Introduction to Parallel Computing
Problem Assignment #2

Mandelbrot Set

Presentation & Demo

Yihong Gu
Tsinghua University
yihong15.math@gmail.com

Outlines

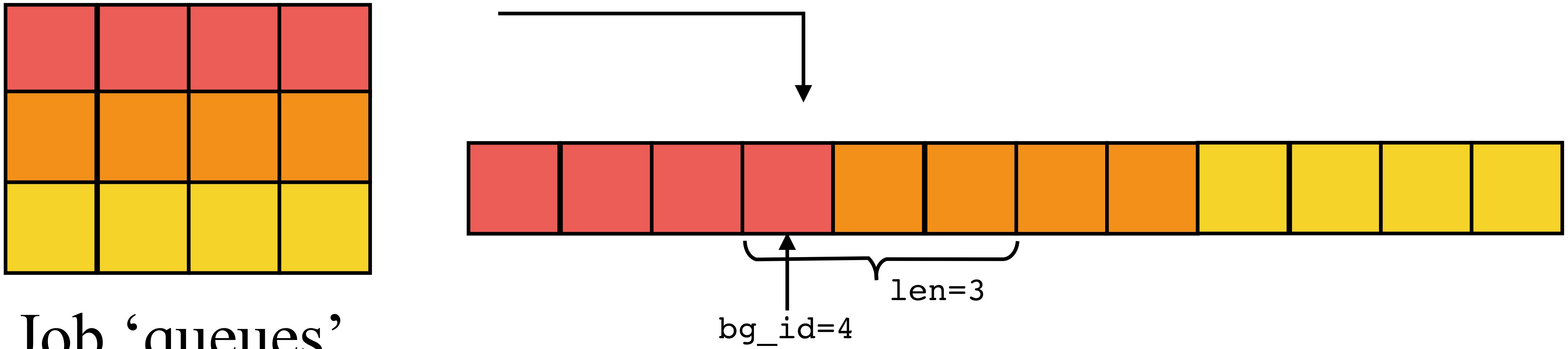
- Hierarchical Master-Slave Architecture
  - Data Preprocessing
  - Dynamic Scheduling using MPI
  - Bootstrap
- Optimization
  - Bulb Check

Outlines

- Hierarchical Master-Slave Architecture
  - Data Preprocessing
  - Dynamic Scheduling using MPI
  - Bootstrap
- Optimization
  - Bulb Check

Data Preprocessing

- ‘Flatten’



- Job ‘queues’
  - message of chunk (2 ints): (bg_id, len)
    - e.g. (4, 3)

Dynamic Scheduling using MPI

Master:

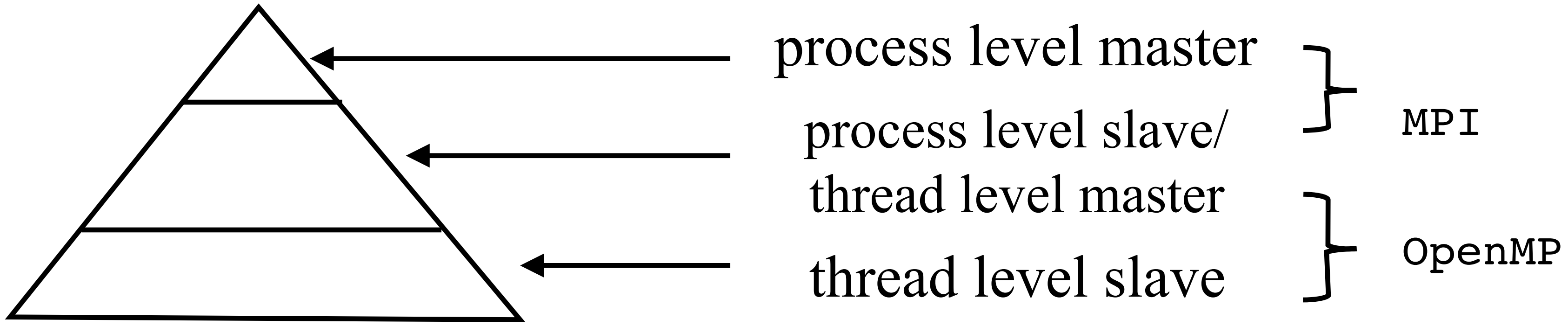
1. Send initial messages to all slaves (MPI_Recv)
2. Receive messages from all slaves (MPI_Isend)
   - `source=MPI_ANY_SOURCE`
3. (Suppose receive message from slave i)
   - Check whether all jobs have been done
     - Yes: send overall finish message
     - No: allocate a new job chunk (MPI_Isend)
4. Repeat [2]-[3] until finished

Slave:

1. Receive a job chunk message (MPI_Recv)
2. Do job using OpenMP
3. Send a job chunk finish message to Master
4. Repeat [1]-[3] until receive a overall finish message from Master

Hierarchical Dynamic Scheduling

- Hierarchical Dynamic Scheduling
  - `PROCESS_CHUNK_SIZE=(Width+Height)*5`
  - `THREAD_CHUNK_SIZE=50`



process level master

process level slave/
thread level master

thread level slave

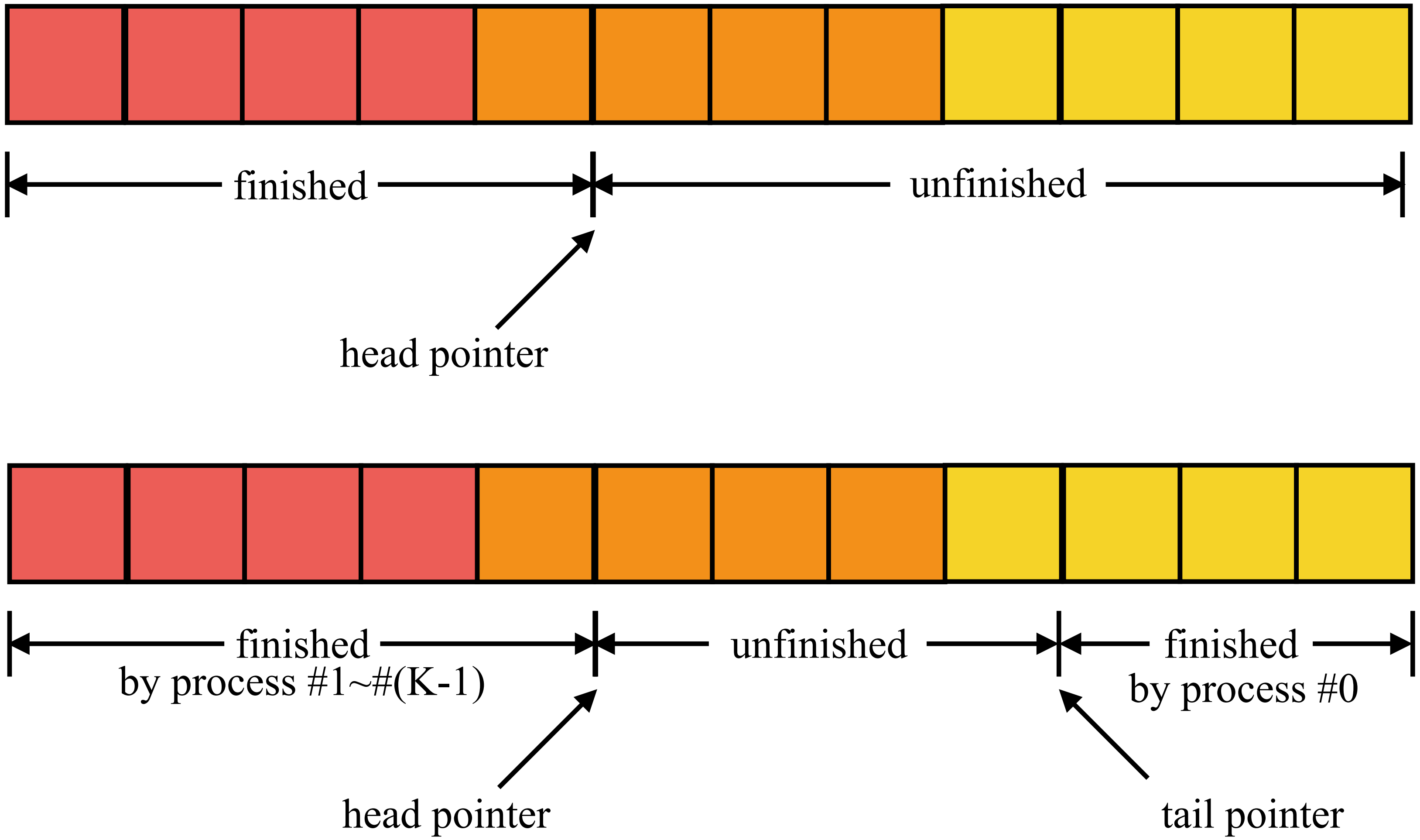
MPI

OpenMP

- The implementation of ‘master’
  - Process ? rank=0 -> master
    - waste
  - Thread ? rank=0 + PThread
    - communication problem: one process can’t using MPI_Recv and MPI_Isend at the same time.

Bootstrap

- Recall job ‘queue’
-



Bootstrap (cond)

Master:

1. Send initial messages to all slaves (MPI_Recv)
2. Receive messages from all slaves (MPI_Isend)
   - `source=MPI_ANY_SOURCE`
3. (Suppose receive message from slave i)
   - Check whether `head` = `tail`
     - Yes: send overall finish message
     - No: allocate a new job chunk (MPI_Isend)
4. Repeat [2]-[3] until finished

Slave #0:

1. Change the `tail` pointer and get job chunk itself
2. Do job using OpenMP
3. Repeat [1]-[2] until `head` = `tail`

Slave #1 ~ #(K-1): remains same

Bootstrap (cond)

Master:

1. Send initial messages to all slaves (MPI_Recv)
2. Receive messages from all slaves (MPI_Isend)
   - `source=MPI_ANY_SOURCE`
3. (Suppose receive message from slave i)

   `mutex #0`

   - Check whether `head` = `tail`
     - Yes: send overall finish message
     - No: allocate a new job chunk (MPI_Isend)
4. Repeat [2]-[3] until finished

Slave #0:

`mutex #0`

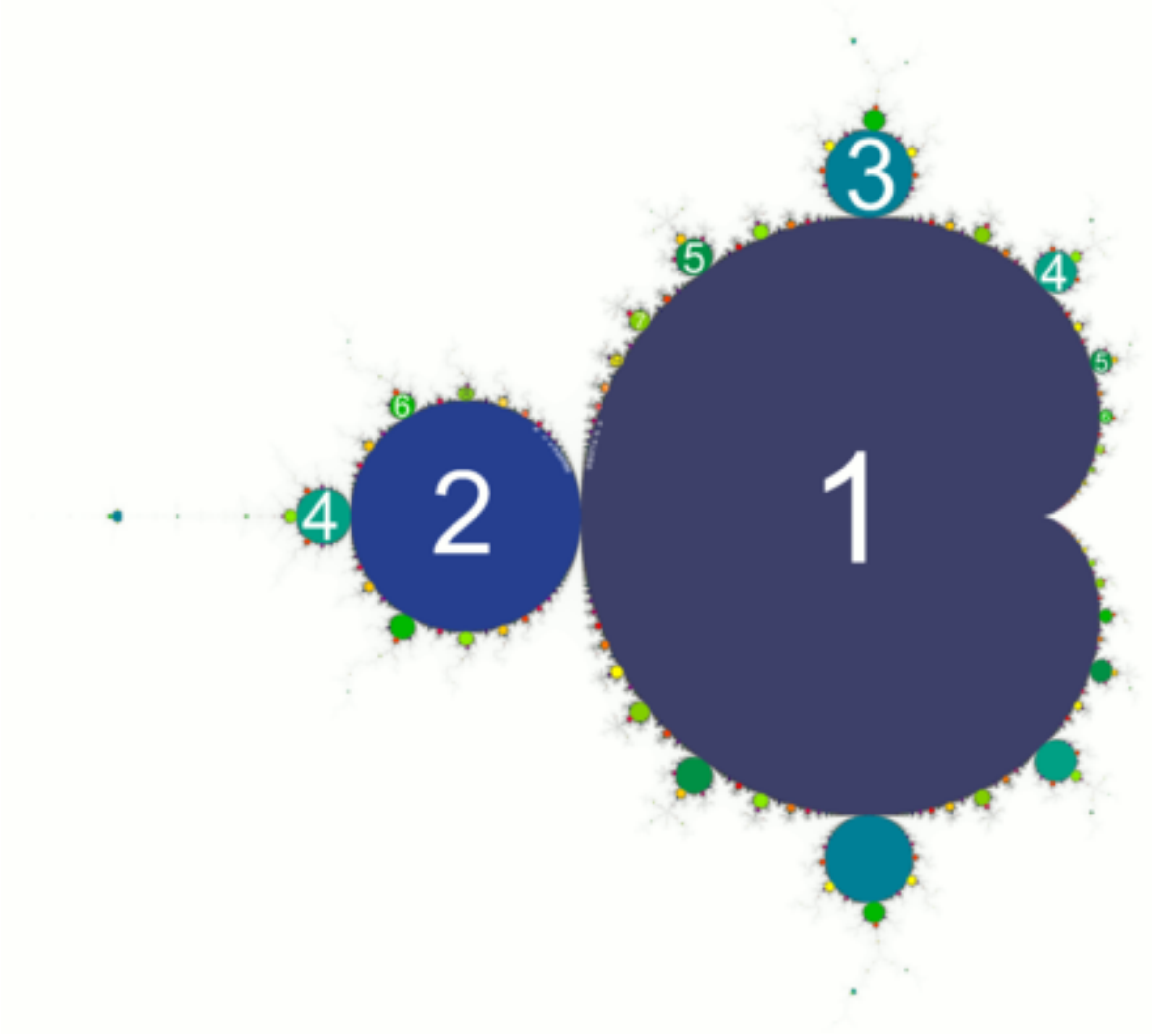
1. Change the `tail` pointer and get job chunk itself
2. Do job using OpenMP
3. Repeat [1]-[2] until `head` = `tail`

Slave #1 ~ #(K-1): remains same

Outlines

- Hierarchical Master-Slave Architecture
  - Data Preprocessing
  - Dynamic Scheduling using MPI
  - Bootstrap
- Optimization
  - Bulb Check

Bulb Check



- Get Analytical Solution of Bulb #1 and #2.

$$q = \left(x - \frac{1}{4}\right)^2 + y^2,$$

$$q\left(q + \left(x - \frac{1}{4}\right)\right) < \frac{1}{4}y^2.$$

Thank You

Q&A