Accelerating the Material Point Method on Emerging Computing Architectures

Sagar Dolas

Technical University of Delft
Deltares

sagardolas.cosse@gmail.com

March 6, 2017

Today's Agenda

1. Motivation
2. Problem Formulation
3. High Performance Computing
4. Glimpse of Results
5. Future Work

Motivation

Installation of Monopile - Dutch National Waters

Installation of Monopile - Dutch National Waters

Why this simulation so important ?

Why this simulation so important ?

Why this simulation so important ?

Joint Industry Project for Simulation of Installation of Monopile

Why this simulation so important ?

Joint Industry Project for Simulation of Installation of Monopile

1. Prolong useful life of pile foundation.

Why this simulation so important ?

Joint Industry Project for Simulation of Installation of Monopile

1. Prolong useful life of pile foundation.
2. Decisive for optimization of pile design.

Why this simulation so important ?

Joint Industry Project for Simulation of Installation of Monopile

1. Prolong useful life of pile foundation.
2. Decisive for optimization of pile design.
3. Improved installation techniques.

Why this simulation so important ?

Joint Industry Project for Simulation of Installation of Monopile

1. Prolong useful life of pile foundation.
2. Decisive for optimization of pile design.
3. Improved installation techniques.

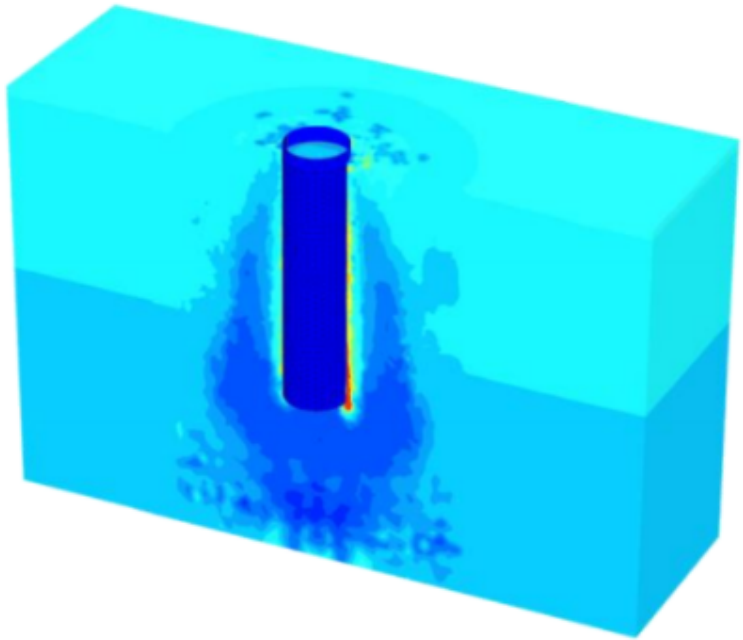


Figure: Simulation of monopile penetration with the Material Point Method

Why this simulation so important ?

Joint Industry Project for Simulation of Installation of Monopile

1. Prolong useful life of pile foundation.
2. Decisive for optimization of pile design.
3. Improved installation techniques.

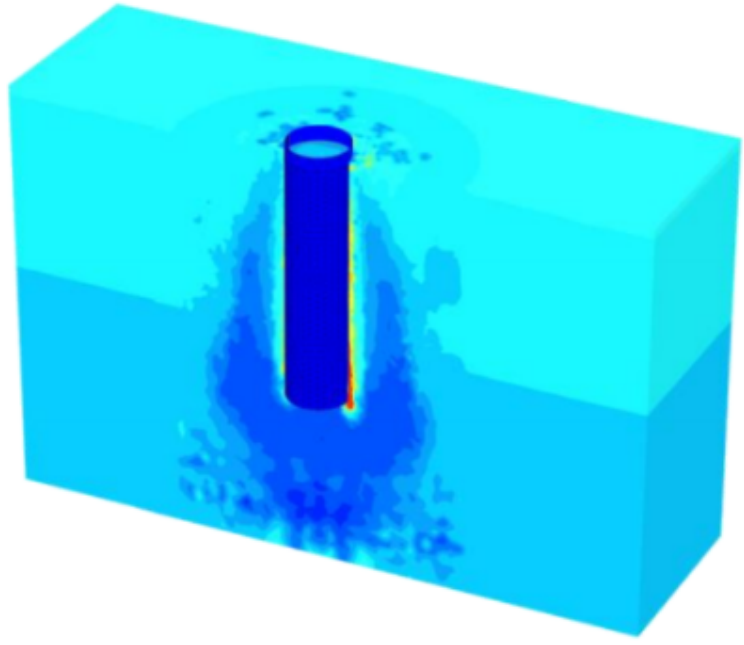


Figure: Simulation of monopile penetration with the Material Point Method

Software Tool

Anura3D is software developed at Deltares for such simulations

Problem Formulation

The Material Point Method

The Material Point Method

1. Numerical technique to simulate behavior of solids, liquids, gases or any other continuum material.

The Material Point Method

1. Numerical technique to simulate behavior of solids, liquids, gases or any other continuum material.
2. Multiphase interaction handled properly.

The Material Point Method

1. Numerical technique to simulate behavior of solids, liquids, gases or any other continuum material.
2. Multiphase interaction handled properly.
3. Key feature of MPM is use of material points, which are allowed to move freely through the background mesh.

The Material Point Method

1. Numerical technique to simulate behavior of solids, liquids, gases or any other continuum material.
2. Multiphase interaction handled properly.
3. Key feature of MPM is use of material points, which are allowed to move freely through the background mesh.
4. Avoids extreme mesh distortion as in FEM.

The Material Point Method

1. Numerical technique to simulate behavior of solids, liquids, gases or any other continuum material.
2. Multiphase interaction handled properly.
3. Key feature of MPM is use of material points, which are allowed to move freely through the background mesh.
4. Avoids extreme mesh distortion as in FEM.

MPM Algorithm

The Material Point Method

1. Numerical technique to simulate behavior of solids, liquids, gases or any other continuum material.
2. Multiphase interaction handled properly.
3. Key feature of MPM is use of material points, which are allowed to move freely through the background mesh.
4. Avoids extreme mesh distortion as in FEM.

MPM Algorithm

1. **Physical quantities are mapped from material points to underlying grid.**

The Material Point Method

1. Numerical technique to simulate behavior of solids, liquids, gases or any other continuum material.
2. Multiphase interaction handled properly.
3. Key feature of MPM is use of material points, which are allowed to move freely through the background mesh.
4. Avoids extreme mesh distortion as in FEM.

MPM Algorithm

1. **Physical quantities are mapped from material points to underlying grid**.
2. Equation of motion is solved over the background mesh to find the current acceleration.

The Material Point Method

1. Numerical technique to simulate behavior of solids, liquids, gases or any other continuum material.
2. Multiphase interaction handled properly.
3. Key feature of MPM is use of material points, which are allowed to move freely through the background mesh.
4. Avoids extreme mesh distortion as in FEM.

MPM Algorithm

1. **Physical quantities are mapped from material points to underlying grid.**
2. Equation of motion is solved over the background mesh to find the current acceleration.
3. **Variables of the material points are update via remapping from background grid.**

Particle to Grid Interaction

Particle to Grid Interaction

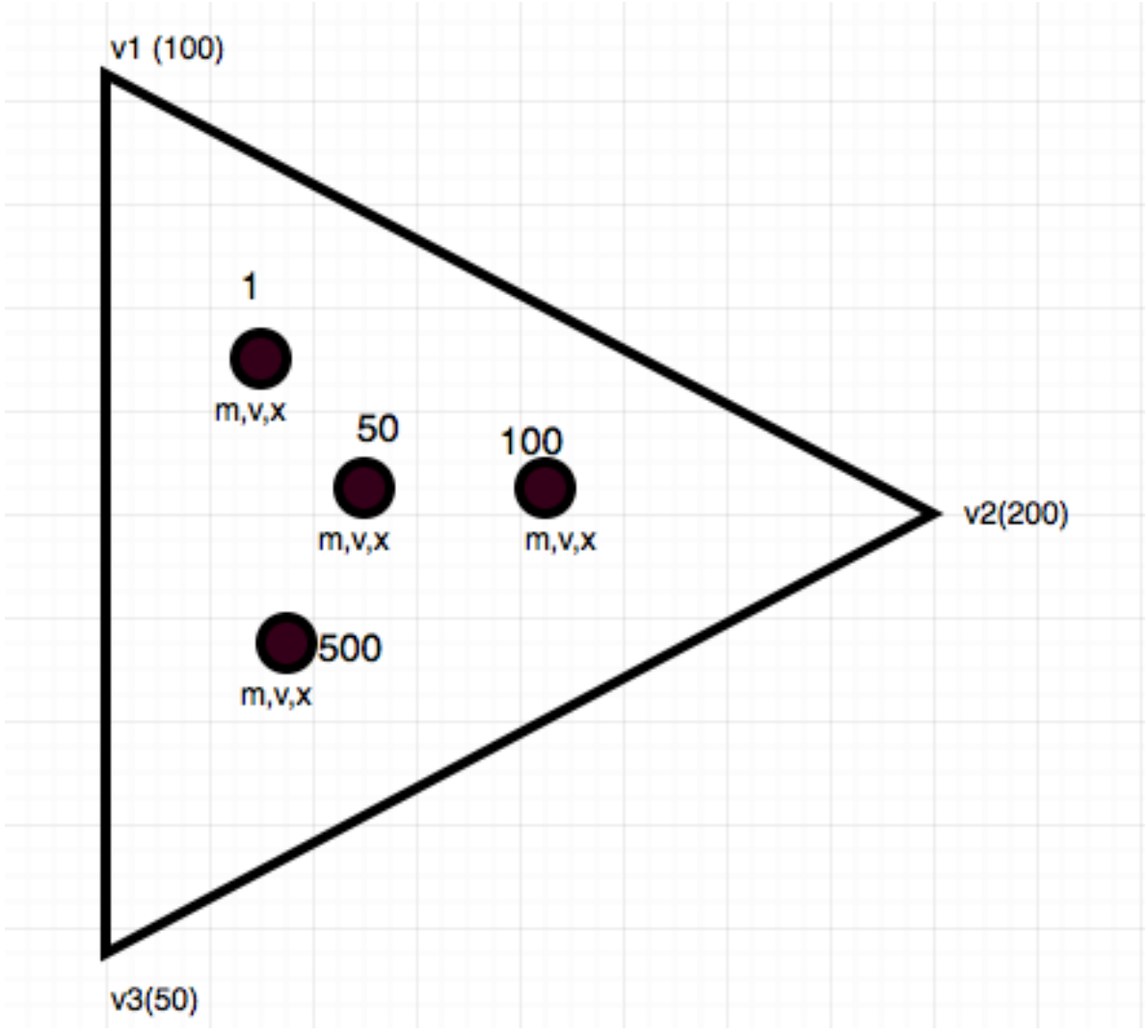


Figure: Particles in Grid

Particle to Grid Interaction

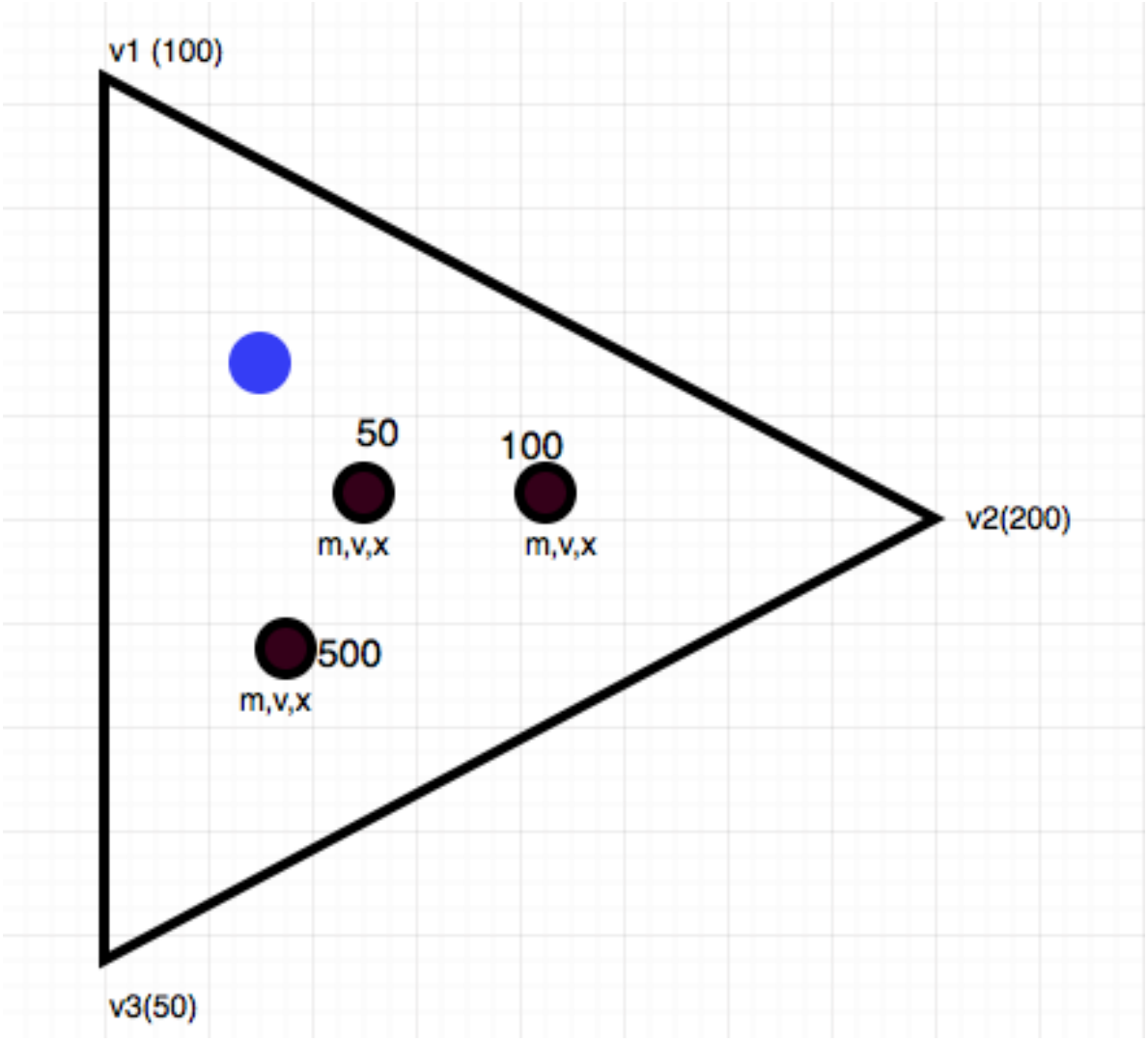
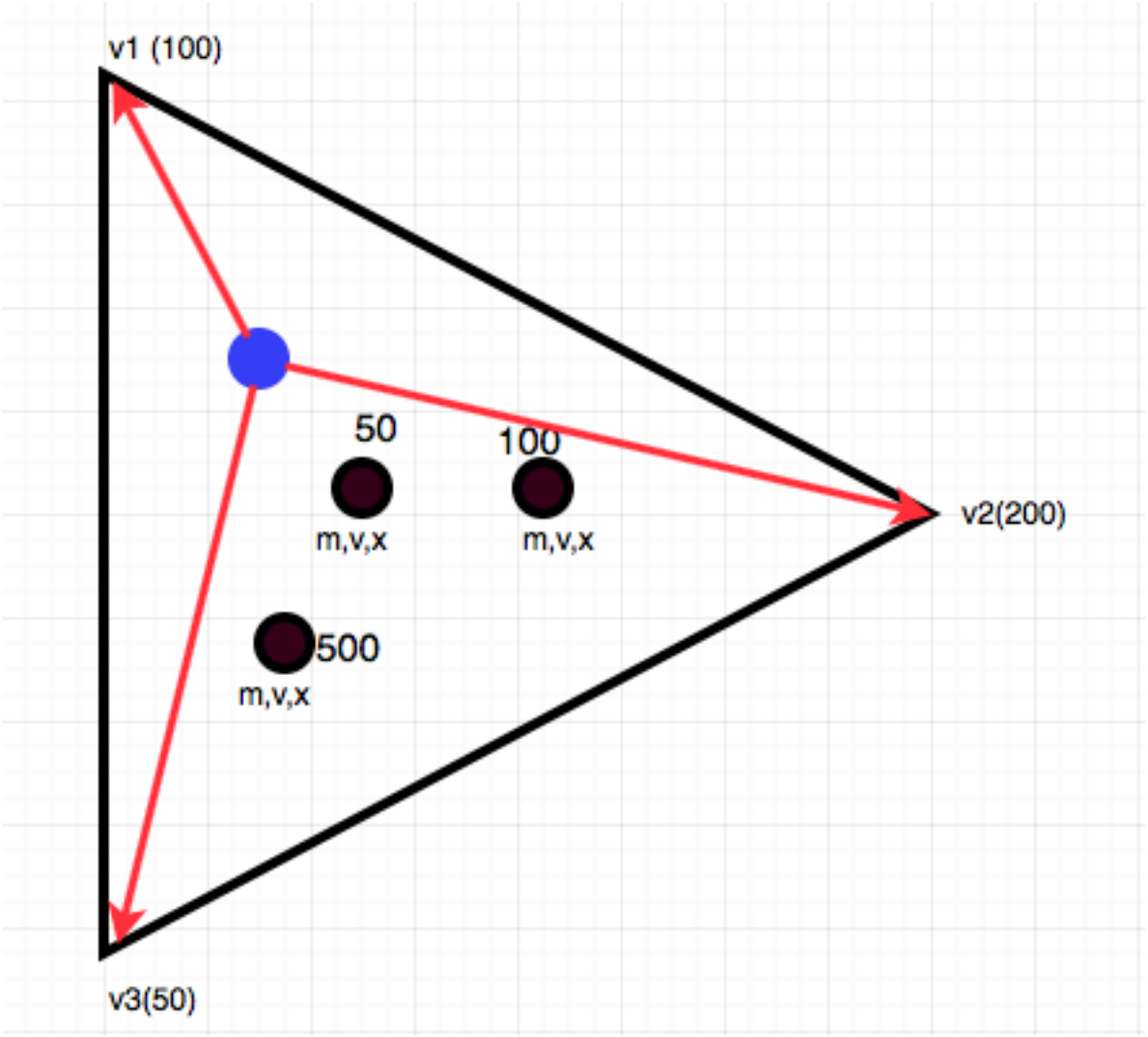


Figure: Blue Particle Starts to Interpolate

Interpolation using Shape functions

Interpolation using Shape functions



Particle to Grid Interaction in Memory

Particle to Grid Interaction in Memory

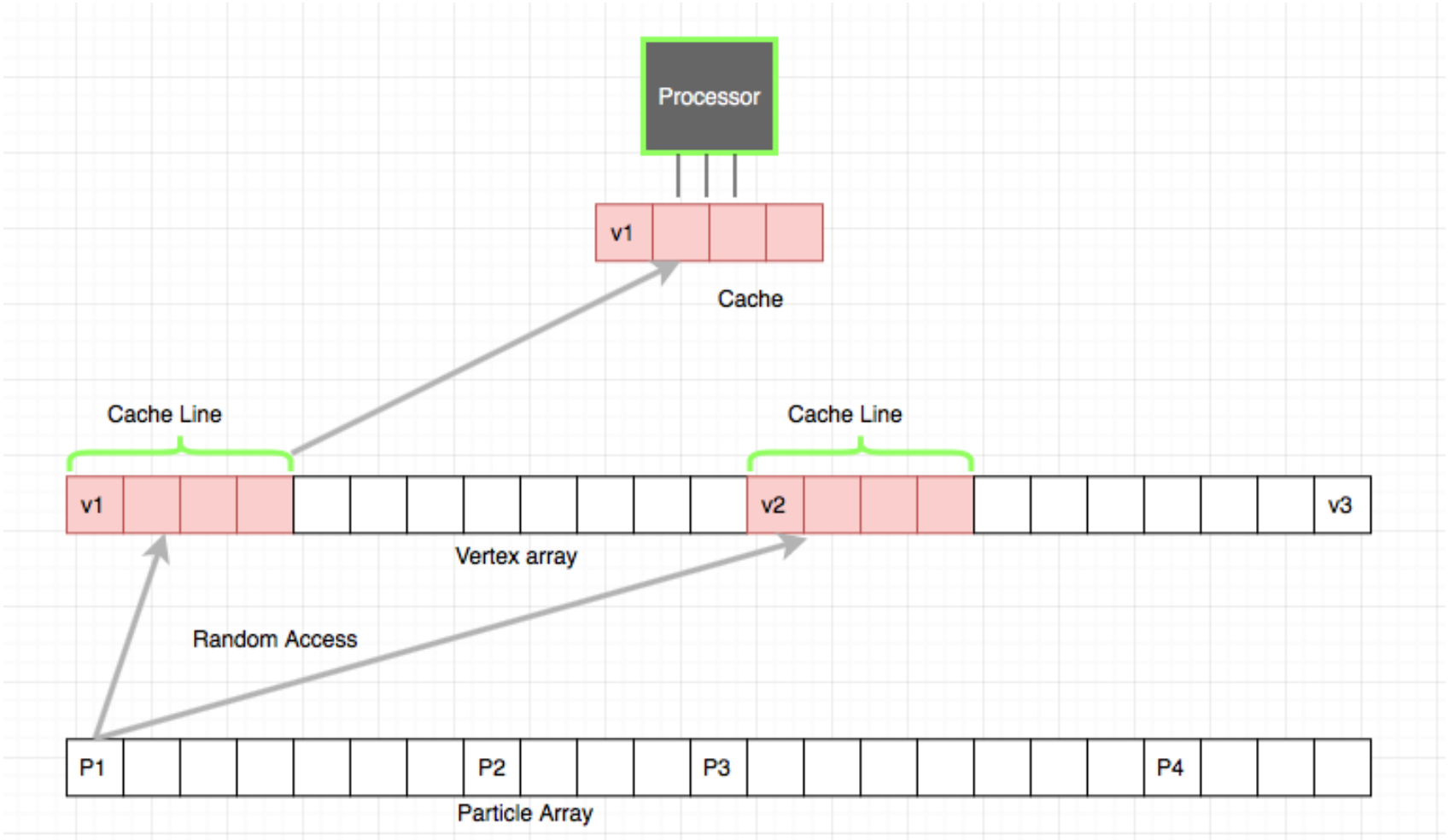
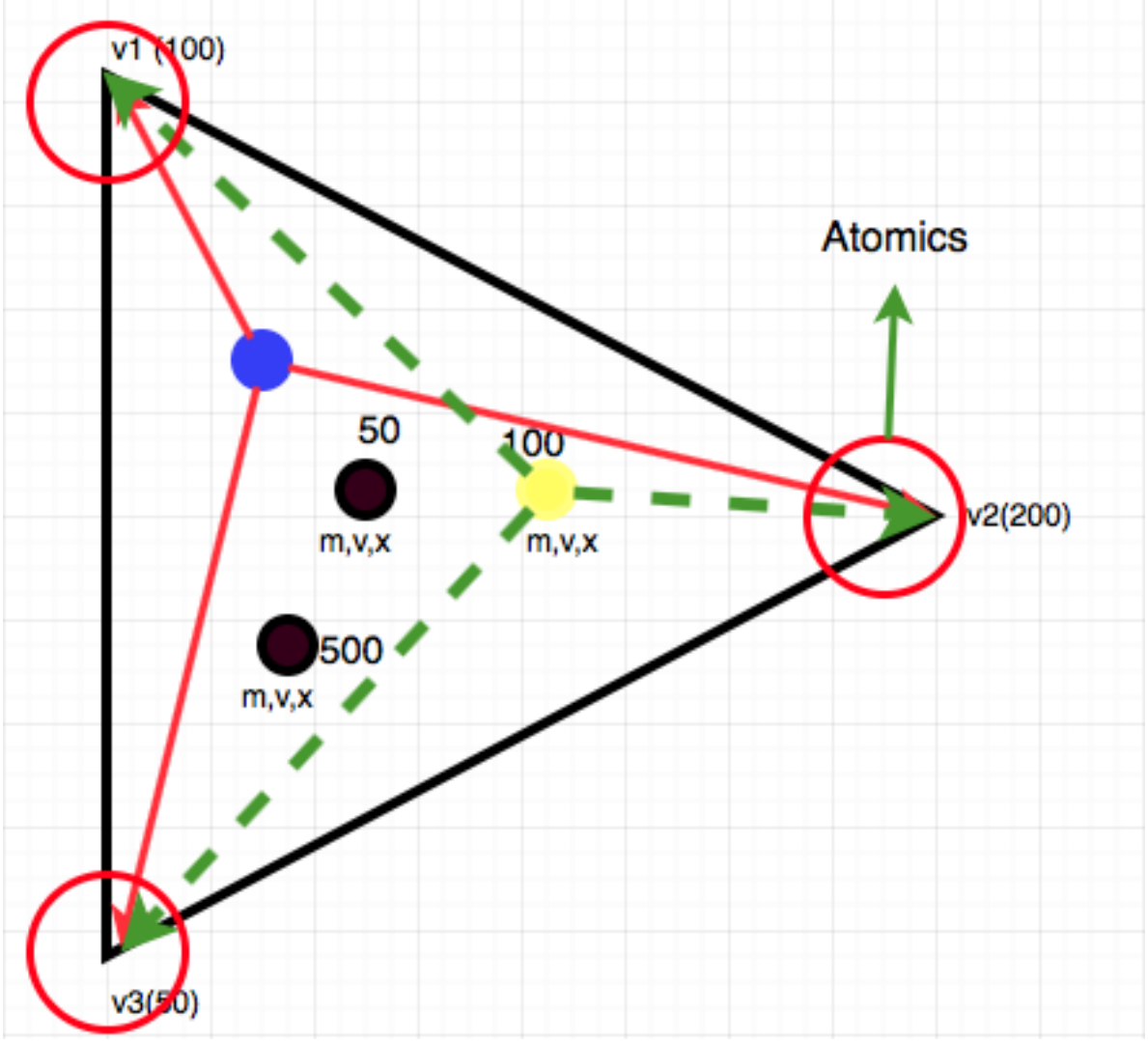


Figure: Particles updating the nodes

Particle Parallel

Particle Parallel



Element wise updating

Element wise updating

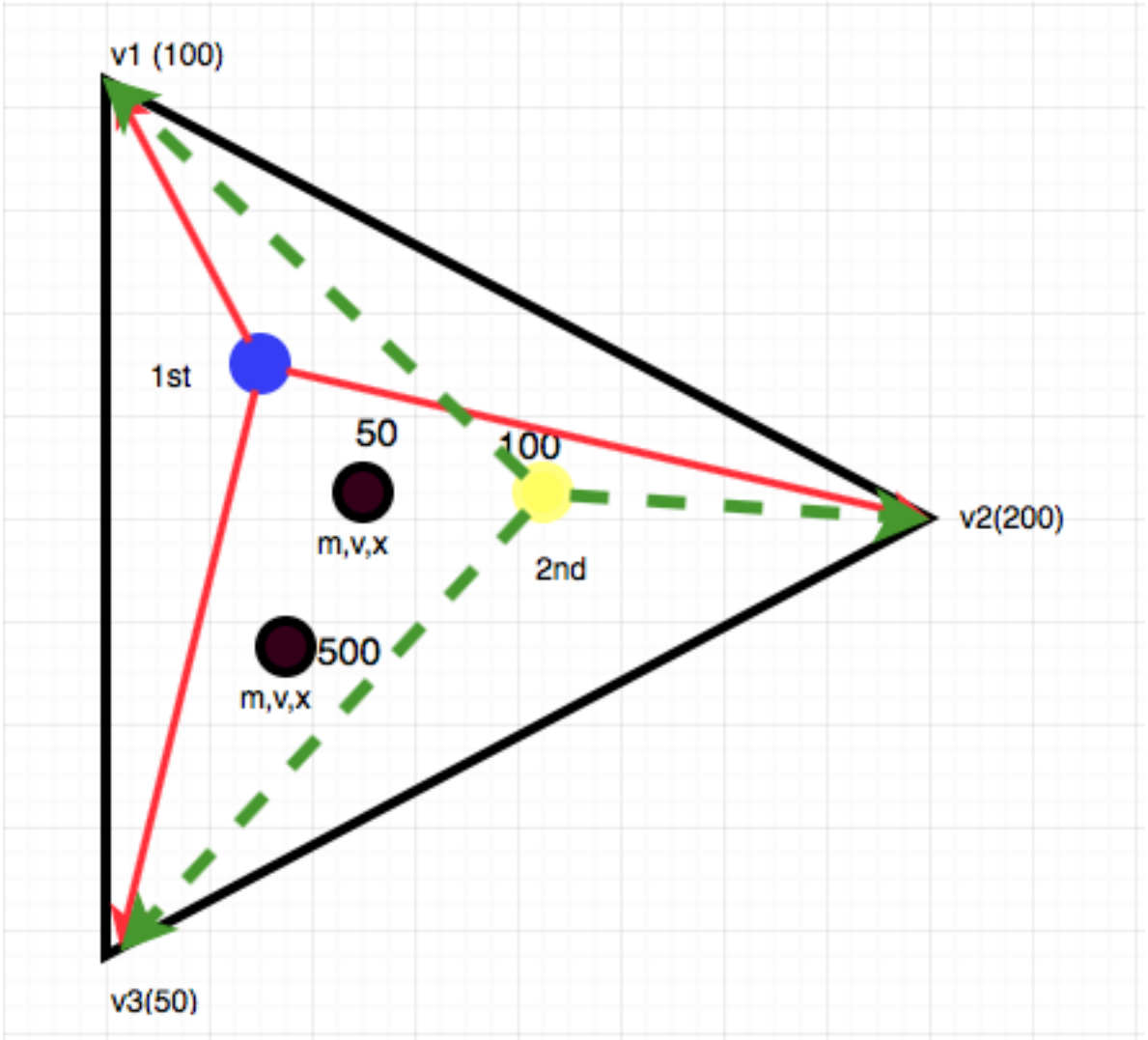


Figure: Element wise updating

Challenges in parallelization over elements

Challenges in parallelization over elements

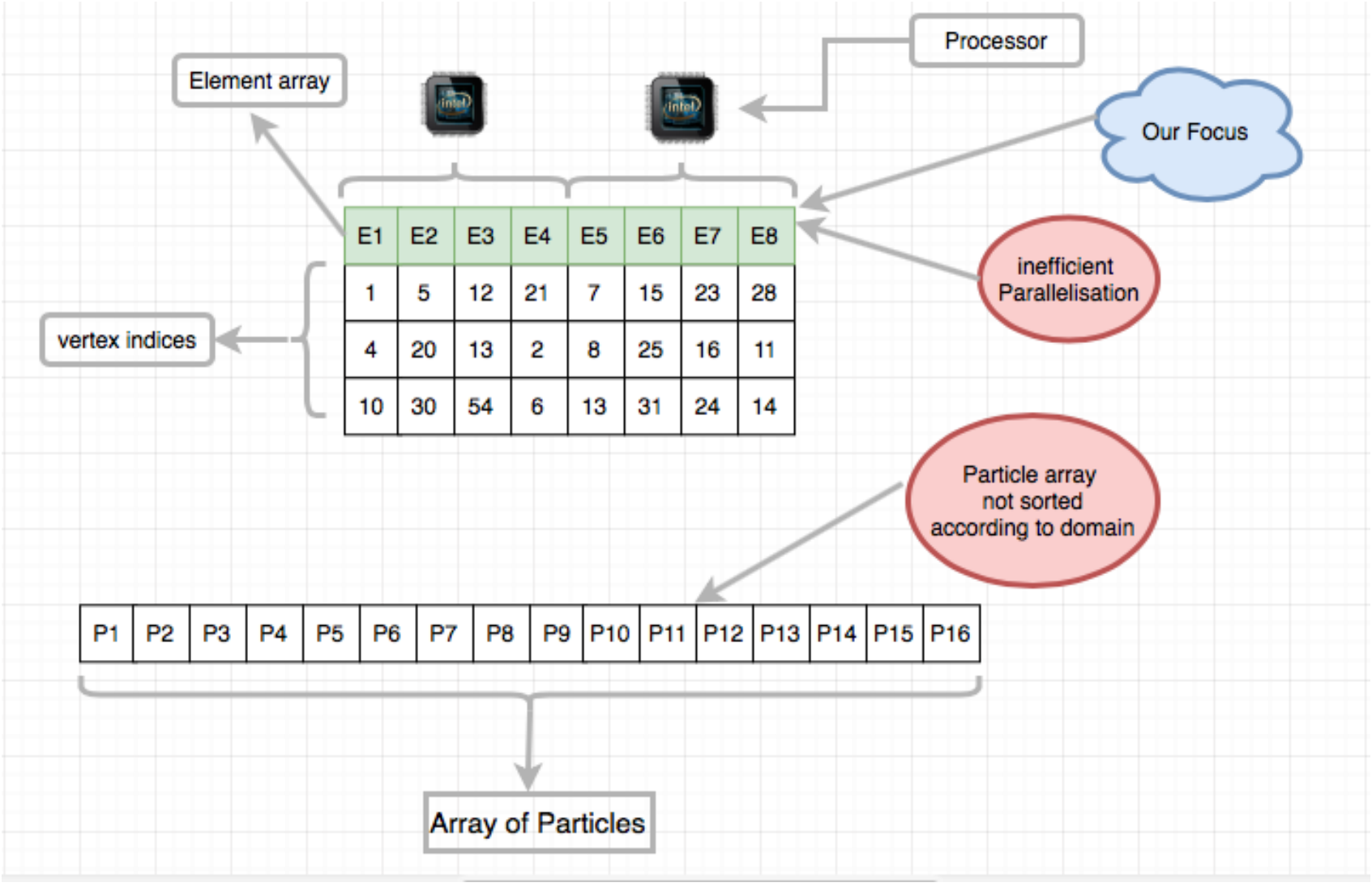


Figure: Challenge in Element wise parallelization

How can we Solve this ??

One Solution

One Solution

```
$$STARTELEMCON
204 207 186 203 206 197 195 202 205 193
199 181 203 186 190 191 200 192 182 193
185 171 189 186 178 180 187 184 179 188
117 99 96 95 108 98 107 106 97 94
99 96 95 78 98 94 97 88 87 85
113 91 95 96 102 93 104 103 92 94
91 95 96 73 93 94 92 82 83 84
45 42 63 59 44 53 54 52 49 62
41 59 36 45 50 47 39 43 52 40
36 41 45 11 39 43 40 27 28 30
77 81 95 73 80 89 86 74 76 83
77 81 73 55 80 76 74 65 67 64
113 135 131 109 125 134 122 111 121 119
60 36 55 59 48 46 57 58 47 56
55 60 59 81 57 58 56 67 70 71
117 114 135 113 115 124 126 116 112 125
117 114 113 96 115 112 116 107 105 103
149 153 167 145 151 161 158 146 148 155
149 153 145 127 151 148 146 137 139 136
189 185 186 203 187 184 188 196 194 193
189 186 207 203 188 197 198 196 193 205
185 203 181 186 194 191 183 184 193 182
181 185 186 167 183 184 182 174 176 175
22 11 42 45 16 29 33 35 30 44
113 95 117 96 104 106 116 103 94 107
132 127 150 153 129 138 141 143 139 152
91 113 109 96 102 111 100 92 103 101
171 185 167 186 178 176 169 179 184 175
60 78 81 55 69 79 70 57 66 67
109 127 114 135 118 120 110 121 130 124
168 171 150 163 170 160 159 164 166 156
168 171 163 186 170 166 164 177 179 173
31 11 22 45 19 16 25 37 30 35
163 145 167 150 154 155 165 156 147 157
73 78 55 81 75 66 64 76 79 67
```

We want this

```
$$STARTELEMCON
1 2 3 4 5 6 7 8 9 10
1 11 2 4 12 13 5 8 14 9
3 15 4 2 16 17 10 6 18 9
19 2 11 4 20 13 21 22 9 14
23 4 15 2 24 17 25 26 9 18
27 28 15 2 29 30 31 32 33 18
15 23 2 28 25 26 18 30 34 33
19 23 2 4 35 26 20 22 24 9
23 36 19 2 37 38 35 26 39 20
36 23 28 2 37 34 40 39 26 33
41 36 27 2 42 43 44 45 39 32
28 36 2 27 40 39 33 29 43 32
41 36 46 27 42 47 48 44 43 49
27 50 28 46 51 52 29 49 53 54
28 36 27 46 40 43 29 54 47 49
50 46 55 28 53 56 57 52 54 58
36 28 55 46 40 58 59 47 54 56
60 55 28 50 61 58 62 63 57 52
50 46 64 55 53 65 60 57 56 67
60 55 50 64 61 57 63 68 67 66
69 64 46 55 70 65 71 72 67 56
60 73 55 64 74 75 61 68 76 67
73 60 77 64 74 78 79 76 68 80
69 73 64 55 81 76 70 72 75 67
69 73 82 64 81 83 84 70 76 85
86 64 82 73 87 85 88 89 76 83
77 73 64 86 79 76 80 90 89 87
86 82 91 73 88 92 93 89 83 94
91 73 77 86 94 79 95 93 89 90
96 82 73 91 97 83 98 99 92 94
100 86 82 91 101 88 102 103 93 92
104 91 86 100 105 93 106 107 103 101
96 82 91 100 97 92 99 108 102 103
91 104 109 100 105 110 111 103 107 112
91 109 96 100 111 113 99 103 112 108
```

P0

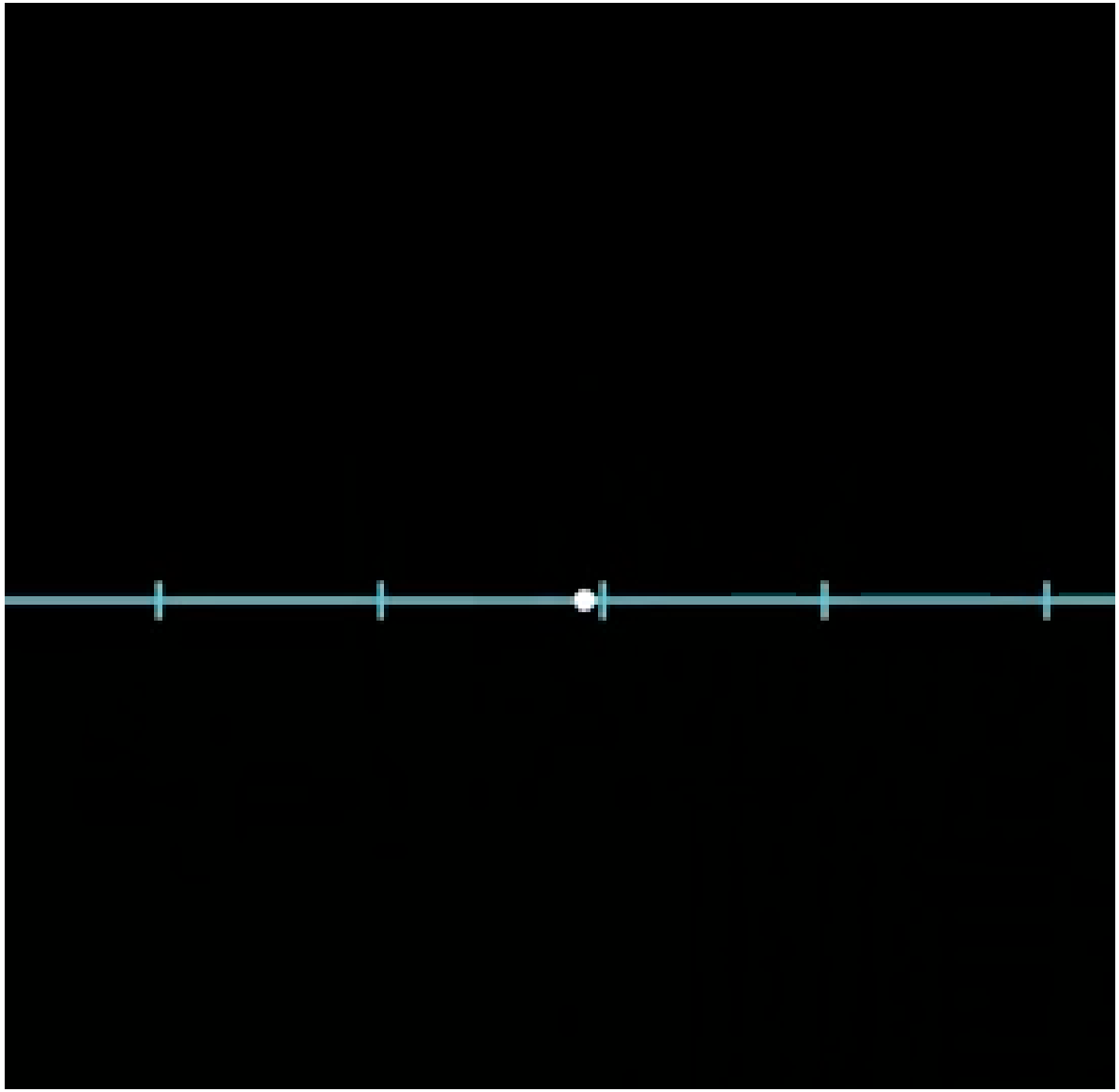
P1

The Space Filling Curves

Moving Towards High Performance Computing

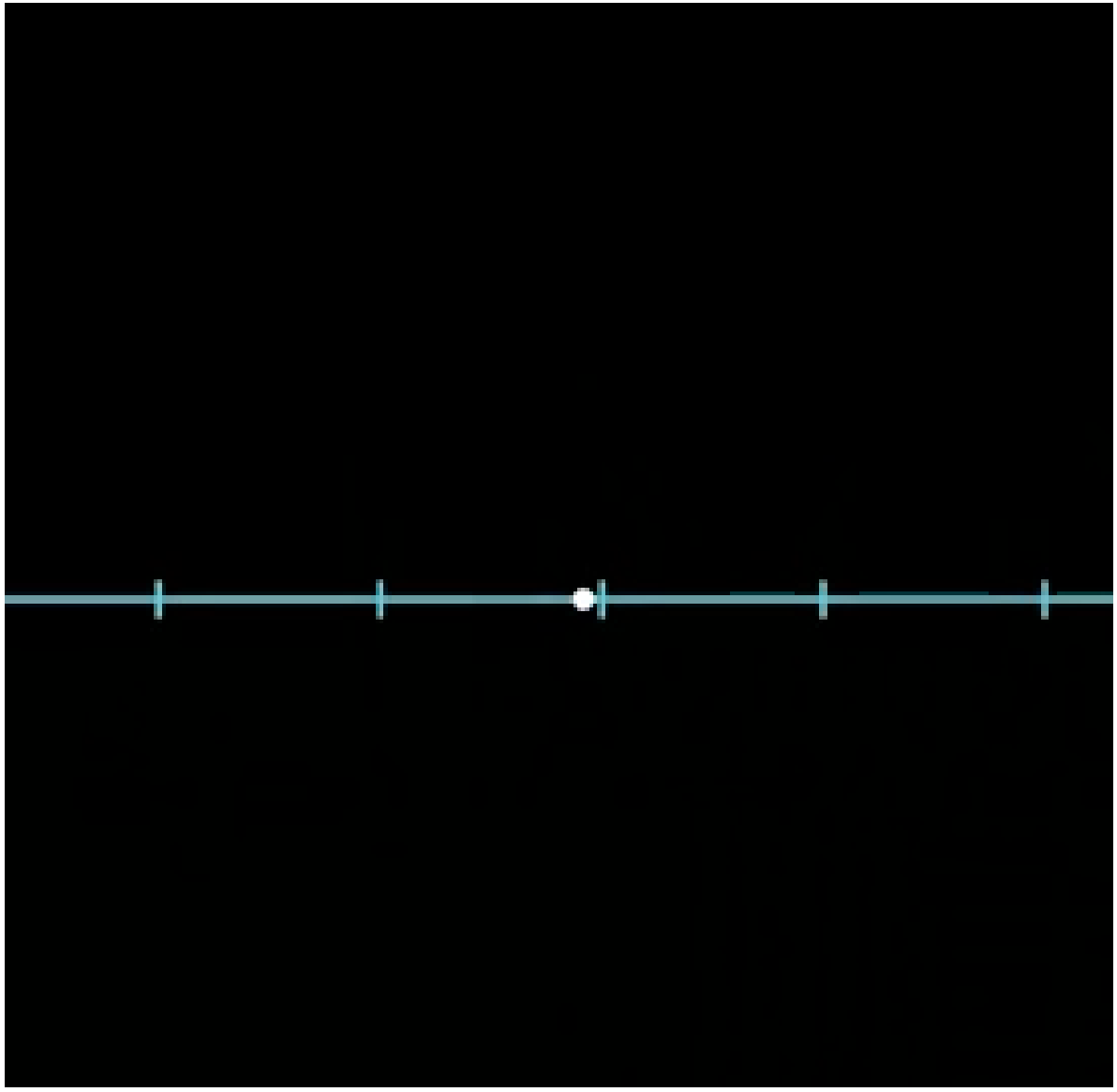
Thought Process

Thought Process



Keeping spatial close entities close in memory

Keeping spatial close entities close in memory



What is space filling curve

What is space filling curve

1. SFC is mapping from n-dimensional space to 1 dimensional space.

What is space filling curve

1. SFC is mapping from n-dimensional space to 1 dimensional space.
2. Mesh renumbering strategy to preserve spatial locality.

What is space filling curve

1. SFC is mapping from n-dimensional space to 1 dimensional space.
2. Mesh renumbering strategy to preserve spatial locality.
3. *de facto* standard for renumbering unstructured finite element mesh.

What is space filling curve

1. SFC is mapping from n-dimensional space to 1 dimensional space.
2. Mesh renumbering strategy to preserve spatial locality.
3. *de facto* standard for renumbering unstructured finite element mesh.

Space Filling Curve Toolbox in *C++*

We have developed standalone preprocessor application in C++ which takes arbitrary mesh file and produces another mesh file with space filling indexing of elements and vertexes.

Space filling curve algorithm for arbitrary unstructured 3D finite element mesh

Space filling curve algorithm for arbitrary unstructured 3D finite element mesh

1. Input the mesh data.

Space filling curve algorithm for arbitrary unstructured 3D finite element mesh

1. Input the mesh data.
2. Store the element and vertex data into appropriate data structure

Space filling curve algorithm for arbitrary unstructured 3D finite element mesh

1. Input the mesh data.
2. Store the element and vertex data into appropriate data structure
3. Find out the centroids of each element.

Space filling curve algorithm for arbitrary unstructured 3D finite element mesh

1. Input the mesh data.
2. Store the element and vertex data into appropriate data structure
3. Find out the centroids of each element.
4. Compute the bounding box of the computational domain.

Space filling curve algorithm for arbitrary unstructured 3D finite element mesh

1. Input the mesh data.
2. Store the element and vertex data into appropriate data structure
3. Find out the centroids of each element.
4. Compute the bounding box of the computational domain.
5. **Compute SFC index of each element based on its centroid**

Space filling curve algorithm for arbitrary unstructured 3D finite element mesh

1. Input the mesh data.
2. Store the element and vertex data into appropriate data structure
3. Find out the centroids of each element.
4. Compute the bounding box of the computational domain.
5. **Compute SFC index of each element based on its centroid**
6. Sort the element according to their SFC index.

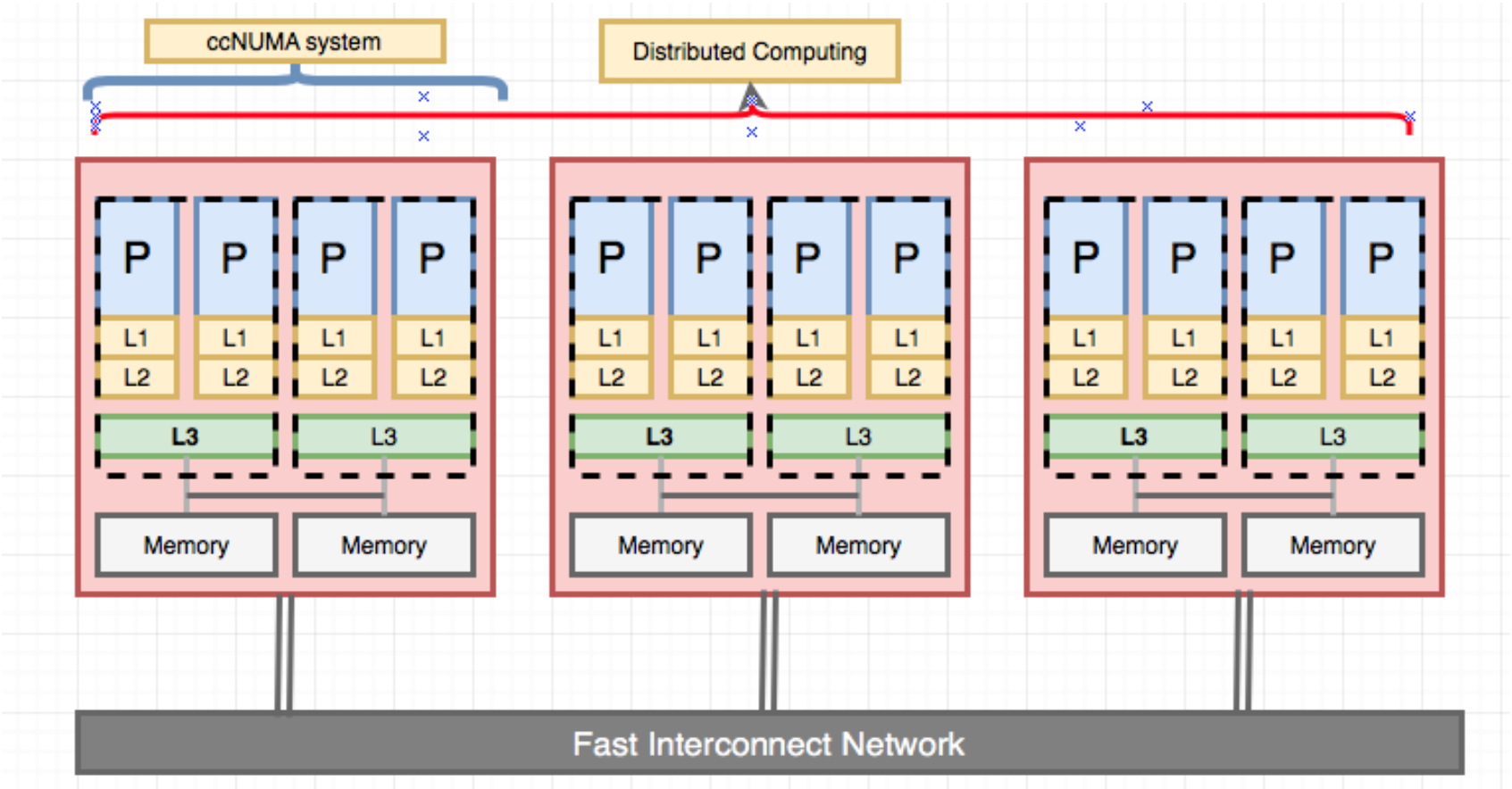
Space filling curve algorithm for arbitrary unstructured 3D finite element mesh

1. Input the mesh data.
2. Store the element and vertex data into appropriate data structure
3. Find out the centroids of each element.
4. Compute the bounding box of the computational domain.
5. **Compute SFC index of each element based on its centroid**
6. Sort the element according to their SFC index.
7. Place vertexes according to the element order.

Is space filling curve enough for achieving high performance ?

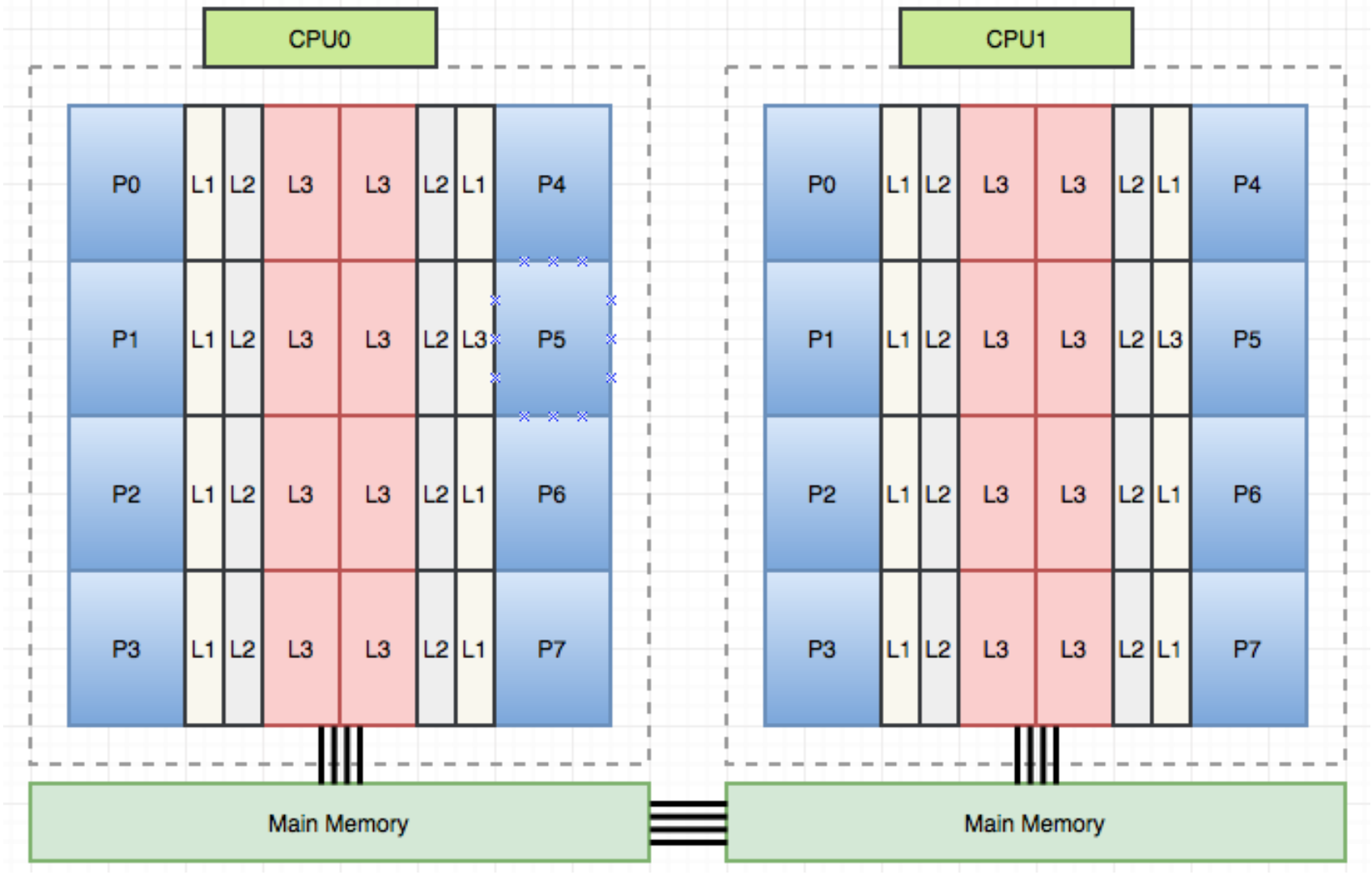
Hybrid supercomputer architecture

Hybrid supercomputer architecture



Close Look at *ccNUMA* system

Close Look at *ccNUMA* system



Key Issues in ccNUMA system

Key Issues in ccNUMA system

Key Issues in ccNUMA system

1. ccNUMA systems utilize complex memory hierarchies to maintain scalability.

Key Issues in ccNUMA system

1. ccNUMA systems utilize complex memory hierarchies to maintain scalability.
2. Modern HPC clusters are not symmetric multiprocessors (SMPs), they have asymmetry.

Key Issues in ccNUMA system

1. ccNUMA systems utilize complex memory hierarchies to maintain scalability.
2. Modern HPC clusters are not symmetric multiprocessors (SMPs), they have asymmetry.
3. Ignoring the architectural aspects can significantly harm performance.

Key Issues in ccNUMA system

1. ccNUMA systems utilize complex memory hierarchies to maintain scalability.
2. Modern HPC clusters are not symmetric multiprocessors (SMPs), they have asymmetry.
3. Ignoring the architectural aspects can significantly harm performance.
4. Processor/ thread affinity play major in performance of compute intensive kernels.

Key Issues in ccNUMA system

1. ccNUMA systems utilize complex memory hierarchies to maintain scalability.
2. Modern HPC clusters are not symmetric multiprocessors (SMPs), they have asymmetry.
3. Ignoring the architectural aspects can significantly harm performance.
4. Processor/ thread affinity play major in performance of compute intensive kernels.
5. Implicit memory allocation policies helps to keep data as local as possible.

Key Issues in ccNUMA system

1. ccNUMA systems utilize complex memory hierarchies to maintain scalability.
2. Modern HPC clusters are not symmetric multiprocessors (SMPs), they have asymmetry.
3. Ignoring the architectural aspects can significantly harm performance.
4. Processor/ thread affinity play major in performance of compute intensive kernels.
5. Implicit memory allocation policies helps to keep data as local as possible.
6. Congestion of memory controllers.

Our focus for present study

Our focus for present study

1. Thread / Processor affinity.

Our focus for present study

1. Thread / Processor affinity.
2. Data placement using first touch principle.

Our focus for present study

1. Thread / Processor affinity.
2. Data placement using first touch principle.
3. Intel's Hyper-threading.

Our focus for present study

1. Thread / Processor affinity.
2. Data placement using first touch principle.
3. Intel's Hyper-threading.
4. Floating point balance.

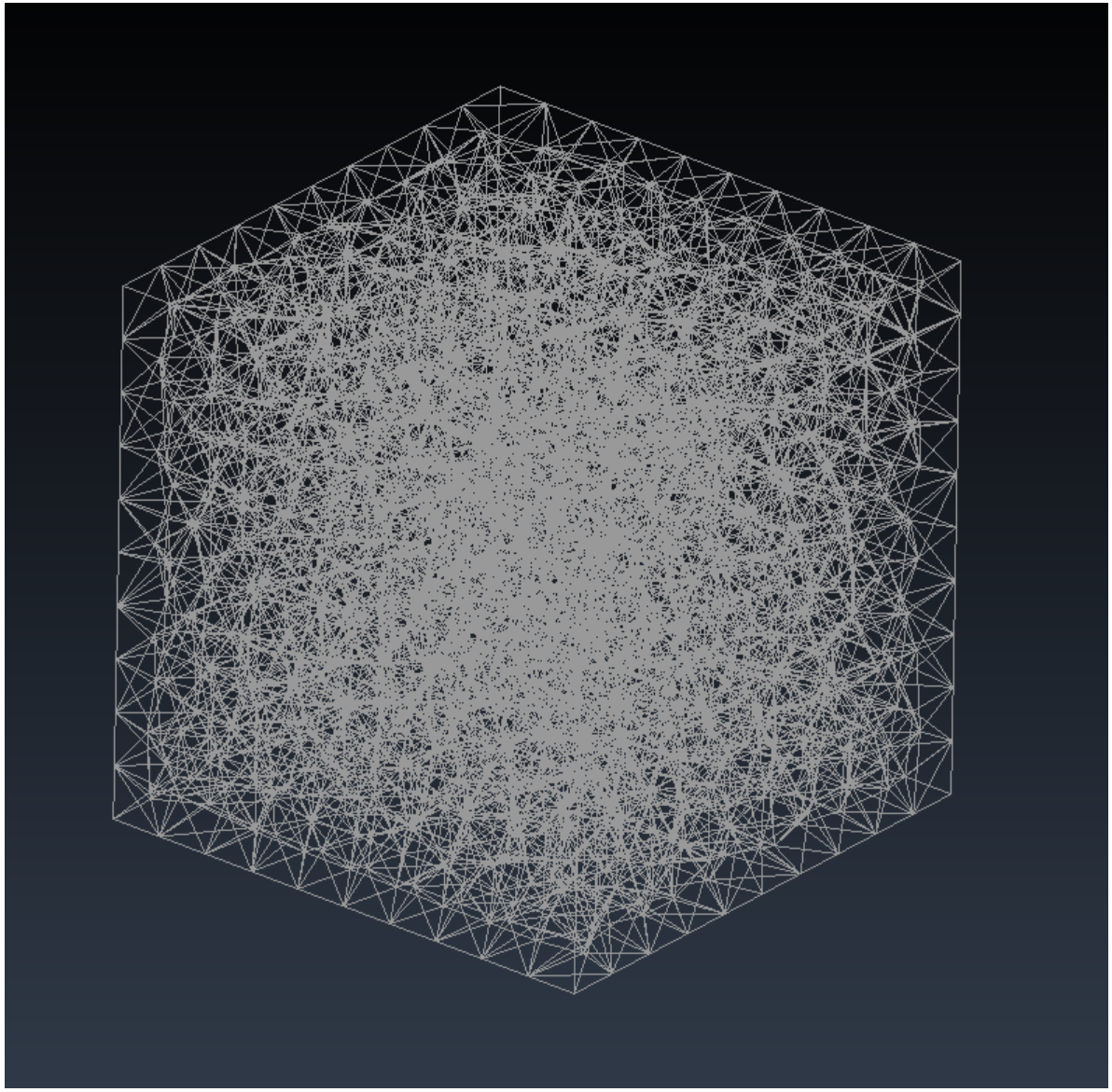
Computational Kernel

We will study the effect of above issues on simple computational kernels and extend this concepts to current 3D MPM code.

First look at results

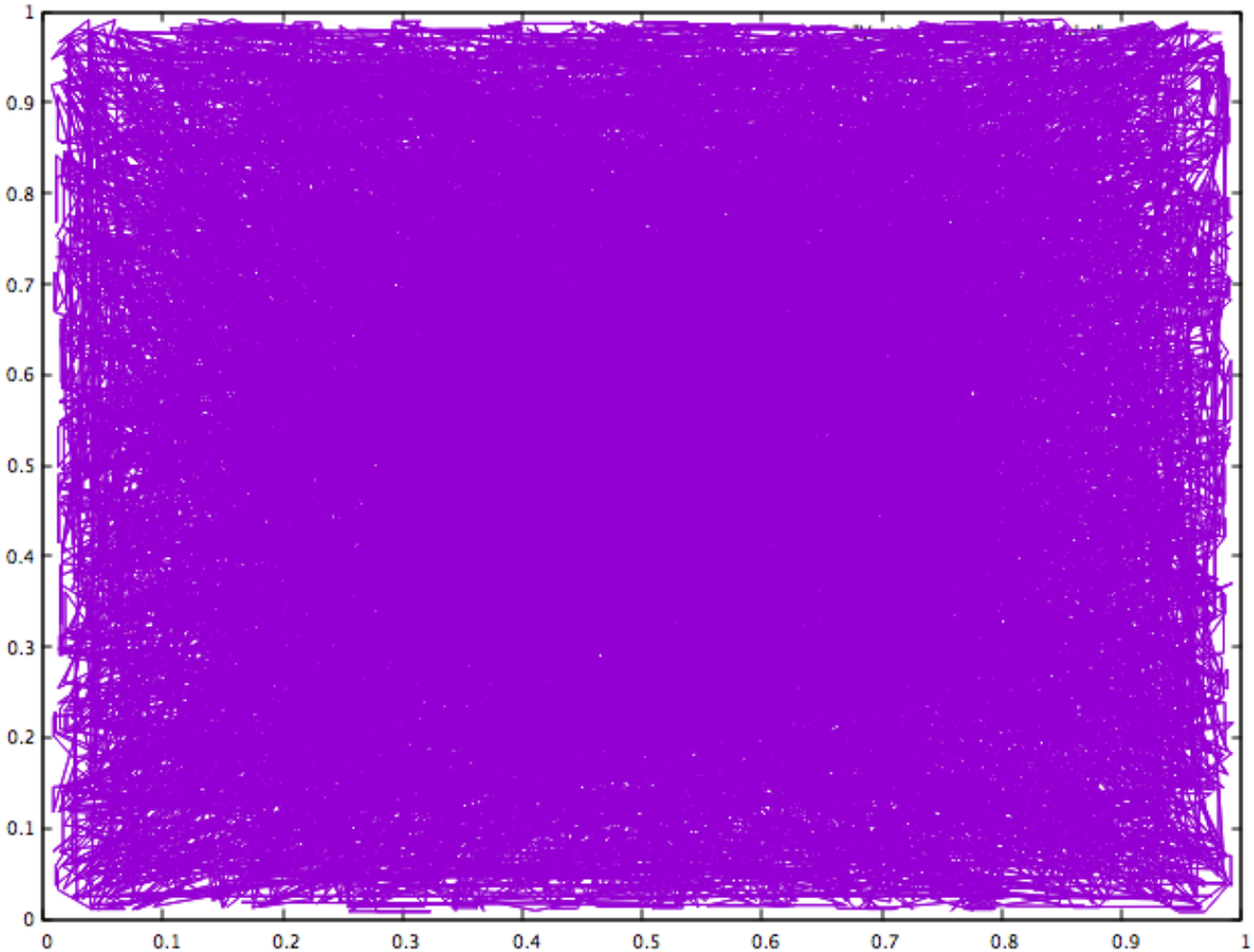
The Space filling curve generation for arbitrary mesh

The Space filling curve generation for arbitrary mesh



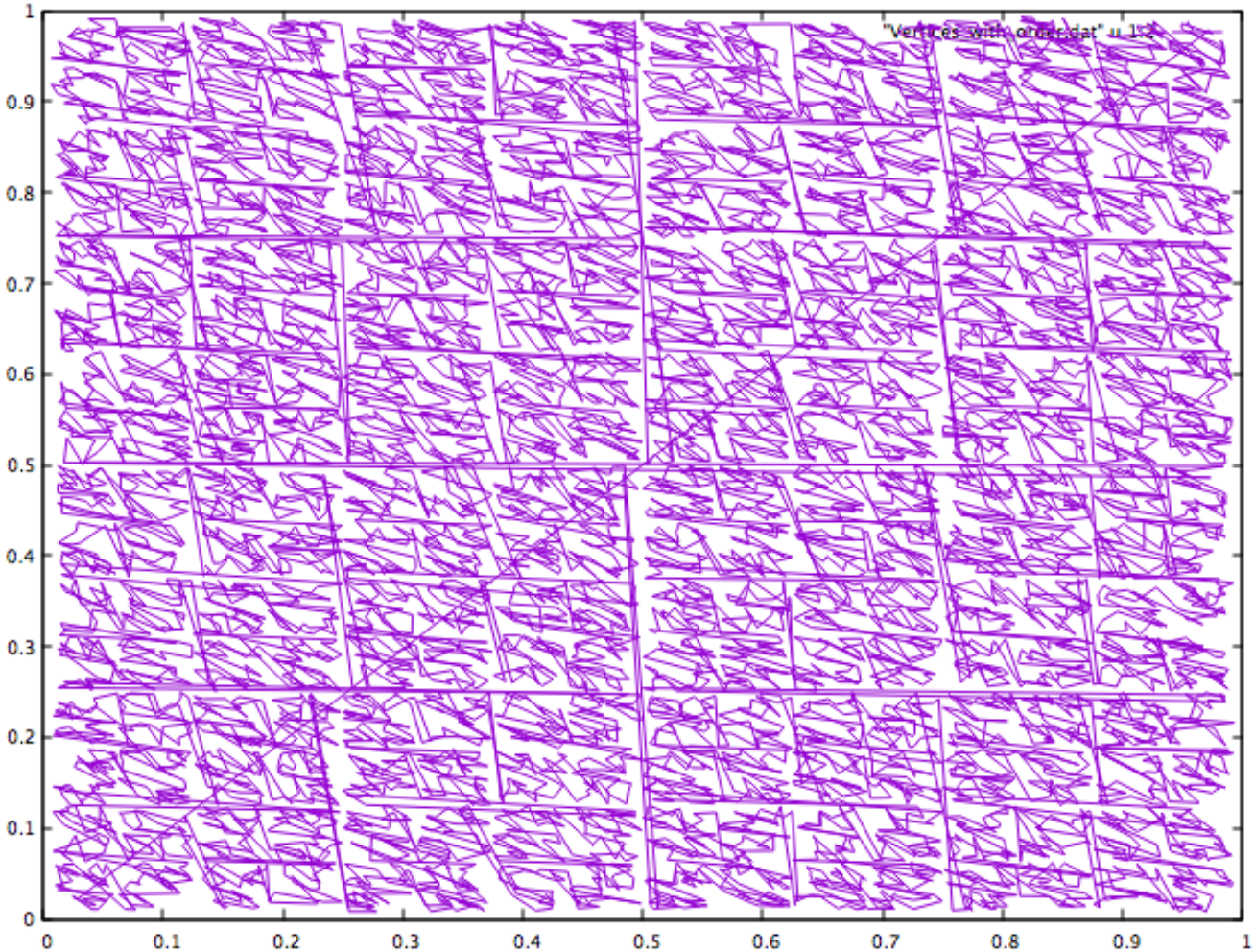
The elements in memory

The elements in memory



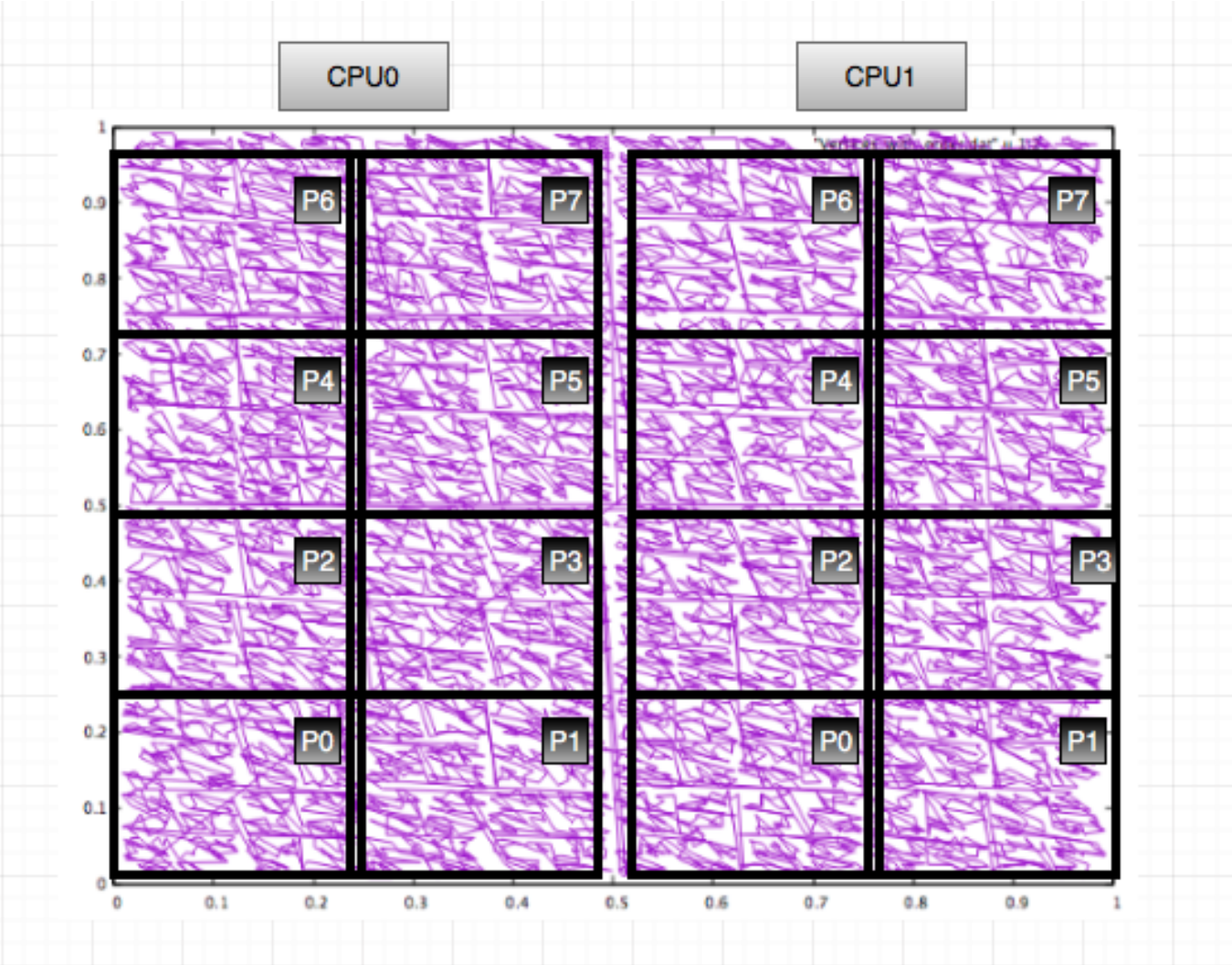
The elements in memory after space filling curve numbering

The elements in memory after space filling curve numbering



The efficient partition over multi-core architectures

The efficient partition over multi-core architectures



Optimization of *SAXPY* and *DAXPY* kernels

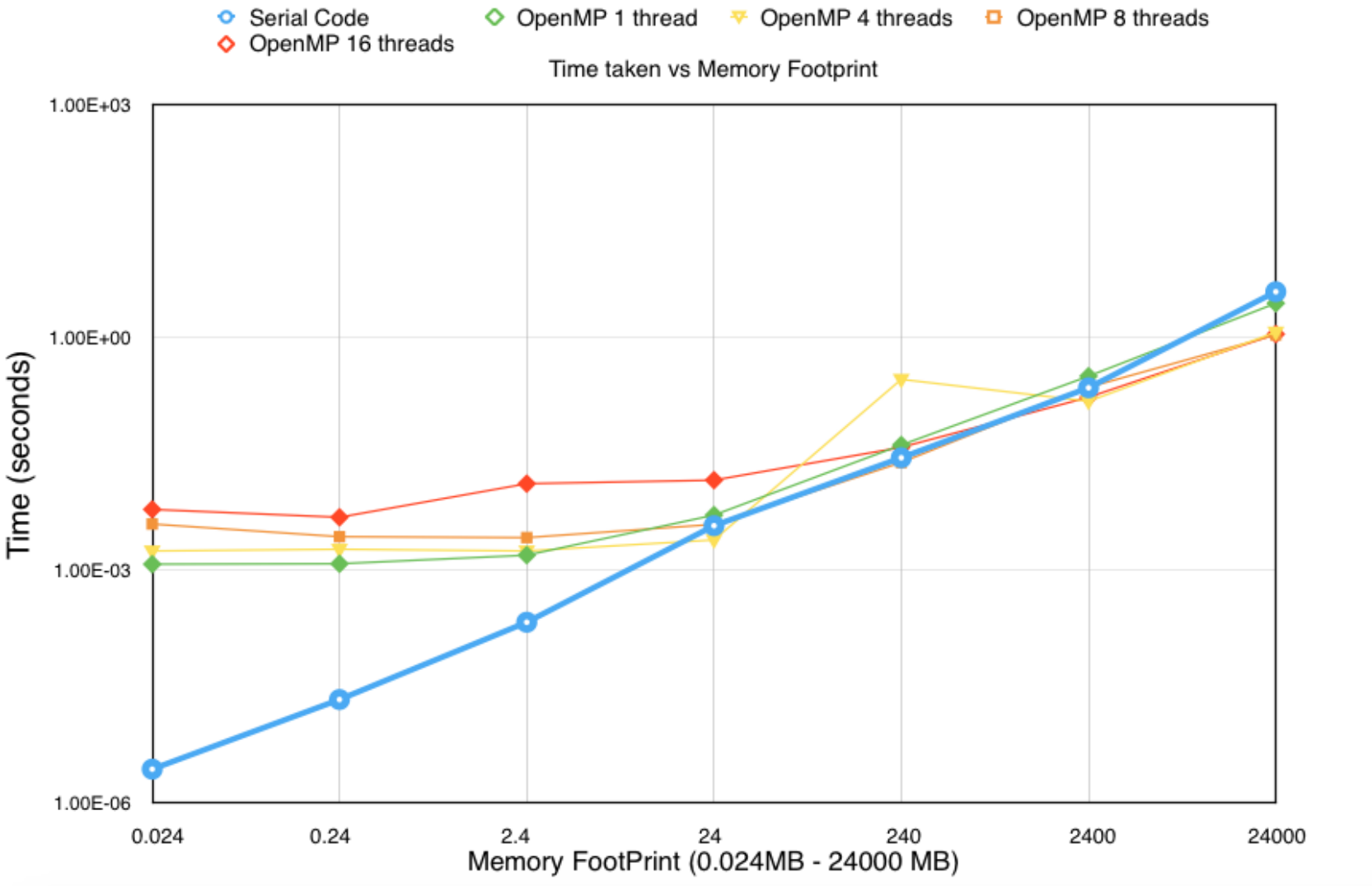
Optimization of *SAXPY* and *DAXPY* kernels

Machine specifications

1. **Architecture** : x86_64
2. **CPUs** : 0-15
3. **Number of Sockets** : 2
4. **Cores per sockets**: 8
5. **CPU Max GHz** : 3.10 GHz
6. **L1 Cache** : 32 KB
7. **L2 Cache** : 256 KB
8. **L3 Cache** : 40 MB
9. **NUMA Node(0) CPUs** : 0-7
10. **NUMA Node(1) CPUs** : 7-15
11. **Operating System** : Linux

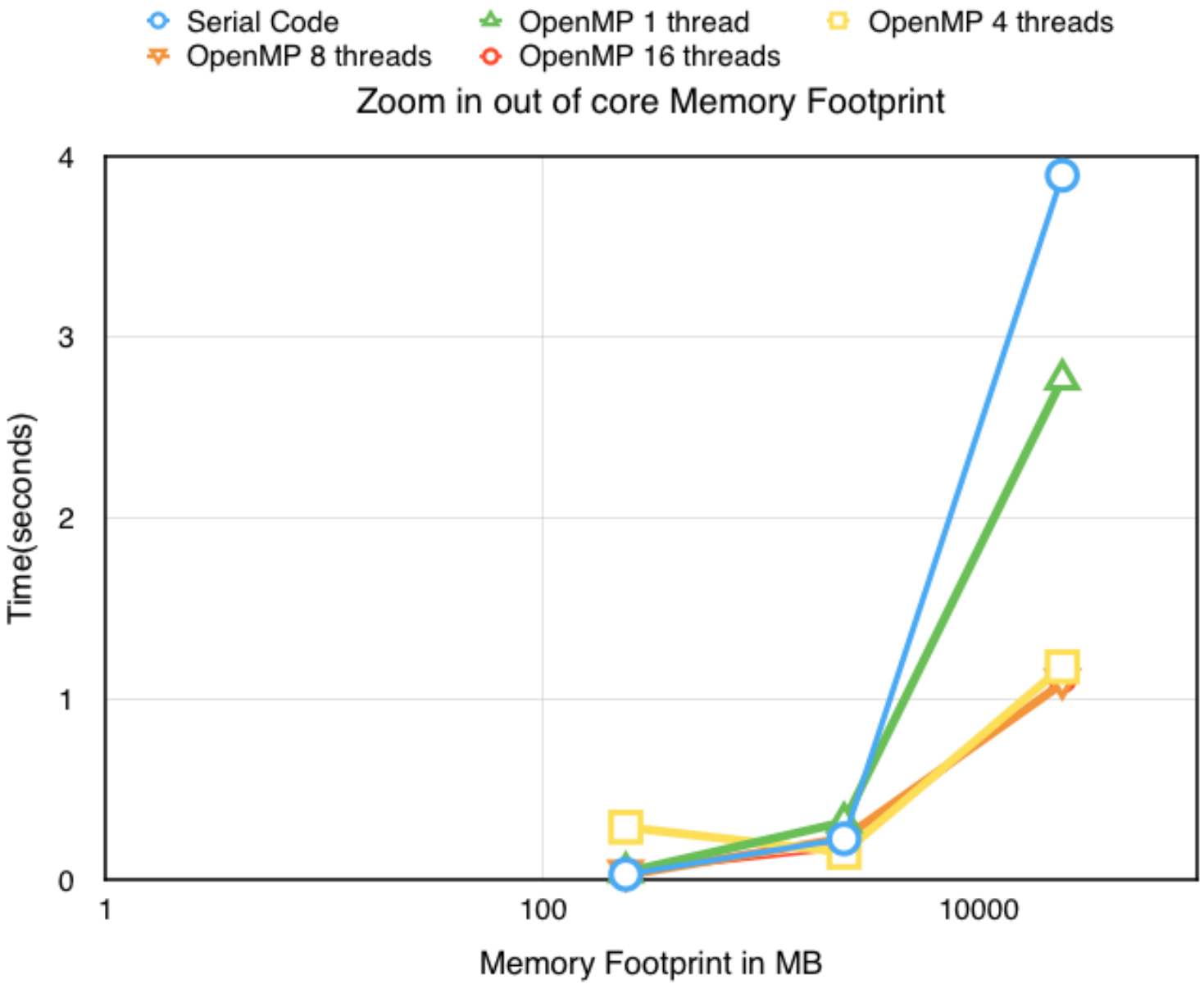
Time measurement without any optimizations

Time measurement without any optimizations



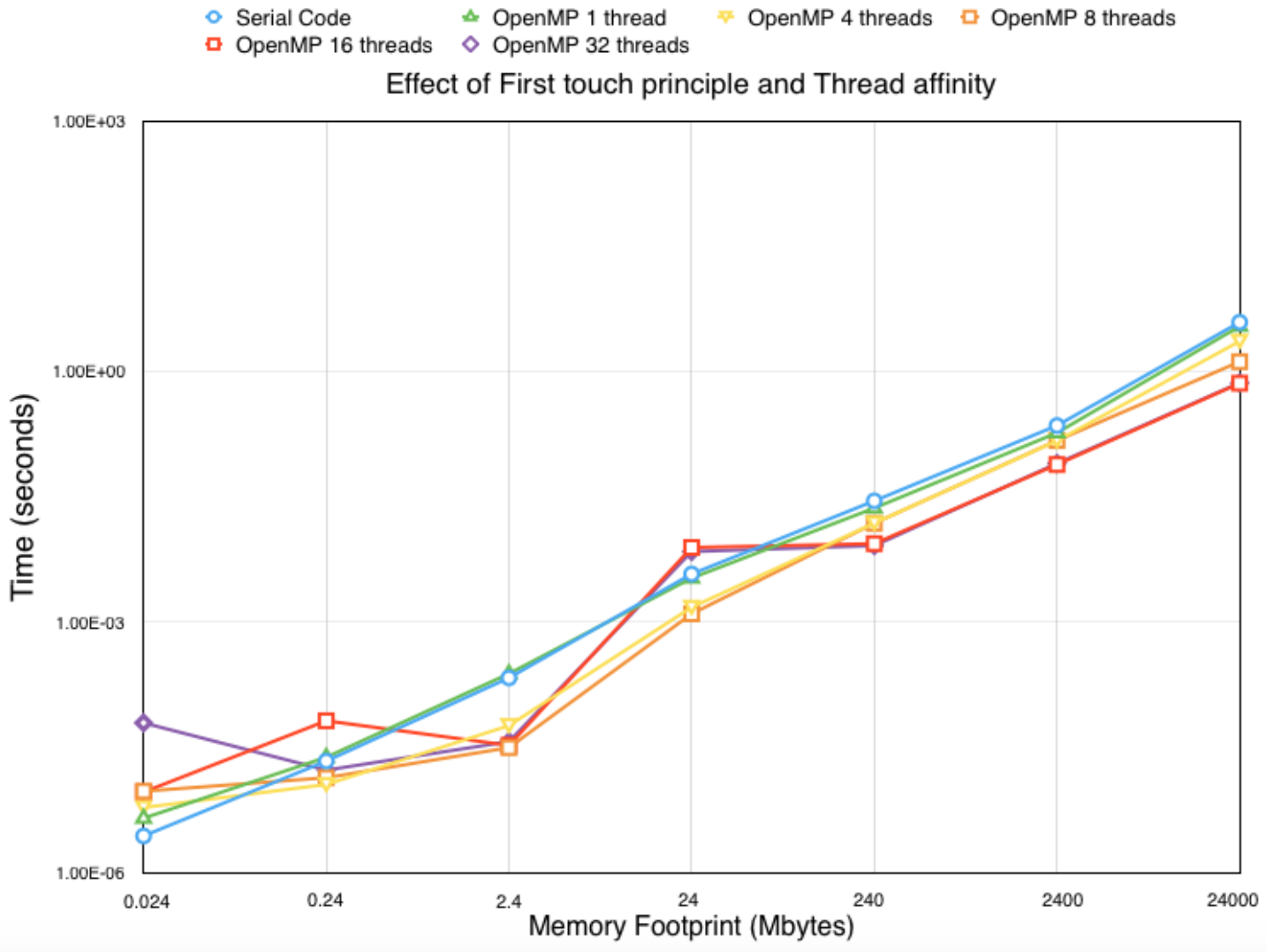
Zoom in for out of core memory

Zoom in for out of core memory



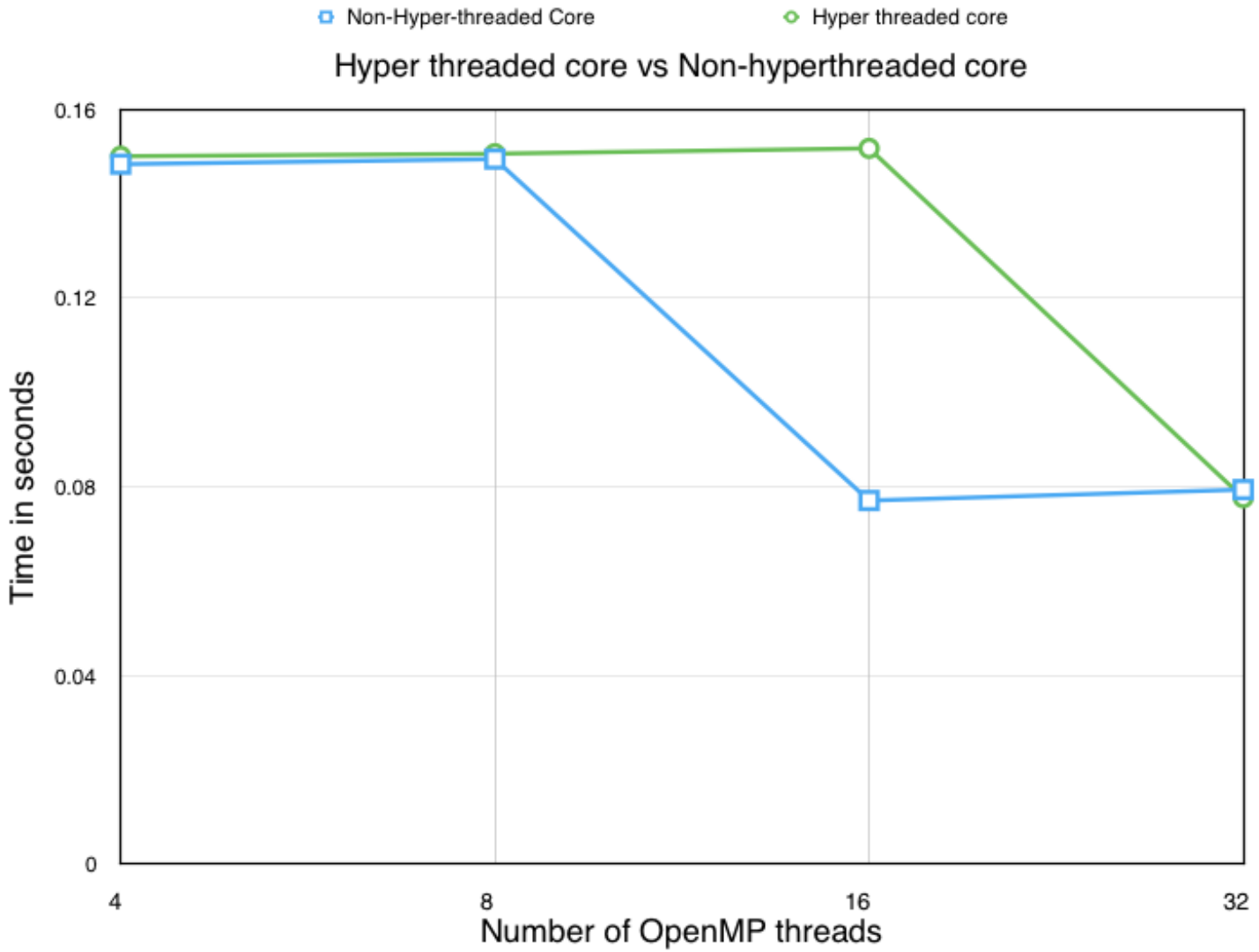
Parallel scalability with thread affinity and data placement

Parallel scalability with thread affinity and data placement



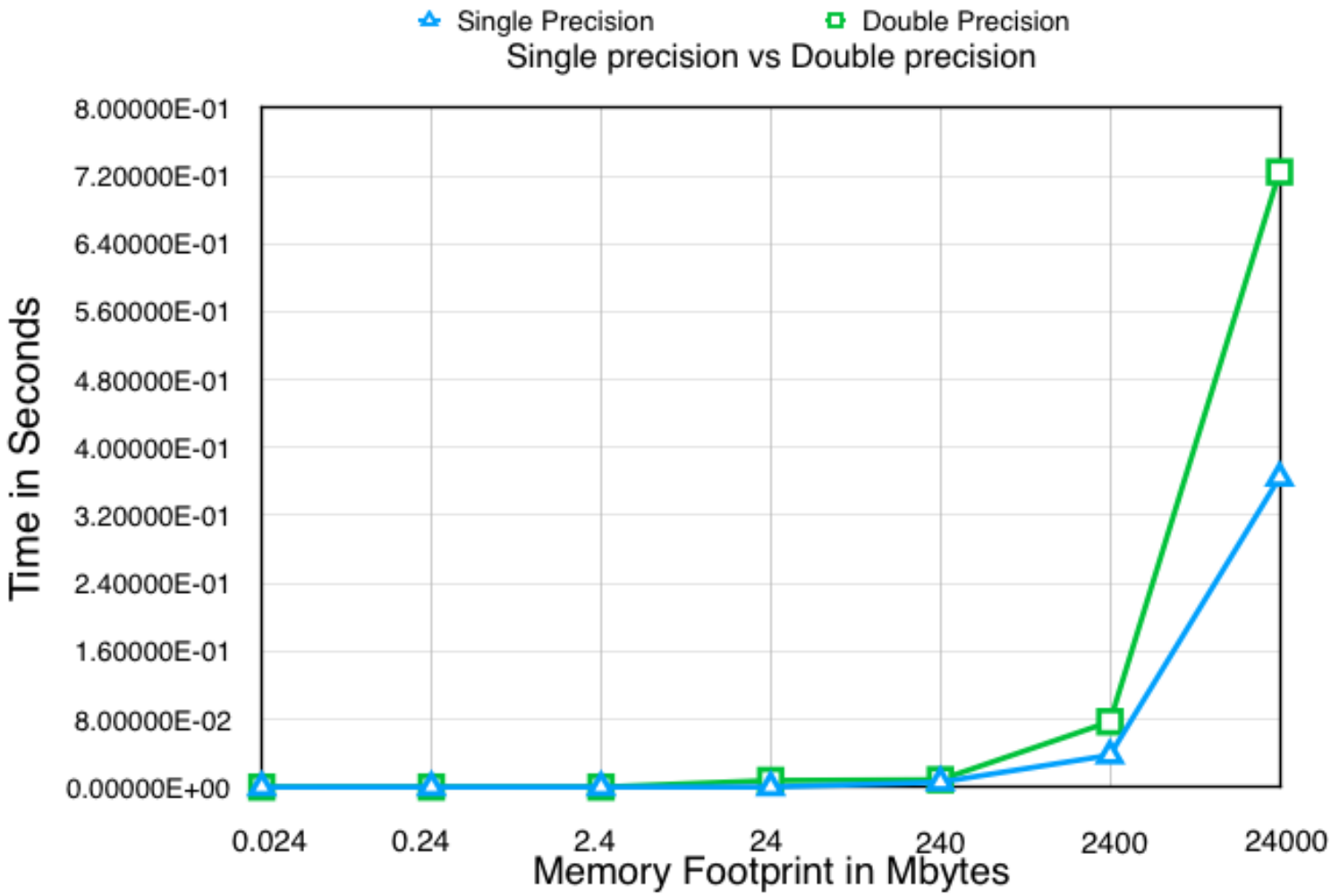
Effect of hyper-threading

Effect of hyper-threading



Floating point balance

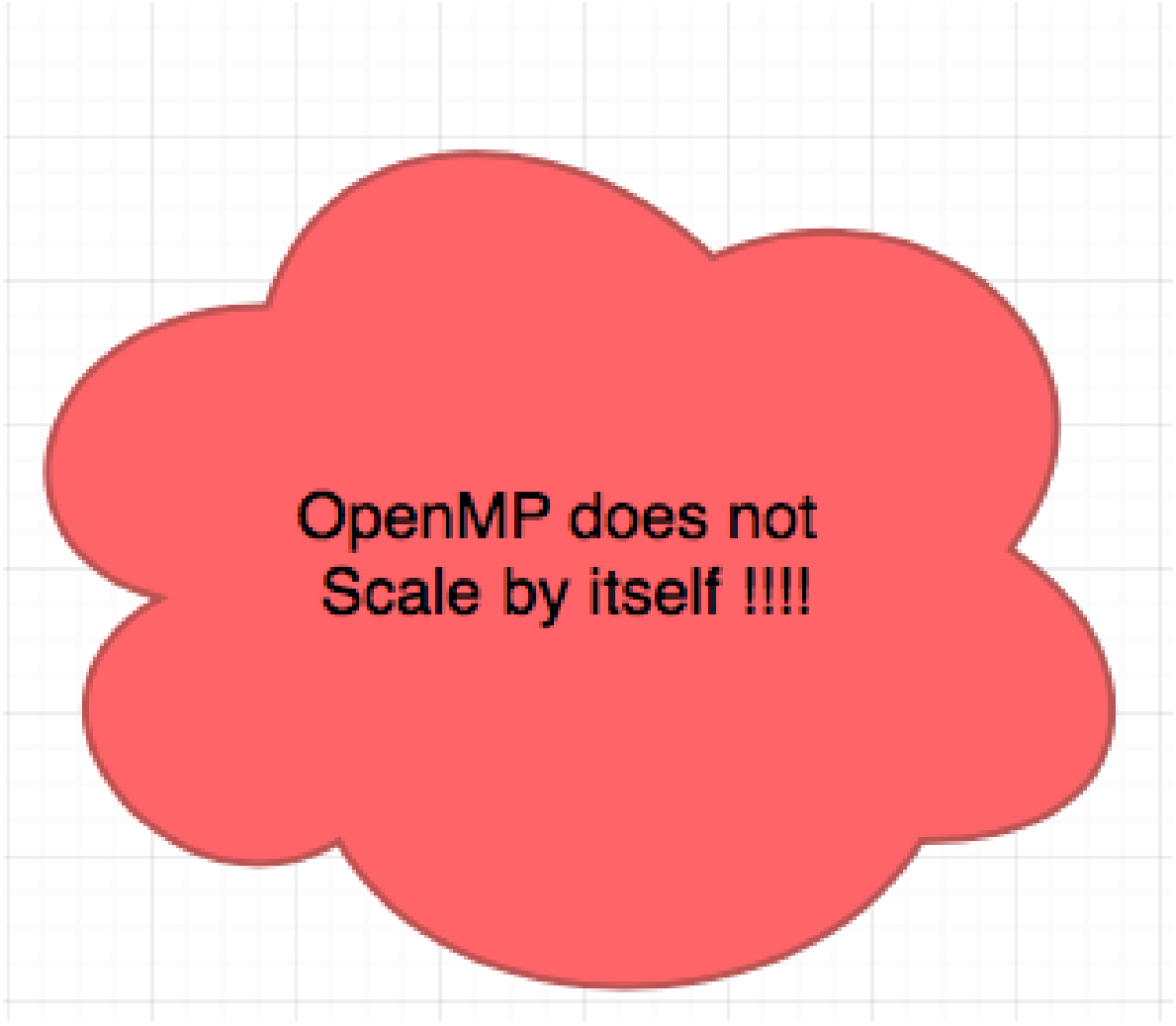
Floating point balance



Conclusion and Future Work

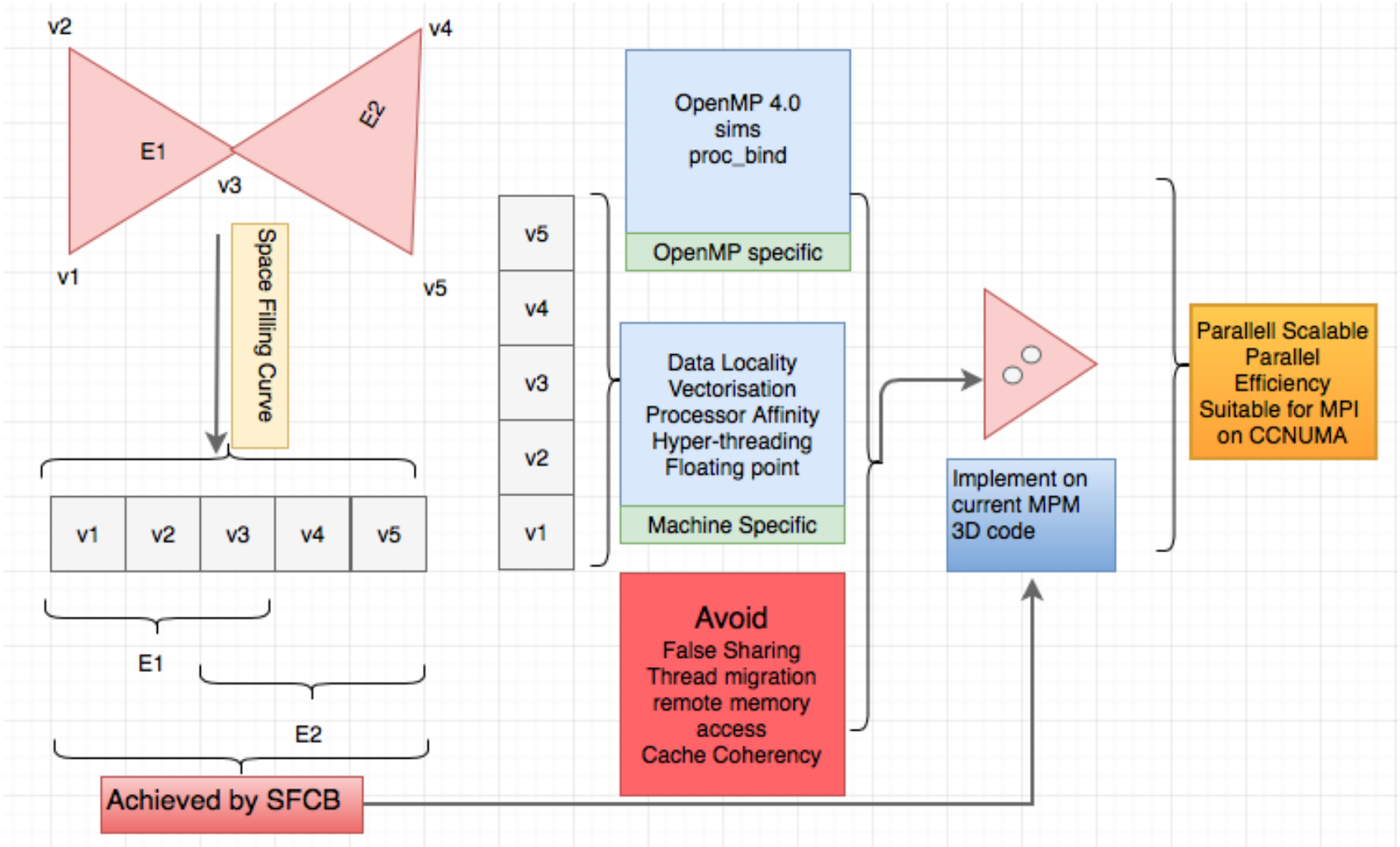
Key inference

Key inference



Moving forward

Moving forward



Thank you for your patience