Parallel computing for Image convolution

Yacine Sahli, E10715005

NTUST

2019

Image convolution

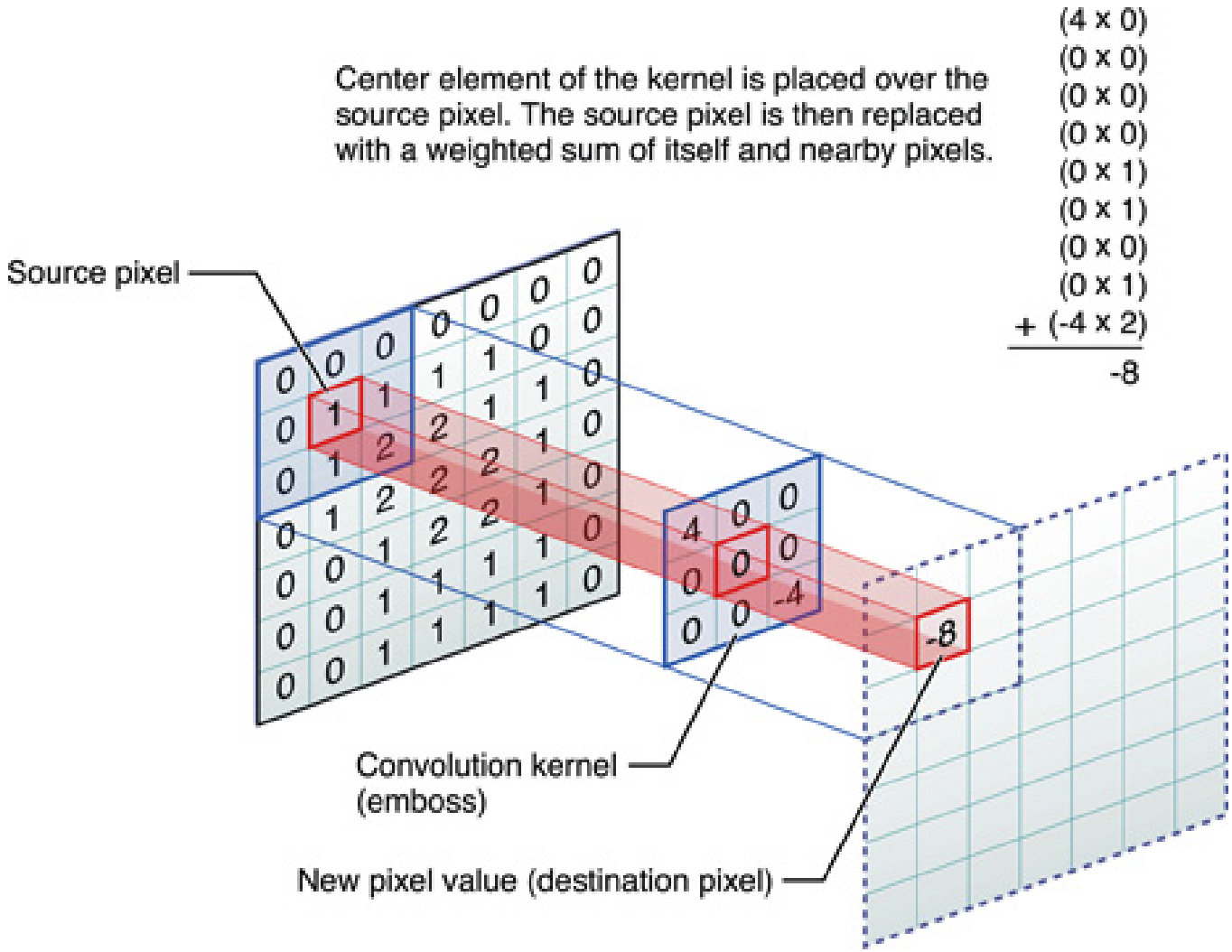
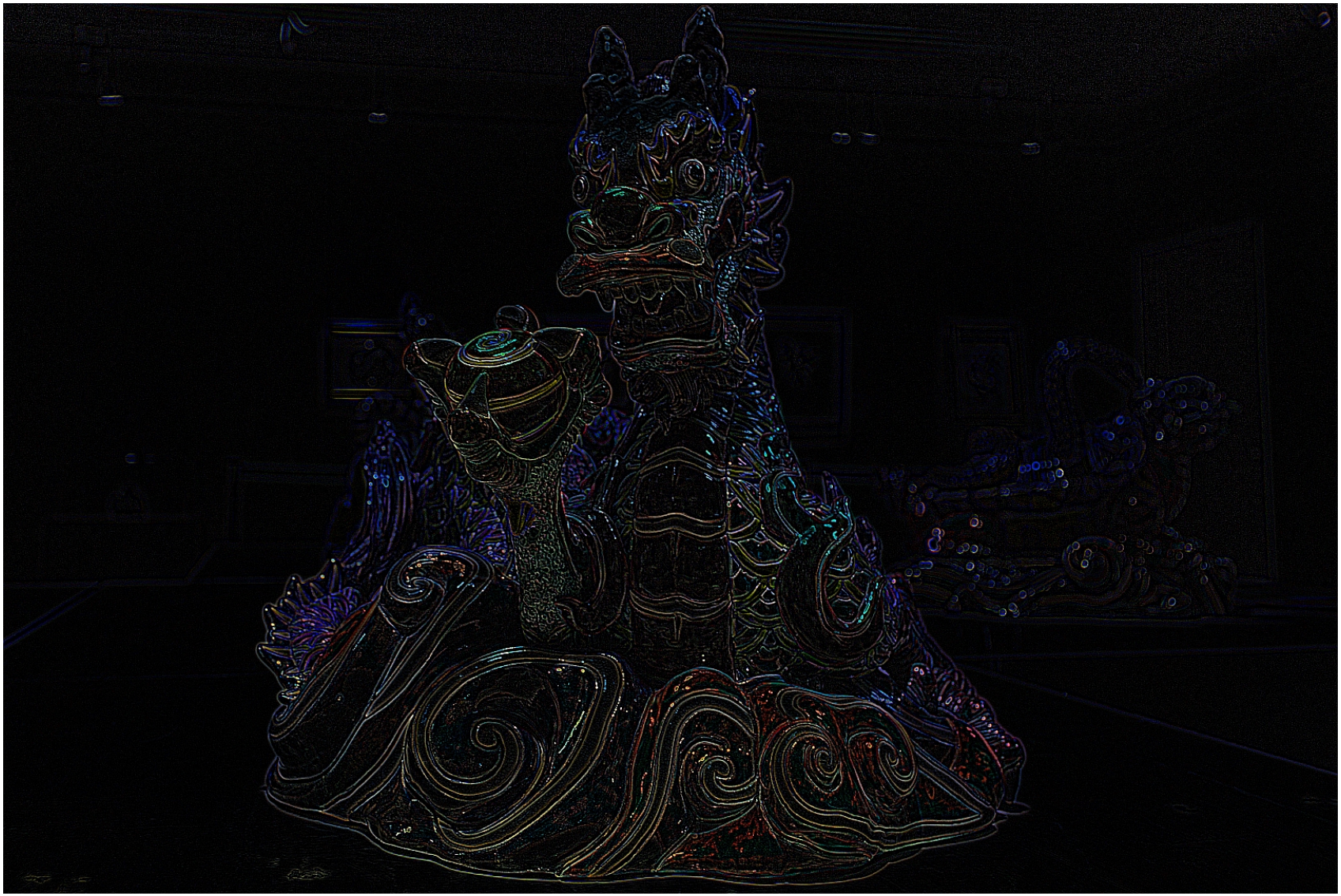


Figure 1: Image Convolution

Application

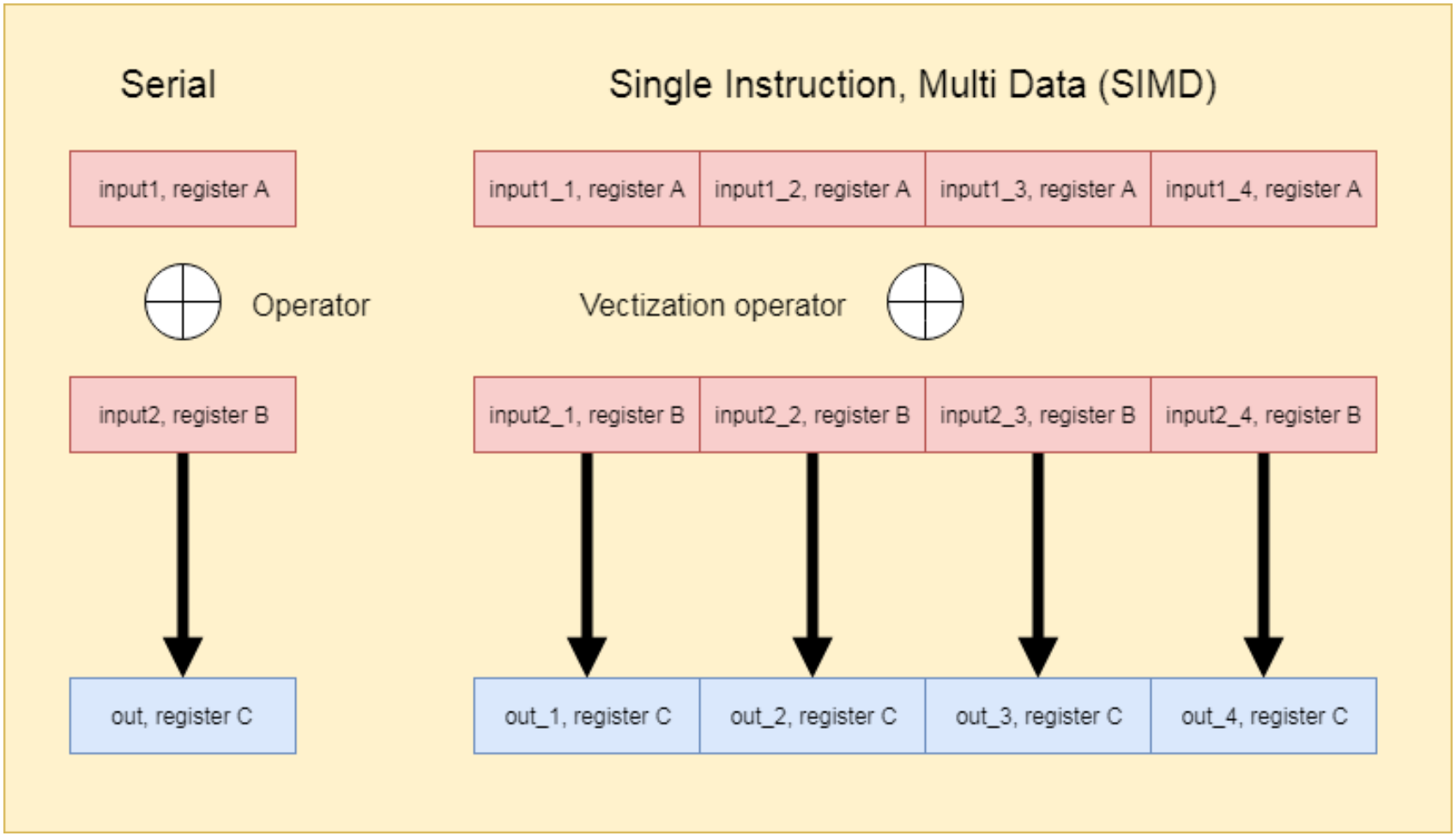
Gaussian blur

HighPass filter



Emboss filter

Parallelization: SIMD



Parallelization: OpenMP

Sheduling

static: smallest overhead

Chunk size

default: Nchunks / N threads

Parallel section

```
#pragma omp parallel
{
#pragma omp /*parallel*/ for
for{;;}
#pragma omp /*parallel*/ for
for{;;}
}
```

Parallelization: OpenMP cont

collapse

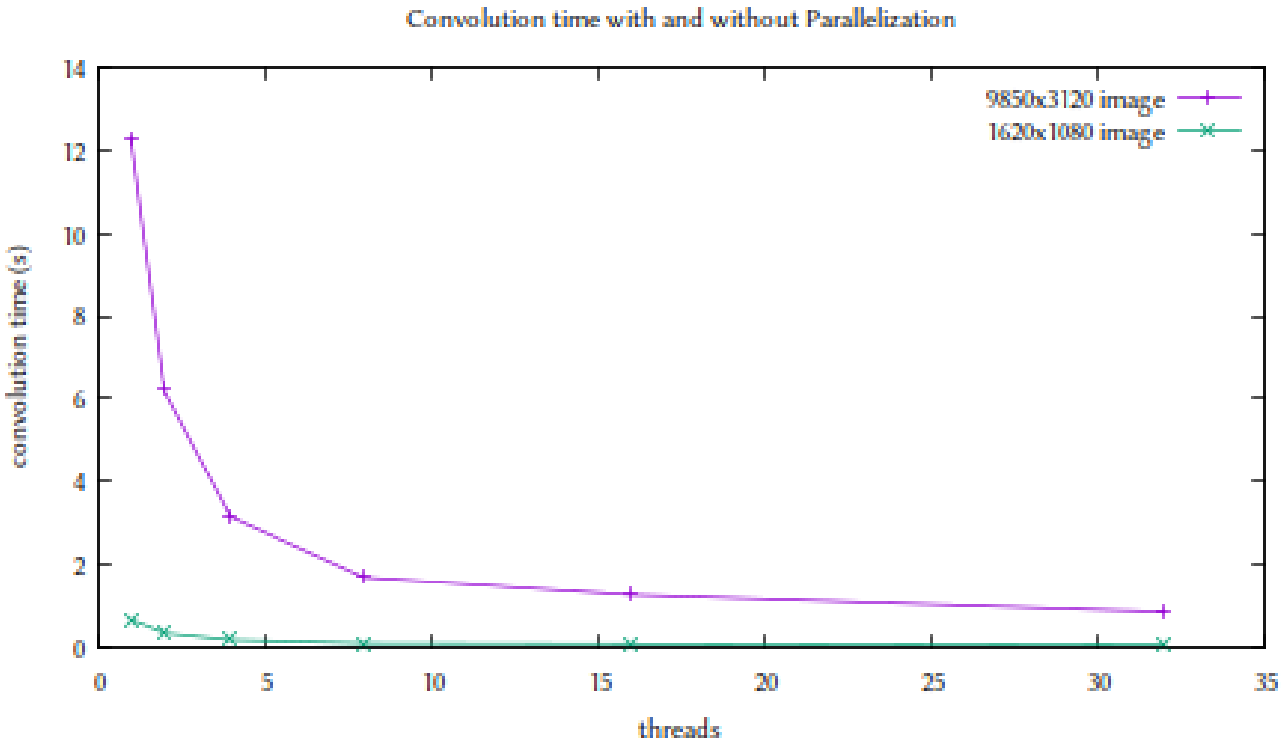
```
for(;;){
    for(;;){

    }
}
```

Use `#pragma omp collapse(N)`

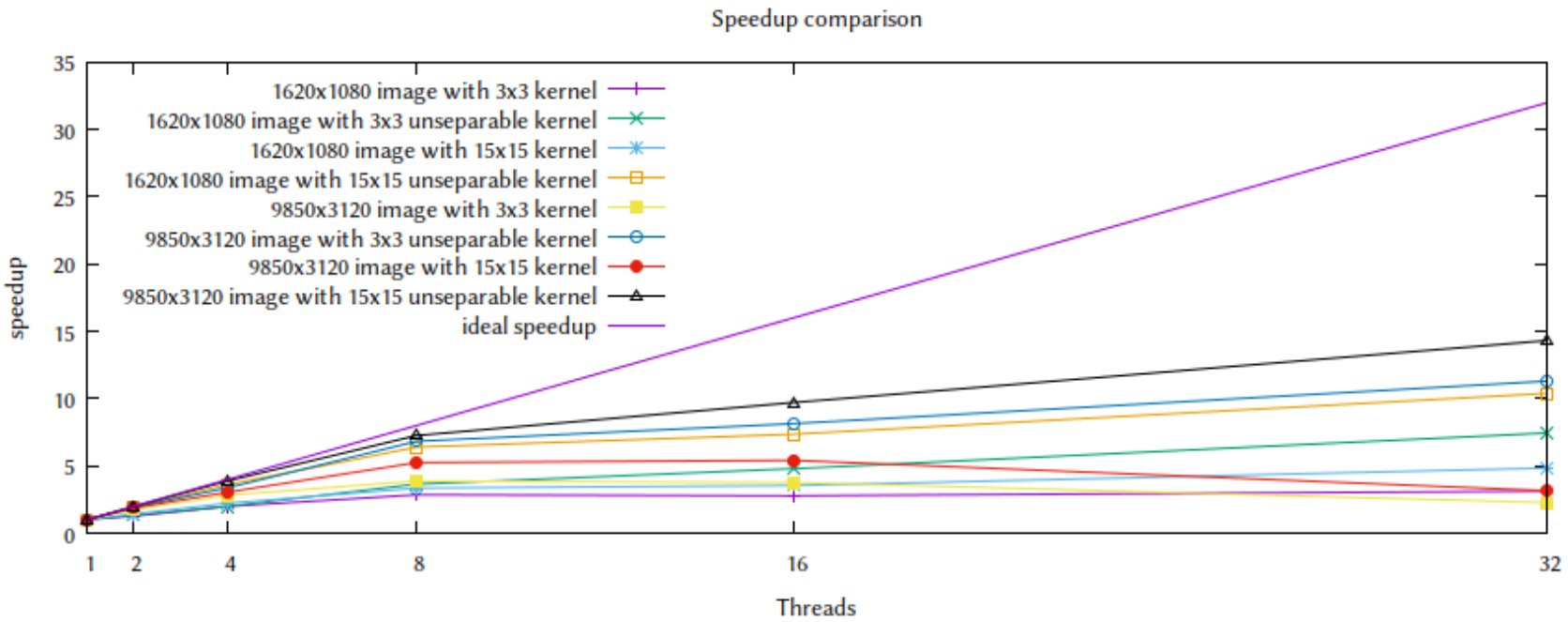
shared()

Share read-only variable. If you write into a variable x, private(x).

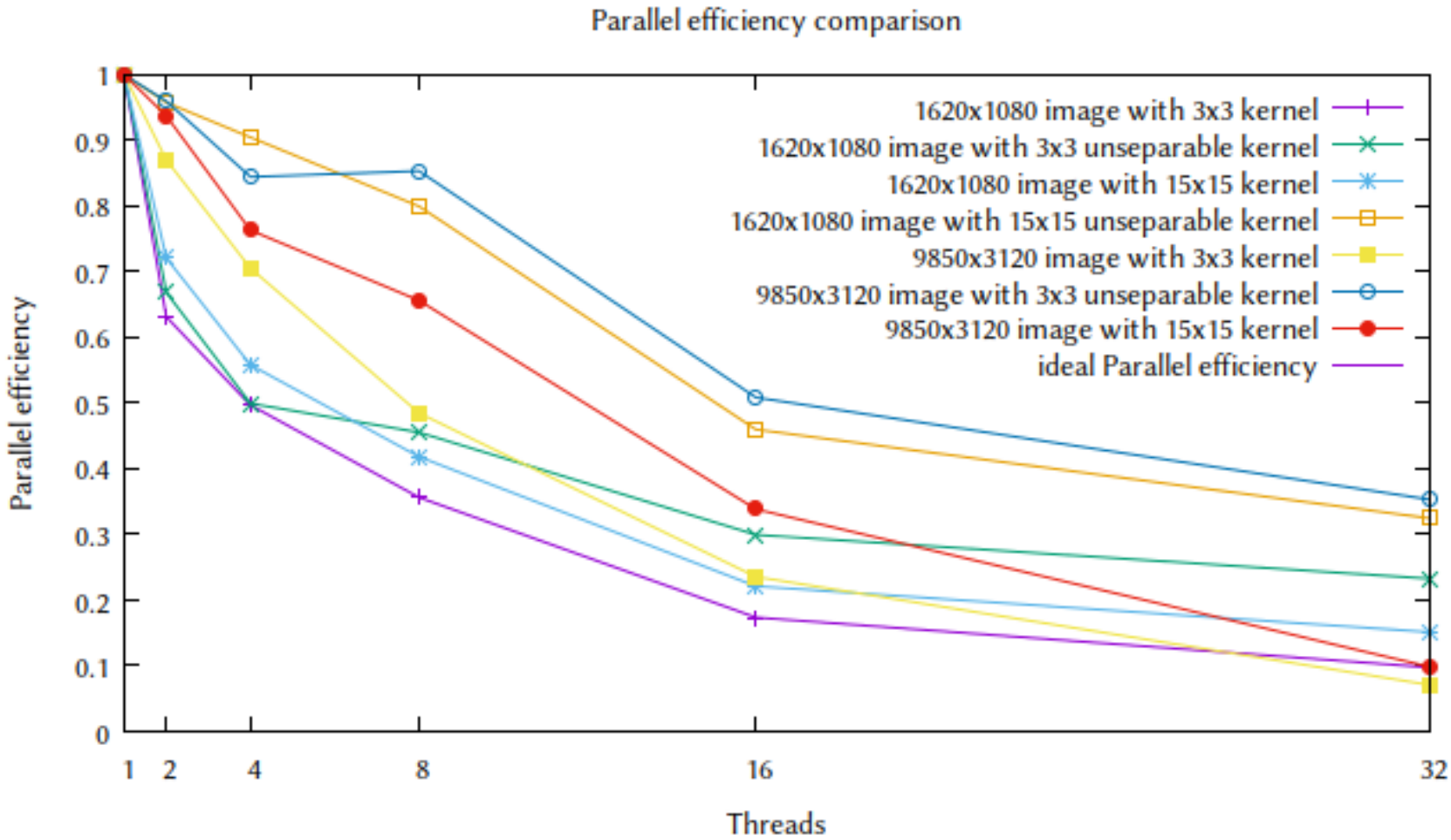
Performance: Serial vs Parallel



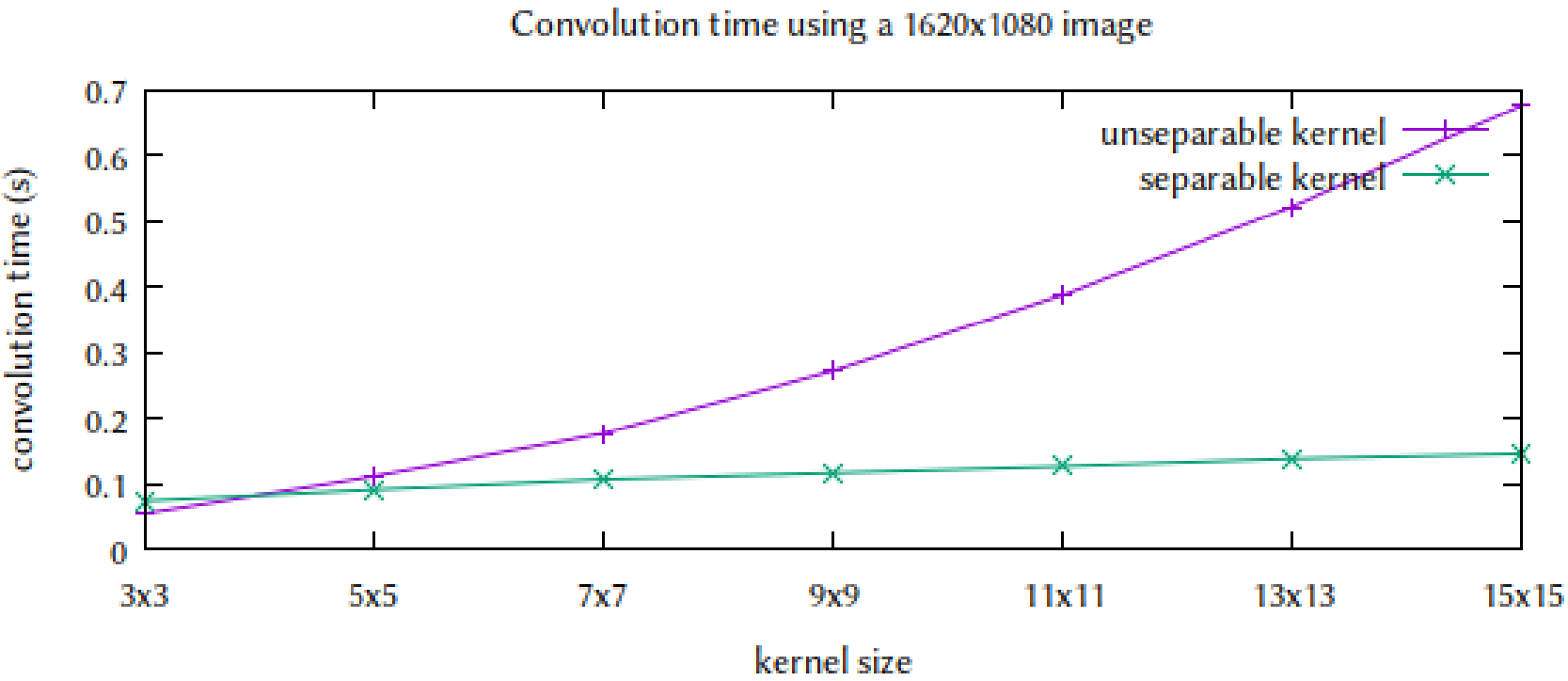
Performance: Speedup



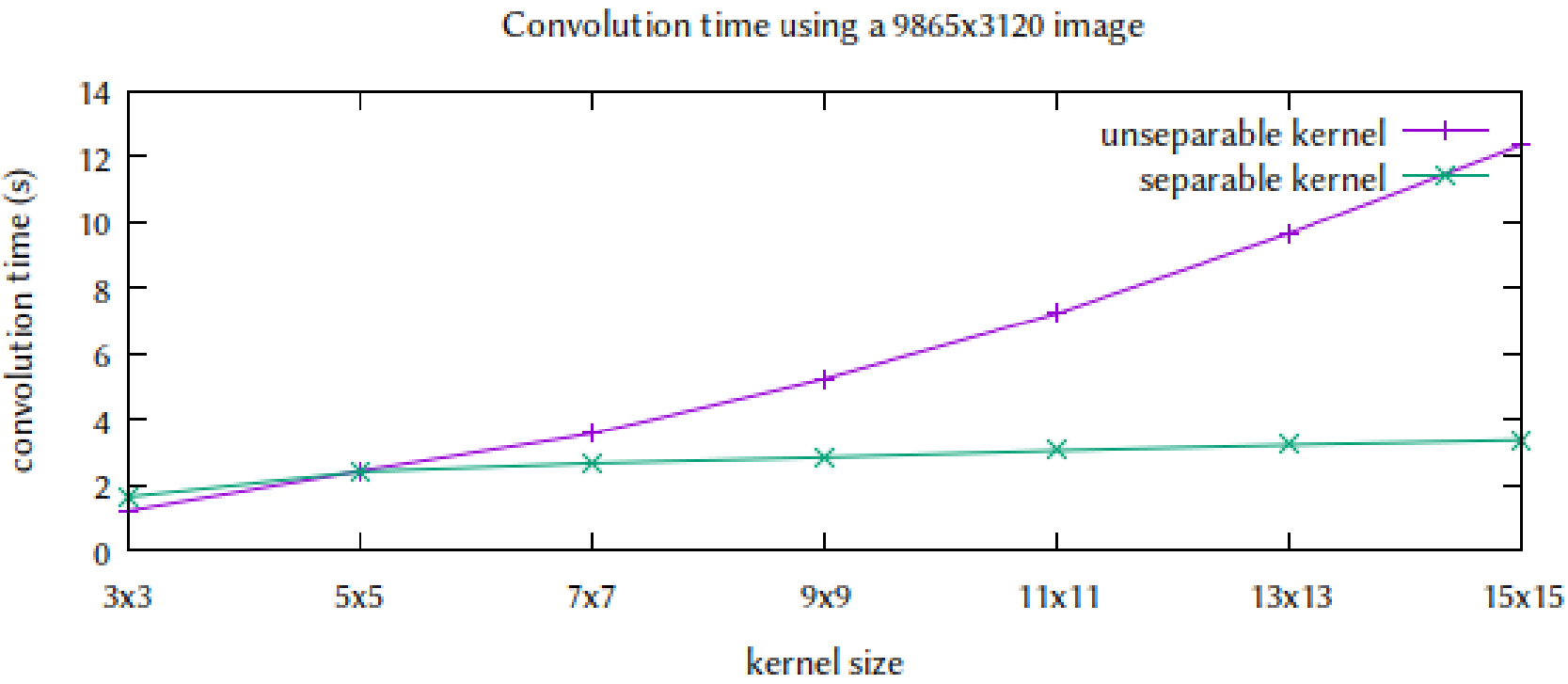
Performance: Parallel efficiency



Performance: Convolution time



Performance: Convolution time



Validations

```
e10715005@cluster2 ~/e10715005/final_project/ImageConvolutionOpenMP $ OMP_NUM_THREADS=1 ./main.exe images/dragon.jpg kernels/box_blur3 tmp1.jpg
1620x1080 3x3 1 0.0982336 0.122991 0.017676
e10715005@cluster2 ~/e10715005/final_project/ImageConvolutionOpenMP $ OMP_NUM_THREADS=4 ./main.exe images/dragon.jpg kernels/box_blur3 tmp2.jpg
1620x1080 3x3 4 0.0988733 0.0624255 0.020923
e10715005@cluster2 ~/e10715005/final_project/ImageConvolutionOpenMP $ diff tmp1.jpg tmp2.jpg
e10715005@cluster2 ~/e10715005/final_project/ImageConvolutionOpenMP $ diff tmp1.jpg images/dragon.jpg
Binary files tmp1.jpg and images/dragon.jpg differ
e10715005@cluster2 ~/e10715005/final_project/ImageConvolutionOpenMP $
```

References

parallel algorithm using python

Matrix convolution using Parallel Programming

3 different parallel programming models for 2D image convolution