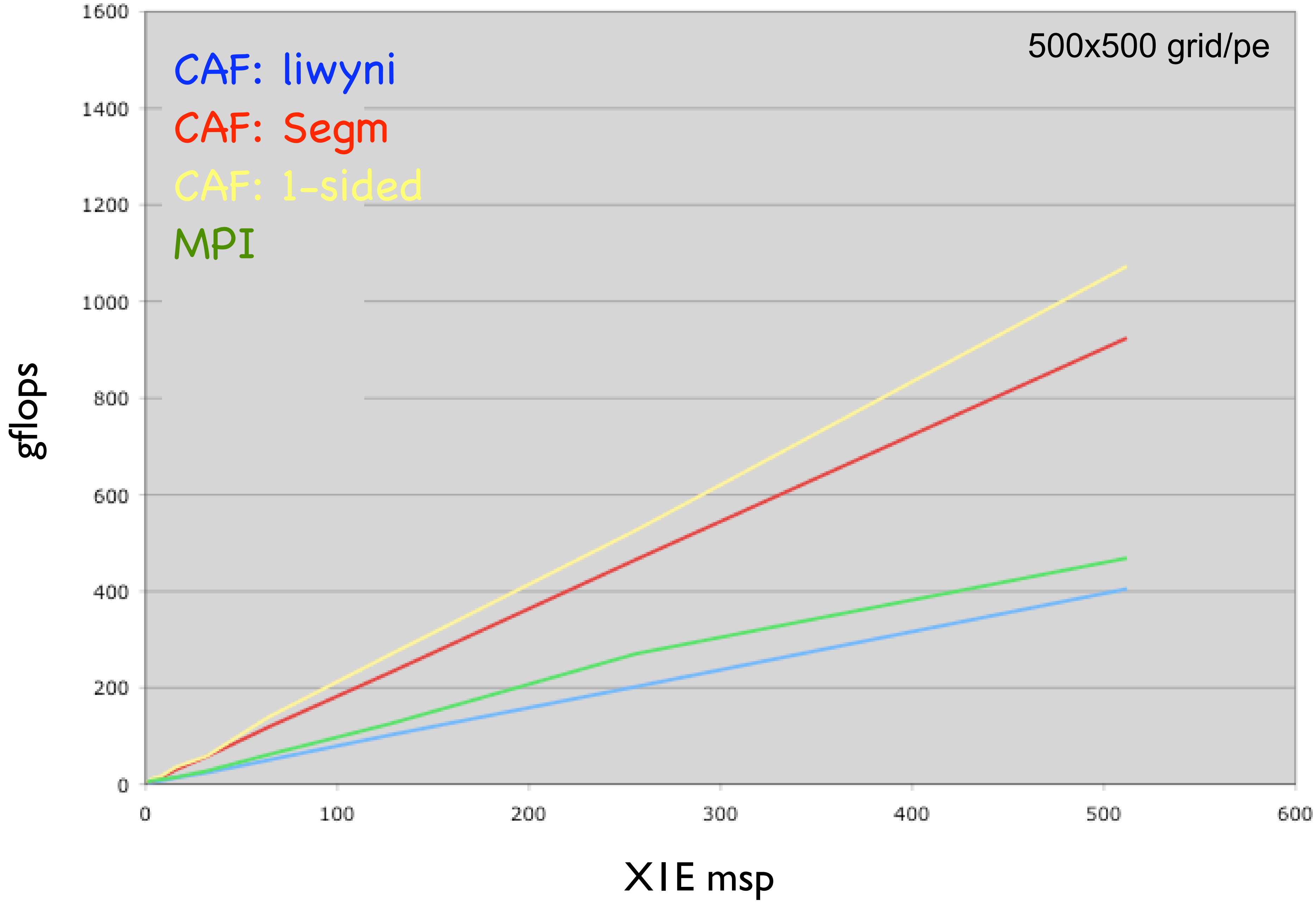
CAF: four extra partners
processes (corners)

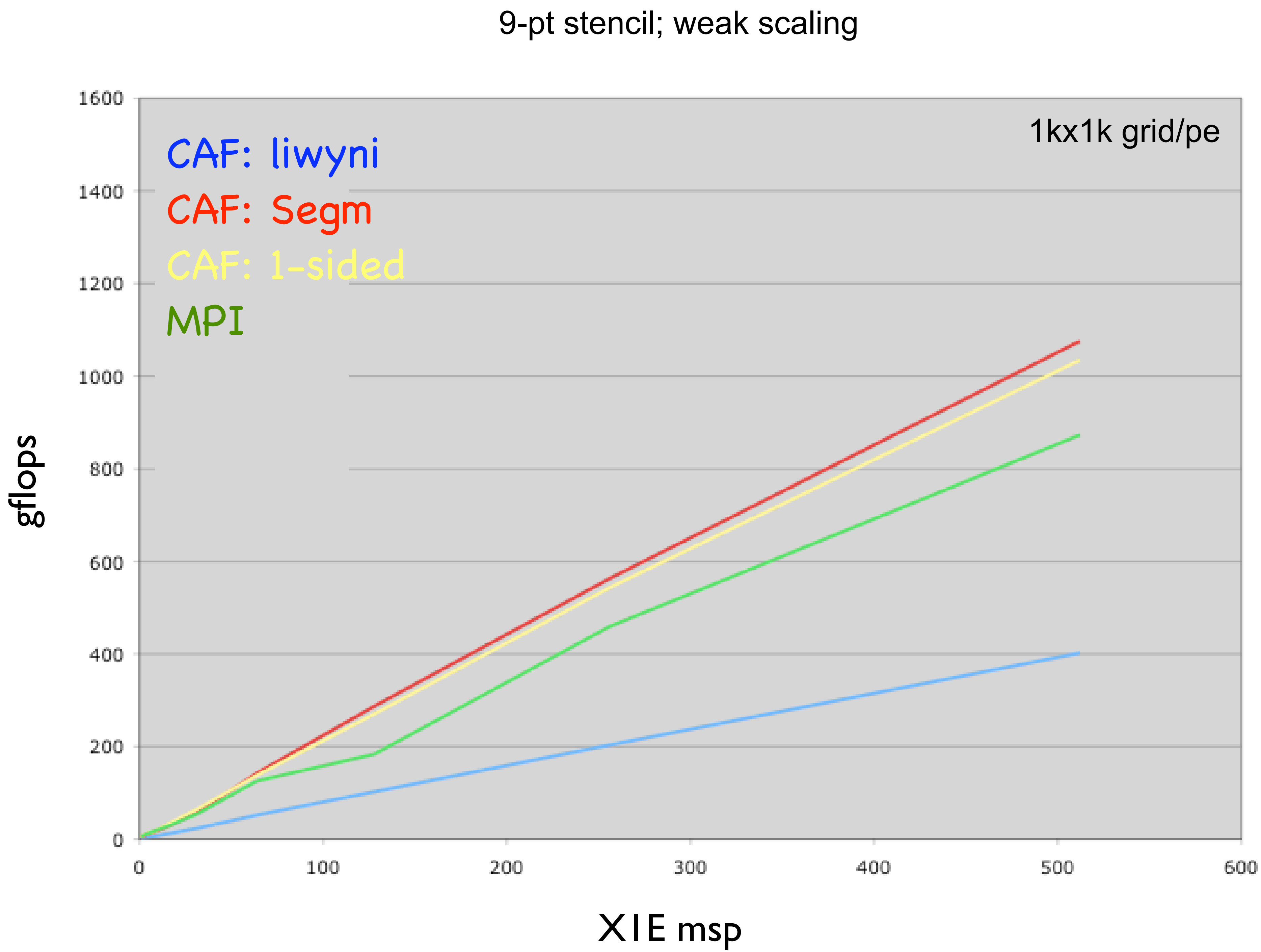


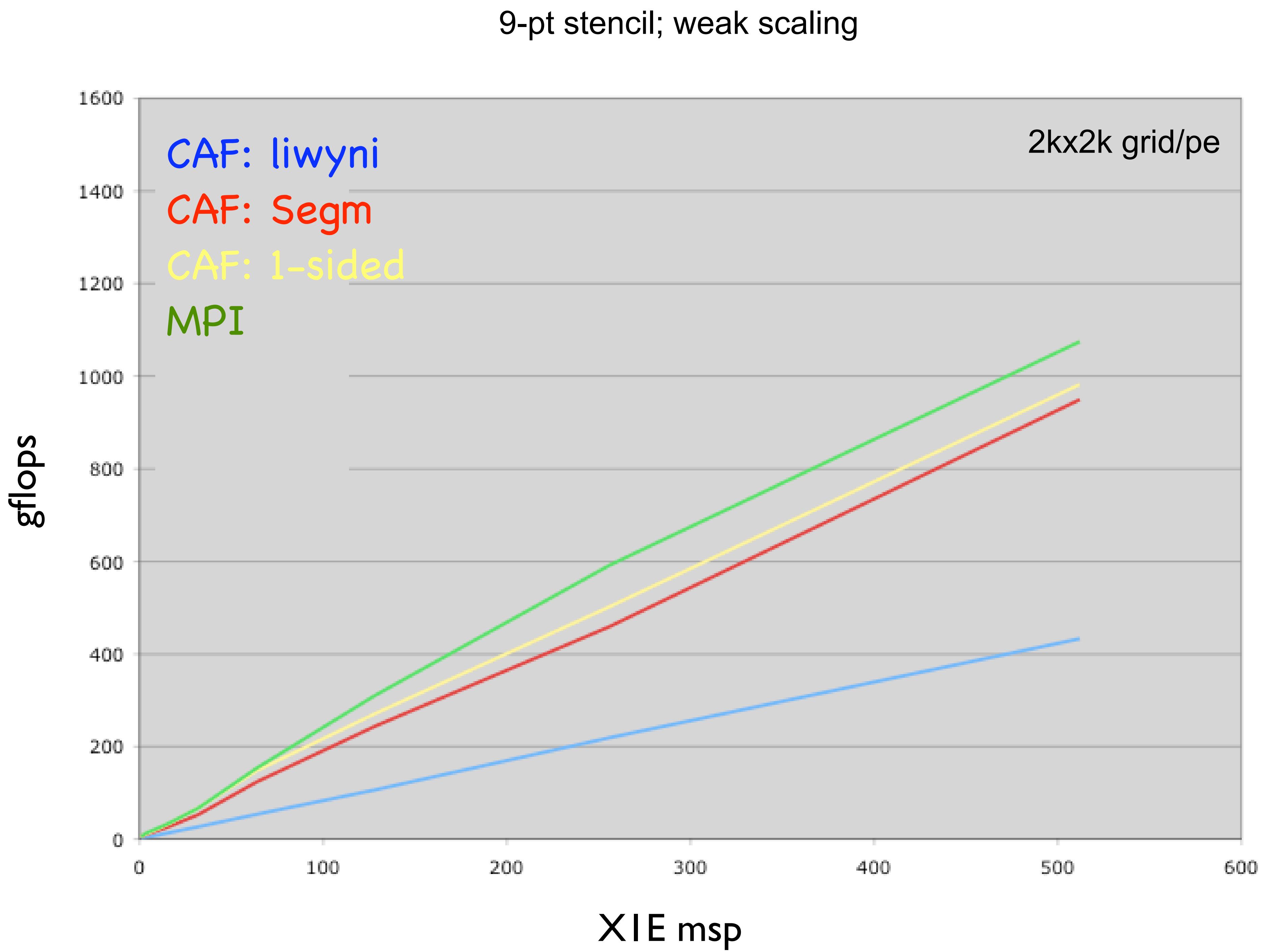
MPI: same number of partners
(with coordination)

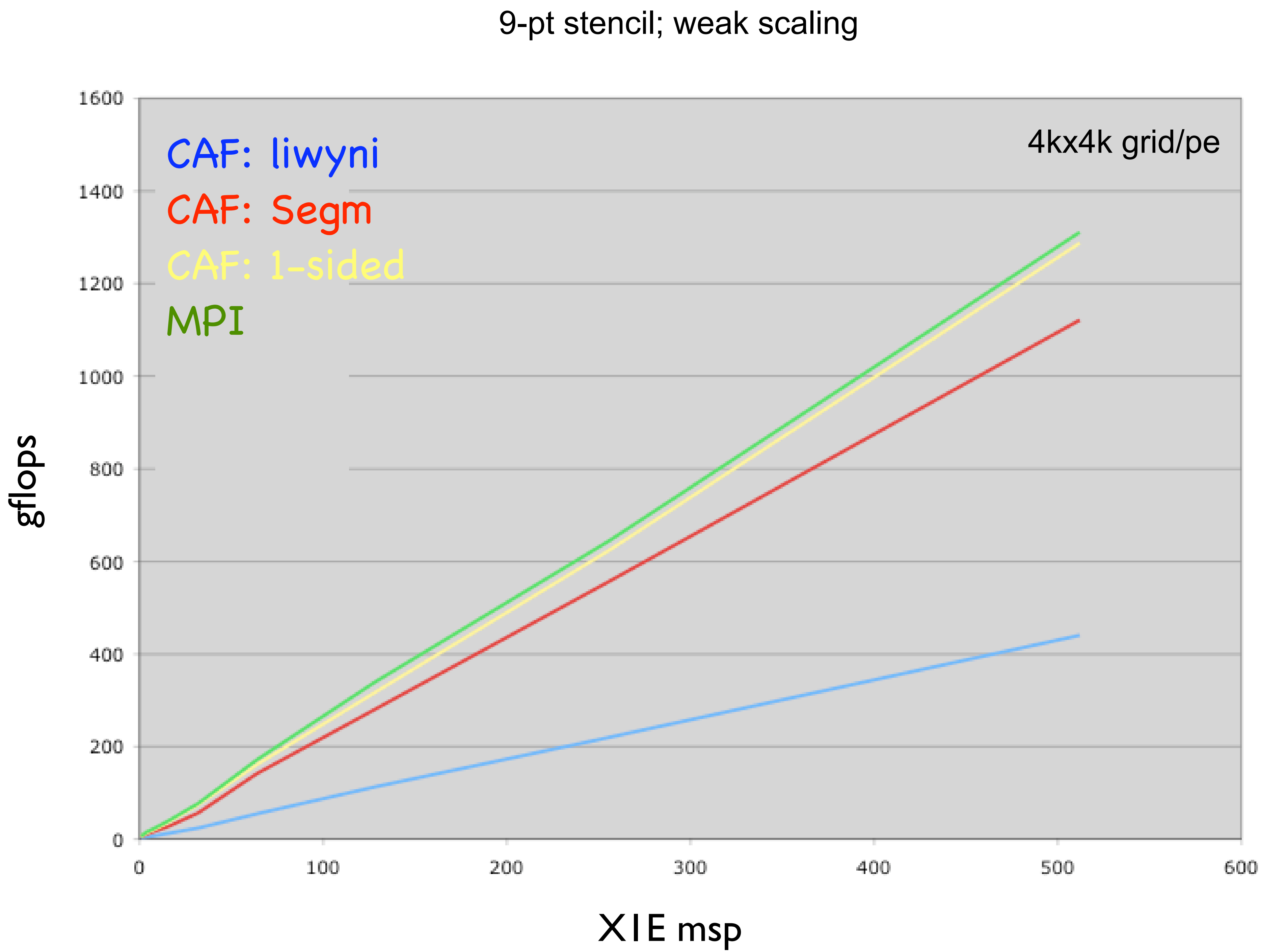

9-pt stencil; weak scaling
100x100 grid/pe
CAF: liwyni
CAF: Segm
CAF: 1-sided
MPI
gflops
1600
1400
1200
1000
800
600
400
200
0
100
200
300
400
500
600
X1E msp

9-pt stencil; weak scaling

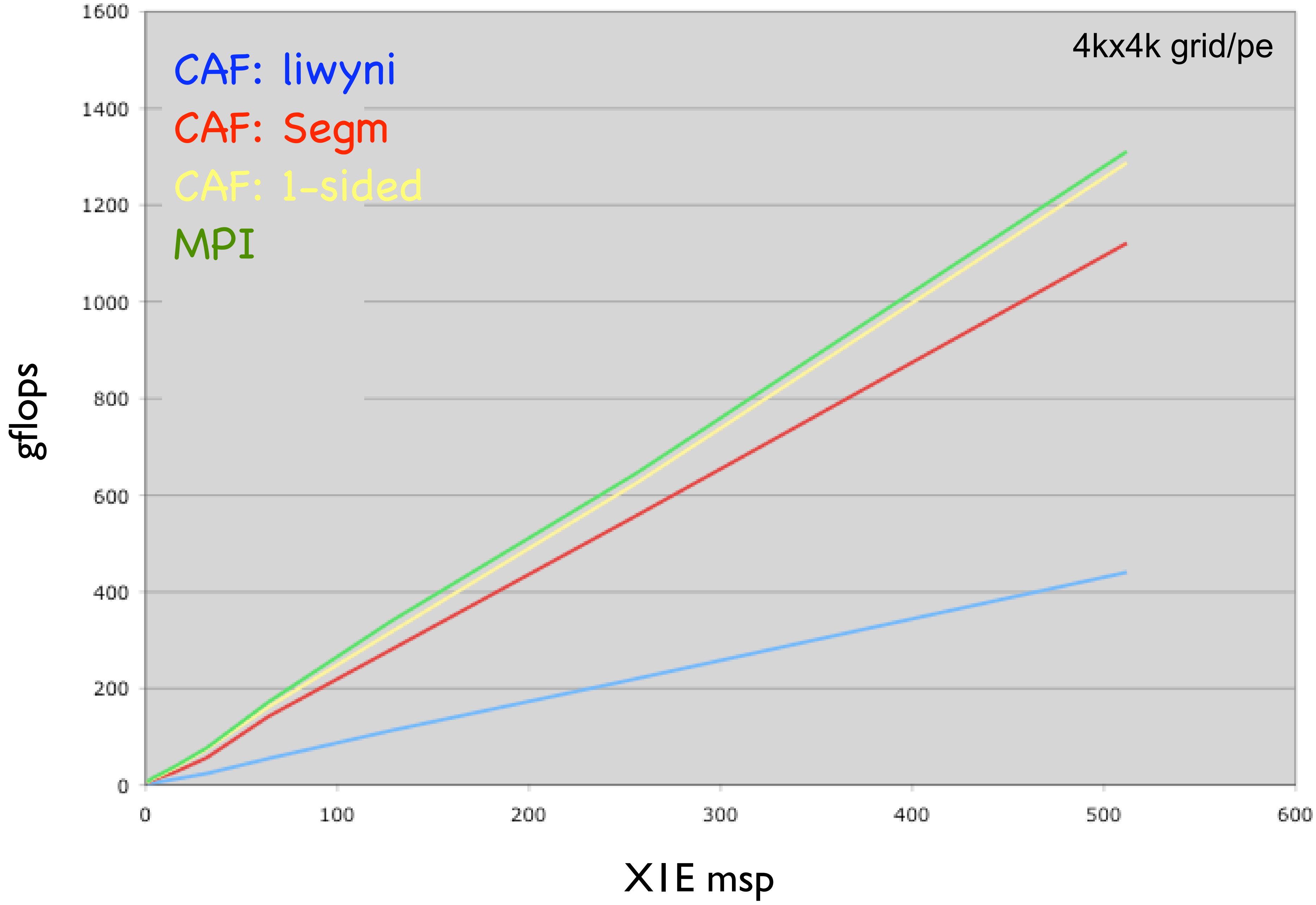



9-pt stencil; weak scaling
1kx1k grid/pe
CAF: liwyni
CAF: Segm
CAF: 1-sided
MPI
gflops
1600
1400
1200
1000
800
600
400
200
0
0
100
200
300
400
500
600
X1E msp

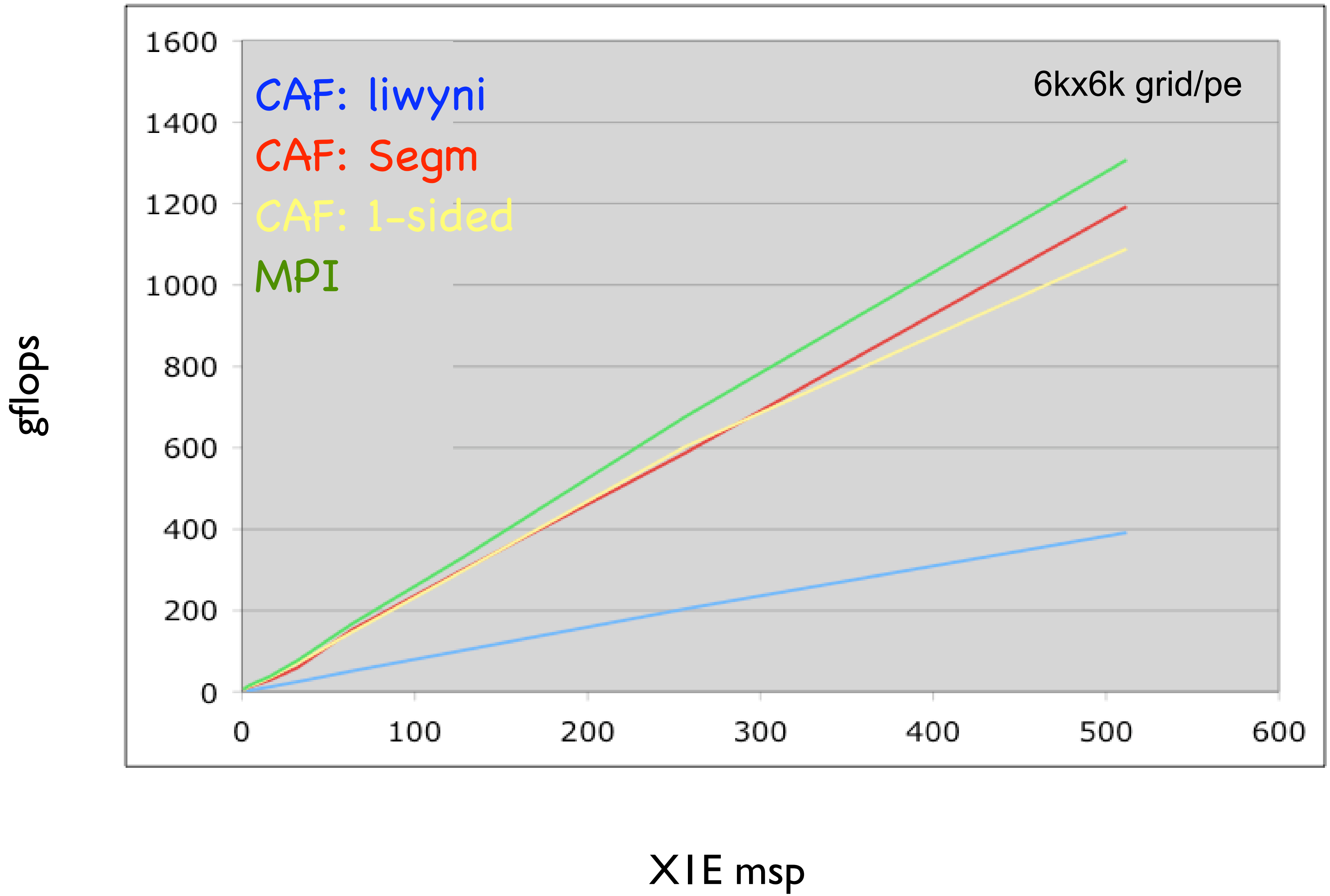

9-pt stencil; weak scaling
2kx2k grid/pe
CAF: liwyni
CAF: Segm
CAF: 1-sided
MPI
gflops
0
200
400
600
800
1000
1200
1400
1600
0
100
200
300
400
500
600
X1E msp


9-pt stencil; weak scaling
4kx4k grid/pe
CAF: liwyni
CAF: Segm
CAF: 1-sided
MPI
gflops
0
200
400
600
800
1000
1200
1400
1600
0
100
200
300
400
500
600
X1E msp

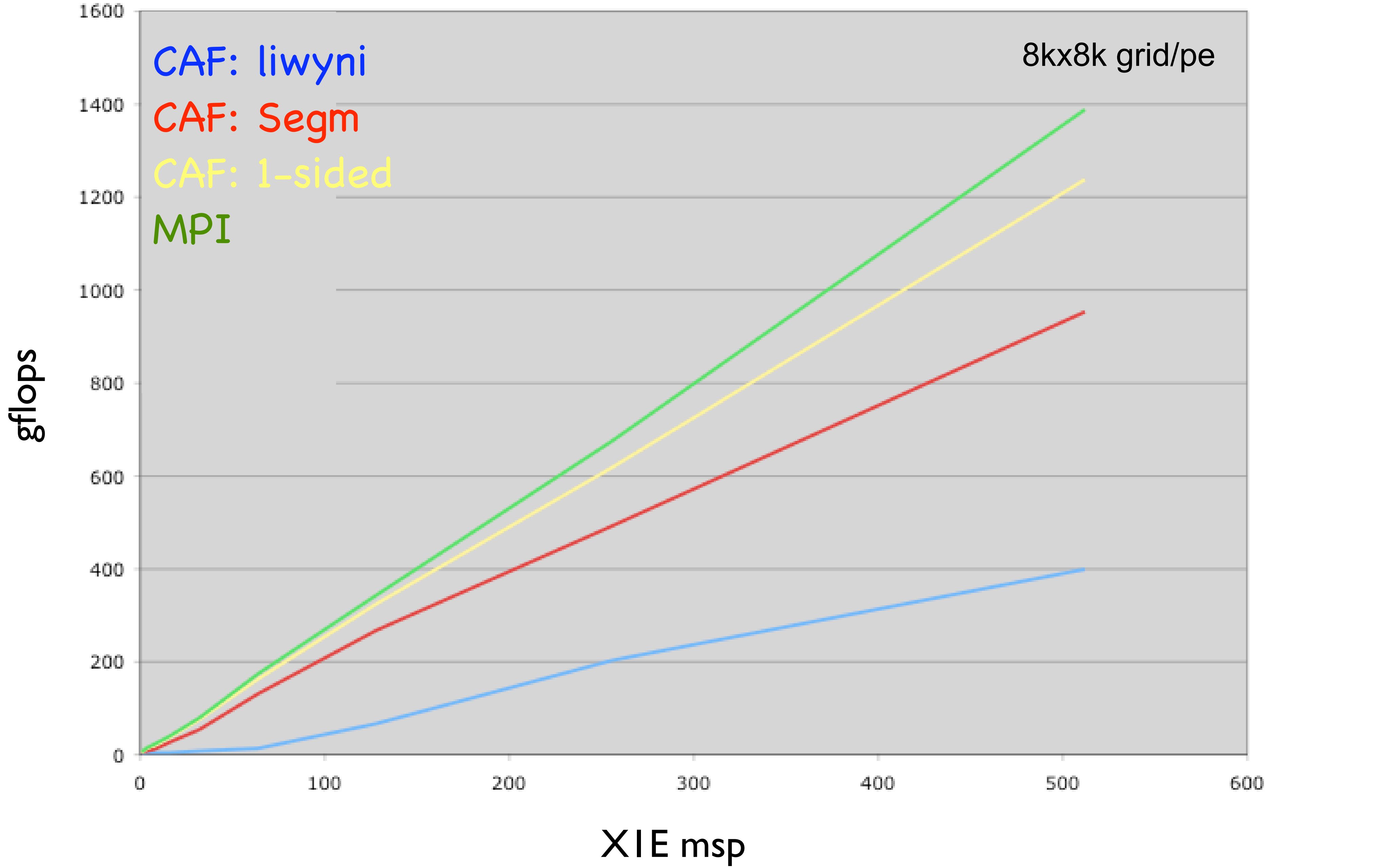
9-pt stencil; weak scaling



9-pt stencil; weak scaling



9-pt stencil; weak scaling



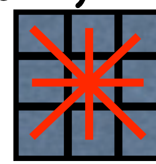
Chapel:
Reduction implementation

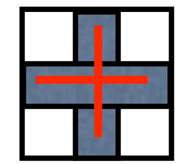
Parallelism

```
const
    PhysicalSpace: domain(2) distributed(Block) = [1..m, 1..n],
    AllSpace = PhysicalSpace.expand(1);

var Coeff, X, Y : [AllSpace] : real;

var Stencil = [ -1..1, -1..1 ];

forall i in PhysicalSpace do
    Y(i) = ( + reduce  [k in Stencil]  Coeff (i+k) * X (i+k) );
```

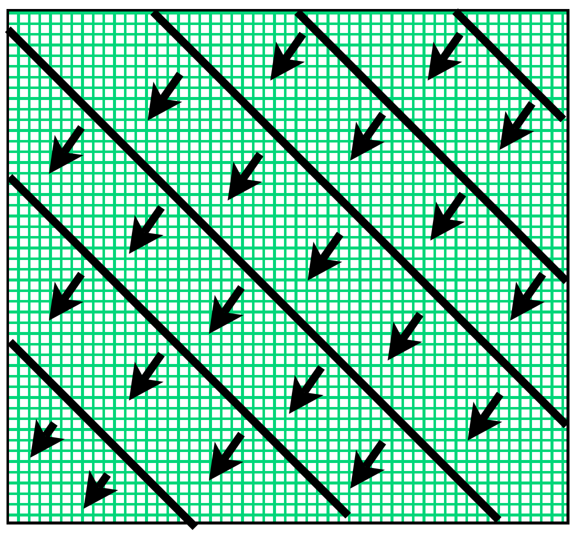
Matrix as a “sparse domain” of 5 pt stencils

```
const
   PhysicalSpace: domain(2) distributed(Block) = [1..m, 1..n],
   AllSpace = PhysicalSpace.expand(1);

var
   Coeff, X, Y : [AllSpace] : real;

var
   Stencil9pt = [ -1..1, -1..1 ],
   Stencil = sparse subdomain (Stencil9pt) = [(i,j) in Stencil9pt]
             if ( abs(i) + abs(j) < 2 ) then (i,j);

forall i in PhysicalSpace do

   Y(i) = ( + reduce [k in Stencil] Coeff (i+k) * X (i+k) );
```

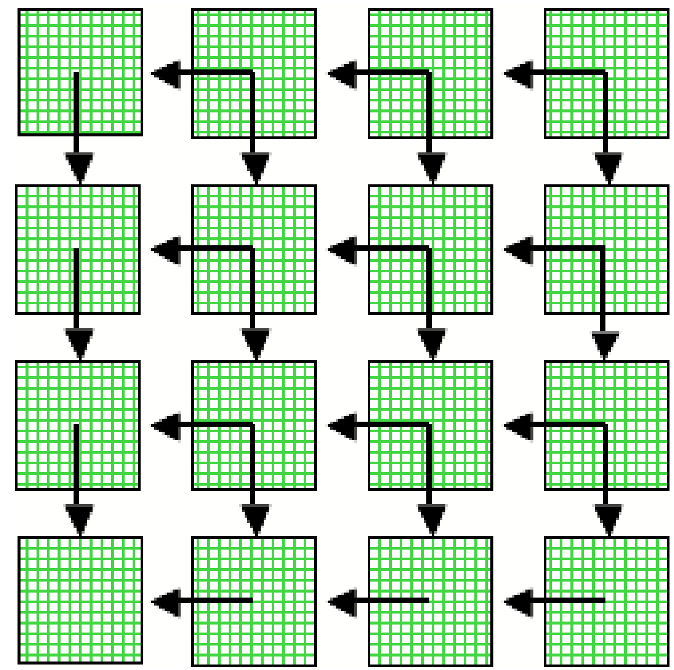




SN transport :
Exploiting the Global-View Model

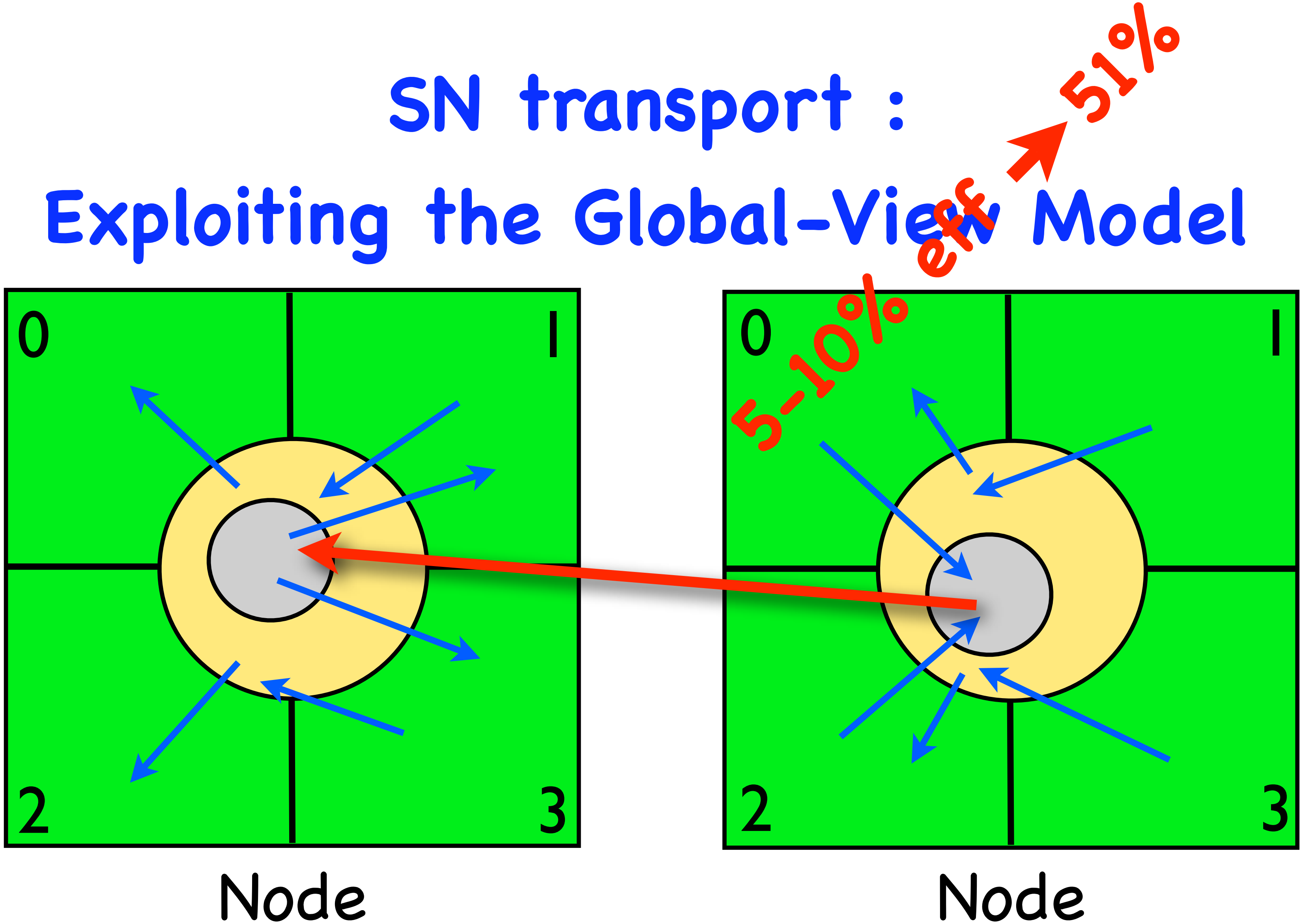


Global-view



Local-view

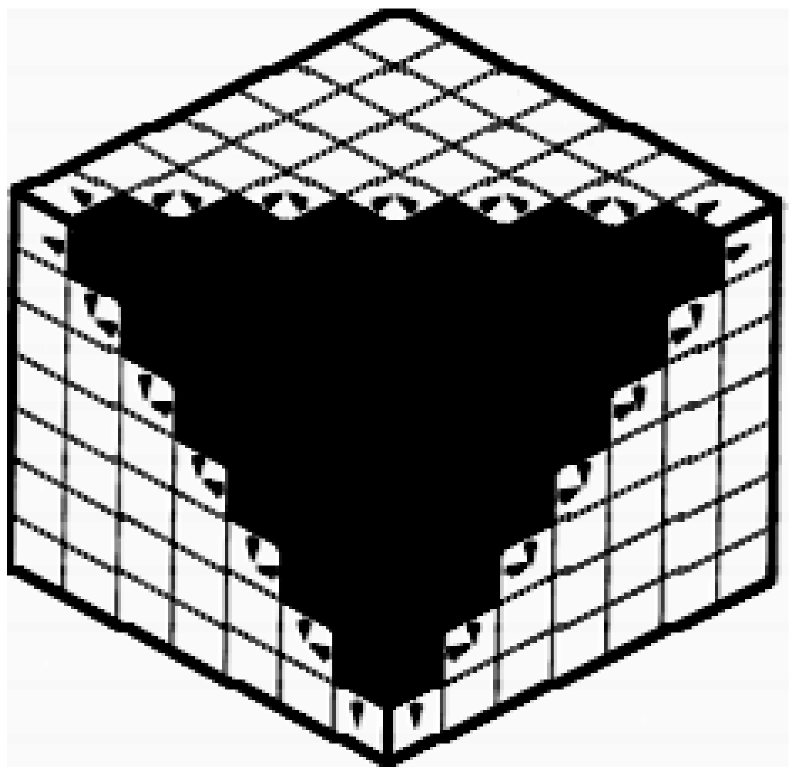
SN transport :
Exploiting the Global-View Model



"Simplifying the Performance of Clusters of Shared-Memory Multi-processor Computers", R.F. Barrett, M. McKay, Jr., S. Suen, BITS: Computing and Communications News, Los Alamos National Laboratory, 2000.

SN transport :
Exploiting the Chapel Memory Model

"S_N Algorithm for the Massively Parallel CM-200 Computer",
Randal S. Baker and Kenneth R. Koch, Los Alamos National Laboratory,
Nuclear Science and Engineering: **128**, 312-320, 1998.



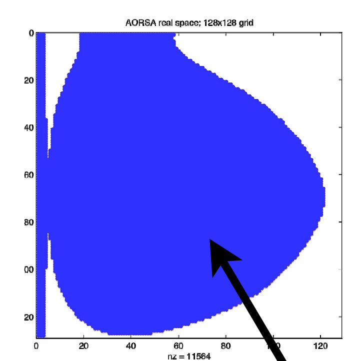
(t3d shmem version, too.)

AORSA arrays in Chapel

```
const

   FourierSpace : domain(2) distributed ( Block Cyclic ) = [1..nodex, 1..nodey];

var
   fgrid,
   mask
         : [FourierSpace] real;

var
   PhysSpace: sparse subdomain (FourierSpace) =
      [i in FourierSpace]  if mask(i) == 1 then i;

var
   pgrid
      : [PhysSpace] real;
ierr = pzgesv ( ..., PhysSpace );
          // ScaLAPACK routine
```

Dense linear solve, so inter-operability needed.

Fourier space



“Real” space

Performance Expectations

~~If we had a compiler we could "know".~~

"Domains" define data structures; coupled with operators.

Distribution options (including user defined)

Multi-Locales

Inter-process communication flexibility

Memory Model

Diversity of Architectures emerging

Strong funding model

Past, Current, and Future work

- "Expressing POP with a Global View Using Chapel: Toward a More Productive Ocean Model", R.F. Barrett, S.R. Alam, and S.W. Poole, ORNL Technical Report TM-2007/122, 2007.
- "Finite Difference Stencils Implemented Using Chapel", Barrett, Roth, and Poole, ORNL Technical Report TM-2007/119, 2007.
- "Strategies for Solving Linear Systems of Equations Using Chapel", Barrett and Poole, Proc. 49th Cray User Group meeting, 2007.
- "Is MPI Hard? An Application Survey", SciComp group & others. submitted.
- "HPLS: Preparing for New Programming Languages for Ultra-scale Applications", ORNL LDRD: Bernholdt, Barrett, de Almeida, Elwasif, Harrison, and Shet.
- "HPCS Languages: An Applications Perspective", Barrett et al, Invited paper & talk, SciDAC 2008.
- "Co-Array Fortran Experinces Solving PDE Using Finite Differencing Schemes", Barrett, Proc. 48th Cray User Group, 2006.
- "UPC on the Cray X1E", Barrett, El-Ghazawi, Yao, 48th Cray User Group, 2006.

Acknowledgments

This research was sponsored by the Office of Mathematical, Information, and Computational Sciences, Office of Science, U.S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

Chapel development team.

ORNL LDRD, DoD, AORSA project team.

SciDAC'08 program committee (Invited paper)

OAK RIDGE National Laboratory