ECE 563
Programming Parallel Machines

Parallelized Graphics Rendering Using Software Implementation Of OpenGLES

Rakesh Shaji Lal
Srivatsan Bhaskar
Bingnan Huang

OpenGLES

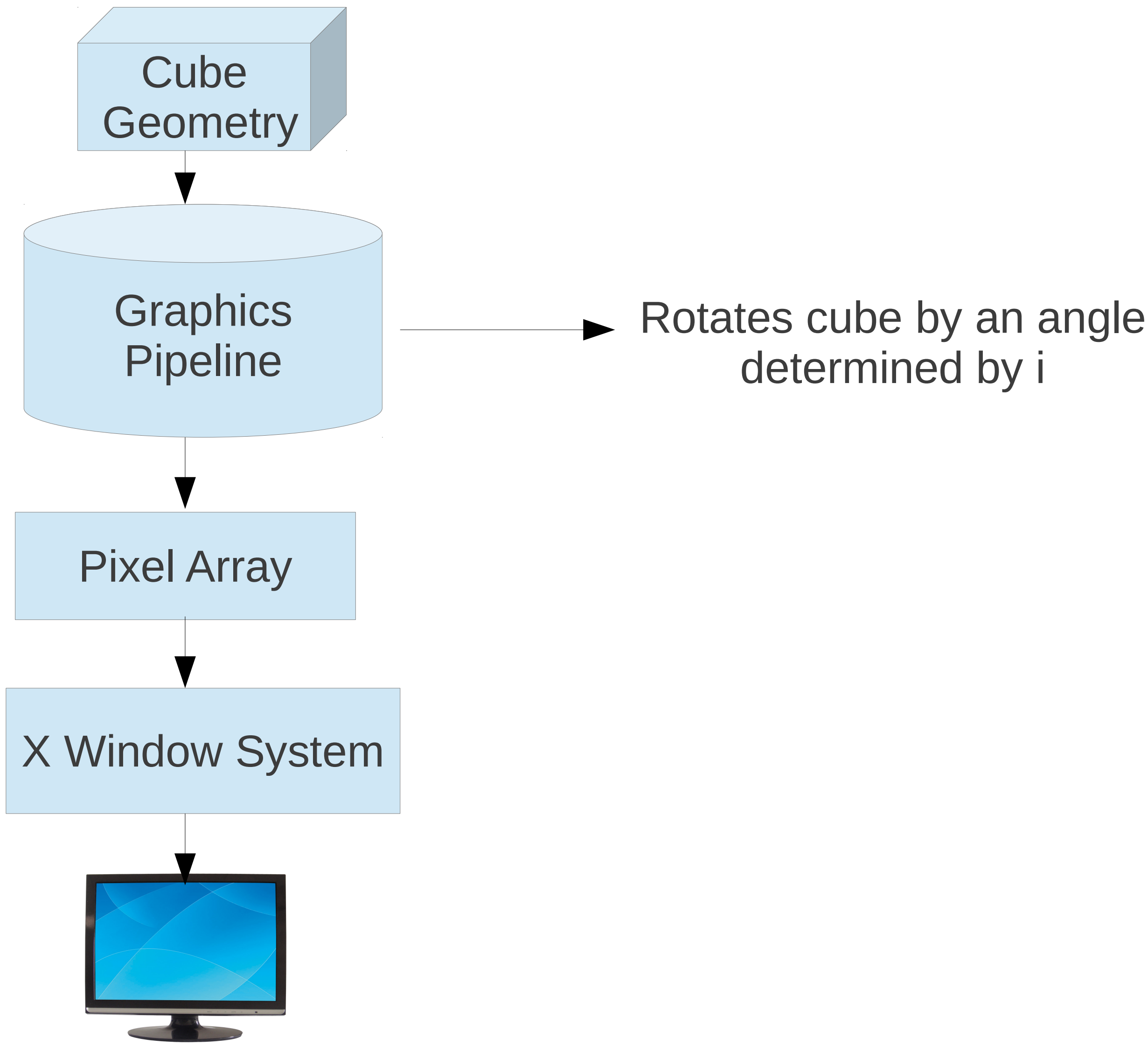
- **Open G**raphics **L**ibrary for **E**mbedded **S**ystems
  - API for rendering graphics in mobile devices
- Often interacts with GPU for accelerated graphics rendering

Objective

- Parallelize the graphics rendering of a test application using software implementation of OpenGLES (implemented in C++)
- Useful for accelerated graphics rendering in the absence of hardware accelerators like GPU

Sequential Code Control Flow

```
for( i =0; i < N; i++)
{
```




```
}
```

Parallelization Methodologies

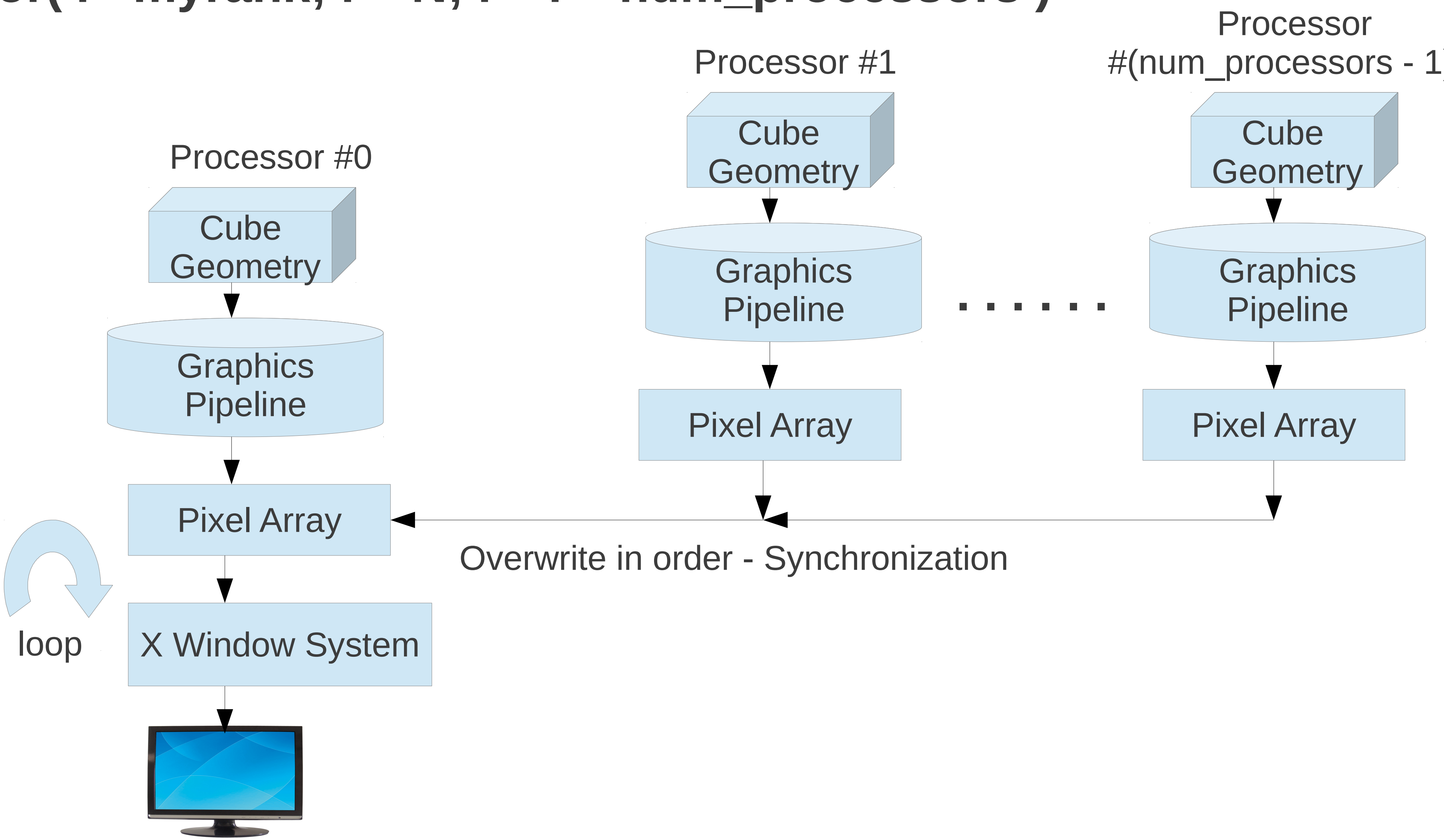
1. Rotation Parallelization using MPI

2. Cube Face Rendering Parallelization Using OpenMP

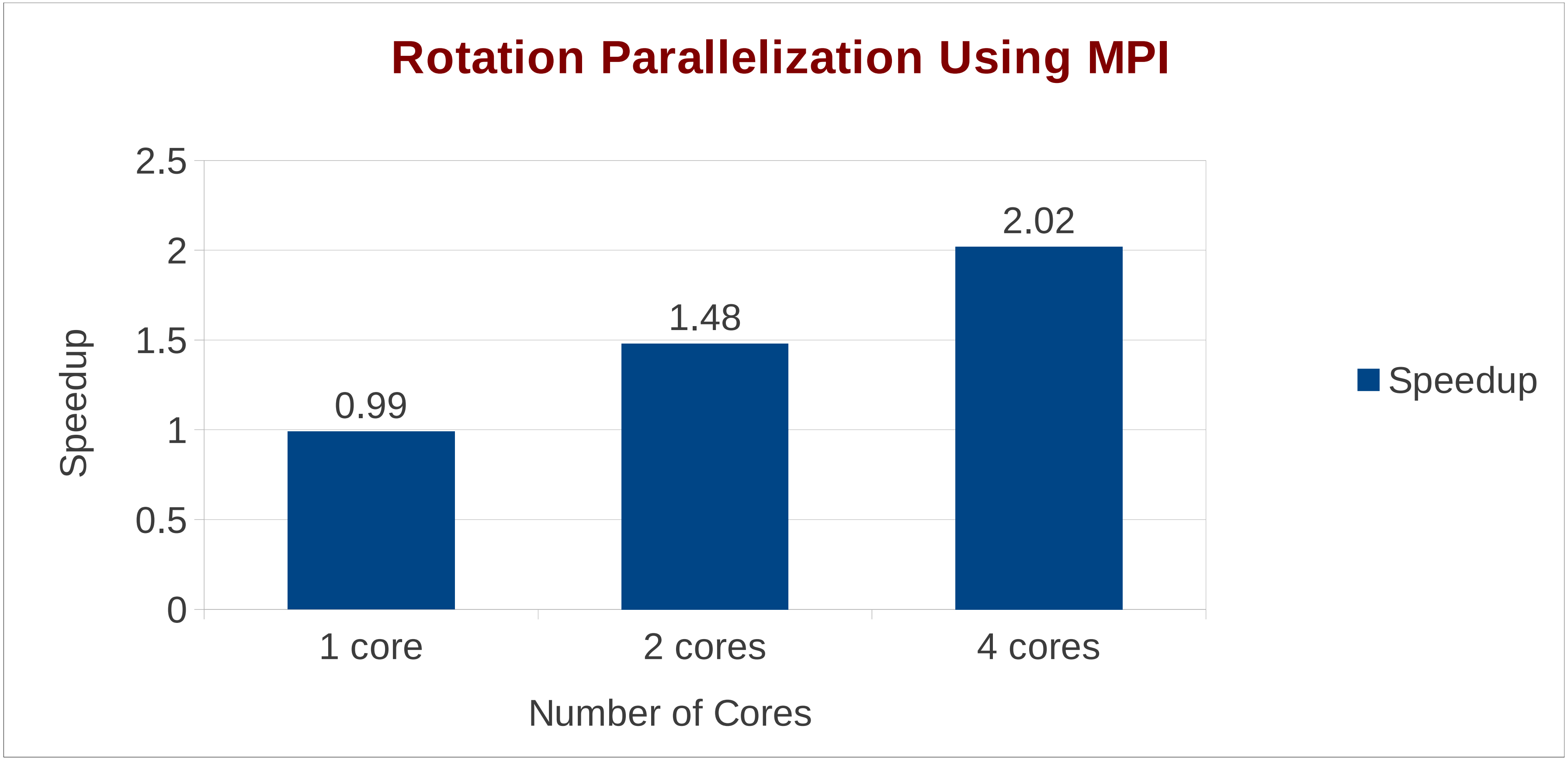
3. Rasterization Parallelization using MPI

Rotation Parallelization using MPI

for(i =myrank; i < N; i = i + num_processors)
{



}



Description	Percentage
Sequential	0.8%
Parallel	99.2%
Synchronization	33.1% of the parallel section

Parallelization Methodologies

1. Rotation Parallelization using MPI

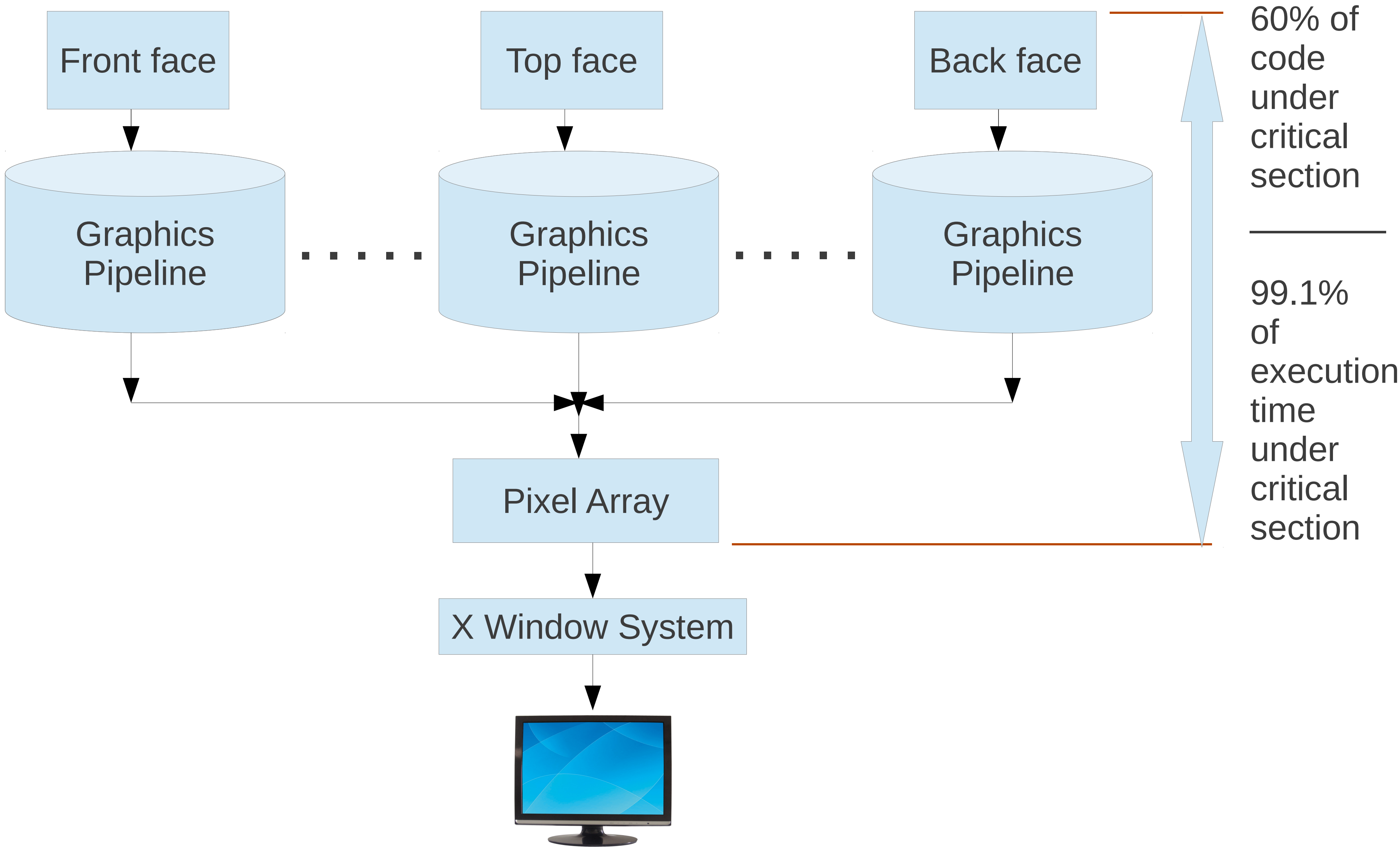
2. Cube Face Rendering Parallelization Using OpenMP

3. Rasterization Parallelization using MPI

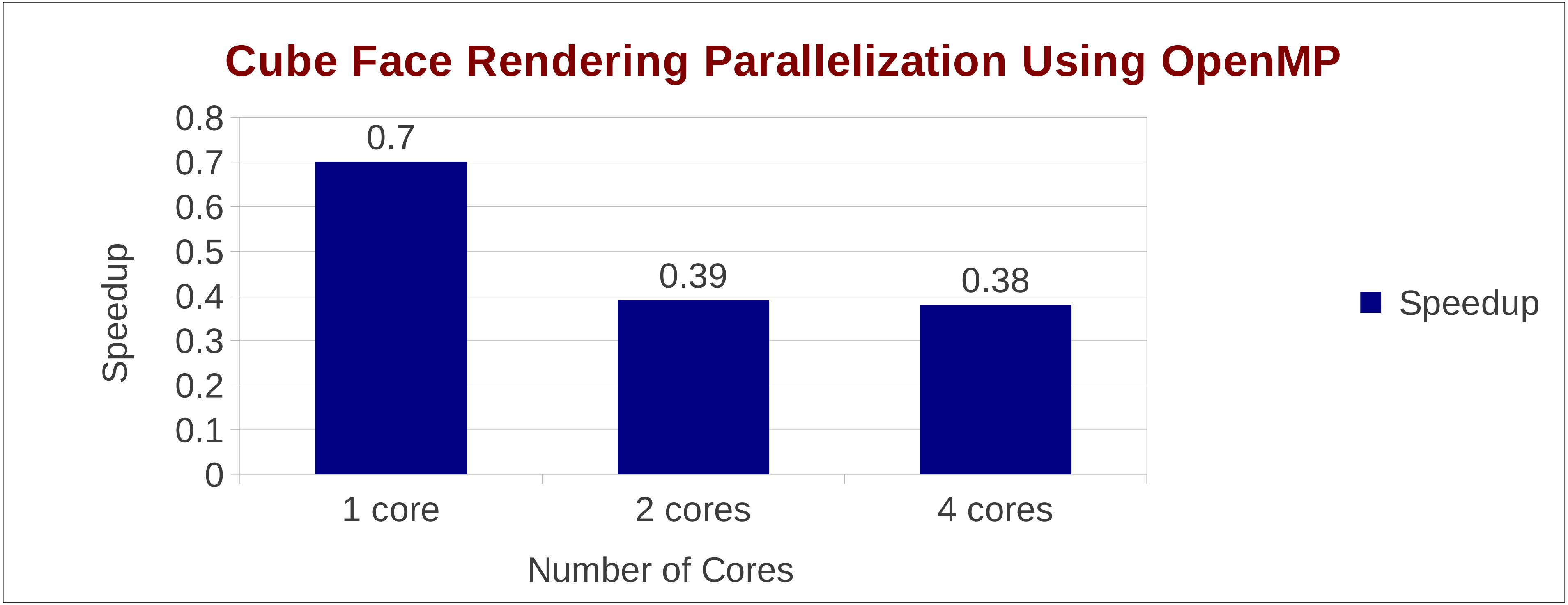
Cube Face Rendering Parallelization using OpenMP

#pragma omp section // Rewrote the matrix operations
// to handle privatized matrices per face

{



}



Description	Time in microseconds
Critical Section in Parallelized Code with 1 core	21110
Same Section in original Sequential Code	9965

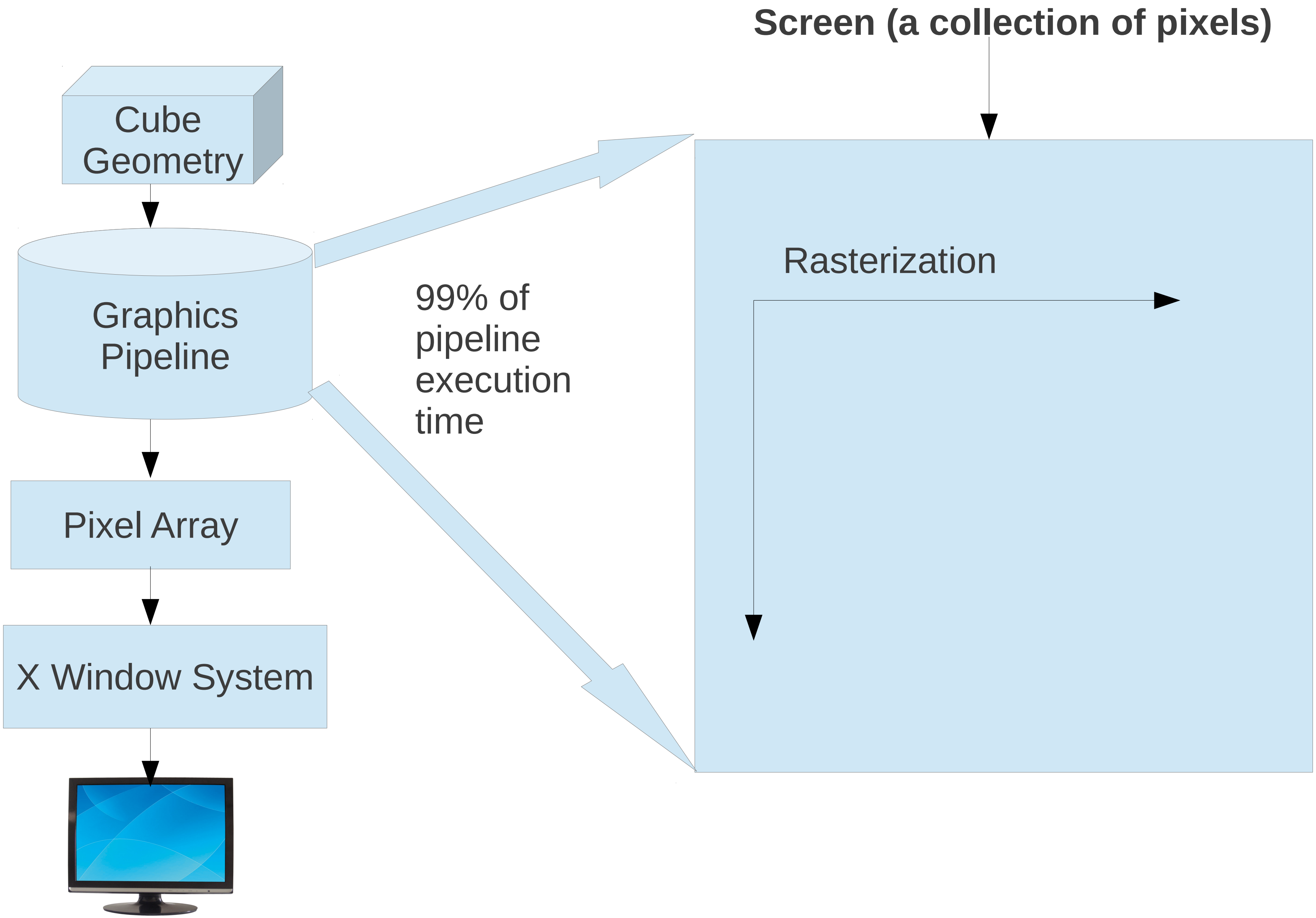
Parallelization Methodologies

1. Rotation Parallelization using MPI

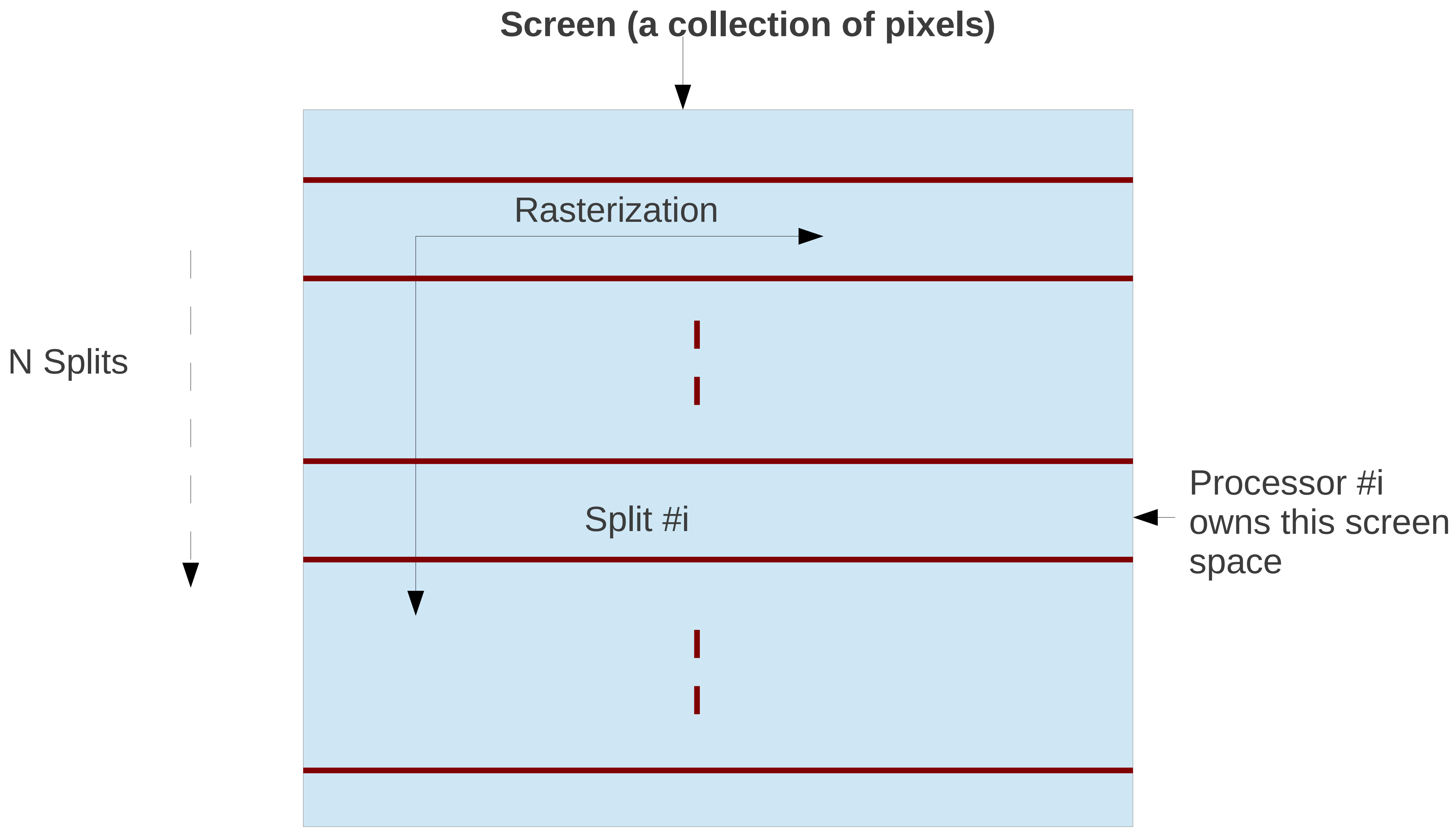
2. Cube Face Rendering Parallelization Using OpenMP

3. Rasterization Parallelization using MPI

Rasterization Parallelization

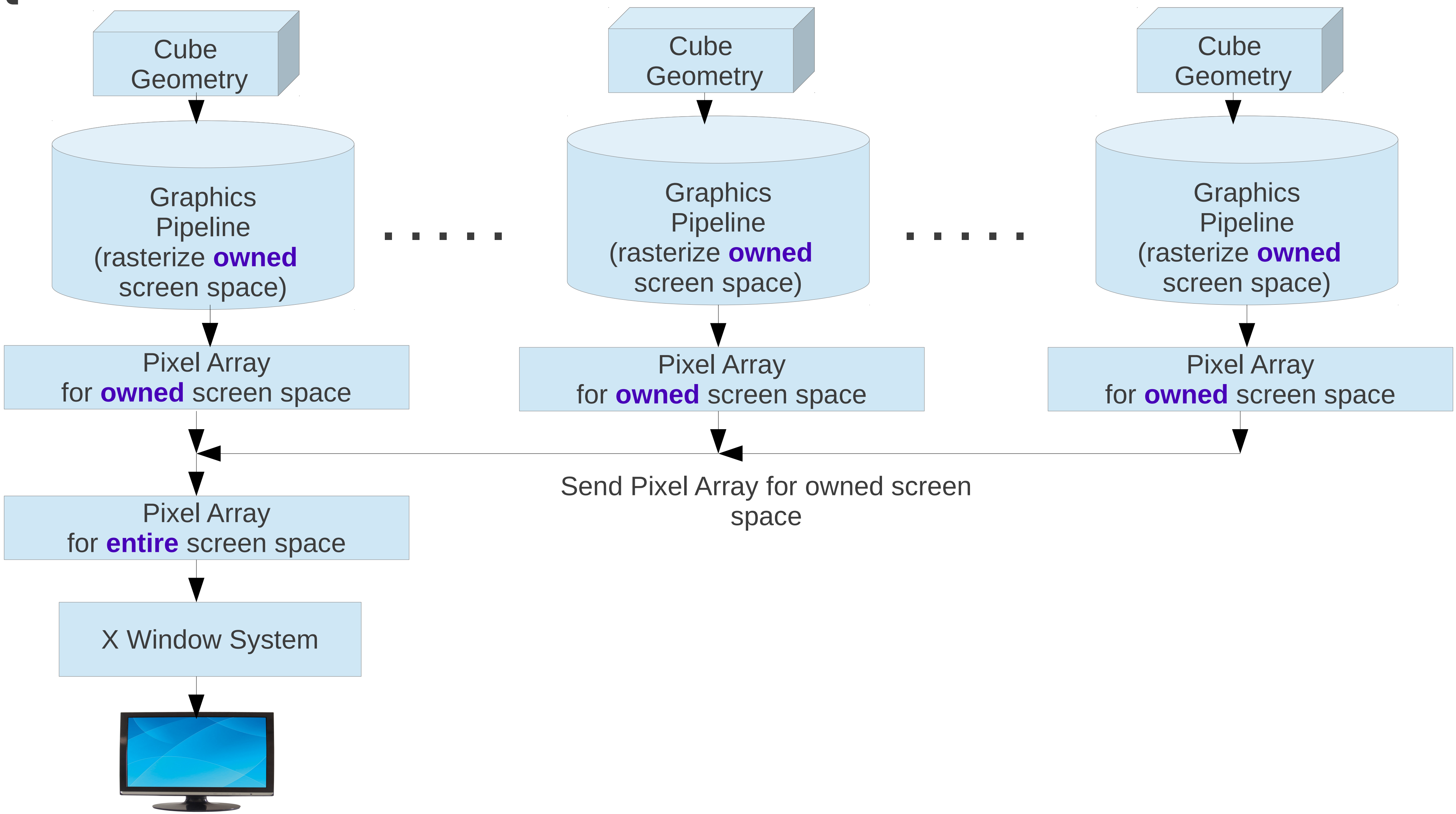


Rasterization Parallelization using MPI

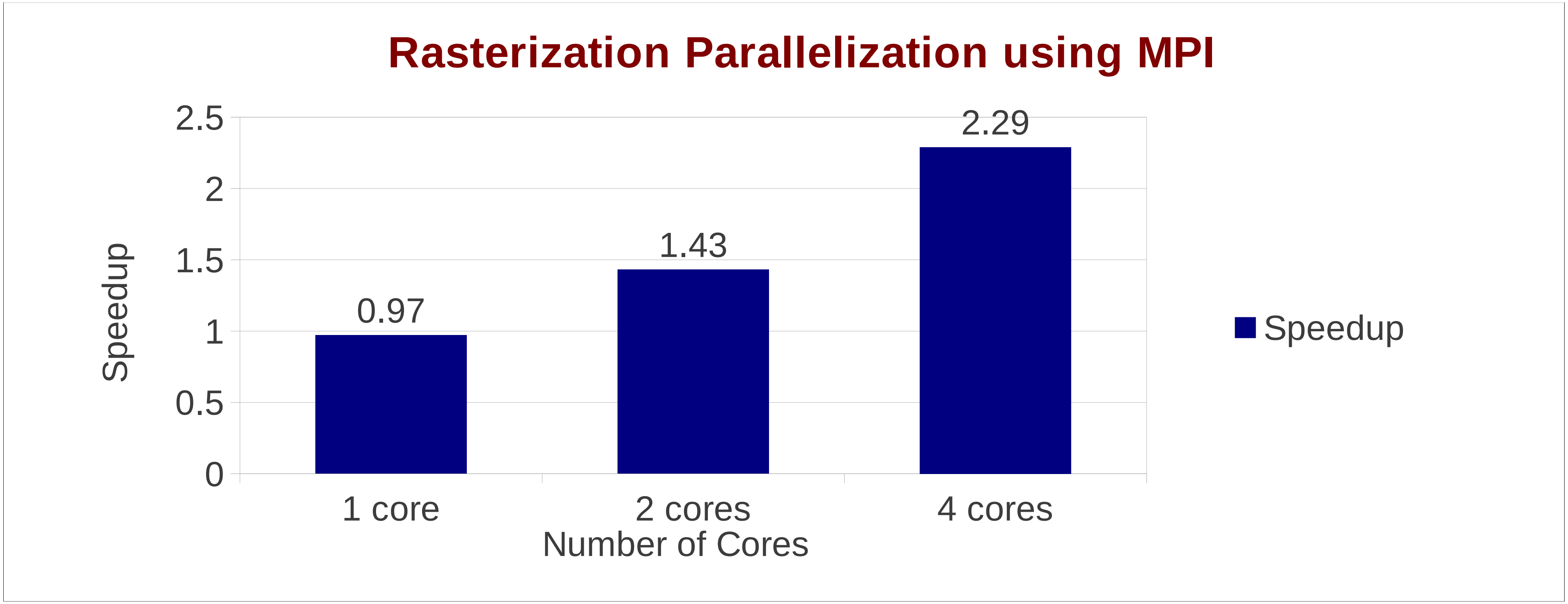


Rasterization Parallelization using MPI

```
for( i =0; i < N; i++)  //10 Induction Variables Substituted
{
```

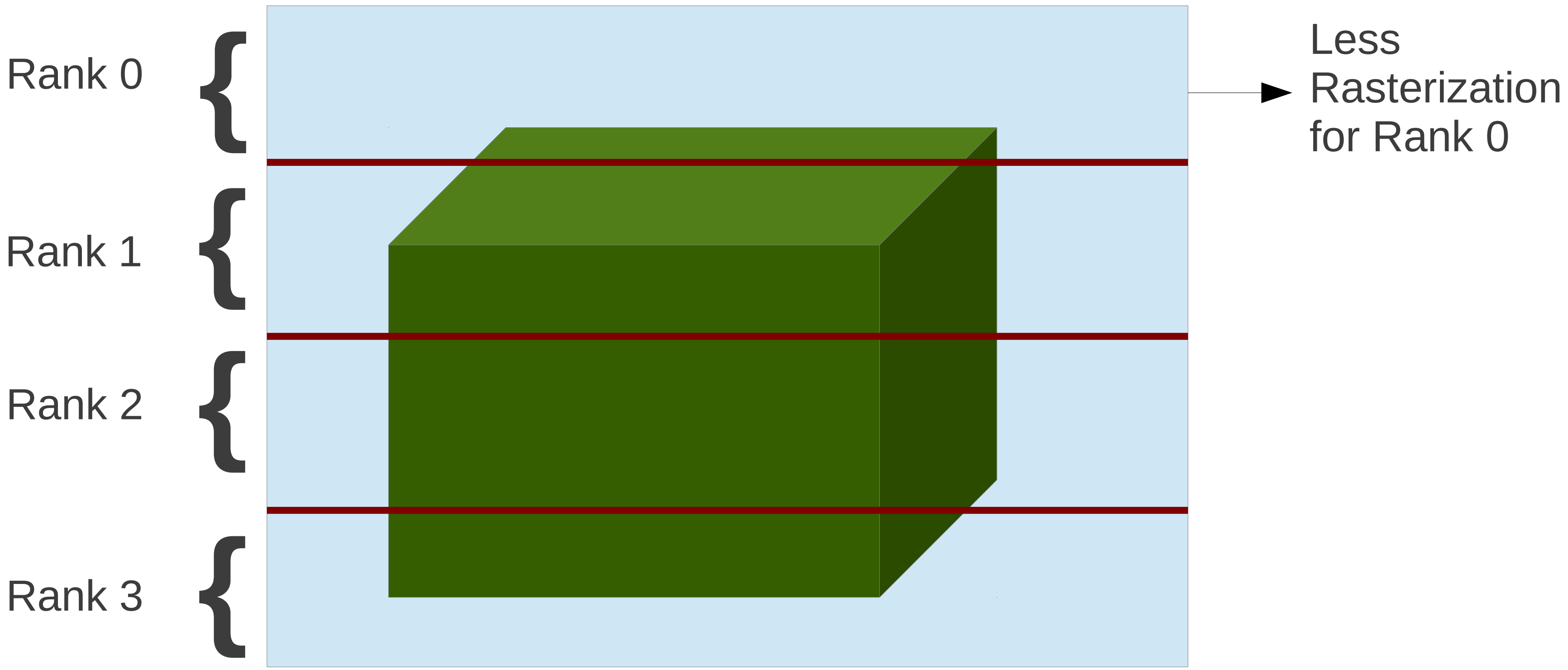



```
}
```

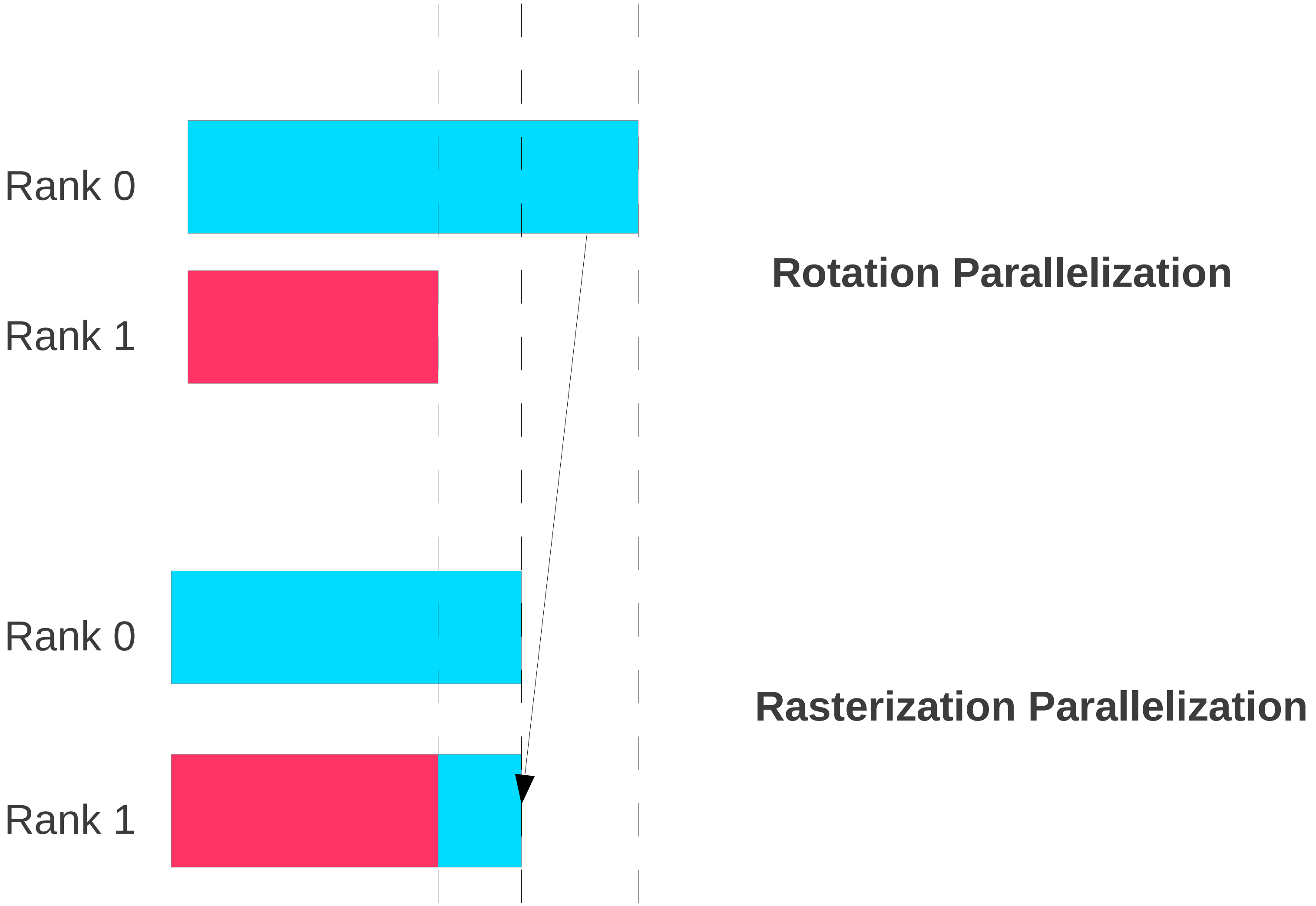

Rasterization Parallelization using MPI
Speedup
Number of Cores
1 core
2 cores
4 cores
0
0.5
1
1.5
2
2.5
0.97
1.43
2.29
Speedup

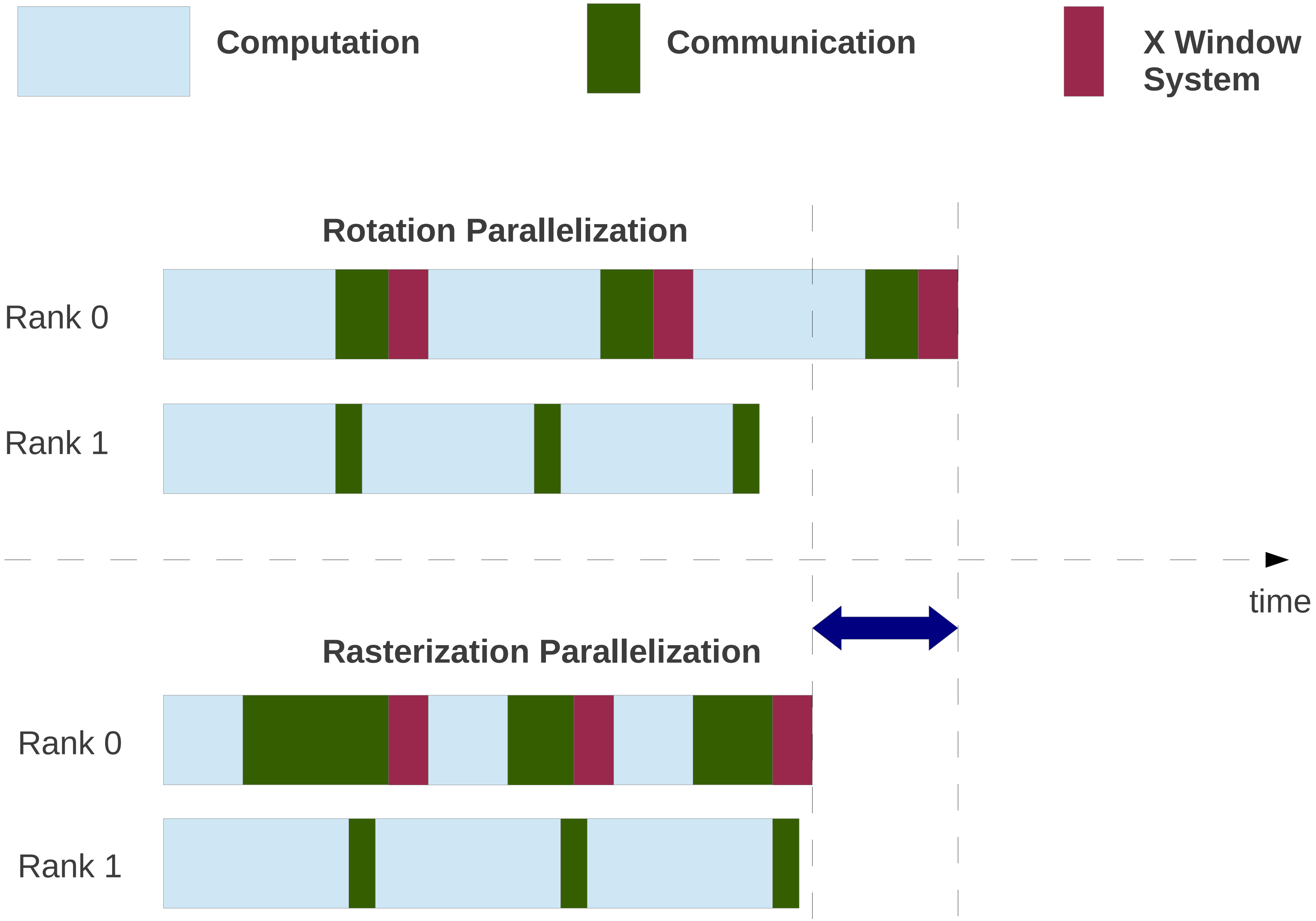
Rotation Parallelization
vs
Rasterization Parallelization

- For 4 cores :
  - Rotation Parallelization Speedup (2.02) < Rasterization Parallelization Speedup(2.29)
- Counter intuitive
  - Rotation Parallelization has more parallel execution than Rasterization Parallelization



Work Offloading




Computation
Communication
X Window System
Rotation Parallelization
Rank 0
Rank 1
time
Rasterization Parallelization
Rank 0
Rank 1

Extra Slides

Induction Variable Substitution Schemes

- Sequential

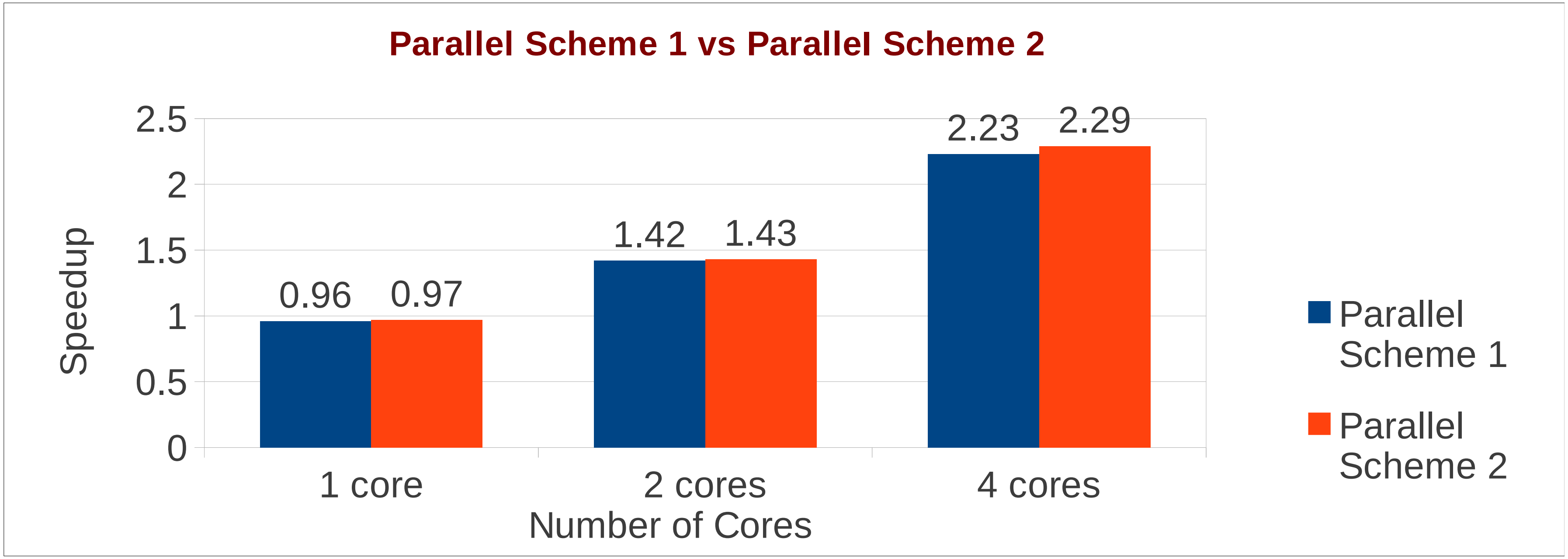
```
k = 0;
for(i = 1; i< N; i++)
    k = k + 2
```

- Parallel Scheme 1

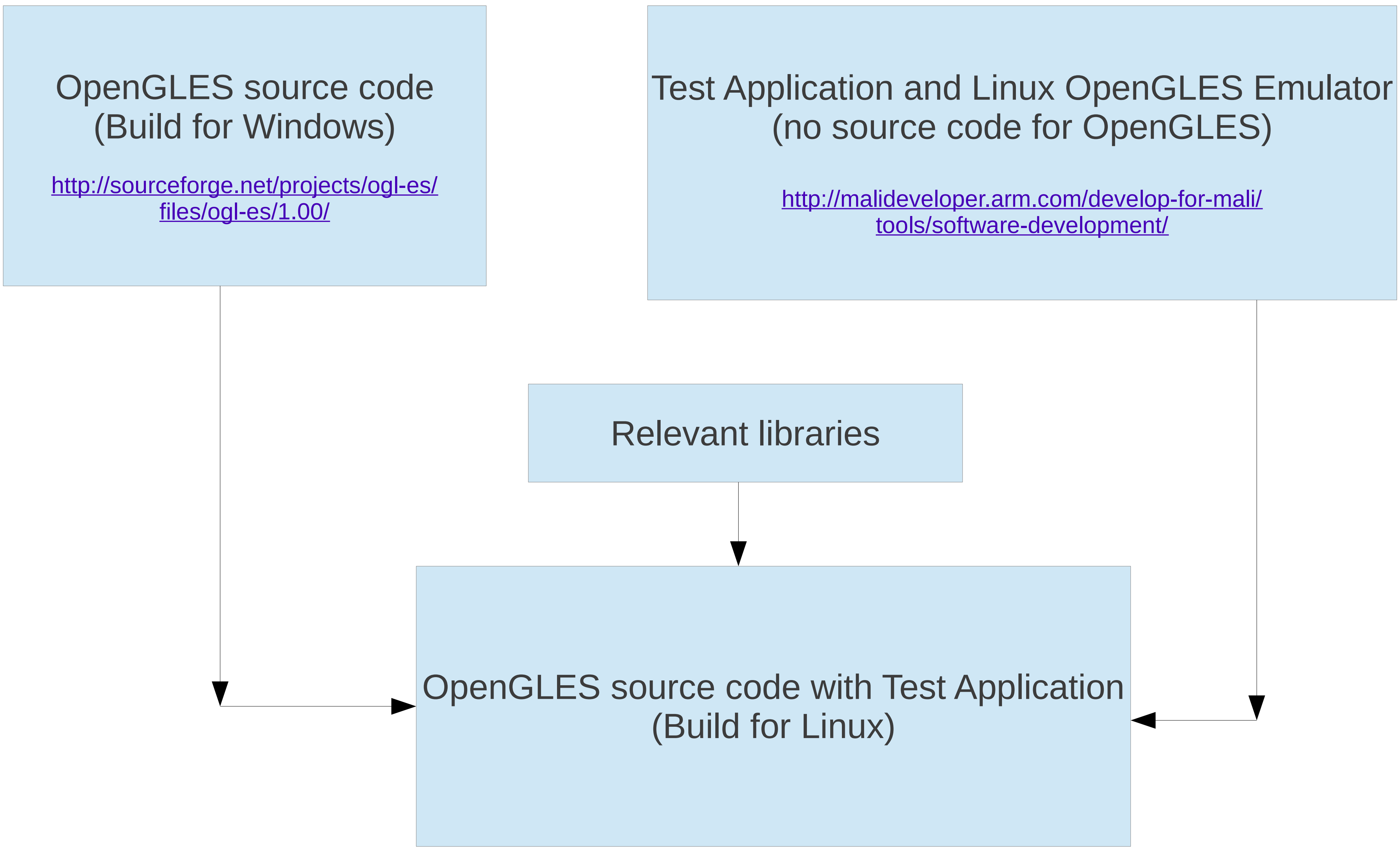
```
for(.....)
    k = i*2;
```

- Parallel Scheme 2

```
k = myrank * chunk_size*2
for(.....)
    k = k +2;
```



Linux Build Creation



Future Work

- Parallelize Rasterization using OpenMP
  - will help eliminate critical section in Cube Face Rendering Parallelization
- Split only the Rasterized screen space evenly among processors using MPI
  - requires OpenGL programming knowledge
- Analyze the parallel engine using more complex test applications
  - Speedup is likely to increase

References

- http://en.wikipedia.org/wiki/OpenGL
- http://en.wikipedia.org/wiki/OpenGL_ES
- http://en.wikipedia.org/wiki/X_Window_System
- http://en.wikipedia.org/wiki/Rasterisation