CCMT

Scalable Performance Prediction of Irregular Workloads in Multi-Phase Particle-In-Cell Applications

Sai Prabhakar Rao Chenna[1], Sivaramakrishnan Balachandar[2], Greg Stitt[1], Herman Lam[1]

[1]Department of Electrical and Computer Engineering

[2]Mechanical and Aerospace Engineering

Center for Compressible Multiphase Turbulence (CCMT)

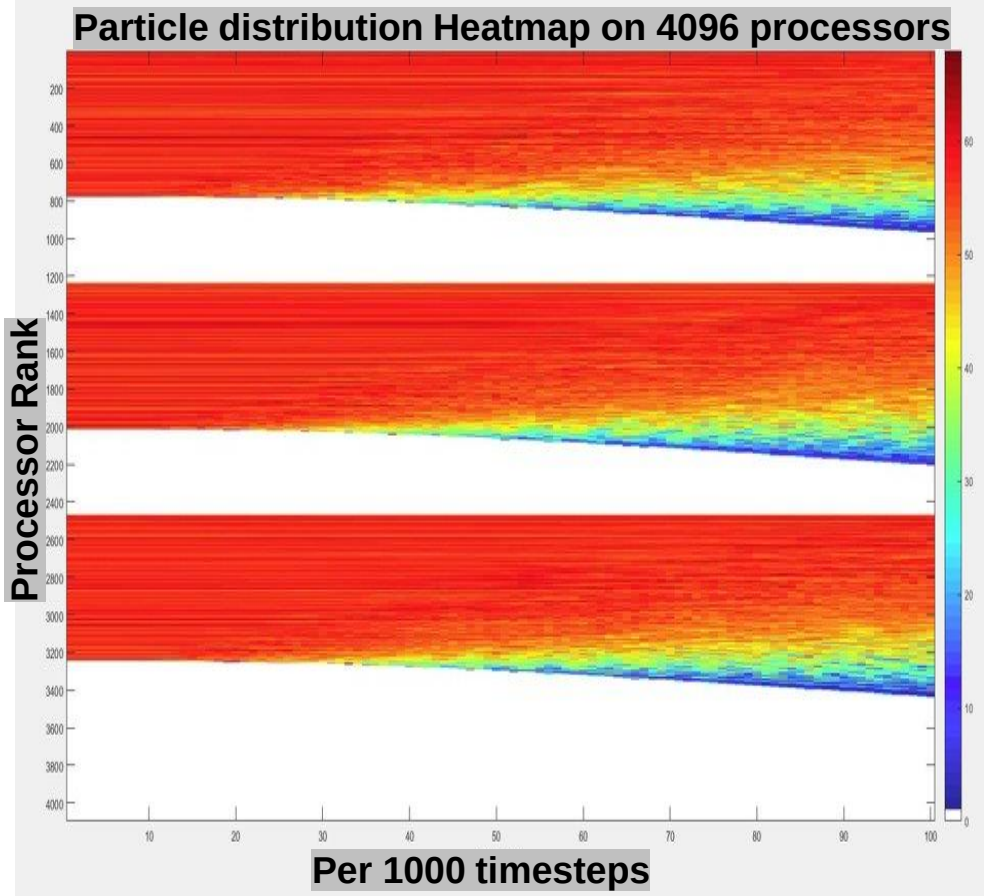
University of Florida, Gainesville, Florida 32611

Outline

- **Motivation & Goal**
  - Need for Scalable Performance prediction of Irregular workloads on Large-scale systems
  - Multi-Phase Particle-In-Cell (MP-PIC) Applications
- **Approach**
  - Trace-driven Workload prediction
  - Performance Modeling & Simulation
  - Advantages & Limitations
- **Evaluation**
  - CMT-nek HPC application
  - Particle-decomposition algorithms
  - Hele-shaw simulation case-study
  - Performance predictions
- **Conclusions & going forward**

Motivation

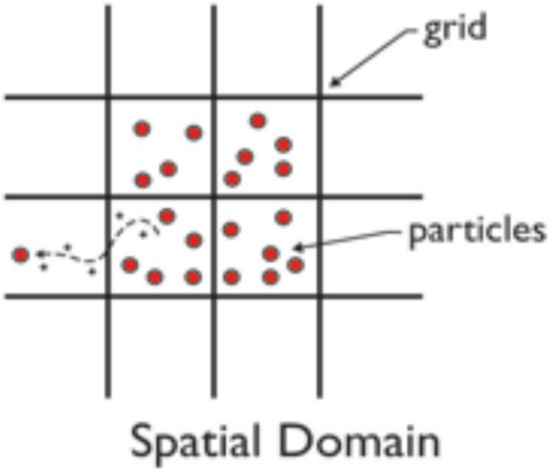
- Performance prediction is critical in order to identify performance bottlenecks on large-scale systems
  - Cycle-accurate simulations infeasible owing to system complexity
  - Analytical performance modeling approaches useful to generate faster performance predictions
- Most applications employ static workload distribution when scaling on large-scale systems
  - Some applications generate dynamic workload during execution e.g. MP-PIC – thereby making it harder to predict performance

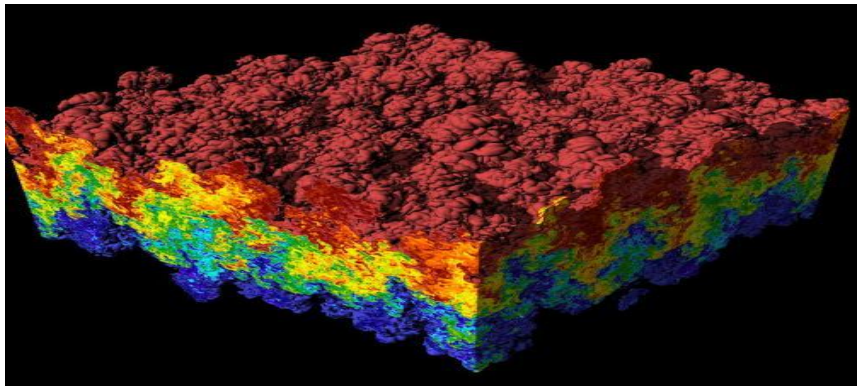


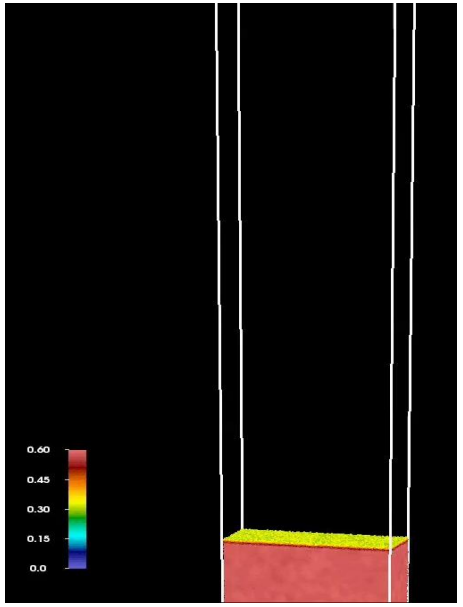
- Need an accurate, scalable prediction framework for such irregular applications

Multi-Phase Particle-In-Cell(MP PIC)

- Multi-phase PIC:
  - Used to model fluid-particle and particle-particle interaction in CFD applications
  - Eulerian frame of reference for fluid phase
  - Lagrangian frame of reference for particle phase
- Applications:
  - Cyclone simulation
  - Fluidized bed reactors
  - Chemical looping combustion

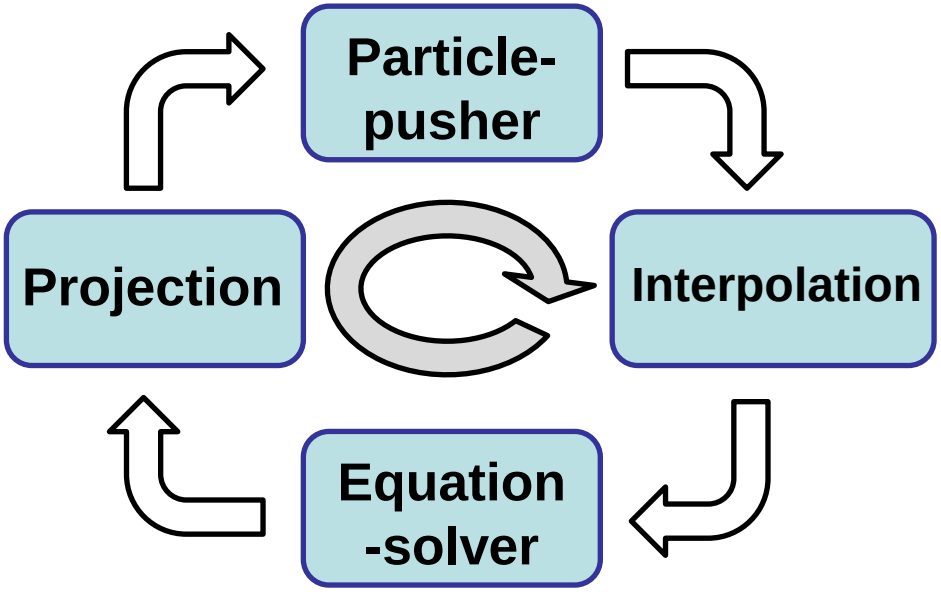






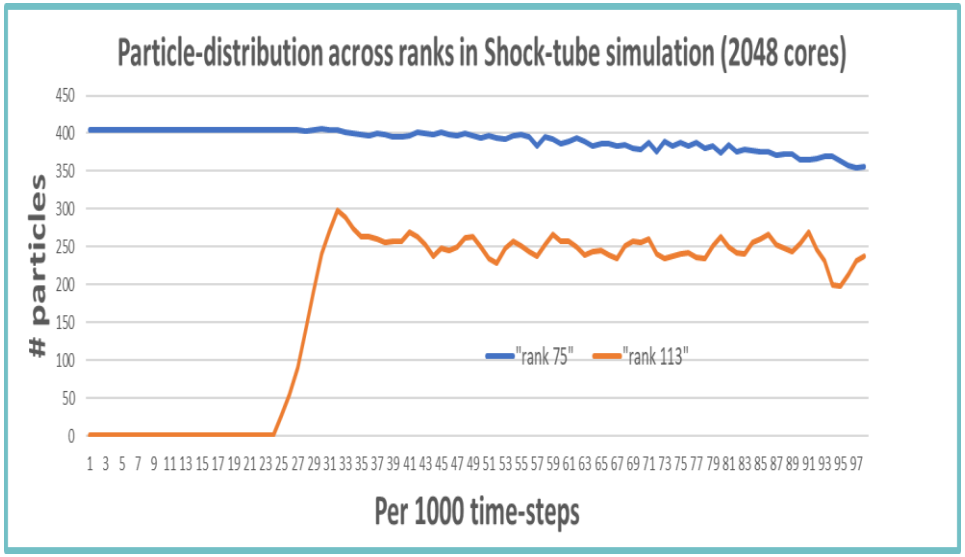
MP-PIC: Particle solver loop

- Key stages of operations
  - **Particle-pusher:** Updating particle location based on the change in position
  - **Interpolation:** Interpolate eulerian (fluid) properties from the grid-points on to the lagrangian particle
  - **Equation solver:** Calculate the lagrangian properties at each individual particle location
  - **Projection:** Project back lagrangian properties back to eulerian grid.



MP-PIC: Computation workload

- Computational workload is categorized into:
  - Element workload (# of spectral elements/processor) – ***static***
    - Remains fixed upon initial domain decomposition
  - Particle workload (# of particles/processor) - ***dynamic***:
    - **Varies among processors:** depends on problem and mapping algorithm
    - **Varies during simulation:** depends on particle velocity (problem-dependent)
- Difficult to predict performance cost due to dynamic workload variation

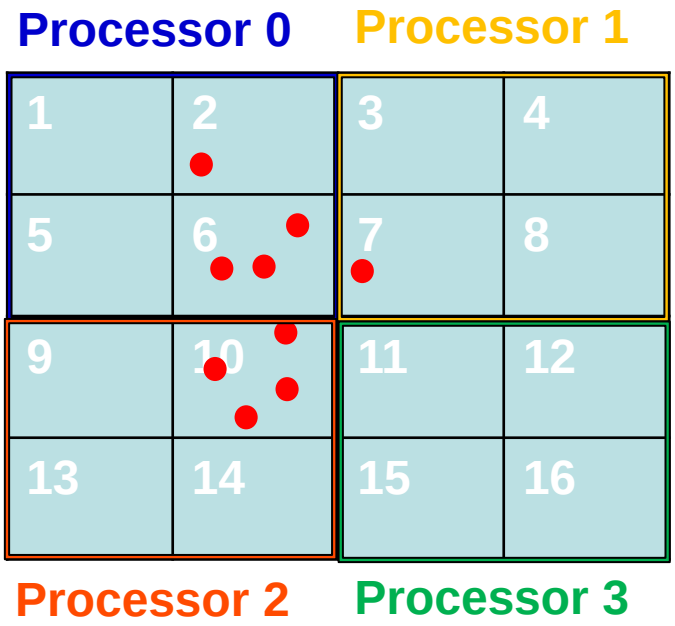


- Particle workload is a significant parameter affecting application performance
  - Particle pusher ***O(Np)***
  - Interpolation ***O(Np+Ng)***
  - Projection ***O(Np+Ng)***
  - Collision forces ***O(Np^2+Np*Ng)***

Np – real particles ; Ng – ghost particles

MP PIC: Particle workload

- Particle workload(# of particles/processor) is ***dynamic***:

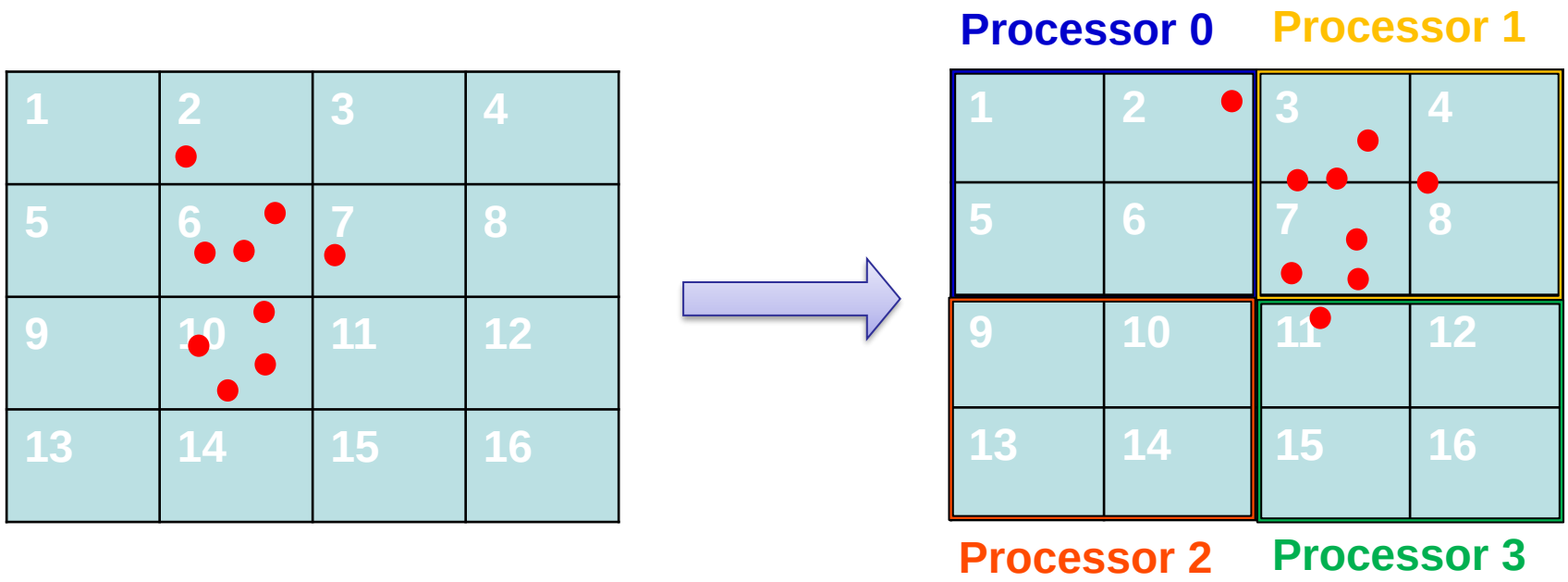


Processor	# of elements	# of particles
0	4	4
1	4	1
2	4	4
3	4	0

- **Varies among processors:** depends on problem and mapping algorithm

MP PIC: Particle workload

- Particle workload(# of particles/processor) is ***dynamic***:

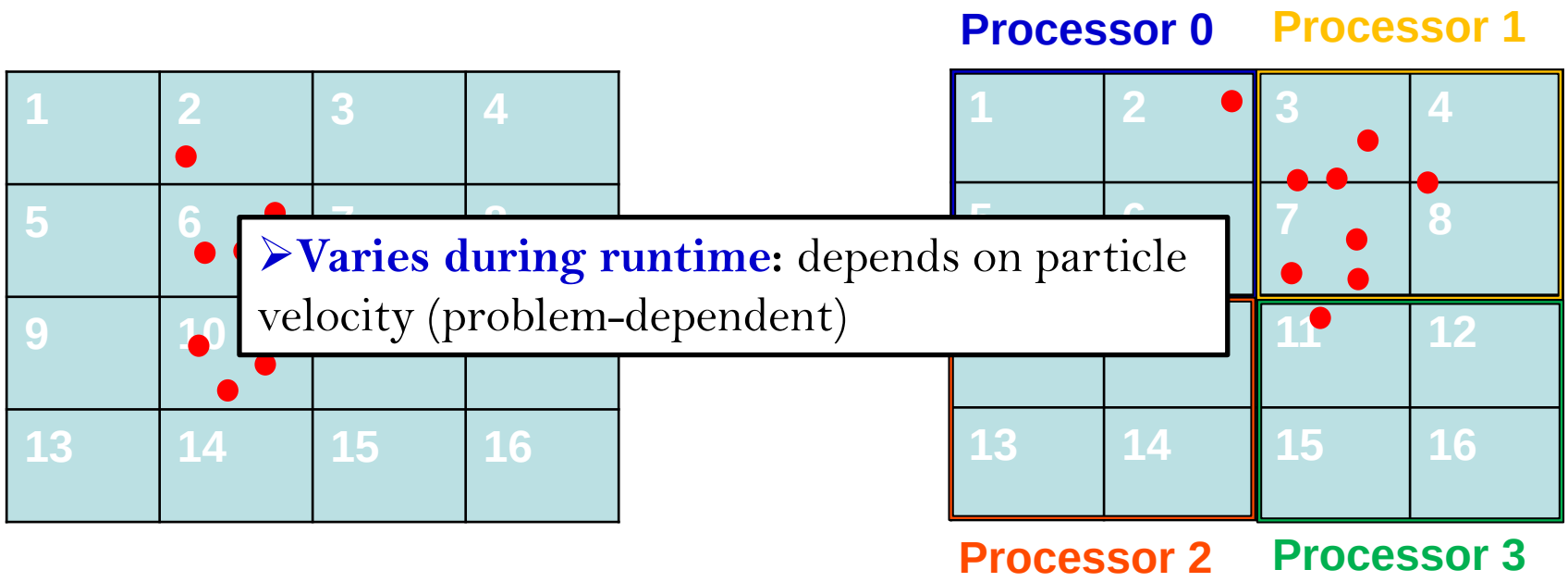


Processor	# of elements	# of particles
0	4	4
1	4	1
2	4	4
3	4	0

Processor	# of elements	# of particles
0	4	1
1	4	7
2	4	0
3	4	1

MP PIC: Particle workload

- Particle workload(# of particles/processor) is ***dynamic***:



Processor	# of elements	# of particles
0	4	4
1	4	1
2	4	4
3	4	0

Processor	# of elements	# of particles
0	4	1
1	4	7
2	4	0
3	4	1

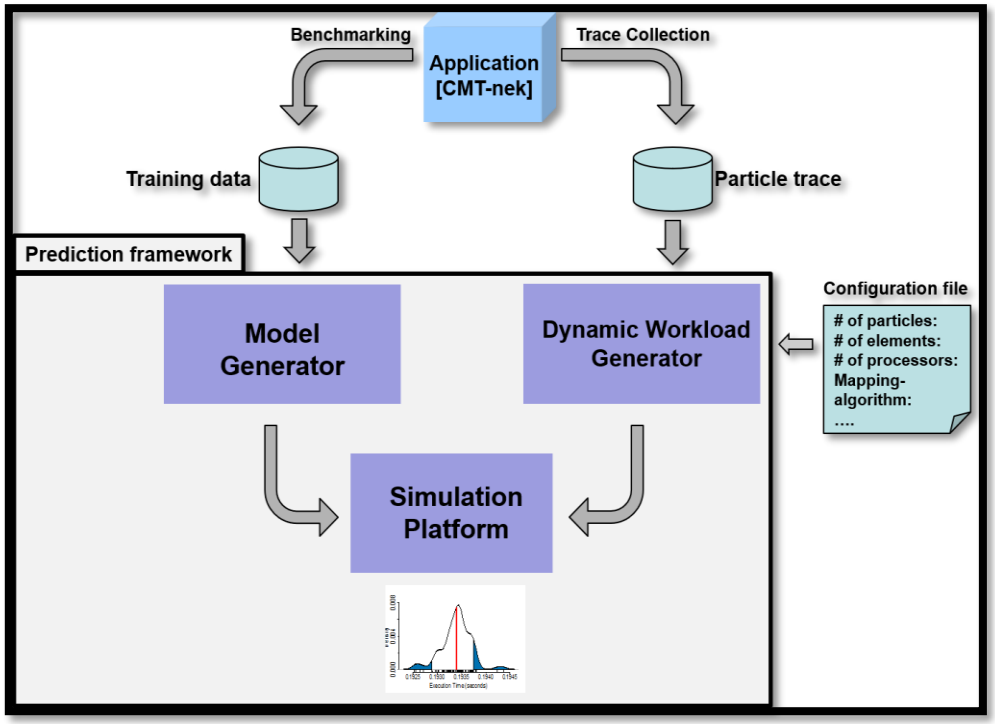
Approach

❑ Problem:

- Performance Prediction of MP-PIC applications is difficult owing to dynamic workload fluctuations

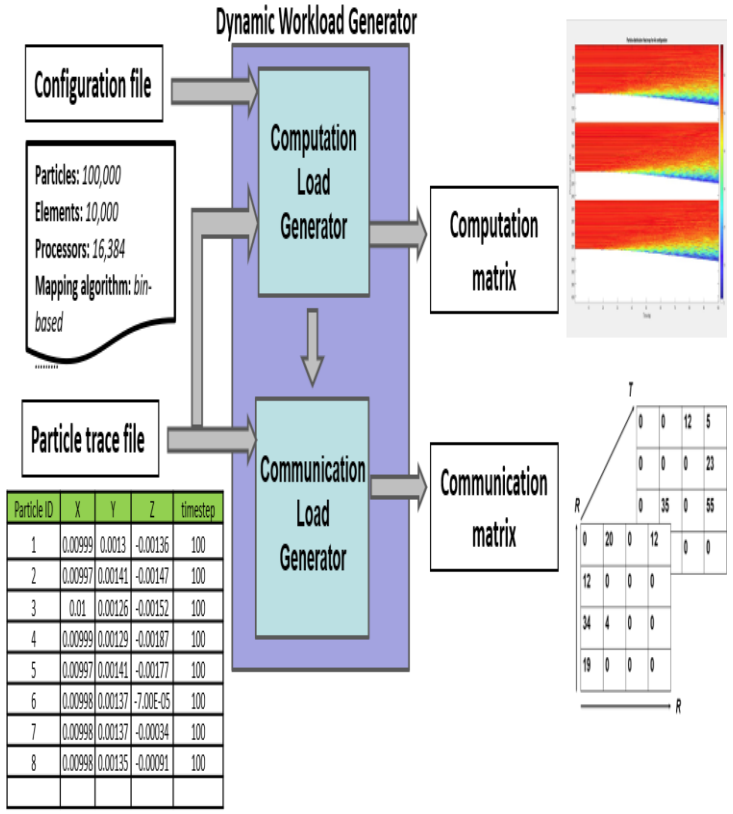
❑ Solution:

- **Trace-driven** performance prediction framework that:
- generates **dynamic workload** on a target system using **application trace**
- Provides a **fast modelling and simulation** platform to predict application performance



Dynamic Workload Generator (DWG)

- **Goal:**
  - Accurately predict the particle workload on each processor during the simulation
- **Principle:**
  - Particle decomposition algorithm relies on particle location to map particles onto processor
  - Particle location is independent on system configuration – problem dependent
  - Hence single application trace is sufficient to predict particle workload on any system configuration
- **Dynamic Workload Generator:**
  - Generate the particle workload per processor by mimicking the particle-mapping algorithm on the input particle trace



Dynamic Workload Generator (DWG)

Input:

- Configuration file:
  - Application configuration: Particle count, mapping algorithm
  - System configuration: Processor count
- Trace file:
  - Particle location sampled at fixed intervals across execution

Output:

- Computation matrix
  - Specifying total particles per processor at a given iteration
- Communication matrix:
  - Number of particles crossing processor domain between consecutive sample intervals

Input

Particle trace

Particle ID	location	timestep
(8,638,0)	(0.13009,0.00007,0.0023)	200
(8,678,0)	(0.13008,0.00138,0.00172)	200
(8,1109,0)	(0.13003,0.00286,0.00148)	200
(8,1262,0)	(0.13008,0.00097,0.00208)	200
(8,1276,0)	(0.13008,0.00134,0.00089)	200
(8,1307,0)	(0.13008,0.00122,0.00306)	200
(8,1442,0)	(0.13002,0.00017,0.00178)	200
(27,594,0)	(0.13003,0.0027,0.00049)	200
(27,879,0)	(0.13006,0.0022,0.00004)	200
(27,971,0)	(0.13007,0.00326,0.00078)	200

Configuration file

```
# of elements:
# of particles:
Box-dimensions:
Particle-algorithm:
Trace-file:
Element-file:
....
```

Output

Computation cost

Processor-ID	# of particles	Time-step
1346	9735	200
1347	10018	200
1348	9744	200
1349	9726	200
1350	9736	200
1351	10026	200
1352	9731	200
1353	9742	200

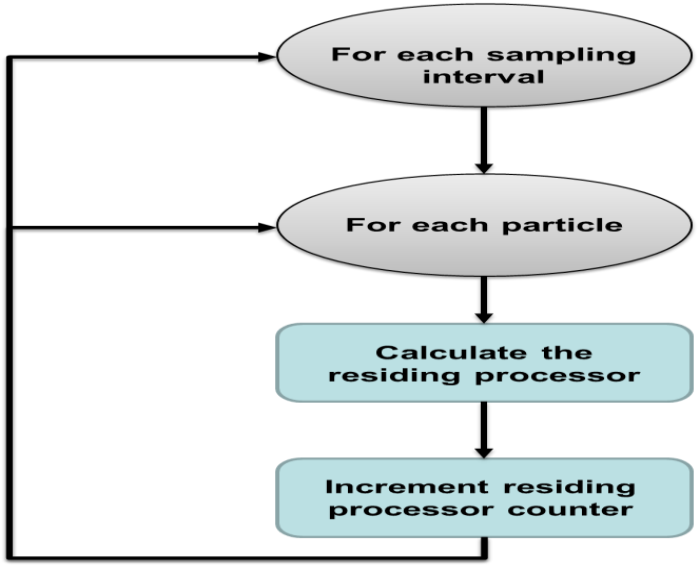
Communication cost

Source rank	Destination rank	# of particles	Time-step
860	857	100	200
881	867	300	200
894	897	100	200
862	863	100	200
885	897	200	200

Dynamic Workload Generator (DWG)

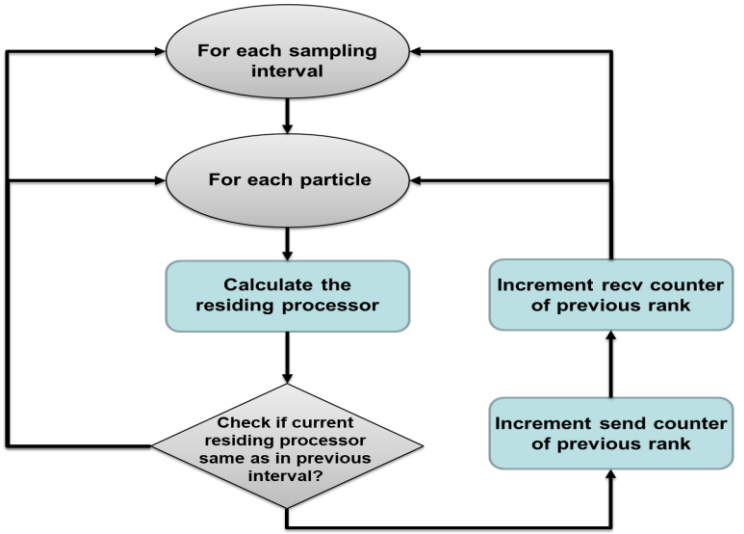
Computation Load Generator:

- Calculate processor owning the particle by mimicking the mapping algorithm
- Increment the particle counter for the corresponding processor
- Repeat across all the sampling intervals



Communication Load Generator:

- Check if the owner (processor) of particle same across two consecutive sampling intervals
- If not, increment the send counter of the previous owner and receive counter of the current owner for the given sampling interval



Dynamic Workload Generator: Workflow

Input

Particle trace

Particle ID	location	timestep
(8,638,0)	(0.13009,0.00007,0.0023)	200
(8,678,0)	(0.13008,0.00138,0.00172)	200
(8,1109,0)	(0.13003,0.00286,0.00148)	200
(8,1262,0)	(0.13008,0.00097,0.00208)	200
(8,1276,0)	(0.13008,0.00134,0.00089)	200
(8,1307,0)	(0.13008,0.00122,0.00306)	200
(8,1442,0)	(0.13002,0.00017,0.00178)	200
(27,594,0)	(0.13003,0.0027,0.00049)	200
(27,879,0)	(0.13006,0.0022,0.00004)	200
(27,971,0)	(0.13007,0.00326,0.00078)	200

Configuration file

of elements:
# of particles:
Box-dimensions:
Particle-algorithm:
Trace-file:
Element-file:
….

Output

Computation cost

Processor-ID	# of particles	Time-step
1346	9735	200
1347	10018	200
1348	9744	200
1349	9726	200
1350	9736	200
1351	10026	200
1352	9731	200
1353	9742	200

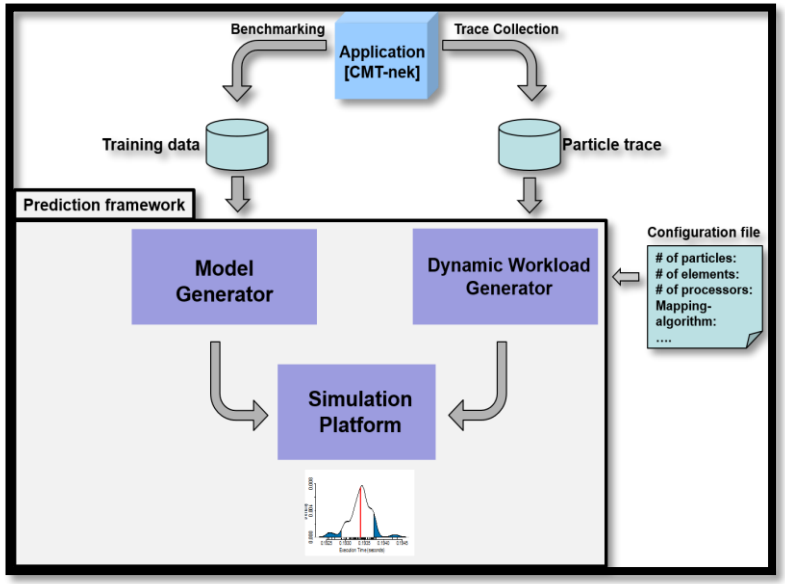
Communication cost

Source rank	Destination rank	# of particles	Time-step
860	857	100	200
881	867	300	200
894	897	100	200
862	863	100	200
885	897	200	200

Modeling and Simulation

❑ Model Generation:

- ➢ Empirical modelling approach to generate analytic performance models
- ➢ Framework supports multiple regression methods – linear regression, symbolic regression
- ➢ Linear regression – efficient for single, two parameter models
- ➢ Symbolic regression[1] – to generate fast and accurate multi-parameter models



❑ Simulation Platform:

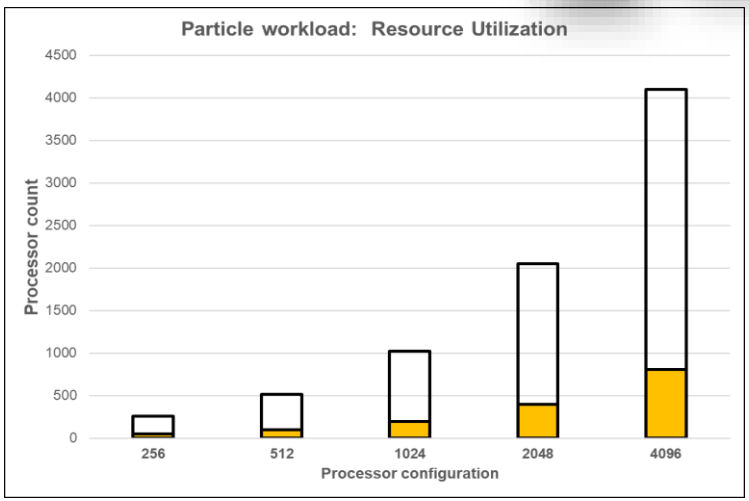
- ➢ Input: Dynamic workload, performance models, system configuration
- ➢ Output: Predicted Application time
- ➢ Simulator: BE-SST[2]
- ➢ Currently, does not support trace-driven simulation

1 Chenna, Sai P., Greg Stitt, and Herman Lam. "Multi-Parameter Performance Modeling using Symbolic Regression." 2019 International Conference on High Performance Computing & Simulation (HPCS). IEEE, 2019.
2 Ramaswamy, Ajay, et al. "Scalable behavioral emulation of extreme-scale systems using structural simulation toolkit." Proceedings of the 47th International Conference on Parallel Processing. 2018

Advantages & Limitations

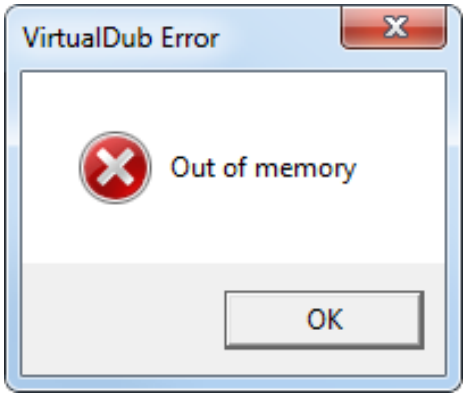
❑ Advantages:

- **Scalability Prediction:**
  - Identify scalability bottlenecks
  - Evaluate optimal processor count for large-scale PIC simulation
- **Algorithm evaluation:**
  - Platform to evaluate multiple particle decomposition algorithms
  - Low-cost implementation provides quick "proof of concept"
- **Parameter tuning:**
  - Tune application parameters based on their impact on performance



❑ Limitations:

- **Trace collection:**
  - Difficult for large-scale runs – expensive & often infeasible
- **Trace files are huge** – usually 10-100GB
  - File-size ∝ (# of particles * sampling-frequency)

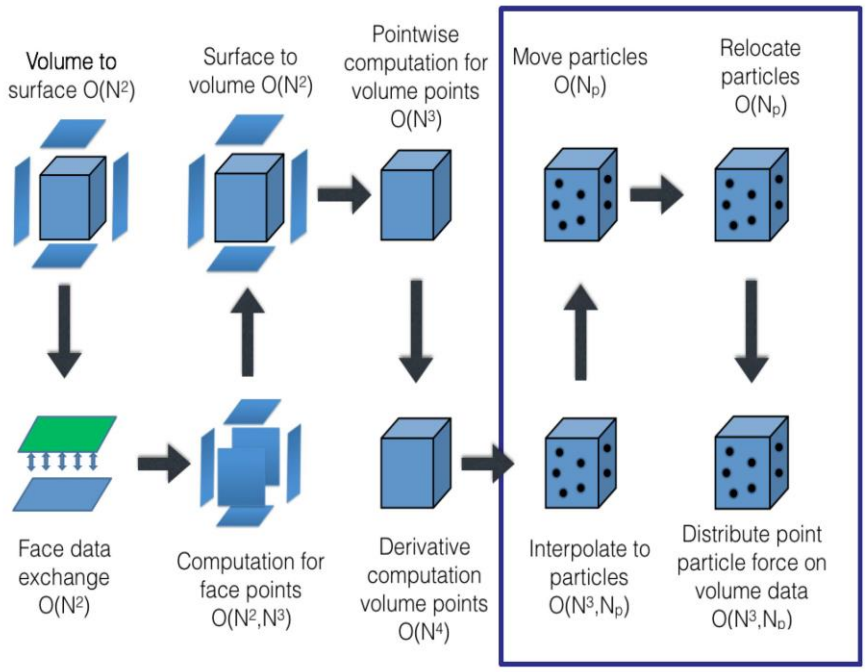


Outline

- Motivation & Goal
  - Need for Scalable Performance prediction of Irregular workloads on large-scale systems
  - Multi-Phase Particle-In-Cell (MP-PIC) Applications
- Approach
  - Trace-driven Workload prediction
  - Performance Modeling & Simulation
  - Advantages & Limitations
- **Evaluation**
  - CMT-nek HPC application
  - Particle decomposition algorithms
  - Hele-shaw simulation case-study
  - Performance predictions
- Conclusions & going forward

Application: CMT-nek (particle-solver)

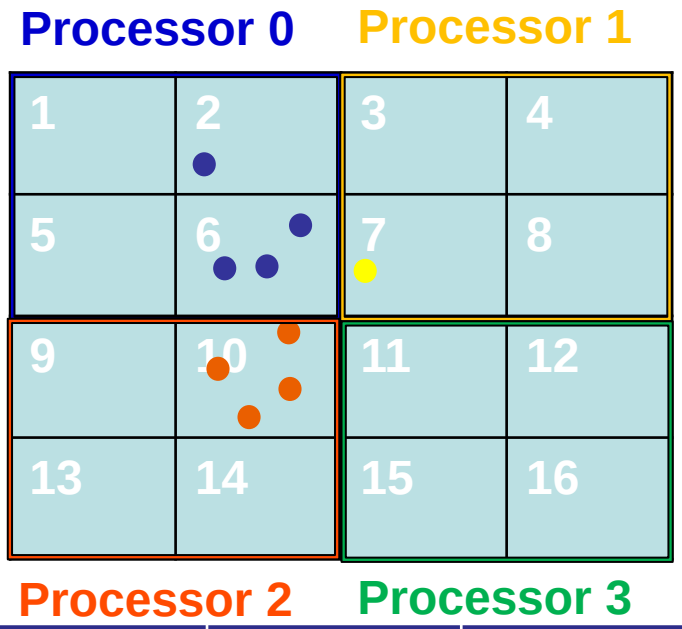
- HPC application, Center for Compressible Multiphase Turbulence University of Florida
  - Proposed solver of compressible Navier-stokes equation for compressible multiphase flows



Architecture: Quartz @ LLNL
- Intel Xeon E5 (Broadwell)
- 36 cores/node, 3018 nodes, 108k cores
- 128GB memory/node

Particle Decomposition: Element-based

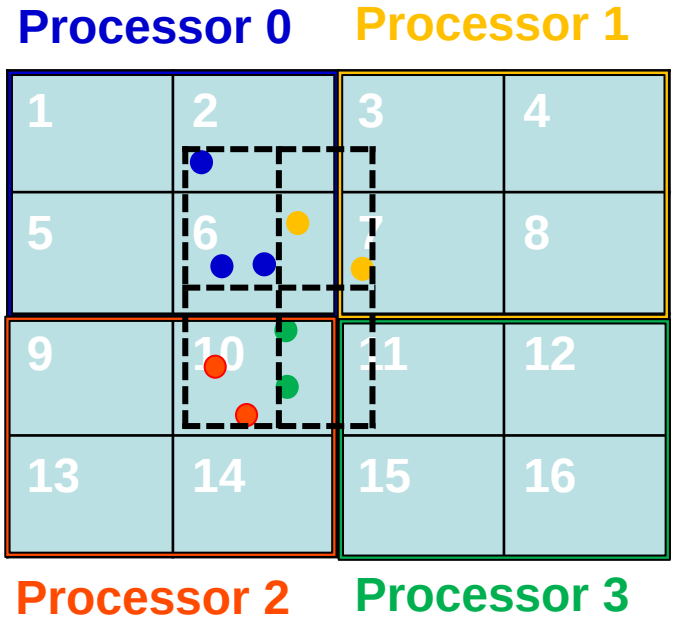
- Standard particle decomposition technique
- Assigns particle to processor which owns corresponding element
- **Pros:**
  - Tight particle-grid interaction
  - Many operations are local to each rank
- **Cons:**
  - Load/memory imbalance



Processor	# of elements	# of particles
0	4	4
1	4	1
2	4	4
3	4	0

Particle Decomposition: Bin-based*

- Decoupling particle-fluid for better scalability
- Each particle is stored with nearby particles in a bin-structure
- **Pros:**
  - Good load/memory balance
  - Better scalability
- **Cons:**
  - Increased inter-processor communication at every iteration
  - Additional computational overhead – bin calculation

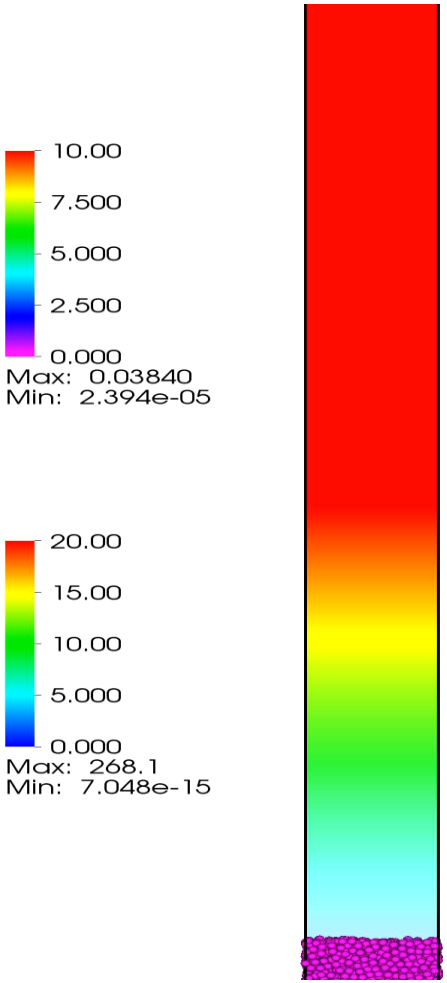


Processor	# of elements	# of particles
0	4	3
1	4	2
2	4	2
3	4	2

* *Zwick, David. "ppiclF: a parallel particle-in-cell library in Fortran." Journal of Open Source Software 4.37 (2019): 1400.*

Case-study: Hele-shaw simulation*

- Case-study: Hele-shaw simulation
- Particles: 599,257
- Elements: 216,225
- Grid-size: 4
- Trace sampling frequency: 100



** R. B. Koneru et al., "A numerical study of particle jetting in a dense particle bed driven by an air-blast," Physics of Fluids, vol. 32, no. 9, p.093301, 2020.*

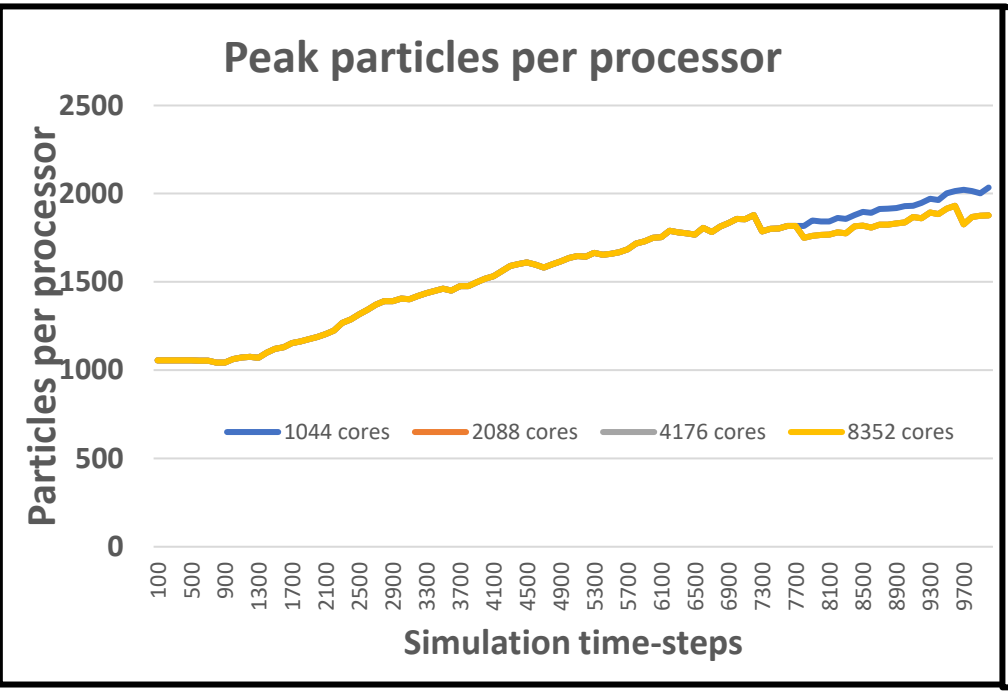
Results: Scalability prediction

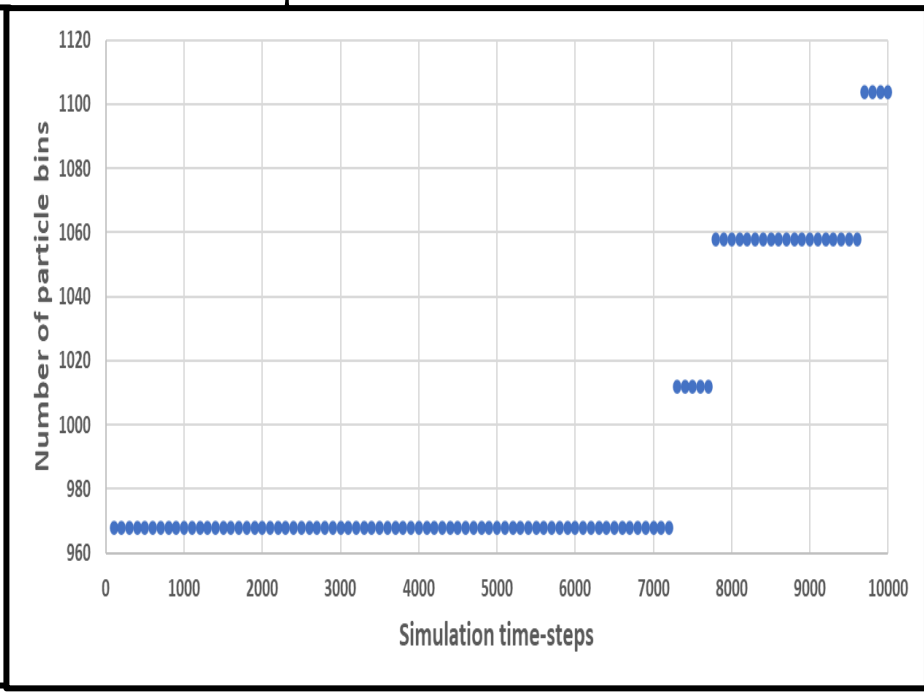
❑ Setup:

- ➢ Particle decomposition: Bin-based mapping
- ➢ Processor count: 1044, 2088, 4176, 8352

❑ Observations:

- ➢ Peak particle-workload did not improve from 2088 processors
- ➢ Maximum number of particle-bins generated – 1104
- ➢ Increasing processor count beyond 1104 wouldn't scale particle workload





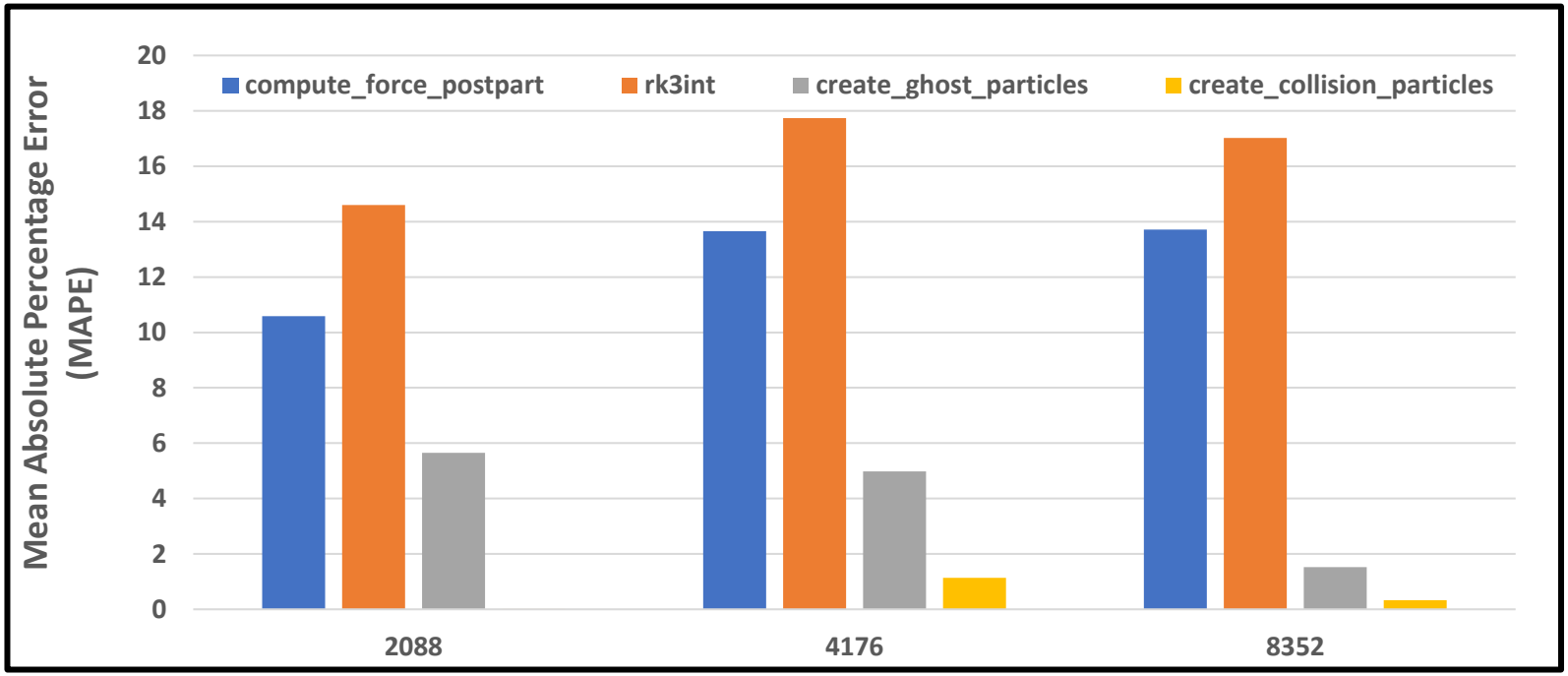
Results: Scalability prediction

❑ Setup:

- Particle decomposition: Bin-based mapping
- Processor count: 1044, 2088, 4176, 8352

❑ Observations:

- Performance prediction of key particle-solver kernels
- Average MAPE error – ***8.42%***
- Peak MAPE error – ***17.7%***



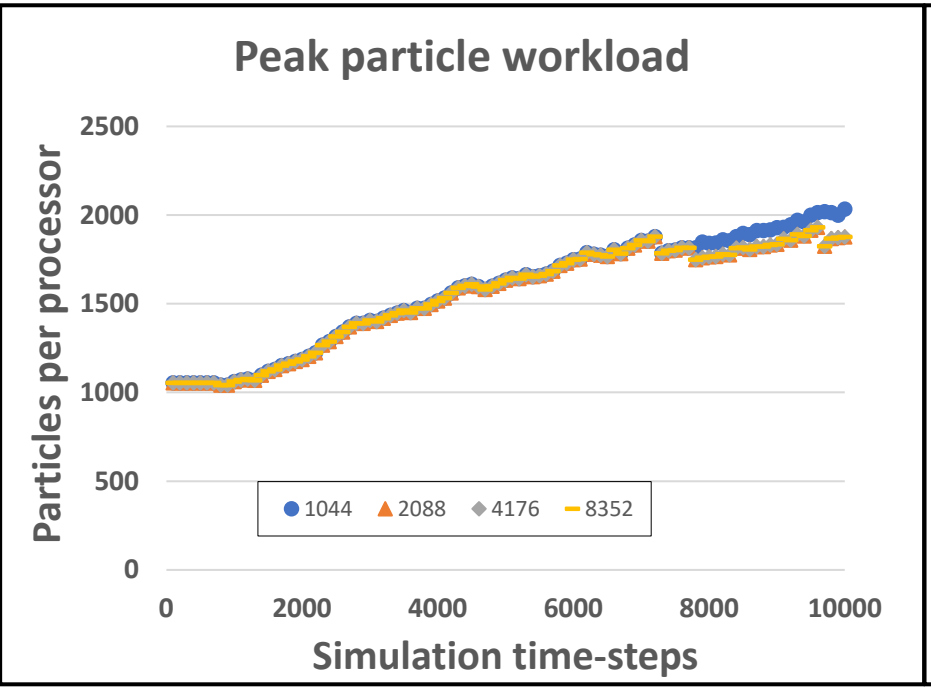
Results: Algorithm evaluation

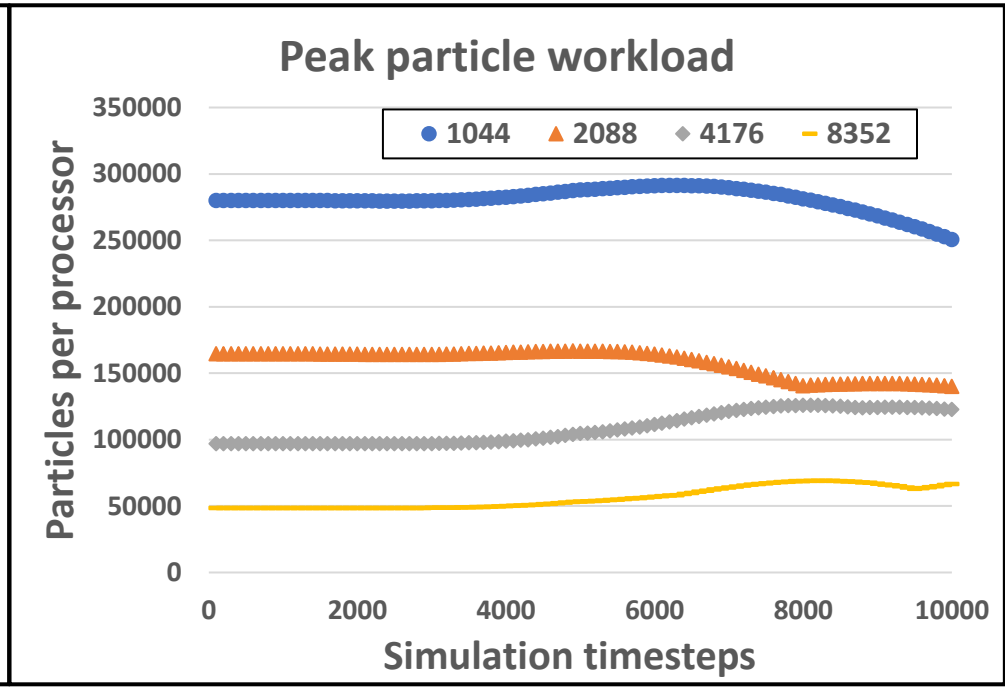
Setup:

- Particle decomposition: Bin-based mapping, element-based mapping
- Processor count: 1044, 2088, 4176, 8352

Observations:

- Bin-based mapping has better particle workload distribution
- Couple of orders magnitude improvement in peak particle workload





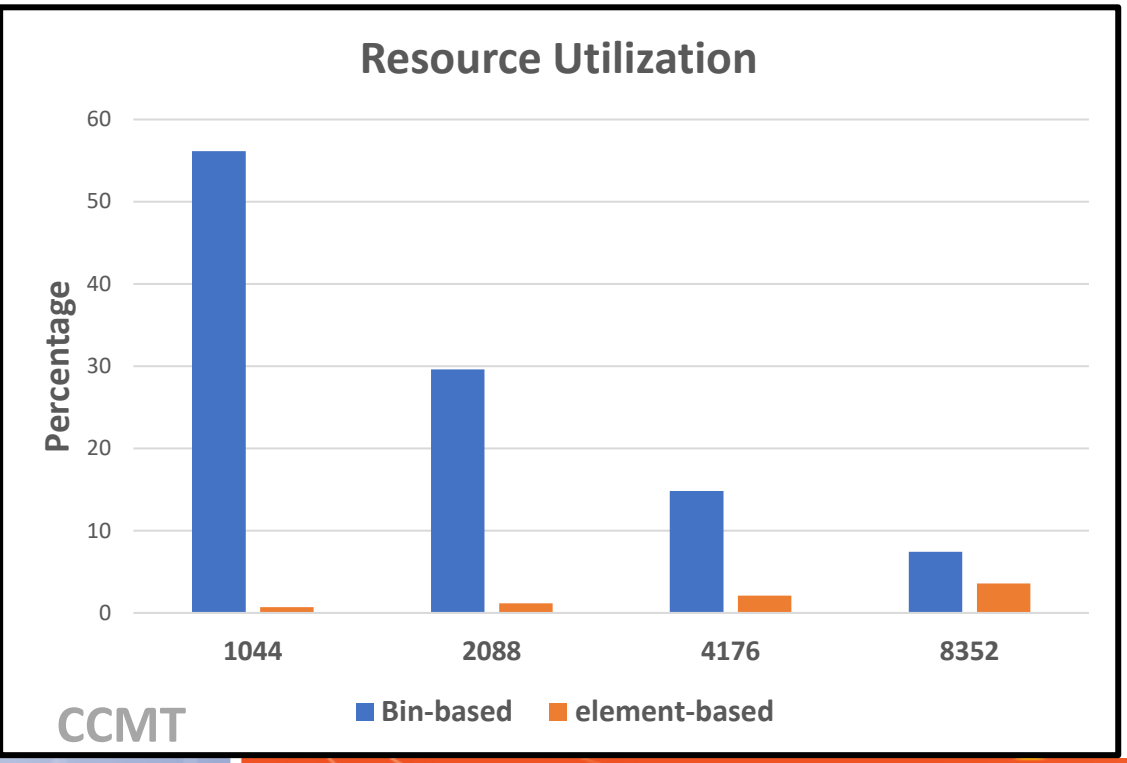
Results: Algorithm evaluation

Setup:

- Particle decomposition: Bin-based mapping, element-based mapping
- Processor count: 1044, 2088, 4176, 8352

Observations:

- Resource Utilization(RU*) is poor in case of element-based mapping for Hele-shaw case-study
- RU worsens for bin-based mapping when increasing processor count beyond 1044
  - Ideal processor count: 1104

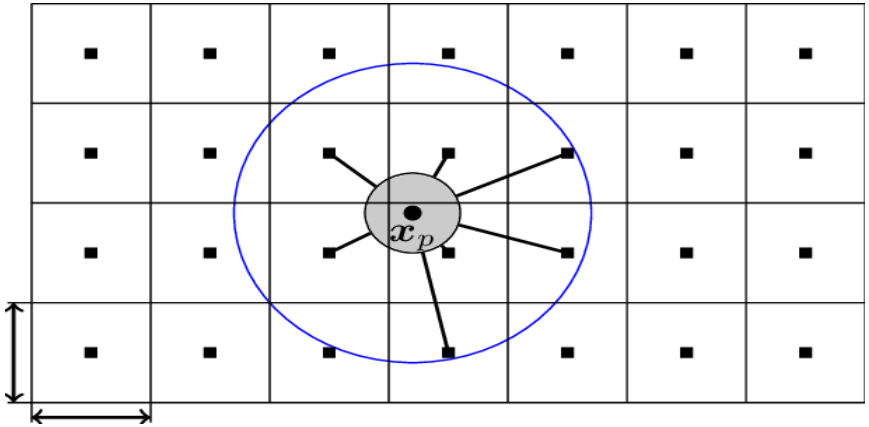


*RU = (# of processor with atleast 1 particle per processor) / total # of processors

Results: Parameter Tuning

Projection filter:

- Defines particle zone of influence on neighboring grid-points
- Filter size has an impact on performance:
  - Determines number of ghost particles per processor (N_g) – ***particle workload***
  - Determines Number of particle bins – ***scaling threshold***
- Identifying performance cost is crucial to evaluate ***accuracy vs cost*** trade-off



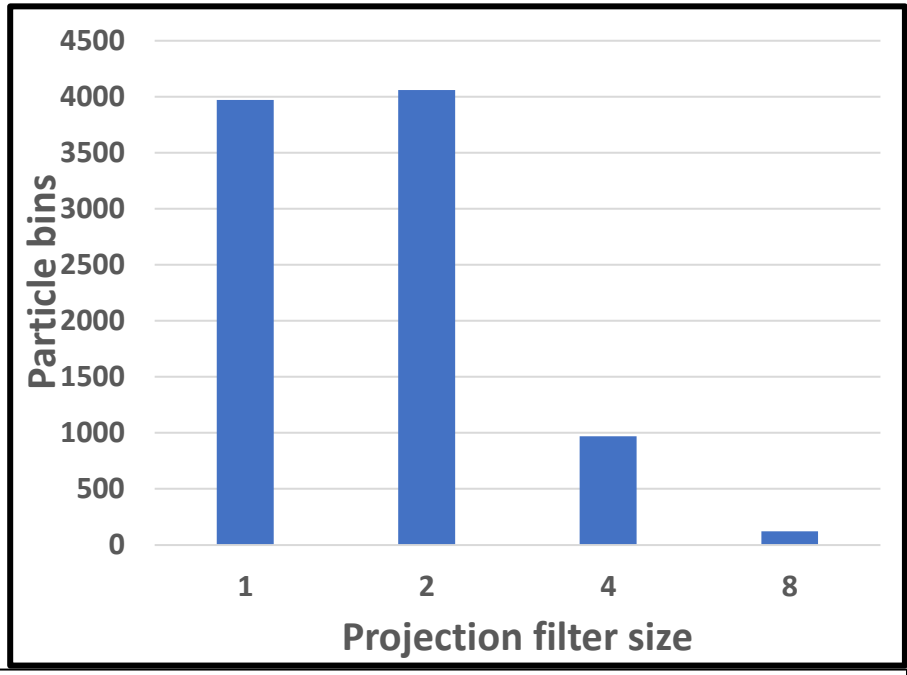


Image courtesy: https://www.researchgate.net/figure/Schematic-representation-of-the-interpolation-operator-in-Eq-14-The-gray-circle_fig1_320796615

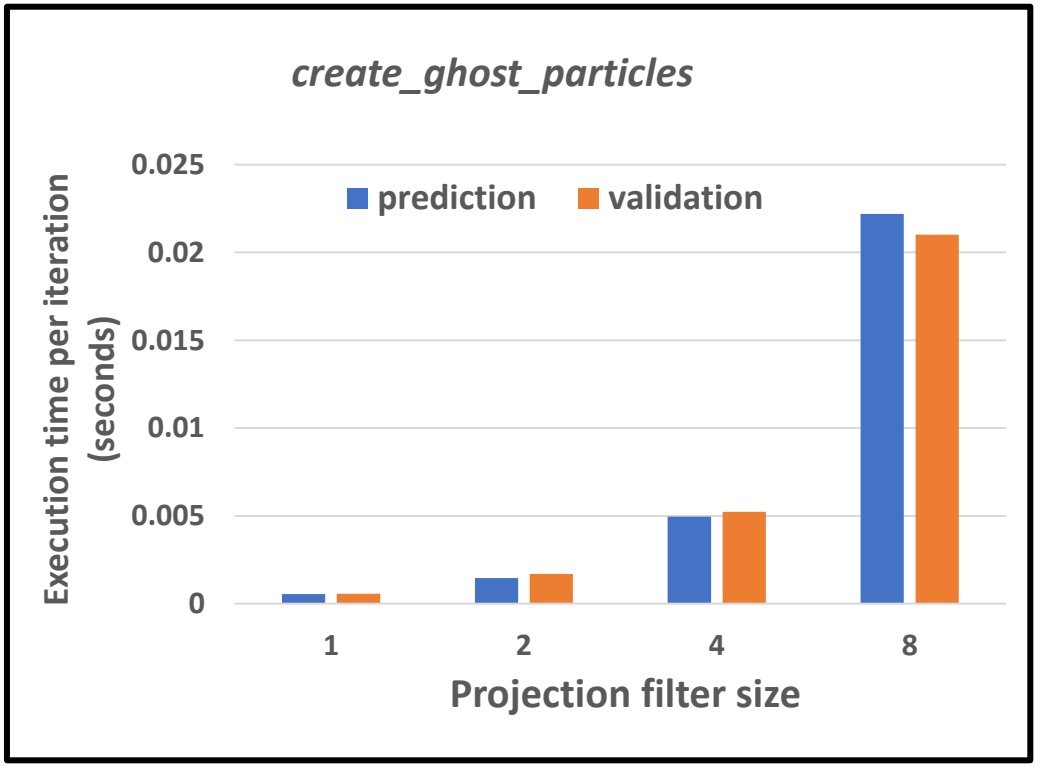
Results: Parameter Tuning

❑ Projection filter:

- Defines particle zone of influence on neighboring grid-points
- Has an impact on performance:
  - Determines number of ghost particles per processor (N_g) – ***particle workload***
  - Determines Number of particle bins – ***scaling threshold***
- Identifying performance cost is crucial to evaluate ***accuracy vs cost*** trade-off

❑ Observations:

- ***create_ghost_particles*** – kernel generating the ghost particles per processor
- Number of ghost particles increase with filter size



Conclusion & Going Forward

- Performance Modeling of MP-PIC applications on Large-scale systems is difficult
  - Dynamic workload fluctuation due to non-homogenous particle distribution
- Presented a performance prediction framework:
  - **DWG** for a problem simulation on any target system
  - Modelling and simulation platform for faster performance predictions
- Demonstrated our prediction framework on MP-PIC application (CMT-nek):
  - ***Scalability prediction*** – identified optimal processor count for particle workload distribution
  - ***Algorithm evaluation*** – Bin-based mapping provides better particle workload distribution
  - ***Performance tuning*** – quantified performance cost of key application parameter
- **Future work:**
  - Include other particle-mapping algorithms in DWG
    - E.g. Dynamical load-balancing*
  - Synthetic trace generation
    - Alleviate trace-collection bottleneck by generating large-scale trace from a low-resolution run
    - Saves trace-collection time and file-size

** Zhai, Keke, et al. "Dynamic load balancing for compressible multiphase turbulence." Proceedings of the 2018 International Conference on Supercomputing. 2018.*

Do you have any questions?