

Advanced Modeling of Particle Accelerators

**J.-L. Vay, T. Drummond, A. Koniges, B. Loring,
C. Mitchell, J. Qiang, O. Ruebel, R. Ryne, H. Vincenti**

Lawrence Berkeley National Laboratory, CA, USA

D. P. Grote

Lawrence Livermore National Laboratory, CA, USA

Axel Hübl

Helmholtz-Zentrum Dresden Rossendorf , Germany

**NERSC Exascale Science Application Program meeting
12/04/2014**



U.S. DEPARTMENT OF
ENERGY

Office of
Science

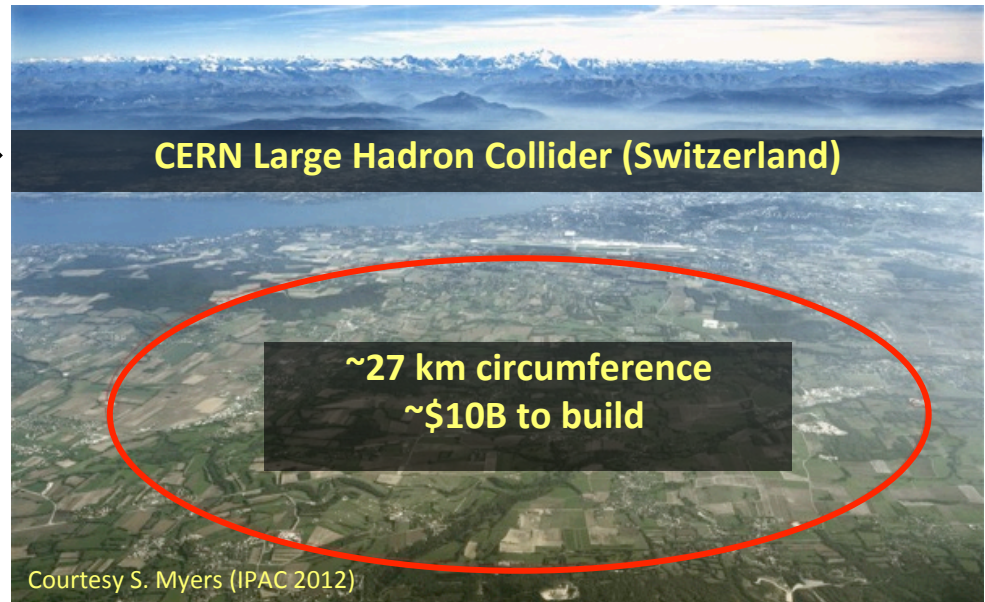


Accelerators are essential tools of science and tech.

There are > 30,000 **particle accelerators** in operation around the world,

serving:

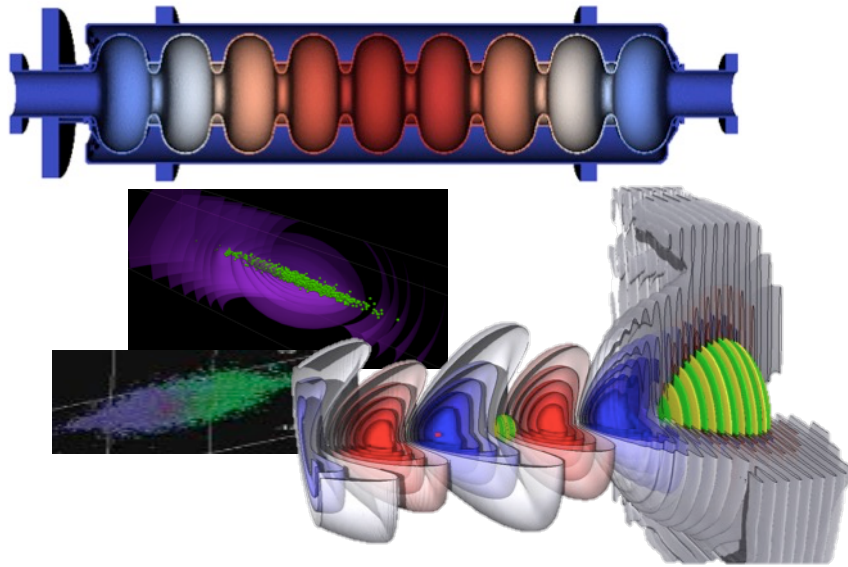
- discovery science, →
- medicine,
- industry,
- energy,
- the environment,
- national security.



Size and cost are a limiting factor for many applications

→ active research worldwide to conceive smaller & cheaper accelerators.

Computer modeling is key to progress



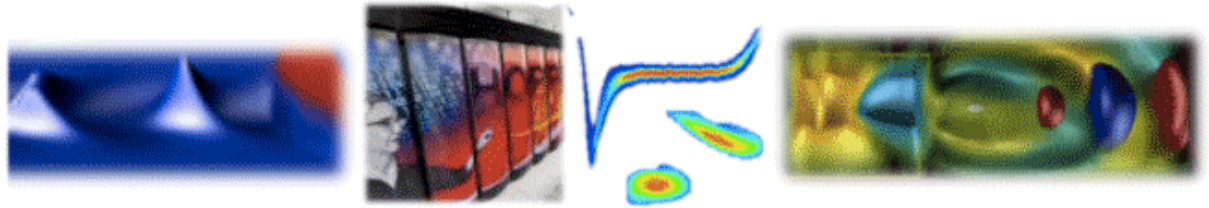
Essential for increasing performance, bringing size and cost down:

- **optimizing** existing accelerators,
- **cost effective** design,
- **game changing** technologies.

^x ■ Trend requires **team work** for **large multi-physics** simulations

increasingly complex accelerators call for
increasingly sophisticated simulation software

BLAST* (Berkeley Lab Accelerator Simulation Toolkit)



BeamBeam3D

BeamBeam3D is a parallel, electrostatic particle-in-cell (PIC) code for modeling strong-strong or strong-weak beam-beam interaction in high-energy colliders.

IMPACT

IMPACT is a 3-D parallel electrostatic PIC framework, incorporating maps, for modeling high-intensity, high-brightness beams in accelerators.

WARP

WARP is a 2- and 3-D, parallel, electrostatic and electromagnetic PIC framework, aware of the accelerator lattice, that is used to model the generation, transport, and neutralization of charged-particle beams. It also has uses in modeling ion traps and laser-plasma accelerators.

Posinst

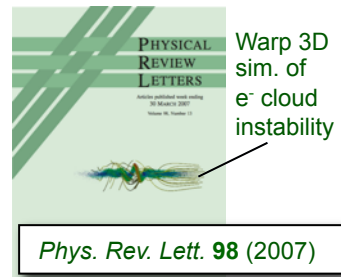
Posinst is a 2-D electrostatic PIC code for studying the buildup of electron clouds.

Many algorithms invented, improved or pioneered in BLAST codes

Algorithm/method	Reference	Originated	Adopted by
Damped EM & particle pushers	<i>Friedman, JCP 1990</i>	Warp	LSP, Elixirs
Warped coordinates PIC in bends	<i>Friedman et al, Phys. Fluid 1992</i>	Warp	
Integrated Maps for rf cavity dynamics	<i>Ryne, LANL Report 1995</i>	ML/IMPACT	D. Abell (nonlinear model)
Stochastic Leap-Frog for Brownian motion	<i>Qiang & Habib, PRE 2000</i>	IMPACT	
Spectral-finite difference multigrid solver	<i>Qiang & Ryne, CPC 2001</i>	IMPACT	
Improved Perfectly Matched Layers	<i>Vay, JCP 2000/JCP 2002</i>	Warp	Osiris
AMR-PIC electrostatic	<i>Vay et al, LPB2002/PoP2004</i>	Warp	
Secondary emission of electrons algorithm	<i>Furman & Pivi, PRST-AB 2003</i>	Posinst	TxPhysics, Warp, spacecraft charging codes
AMR-PIC electromagnetic	<i>Vay et al, CPC 2004</i>	Emi2D	Warp
3D Poisson solver with large aspect ratio	<i>Qiang & Gluckstern, CPC 2004</i>	IMPACT	
Shift-Green function method	<i>Qiang et al, CPC 2004</i>	BBeam3D	
Integrated Green function	<i>Ryne & Qiang</i>	ML/IMPACT	BB3D, IMPACT
Hybrid Lorentz particle pusher	<i>Cohen et al, NIMA 2007</i>	Warp	

and adopted by other codes

Algorithm/method (cont.)	Reference	Originated	Adopted by
Lorentz boosted frame	<i>Vay, PRL 2007</i>	Warp	INF&RNO, JPIC, Osiris, Vorpap
Explicit Lorentz invariant particle pusher	<i>Vay, PoP 2008</i>	Warp	Tristan (astro), QED, Photon-Plasma
New convolution integral w/ smooth kernel	<i>Qiang, CPC 2010</i>	N/A	
Mixed Particle-Field decomposition method	<i>Qiang & Li, CPC 2010</i>	BBeam3D	
PIC with tunable electromagnetic solver	<i>Vay et al, JCP 2011</i>	Warp	Vorpap, Osiris
Efficient digital filter for PIC	<i>Vay et al, JCP 2011</i>	Warp	Vorpap, Osiris
Laser launcher from moving antenna	<i>Vay et al, PoP 2011</i>	Warp	Vorpap, Osiris
Domain decomposition for EM spectral solver	<i>Vay et al, JCP 2013</i>	Warp	



Example:

- Lorentz boosted frame method (performing calculation in optimal frame – next slide)
- first developed at LBNL with application to FEL and e⁻ cloud instability,
- methods (developed in Warp) implemented in Osiris, Vorpap, etc. (Laser Plasma Acc.)
- results reported in DOE-OS news <http://science.energy.gov/news/in-focus/2011/04-21-11-s>

Berkeley Lab Accelerator Simulation Toolkit has a **worldwide** user base

BLAST

B – BeamBeam3D
I – Impact
P – Posinst
W – Warp

United States

1. ANL (B,I,P)
2. BNL (B,I,P,W)
3. Cornell (I,P)
4. FNAL (B,I,P,W)
5. ISU (I)
6. Jlab (B,I,P)
7. LANL (I,P)
8. LBNL (B,I,P,W)
9. LLNL (W)
10. MSU (I,W)
11. NIU (I,W)
12. ODU (I)
13. ORNL (I,P)
14. SLAC (I,P,W)
15. Stanford (I)
16. Tech-X (P)
17. Texas A&M (I)
18. U. Chicago (I)
19. UM (W)
20. UMD (W)
21. UW (I)
22. UCLA (I)
23. WSU (W)
24. Yale U (B,I)

Europe/Asia

1. ASLS (I)
2. CERN (P,W)
3. CIAE (I)
4. DESY (I,W)
5. Diamond (I)
6. ESS (I)
7. Fermi/Elettra (I)
8. Frankfurt (I)
9. GSI (I,W)
10. Hiroshima U. (W)
11. Hong Kong U. (W)
12. IBS (I)
13. IHEP (I,P)
14. IMPCAS (I)
15. IRE (I)
16. KAERI (I)
17. KEK (I)
18. Mumbai Univ. (I)
19. PAL (I)
20. Peking Univ. (I)
21. PSI (I)
22. RRCAT (I)
23. RAL (I)
24. SINAP (I)
25. Technion (W)
26. USTC (I)



Over past year, code developers responded to

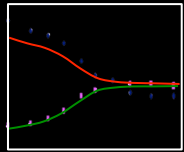
>500 emails received from >40 research institutes/universities & list is growing.

Need for development with minimal disruptions to users.

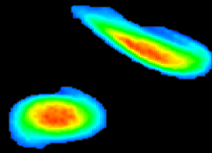
Sample applications of the BLAST codes

Beam dynamics in rings & linacs

Montague resonance



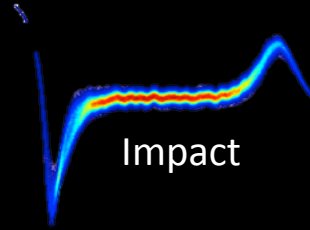
PS



SNS

Impact

High space charge beams



Impact

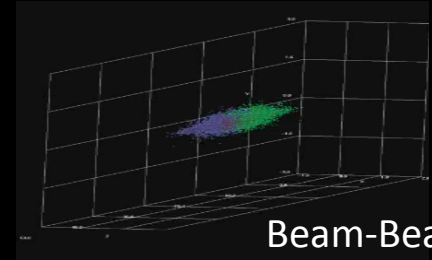


Warp



UMER

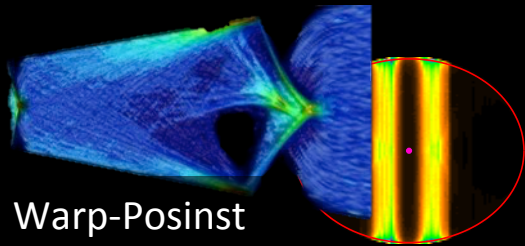
Beam-Beam effects



Beam-Beam3D

LHC, RHIC, Tevatron, KEK-B

Electron cloud effects

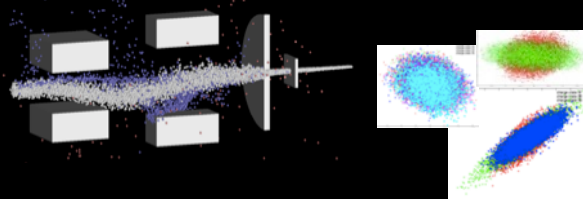


Warp-Posinst

Posinst

SPS

Multi-charge state beams



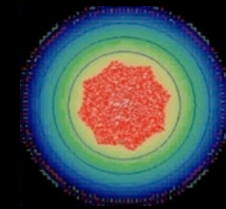
Warp

Impact

LEBT – Project X

FRIB

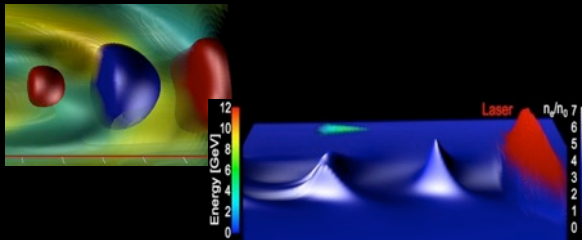
Traps



Warp

Alpha anti-H trap

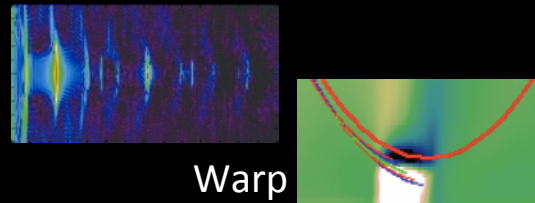
Laser plasma acceleration



Warp

BELLA

3D Radiation

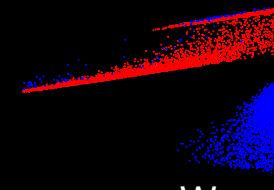


Warp

FEL

CSR

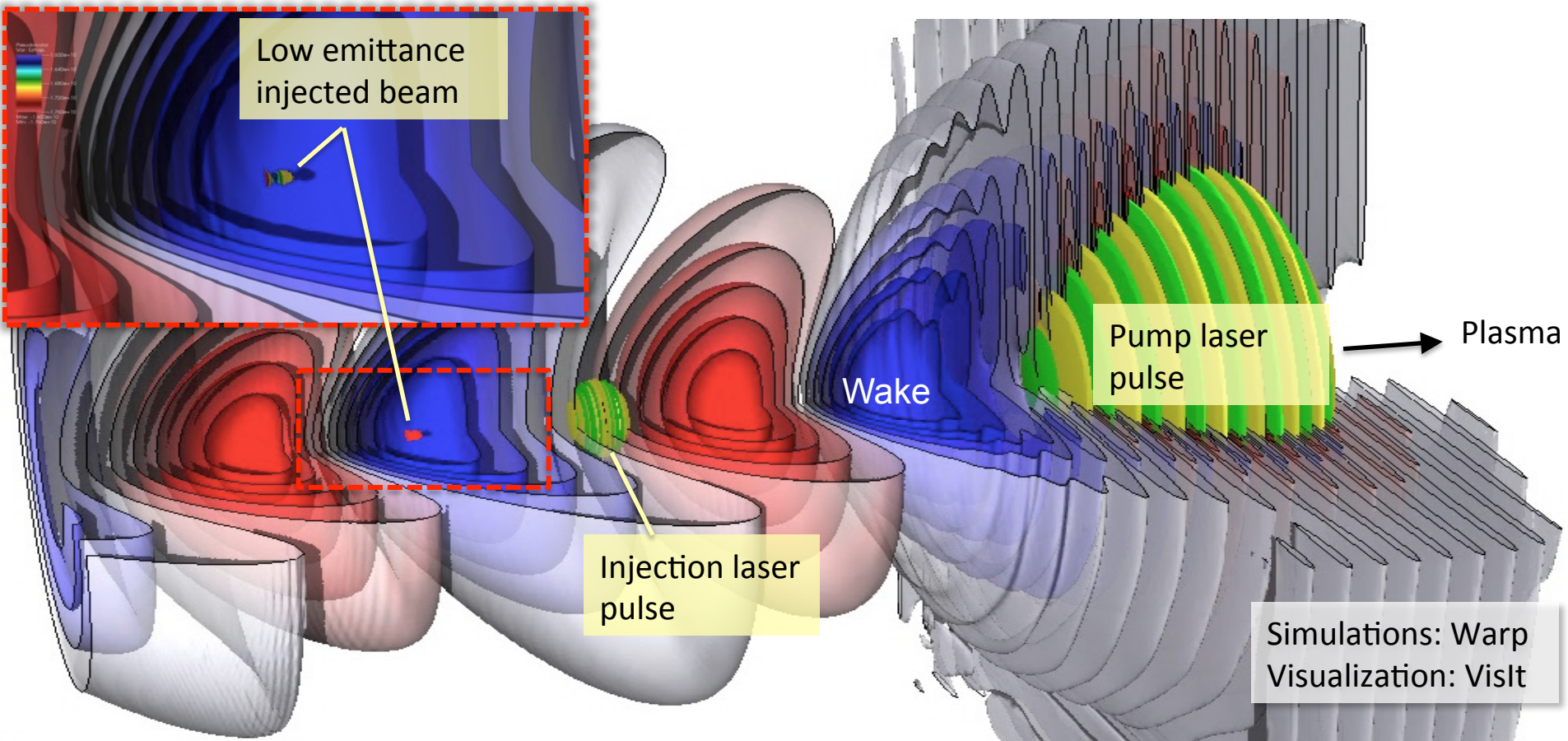
Discover new physics



Warp

Multipacting "Ping-Pong" effect

Large scale simulations validated a new concept of injection of ultra-low emittance beam*



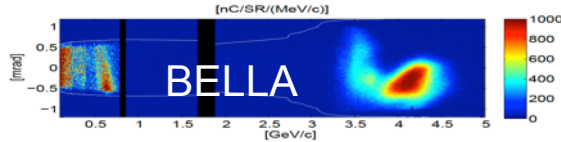
Parametric runs necessitated over a million hours; typical 3-D “medium res.” run \sim 100k core-hours.

Larger simulations needed for high resolution modeling through entire stage.

future-generation accelerators at dramatically lower cost advances in modeling will offer new opportunities

Multi-GeV laser plasma accelerators

Now



Next

- Accelerator: EM + plasma + beam

3-D full PIC

K-BELLA

Collider

$$\sigma_{x,\text{beam}} \sim 1 \mu\text{m}$$

$$L_{\text{acc}} \sim 1\text{m}$$

$$\sigma_{x,\text{beam}} \sim 0.1 \mu\text{m}$$

$$L_{\text{acc}} \sim 100\text{m}$$

Now
Lab frame

~ 1year

~ 10⁴ year

Now
Boosted frame

~ 1hour

~ 1 year

Goal 2024
Boosted frame

<1min

~ 3-4 days

Goal 2024
Boosted frame
+ AMR + env.
10⁶ cores

<10ms

real-time on
clusters

~ 1 hour

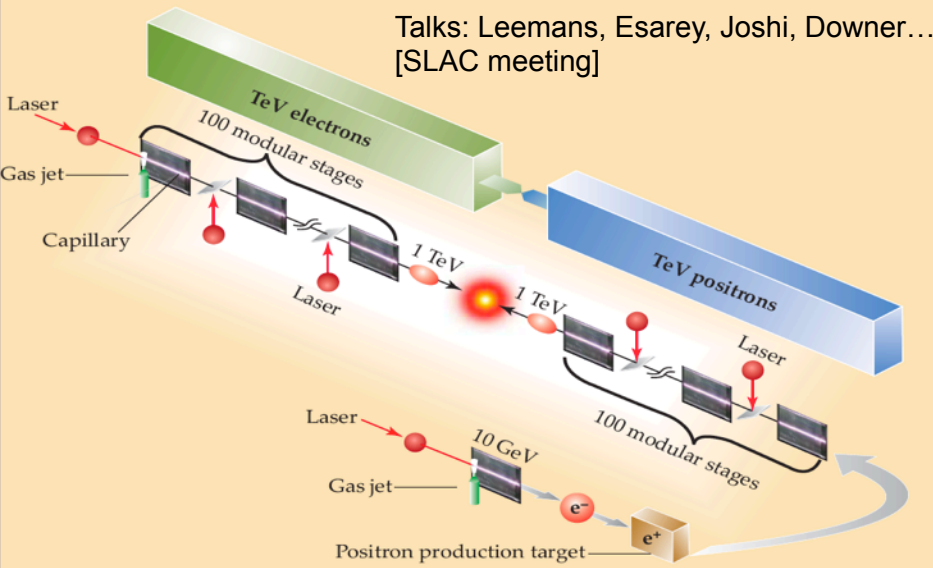
parametric
studies on
supercomputers

Advances in computers & algorithms:

- real-time of single stages on clusters,
- design of colliders on supercomputers.

Laser-plasma (LPA) collider concept:

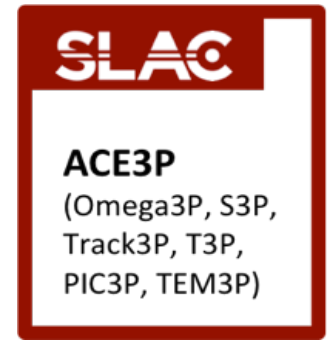
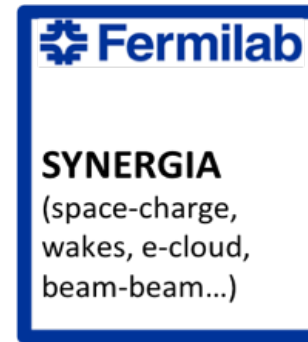
100x 10 GeV stages (1m ea.), injector, focus, e+



New initiative:

Consortium for Advanced Modeling of Particle Accelerators

CAMPA



Points of contact:

LBL: J.-L. Vay, J. Qiang

SLAC: C.-K. Ng, Z. Li

FNAL: J. Amundson, E.G. Stern

PICon GPU



OpenSource Development
HZDR (GPL/LGPL)



General Many-Core Algorithms
(x86 & Xeon Phi port started)

7.2 PFlop/s reported in
ACM Gordon Bell 2013

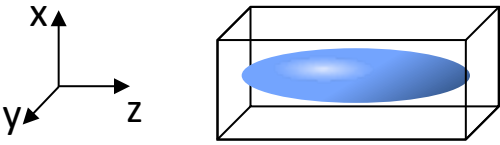
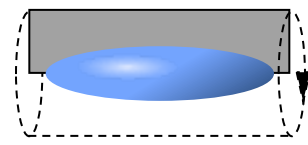
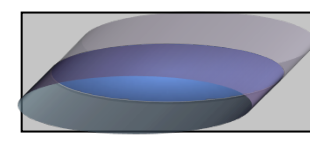
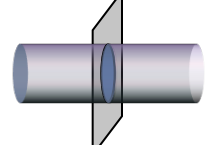
modular, general purpose,
particle-mesh library

implements fully-relativistic
3D3V particle-in-cell algorithm

simulate **laser-plasma interactions**
for next-gen. *particle accelerators*

In situ diagnostics
live rendering, high-throughput
I/O with ADIOS, far-field radiation,
multi-physics, interactive simulations

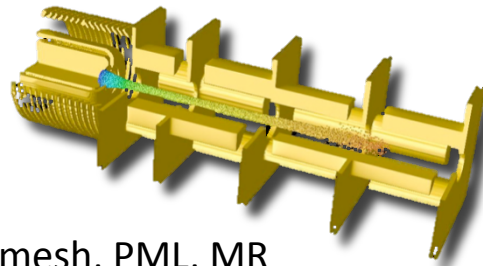
Warp: Particle-In-Cell framework for accelerator modeling

• Geometry:	3-D (x,y,z)	axisym. (r,z)	2-D (x,z)	2-D (x,y)
				
• Reference frame:	lab	moving-window	Lorentz boosted	
	z	z-vt	$\gamma(z-vt); \gamma(t-vz/c^2)$	

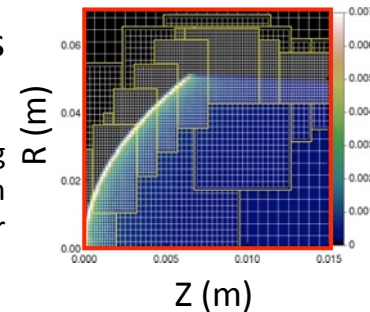
• Field solvers

- electrostatic/magnetostatic - FFT, multigrid; AMR; implicit; cut-cell boundaries

Versatile conductor generator accommodates complicated structures



Automatic meshing around ion beam source emitter



- Fully electromagnetic - Yee mesh, PML, MR

• Accelerator lattice: general; non-paraxial; can read MAD files

- solenoids, dipoles, quads, sextupoles, linear maps, arbitrary fields, acceleration.

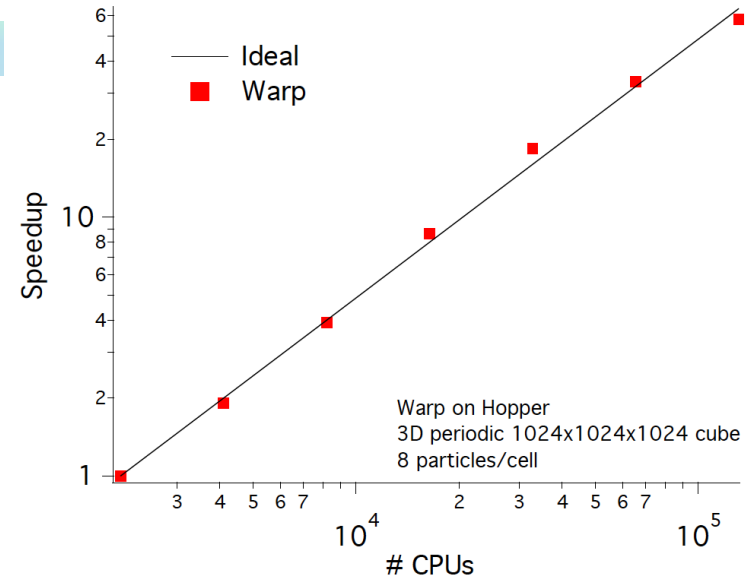
• Particle emission & collisions

- particle emission: space charge limited, thermionic, hybrid, arbitrary,
- secondary e- emission (Posinst), ion-impact electron emission (Txphysics) & gas emission,
- Monte Carlo collisions: ionization, capture, charge exchange.

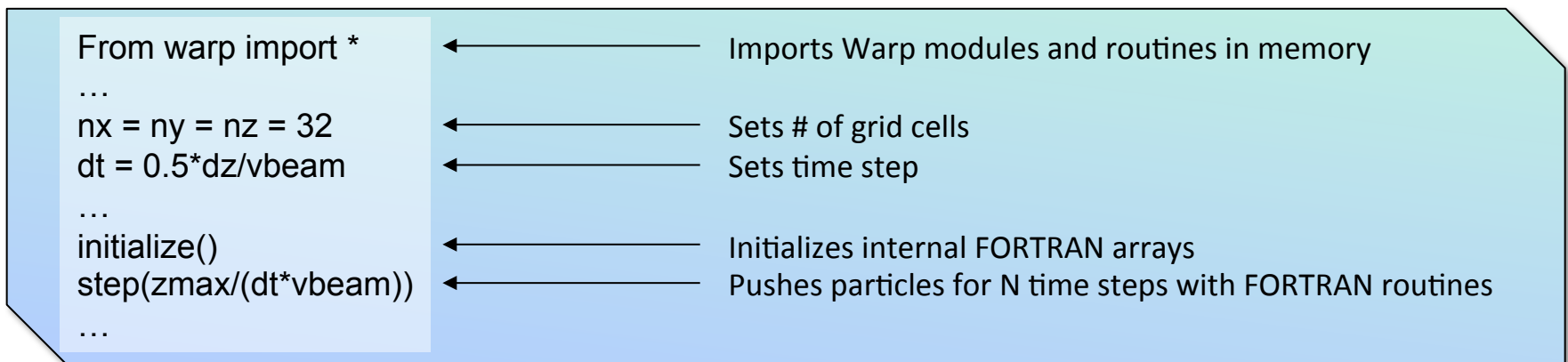
Warp is parallel, combining modern and efficient programming languages

- **Parallellization:** MPI (1, 2 and 3D domain decomposition)

Parallel strong scaling of Warp 3D PIC-EM solver on Hopper supercomputer (NERSC)



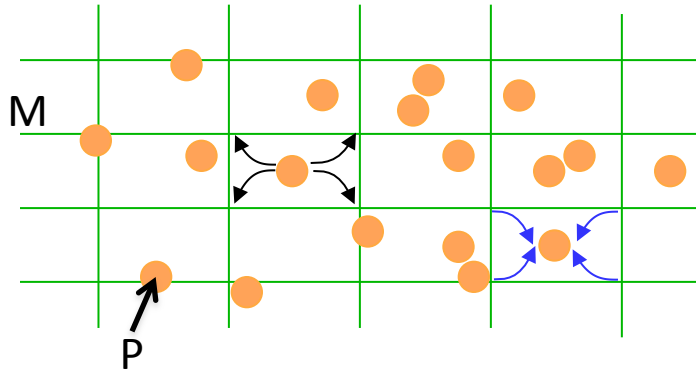
- **Python and FORTRAN*:** “steerable,” input decks are programs



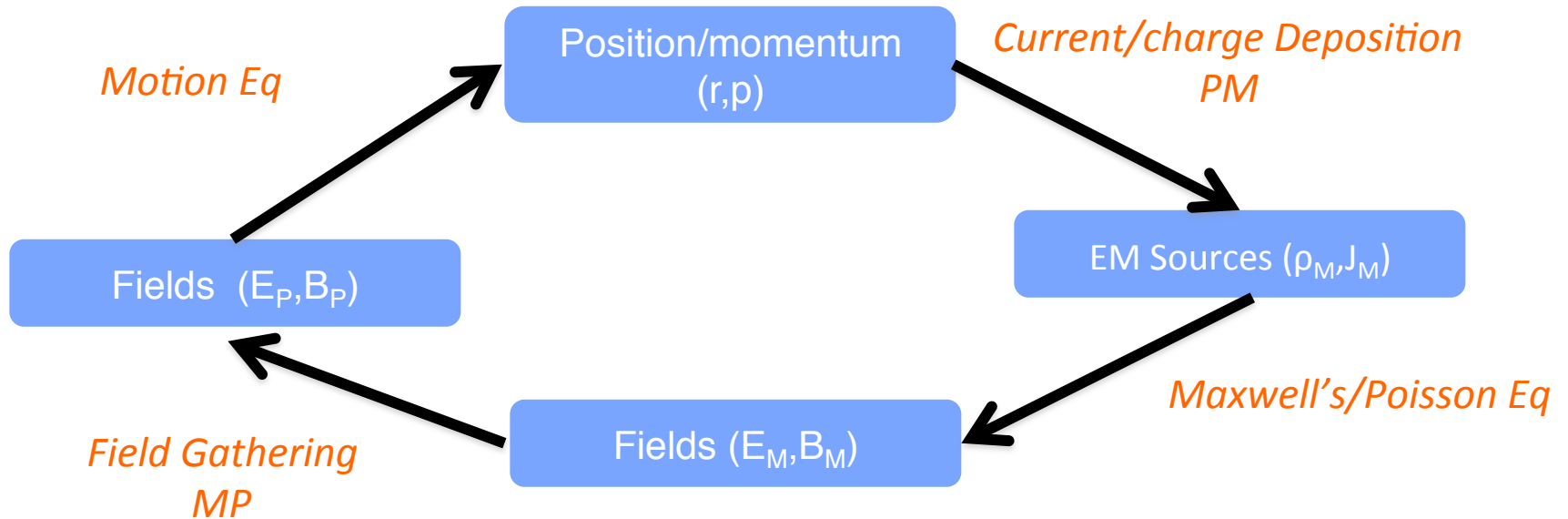
*<http://hifweb.lbl.gov/Forthon> (wrapper supports FORTRAN90 derived types) – dpgrote@lbl.gov

Goal: identify hotspots that limit performances while porting to KL.

PRINCIPLE OF A PARTICLE-IN-CELL CODE



Charged macro-particles (p^+ , e^- , ions, ...) interacting through fields on grid



GENERAL PERFORMANCE CONSIDERATION OF PIC CODES

- In Hybrid MPI/Open MP schemes hot spots may come from:

1. Shared memory part of the code (OpenMP)

- Field Gathering/Current deposition: cache misses (particles not contiguous in memory). Particle sorting/tiling?
- Unvectorized loops (flow-dependency, function call etc.) : re-write/organize some parts of the PIC code?

2. Distributed memory level (MPI)

- Uneven repartition of particles among processors: dynamic load balancing

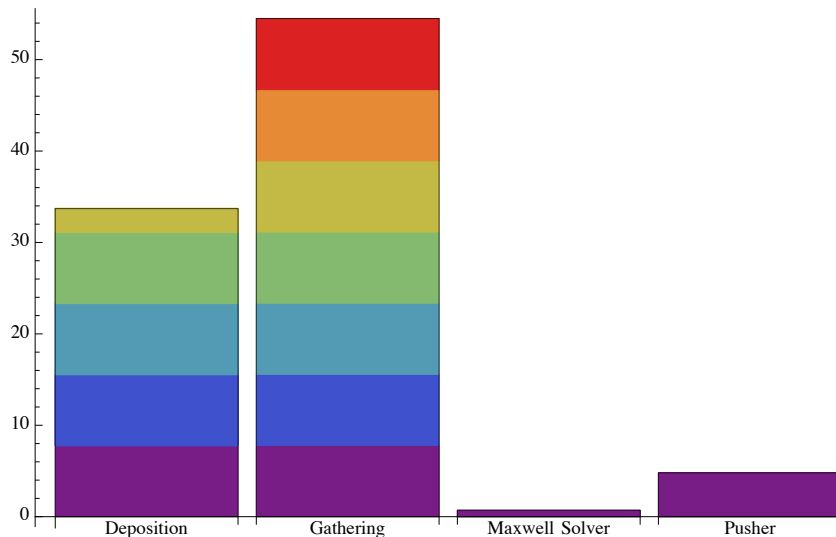
PERFORMANCE ANALYSIS OF WARP CORE ON EDISON

- Parallelization on Distributed memory architectures is very efficient.
 - Local method for solving Maxwell's equation,
 - One potential limitation: load balancing.
Dynamic Load balancing strategies could be implemented in WARP.
- Here we focus on what could limit parallel performance on shared memory architecture (basically on each node with OpenMP \geq 1thread).
 - **Relevant for future KnightsLanding architecture at NERSC.**
- Performance analysis of WARP kernel on EDISON/BABBAGE to identify potential hotspots:
 - First study: sequential mode (OMP_NUM_THREADS=1),
 - Second study: limitations in terms of scalability (OMP_NUM_THREADS $>$ 1).

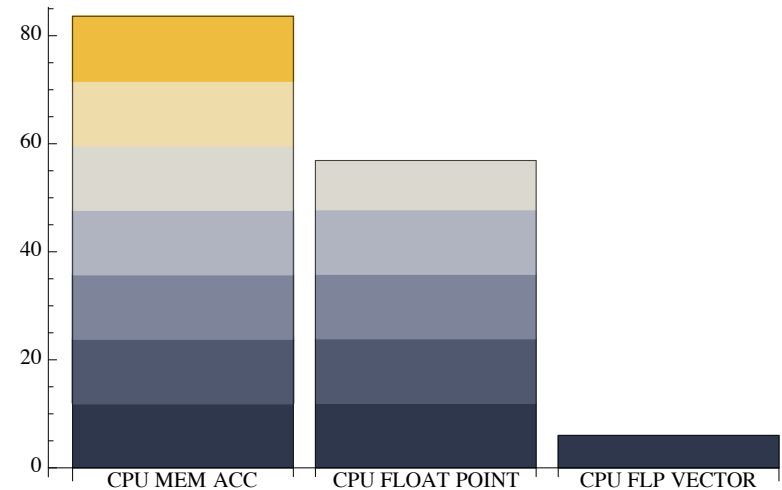
PERFORMANCE ANALYSIS OF WARP CORE ON EDISON

- First study: sequential mode (OMP_NUM_THREADS=1)
 - Total performance of the kernel (standard 2D3V laser-plasma acceleration case):

Time (% total)



Time (% total)

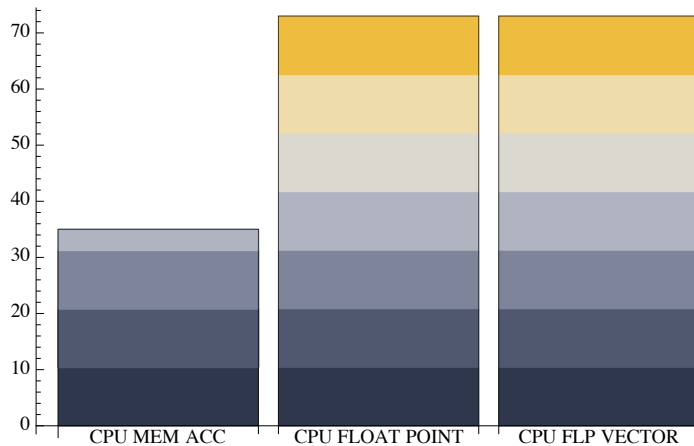


- Total time dominated by deposition/gathering subroutines
- High memory access and Low vectorization

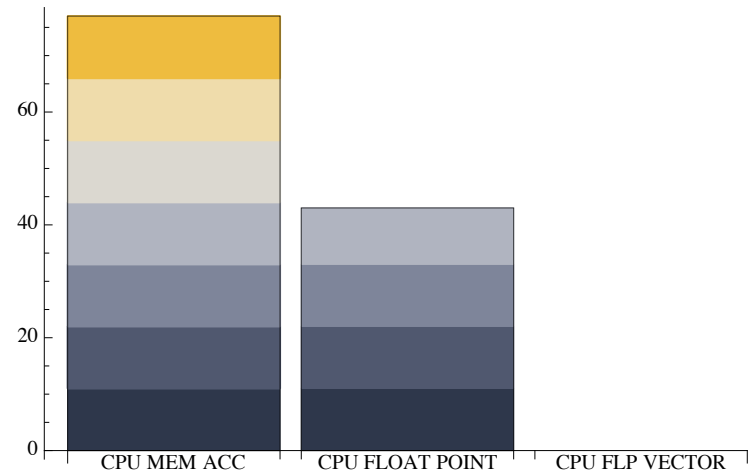
PERFORMANCE ANALYSIS OF WARP CORE ON EDISON

- First study: sequential mode (OMP_NUM_THREADS=1)
 - Performance of particle pusher vs. deposition&gathering

Pusher (% total)



Dep&Gat (% total)



- Moderate memory access,
- loop is well vectorized:
- we can expect good OpenMP performance

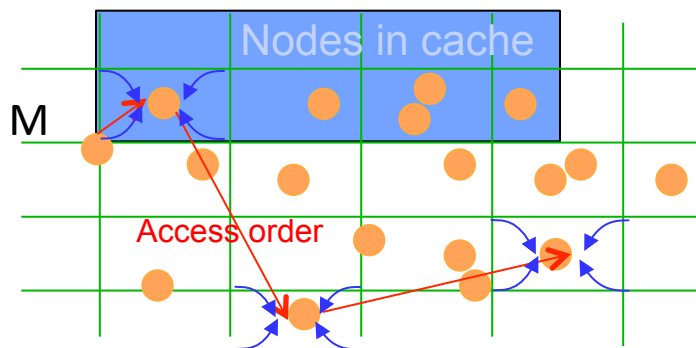
- High memory access (cache misses &/or inefficiently-structured code)
- loop is not vectorized
- poor future OpenMP performance?

PERFORMANCE ANALYSIS OF WARP CORE ON BABBAGE

1. First study: sequential mode (OMP_NUM_THREADS=1)

- Possible improvements for deposition/gathering routines:

Ex : Field gathering process



Code structure

```
DO i=1,nparticles
  DO inode=1,N_adjacentnodes
    Field(particle)= Field(particle)+Weight*Field(inode)
  END DO
END DO
```

- ◆ WITH CURRENT STRUCTURE, PARTICLE ACCESS ORDER INDUCE LOT OF CACHE MISSES
- ◆ PARTICLES MUST BE SORTED SO THAT CPU ACCESS CONTINUOUS DATA IN CACHE: « PARTICLE TILING »

GENERAL PERFORMANCE CONSIDERATION OF PIC CODES

- In Hybrid MPI/Open MP schemes hot spots may come from:

1. Shared memory part of the code (OpenMP)

- Field Gathering/Current deposition: cache misses (particles not contiguous in memory). Particle sorting/tiling?
- Unvectorized loops (flow-dependency, function call etc.) : re-write/organize some parts of the PIC code?

2. Distributed memory level (MPI)

- Uneven repartition of particles among processors: dynamic load balancing

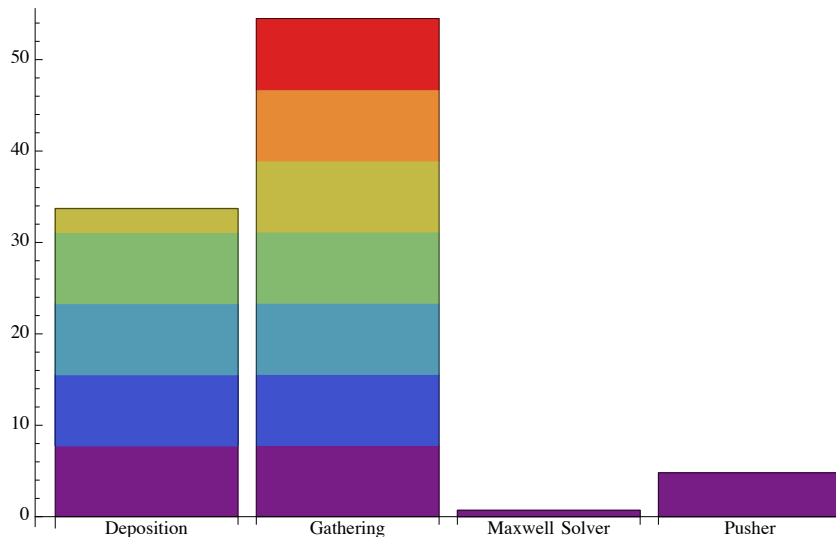
PERFORMANCE ANALYSIS OF WARP CORE ON EDISON

- Parallelization on Distributed memory architectures is very efficient.
 - Local method for solving Maxwell's equation,
 - One potential limitation: load balancing.
Dynamic Load balancing strategies could be implemented in WARP.
- Here we focus on what could limit parallel performance on shared memory architecture (basically on each node with OpenMP \geq 1thread).
 - **Relevant for future KnightsLanding architecture at NERSC.**
- Performance analysis of WARP kernel on EDISON/BABBAGE to identify potential hotspots:
 - First study: sequential mode (OMP_NUM_THREADS=1),
 - Second study: limitations in terms of scalability (OMP_NUM_THREADS $>$ 1).

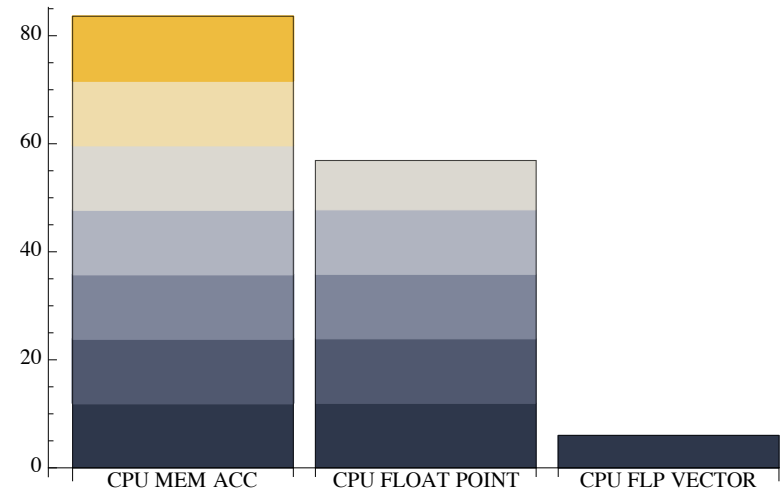
PERFORMANCE ANALYSIS OF WARP CORE ON EDISON

- First study: sequential mode (OMP_NUM_THREADS=1)
 - Total performance of the kernel (standard 2D3V laser-plasma acceleration case):

Time (% total)



Time (% total)

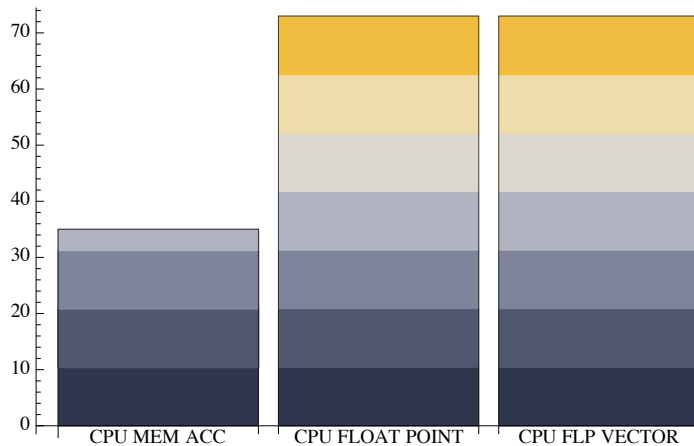


- Total time dominated by deposition/gathering subroutines
- High memory access and Low vectorization

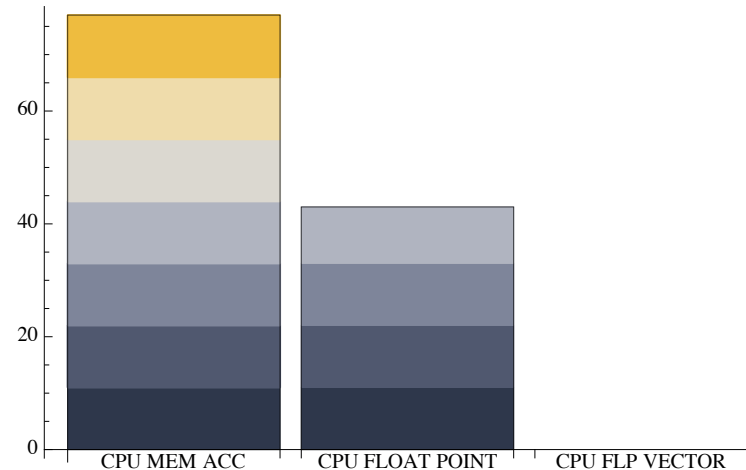
PERFORMANCE ANALYSIS OF WARP CORE ON EDISON

- First study: sequential mode (OMP_NUM_THREADS=1)
 - Performance of particle pusher vs. deposition&gathering

Pusher (% total)



Dep&Gat (% total)



- Moderate memory access,
- loop is well vectorized:
- we can expect good OpenMP performance

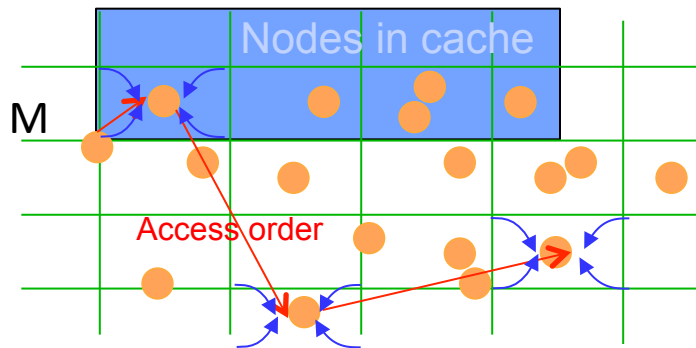
- High memory access (cache misses &/or inefficiently-structured code)
- loop is not vectorized
- poor future OpenMP performance?

PERFORMANCE ANALYSIS OF WARP CORE ON EDISON

1. First study: sequential mode (OMP_NUM_THREADS=1)

- Possible improvements for deposition/gathering routines:

Ex : Field gathering process



Code structure

```
DO i=1,nparticles
  DO inode=1,N_adjacentnodes
    Field(particle)= Field(particle)+Weight*Field(inode)
  END DO
END DO
```

- ◆ WITH CURRENT STRUCTURE, PARTICLE ACCESS ORDER INDUCE LOT OF CACHE MISSES
- ◆ PARTICLES MUST BE SORTED SO THAT CPU ACCESS CONTIGUOUS DATA IN CACHE: « PARTICLE TILING »

PERFORMANCE ANALYSIS OF WARP CORE ON BABBAGE

2. Second study: parallel mode (OMP_NUM_THREADS>1)

- *Particle pusher: preliminary optimization*

Initial structure of particle pusher

```
DO i=1,ntstep
! $OMP PARALLEL
! Push velocities
call pushv(particles) !loop on particles (OMP DO)
.....
! Push positions
call pushx(particles) !loop on particles (OMP DO)
....
!$OMP END PARALLEL
End DO
```

New structure (Merged loops)

```
DO i=1,ntstep
$OMP PARALLEL
! Push velocities
call pushvx(particles) !merged loop on
particles
$OMP END PARALLEL
END DO
```

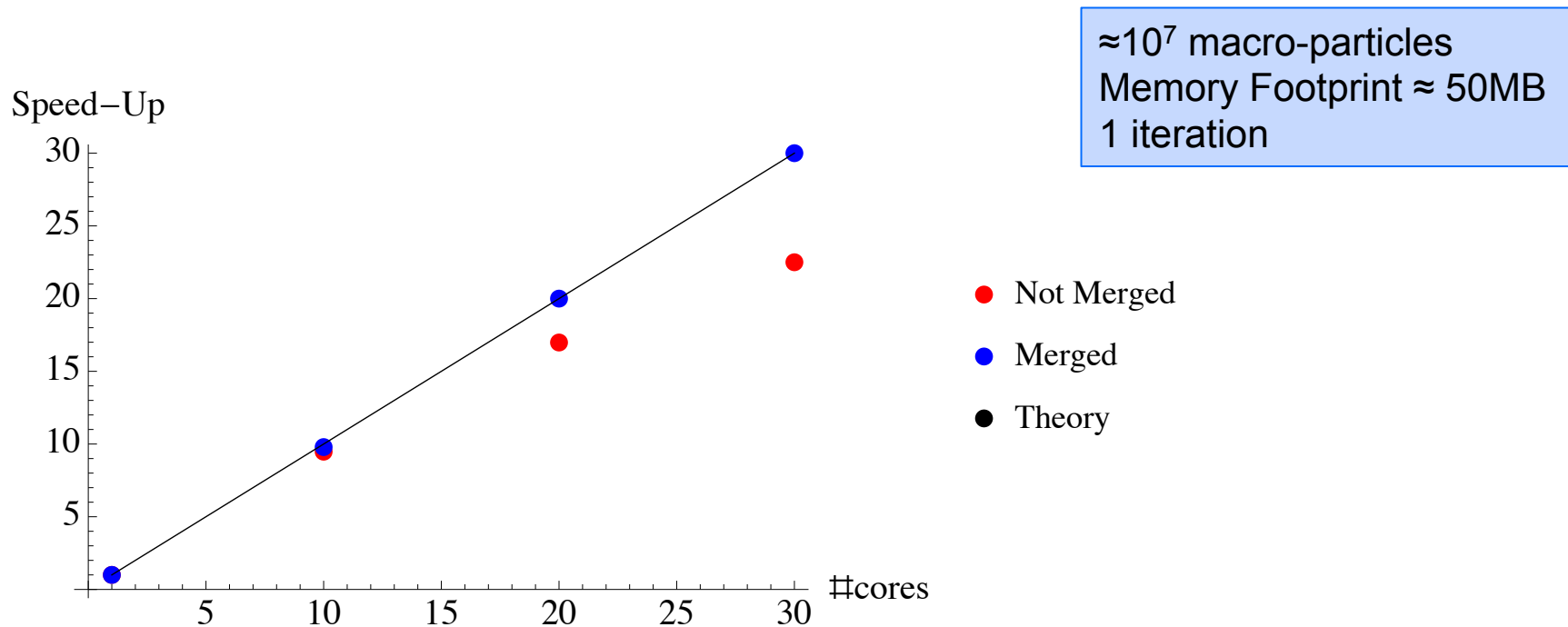
◆ MERGE LOOPS TO AVOID OVERHEADS COSTS

◆ BIGGER LOOPS MAY ALSO INCREASE VECTORIZATION PERFORMANCE AND REDUCE MEM ACC

PERFORMANCE ANALYSIS OF WARP CORE ON BABBAGE

2. Second study: parallel mode (OMP_NUM_THREADS>1)

- *Particle pusher: scaling test after simple optimization*



◆ VERY GOOD SCALING ON A FEW TENS OF CORES ON BABBAGE

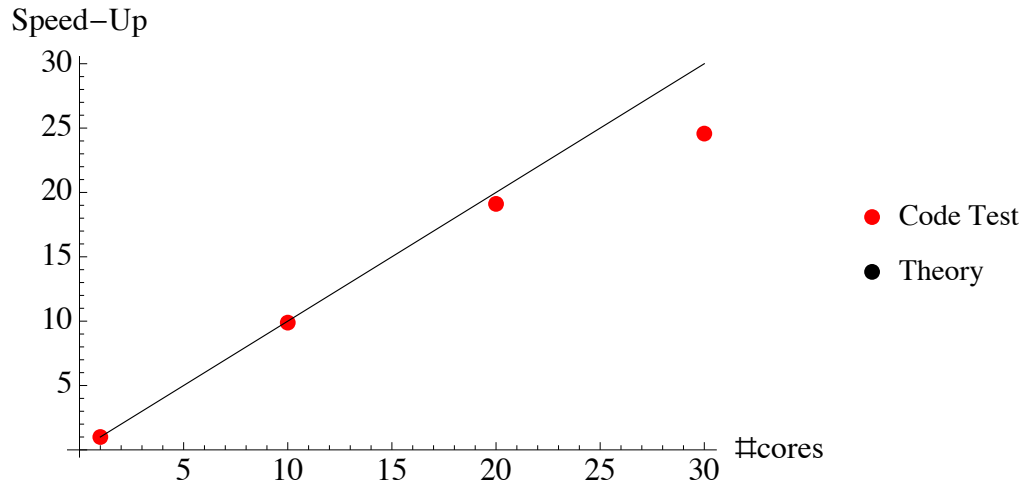
PERFORMANCE ANALYSIS OF WARP CORE ON BABBAGE

2. Second study: parallel mode (OMP_NUM_THREADS>1)

- Field gathering routine:

Code structure for field gathering subroutine

```
!$OMP PARALLEL DO PRIVATE(i,inode)
DO i=1,nparticles
  DO inode=1,N_adjacentnodes
    Field(particle)= Field(particle)+Weight*Field(inode)
  END DO
END DO
!$OMP END PARALLEL DO
```



≈10⁷ macro-particles
Nx=Ny=512
Memory Footprint ≈ 50MB
1 iteration

◆ GOOD SCALING ON A FEW TENS OF CORES ON BABBAGE

PERFORMANCE ANALYSIS OF WARP CORE ON BABBAGE

2. Second study: parallel mode (OMP_NUM_THREADS>1)

- Current deposition routine: code structure

Code structure for current deposition subroutine

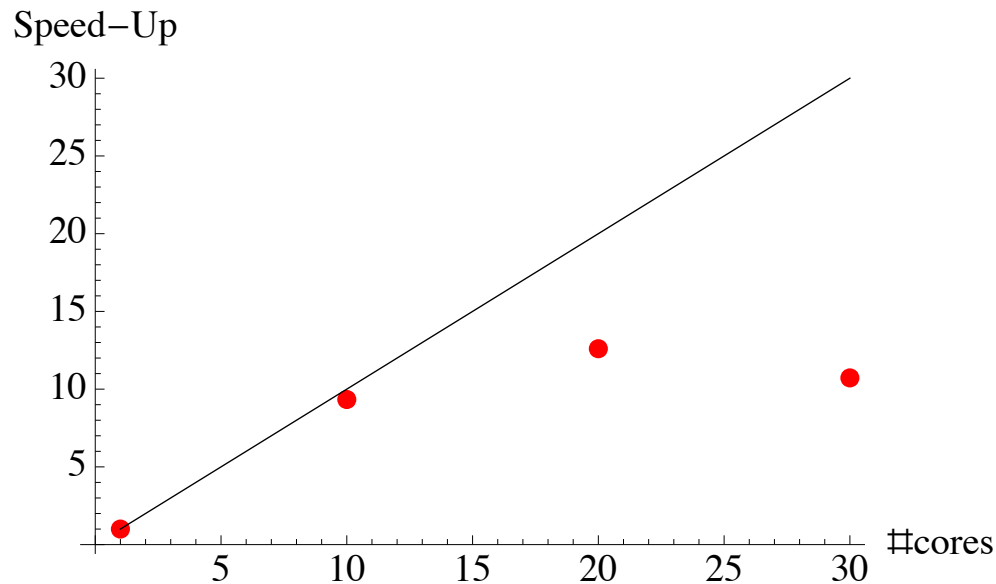
```
!$OMP PARALLEL
!$OMP DO PRIVATE(i,inode,LocalCurrent)
LocalCurrent=0.
DO ip=1,nparticles
  DO inode=1,N_adjacentnodes
    LocalCurrent(inode)=LocalCurrent(inode)+DeltaCurrent(ip)
  END DO
END DO
!$OMP END DO

!$OMP CRITICAL
Current=Current+LocalCurrent
!$OMP END CRITICAL
!$OMP END PARALLEL
```

PERFORMANCE ANALYSIS OF WARP CORE ON BABBAGE

2. Second study: parallel mode (OMP_NUM_THREADS>1)

- Current deposition routine: performance test



$\approx 10^7$ macro-particles
Nx=Ny=512
Memory Footprint \approx 50MB
1 iteration

- Code Test
- Theory

- ◆ BAD SCALING FOR LARGE NUMBER OF CORES (>10)
- ◆ NEEDS FURTHER OPTIMIZATION: LOOP ON NODES INSTEAD OF PARTICLES? → NO REDUCTION (LIKE FIELD GATHERING)

PERFORMANCE ANALYSIS OF WARP CORE ON BABBAGE

2. Second study: parallel mode (OMP_NUM_THREADS>1)

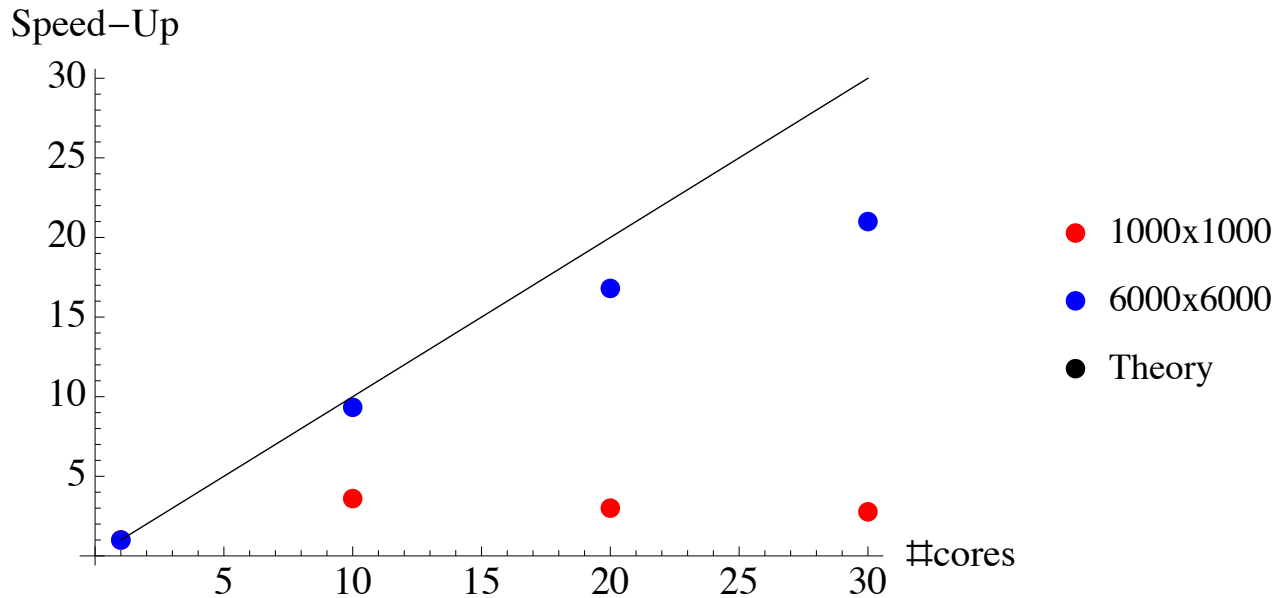
- Maxwell solver: code structure

```
!$OMP PARALLEL DEFAULT(SHARED) PRIVATE(l,j)
!$OMP DO COLLAPSE(2)
do l = 0, nz-1
  do j = 0, nx-1
    By(j,l) = By(j,l) + alphax*dtstdx * (Ez(j+1,l ) - Ez(j ,l )) &
      + betaxz*dtstdx * (Ez(j+1,l+1) - Ez(j ,l+1) &
        + Ez(j+1,l-1) - Ez(j ,l-1)) &
      - alphaz*dtstdz * (Ex(j ,l+1) - Ex(j ,l )) &
      - betazx*dtstdz * (Ex(j+1,l+1) - Ex(j+1,l )) &
        + Ex(j-1,l+1) - Ex(j-1,l ))
  end do
end do
!$OMP END DO
!$OMP END PARALLEL
```

PERFORMANCE ANALYSIS OF WARP CORE ON BABBAGE

2. Second study: parallel mode (OMP_NUM_THREADS>1)

- Maxwell solver: code performance



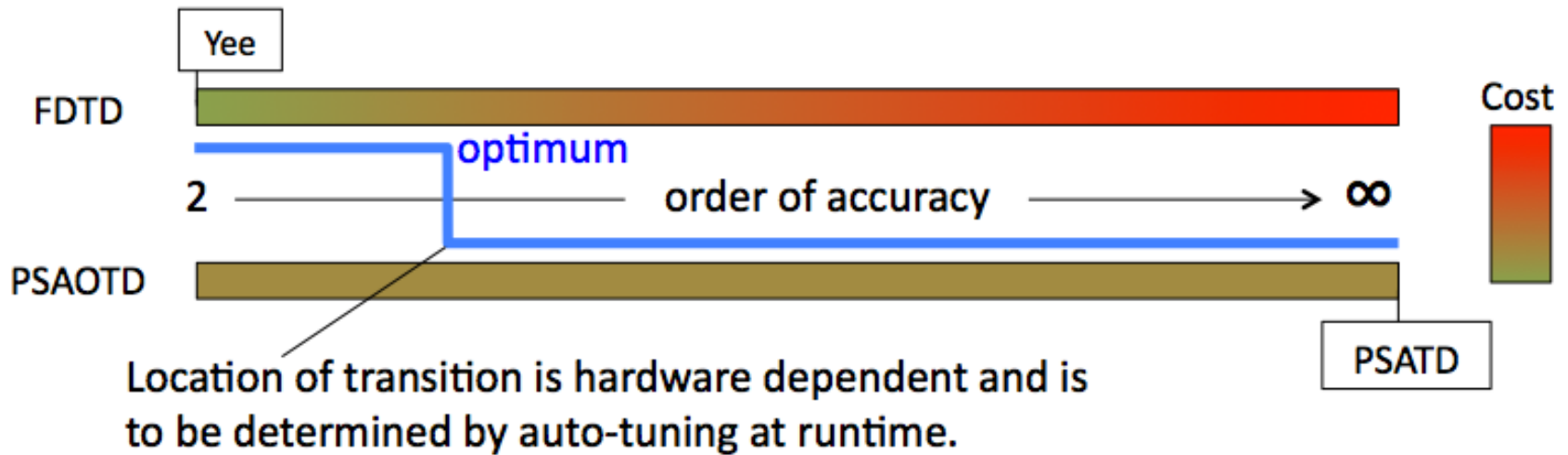
◆ GOOD SCALING ON A FEW TENS OF CORES FOR FIELD MATRIX>10MB

Long term: ultimate flexibility with runtime auto-tuning of order of accuracy vs locality

Larger stencils → higher order of accuracy but lower locality

Limit of infinite order with FFT.

FFT global but Maxwell local → local FFT on nodes or group of nodes*



Warp+VisIt

- The science problem we want to solve as part of NESAP
 - Reduce I/O cost (write/read data to/from persistent storage for storage/post-process.)
 - Analyze and visualize data while it is generated
- High level architecture/algorithm of code for the Intel and Cray people
 - Tightly coupled *in situ* visualization and analysis using VisIt
 - Vis. algo. required for accelerator application range from standard plotting functions to volume visualizations, surface extraction, particle advections and many more
- Potential challenges for getting the code to run well on KL
 - Vis. & analyses have greedy memory use. Limited memory per core and competition for memory between the sim and vis are a potential problem.
 - Need to address coordination & synchronization between simulation & vis which will use distinct thread pools (e.g. NUMA issues as data is transferred from one thread pool to the other).
 - VisIt's threading is new & relatively untested. We expect to identify and address performance issues as we apply complex pipelines used by scientists.
 - Will work closely with SDAV on these issues. Not just an issue for VisIt but for high-performance visualization in general. SDAV has a NESAP proposal focusing on these issues.
 - Simulations have best case scenario for optimization on the MIC. Legacy C++ vis codes have the worst case scenario (e.g. fewer opportunities for vectorization exist. Need to narrow the performance gap)

Warp+VisIt

- Performance data and code profiles:
 - David Camp, Wes Bethel and Hank Childs. "Transitioning Data Flow-Based Visualization Software to Multi-Core Hybrid Parallelism". Proceedings of Data-Flow Execution Models for Extreme Scale Computing (DFM 2013) conference. LBNL-6363E. [\(bibtex\)](#) [\(PDF\)](#)
 - <http://vis.lbl.gov/Vignettes/TrillionZones/>
- How NERSC, Cray, Intel can help
 - Performance and thread profiling
 - Vis writes many small files. Burst buffers slated for Cori can be used for in-situ vis related I/O.
 - *In situ* analysis can benefit from burst buffers as an active cache to save data needed for analysis out-of-core (e.g, for analyses across time-steps, acceleration structures for data search/query, topology graphs etc.)