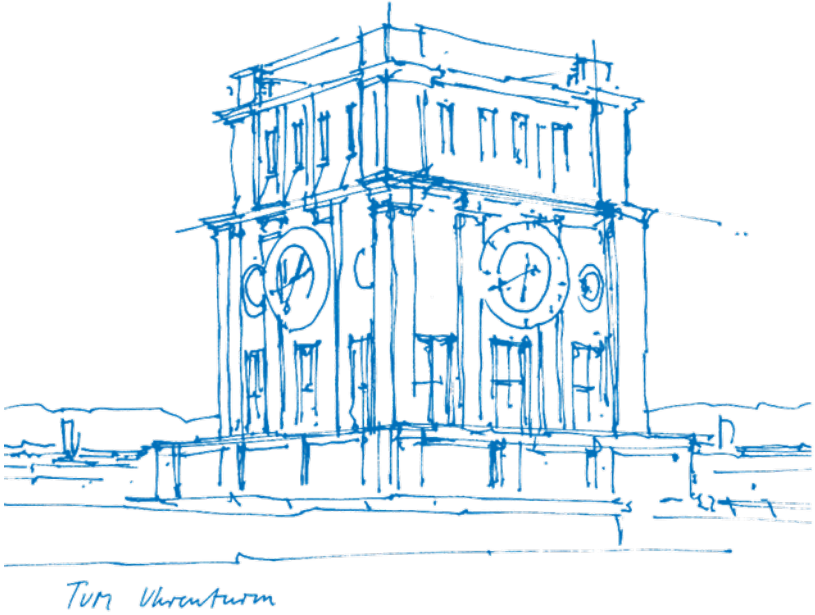
Tutorial 04 – OpenMP

Dennis-Florian Herr, M.Sc.
Chair for Computer Architecture and Parallel Systems (CAPS)
Technical University Munich
24.05.2024



OpenMP: parallel for

OpenMP: Schedules

Web Resource

https://caps-tum.github.io/parprog-visualizations/openmp-schedules/omp_schedules.html

OpenMP: Data Sharing

Explicit Sharing

Code	Shared Data	T1 Data	T2 Data
`int data[] = {0, 0};`	`data = [0, 0];`		
`#pragma omp parallel for shared(data)`			
`for(int i = 0 ; i < 2; i++) {`			
`data[i]++;`	`data = [0→1, 0→1];`	`i = 0;`	`i = 1;`
`}`			
`outputArray(data);`	`data = [1, 1];`		

OpenMP: Data Sharing

Implicit Sharing

Code	Shared Data	T1 Data	T2 Data
`int data[] = {0, 0};`	`data = [0, 0];`		
`#pragma omp parallel for`			
`for(int i = 0 ; i < 2; i++) {`			
`data[i]++;`	`data = [0→1, 0→1];`	`i = 0;`	`i = 1;`
`}`			
`outputArray(data);`	`data = [1, 1];`		

OpenMP: Data Sharing

Data Racing

Code	Shared Data	T1 Data	T2 Data
`int data = 0;`	`data = 0;`		
`#pragma omp parallel for`			
`for(int i = 0 ; i < 2; i++) {`			
`data++;`	`data = 0→?;`	`i = 0;`	`i = 1;`
`}`			
`output(data);`	`data = ?;`		

OpenMP: Data Sharing

Privatization

Code	Shared Data	T1 Data	T2 Data
`int data = 0;`	`data = 0;`		
`#pragma omp parallel for private(data)`		`data = ?;`	`data = ?;`
`for(int i = 0 ; i < 2; i++) {`			
`data++;`	`data = 0;`	`data =?→?; i=0;`	`data =?→?;i=1;`
`}`			
`output(data);`	`data = 0;`		

OpenMP: Data Sharing

First Private

Code	Shared Data	T1 Data	T2 Data
`int data = 0;`	`data = 0;`		
`#pragma omp parallel for firstprivate(data)`		`data = 0;`	`data = 0;`
`for(int i = 0 ; i < 2; i++) {`			
`data++;`	`data=0;`	`data=0→1; i=0;`	`data=0→1;i=1;`
`}`			
`output(data);`	`data = 0;`		

OpenMP: Data Sharing

First Private + Last Private

Code

```
int data = 0;

#pragma omp parallel for firstprivate(data)\
                         lastprivate(data)
for(int i = 0 ; i < 2; i++) {

    data++;

}

output(data);
```

Shared Data | T1 Data | T2 Data

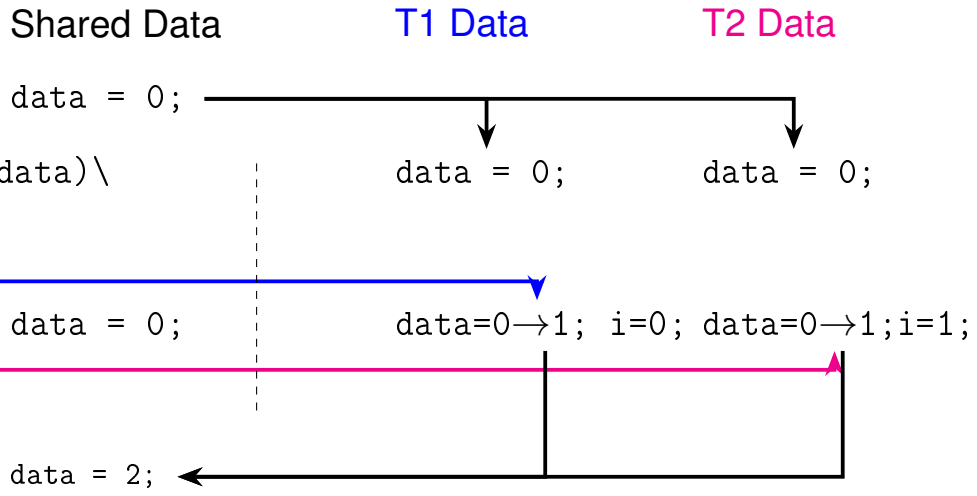
```
data = 0;

data = 0;        data = 0;

data = 0;        data=0→1; i=0; data=0→1;i=1;

data = 1;
```

OpenMP: Data Sharing

Reduction

Code

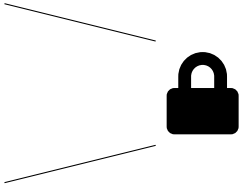
```
int data = 0;

#pragma omp parallel for reduction(+:  data)\

for(int i = 0 ; i < 2; i++) {

    data++;

}

output(data);
```



OpenMP: Synchronization

Critical

```
...
#pragma omp critical
{
    globalVariable++;
}
...
```



Alternatives: `atomic`, `single`, `master`, `etc.`

Also available: `omp_lock_t`

OpenMP: Synchronization

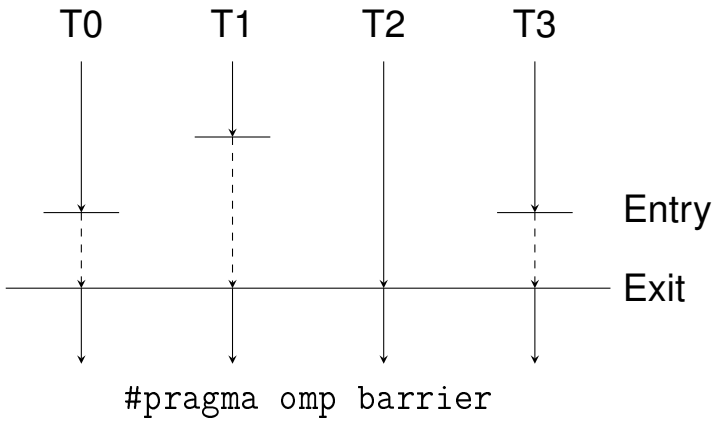
Critical

```
...
#pragma omp critical
{
    globalVariable++;
}
...
```

Alternatives: `atomic`, `single`, `master`, `etc.`

Also available: `omp_lock_t`

Barrier



Moodle Quiz

Moodle Quiz

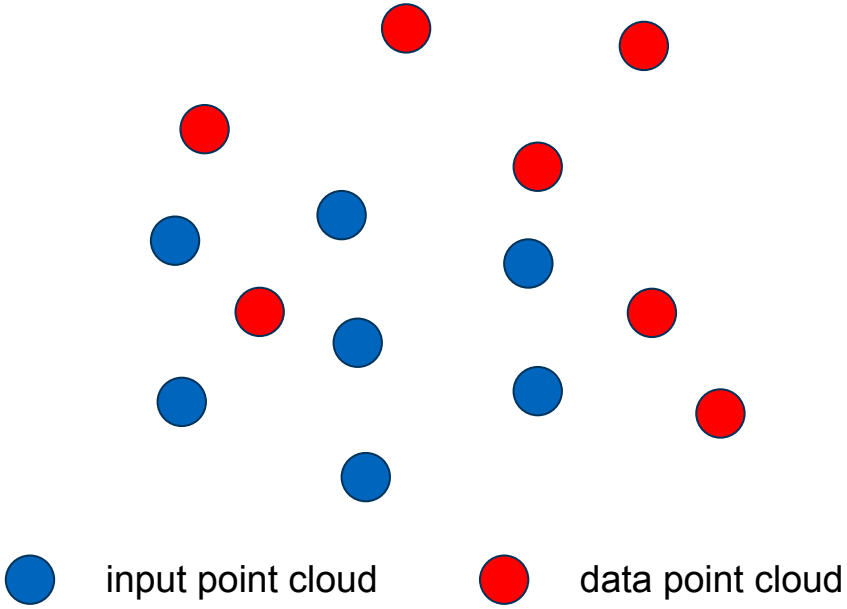
https://www.moodle.tum.de/mod/quiz/view.php?id=2976068

In-Class Exercise


10
9
8
7
6

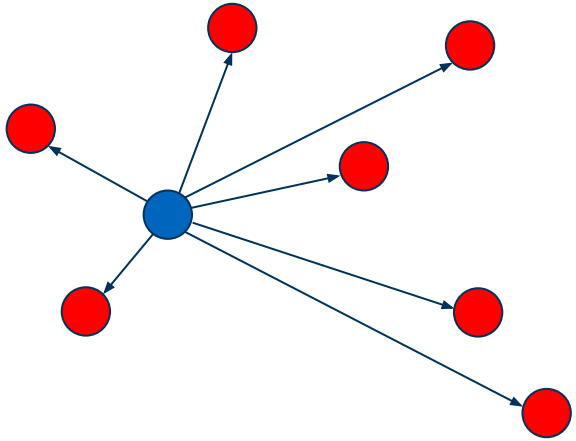
Facebook Harry Potter House Test

Parallelize k-Nearest-Neighbors



Each data point has a class label

Parallelize k-Nearest-Neighbors

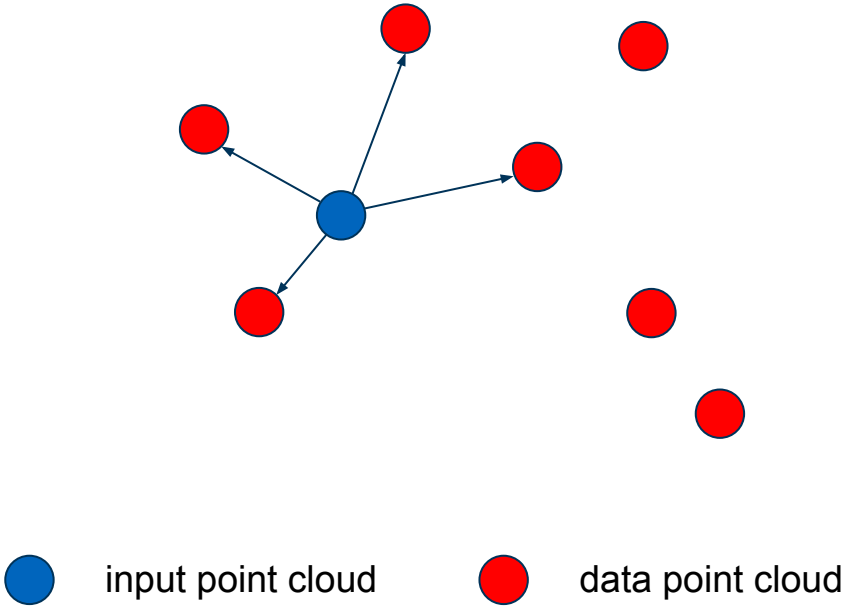


input point cloud

data point cloud

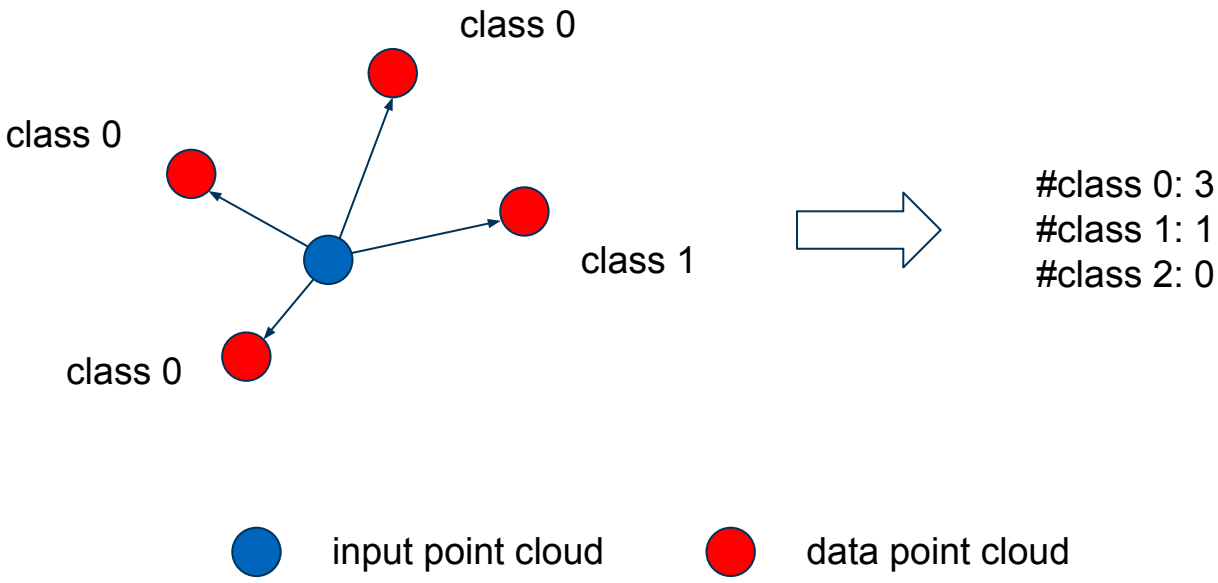
For a point in the input point cloud, find its distance to all the points in the data point cloud

Parallelize k-Nearest-Neighbors



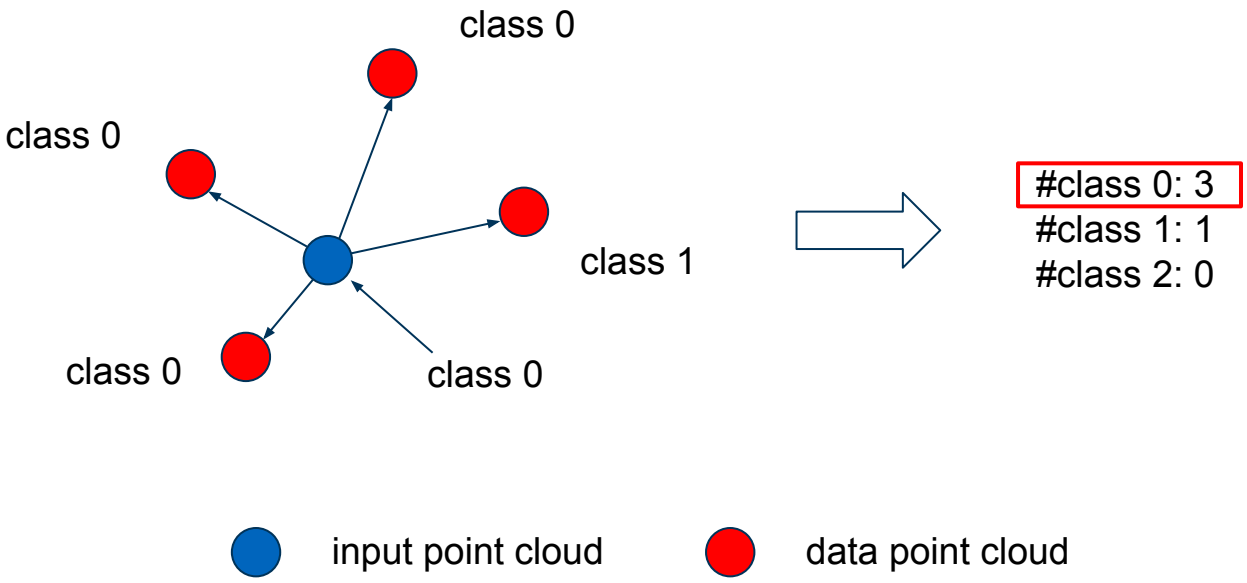
Find the nearest k points in the data point cloud
(here k=4)

Parallelize k-Nearest-Neighbors



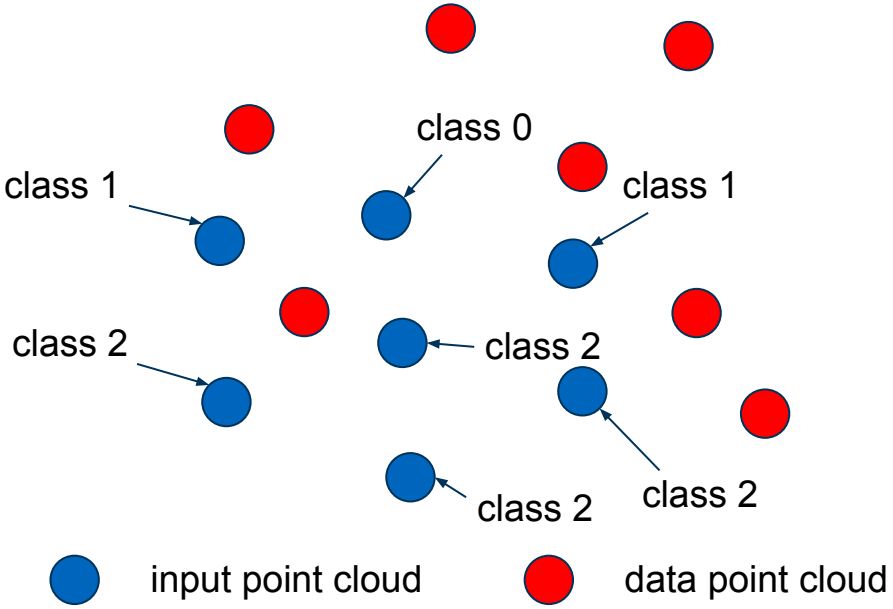
count the number of neighboring points belonging to each class (here only consider 3 classes)

Parallelize k-Nearest-Neighbors



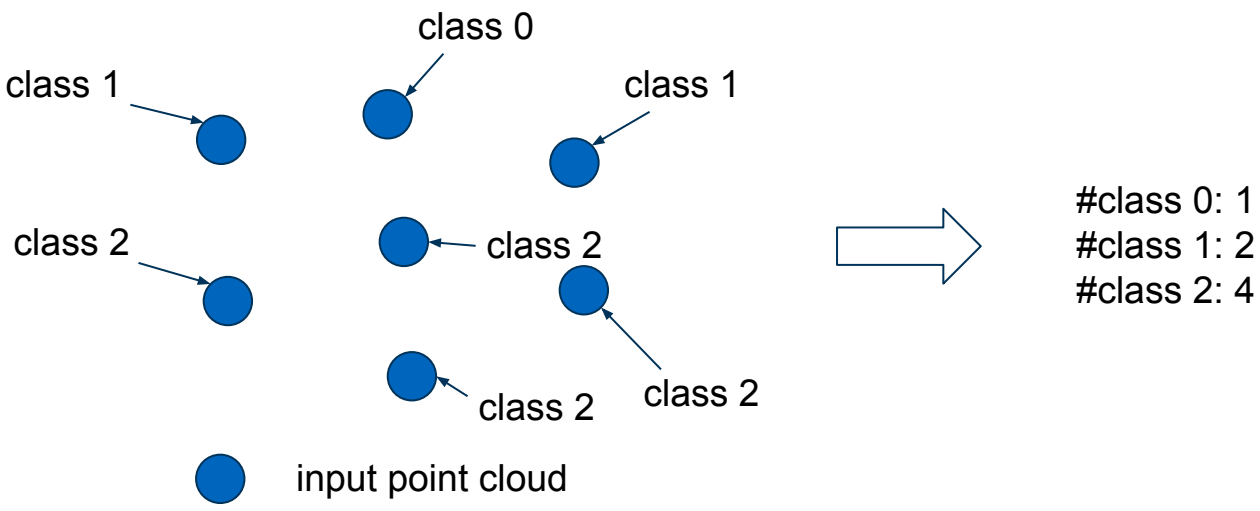
assign the class label that is owned by most neighboring points to the input point

Parallelize k-Nearest-Neighbors



repeat the above process for every point in the input cloud and assign class labels for them

Parallelize k-Nearest-Neighbors



count the number of points that belong to each class
and print as output

Where to find the exercise ?

- Go to the following repository to get the exercise: https://gitlab.lrz.de/lrr-tum/teaching/parprog/ss2024/published-assignments
- If you haven't done this before, use git to clone the exercise to your local machine:

```
cd your_folder
git clone https://gitlab.lrz.de/lrr-tum/teaching/parprog/ss2024/published-assignments.git
```

- Use git pull to get the repository updated

```
cd <repository_folder>
git pull
```

- Go to folder in-class-3 for this week's task, you can also find a README.md there

In-class exercise 3: use OpenMP

- You will find the sequential code in student_submission.cpp
- Your tasks
  - Use the correct OpenMP pragma to parallelize for loop(s)
  - Add the correct arguments to the pragma(s) above
  - Use the correct OpenMP pragma to synchronize shared variable access
- Achieve a speedup of 15 (on the server)
- Our server has 16 cores with 2 way hyperthreading (i.e. 32 threads)

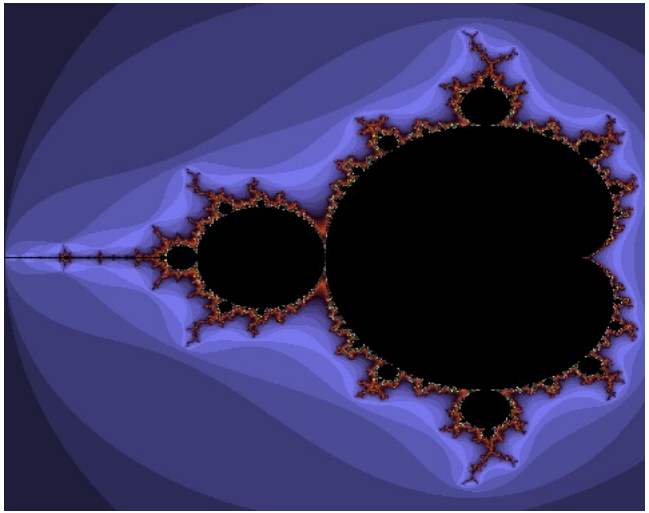
Homework

Homework - Mandelbrot

- Use OpenMP to parallelize `mandelbrot_draw()` :

```
 void mandelbrot_draw( ...  some args ) {
  ...
  for (int i = 0; i < y_resolution; i++){
    for (int j = 0; j < x_resolution;
j++){
      embarrassingly parrallel calculation
of pixels
      ...
} } }
```

- Problem solved : $Z_{n+1} = Z_n^2 + C$



More on Mandelbrot Set : https://mathworld.wolfram.com/MandelbrotSet.html

Homework - Mandelbrot

Speed up Requirement :

- Your solution should have a speedup ≥ 16

Build the program

- Makefile:
  `make`

Usage of the program

- Sequential:
  `./sequential_implementation -r 480x380 -i 1000 -f mandelbrot.ppm`
- Parallel:
  `./student_submission -r 480x380 -i 1000 -f mandelbrot.ppm`

Recap & Questions

Covered today:

- `#pragma omp parallel for`
- Schedules
- Sharing
- Synchronization

Questions

This slide is intentionally left blank.