Bitonic Merge Sort

- Name: 盛爾葳
- email: ewinnie.sheng@gmail.com

Implementation

I/O

1. 若array_size不能被number of processes(np)整除，會用`DBL_MAX`將其padding至可以被整除，確保所有processes的分配到的elements數量(chunksize)都相等。

    > 會想到用padding是因為之前只有測資2、6、7過不了，且實在找不出什麼邏輯問題，便將其process配置的情況都print出來，發現這三筆剛好都有大小不等的chunks，於是決定直接將array_size擴充到np的整數倍，這樣最多也只需要增加31個elements (因為np最大是32)，對效能幾乎沒影響。

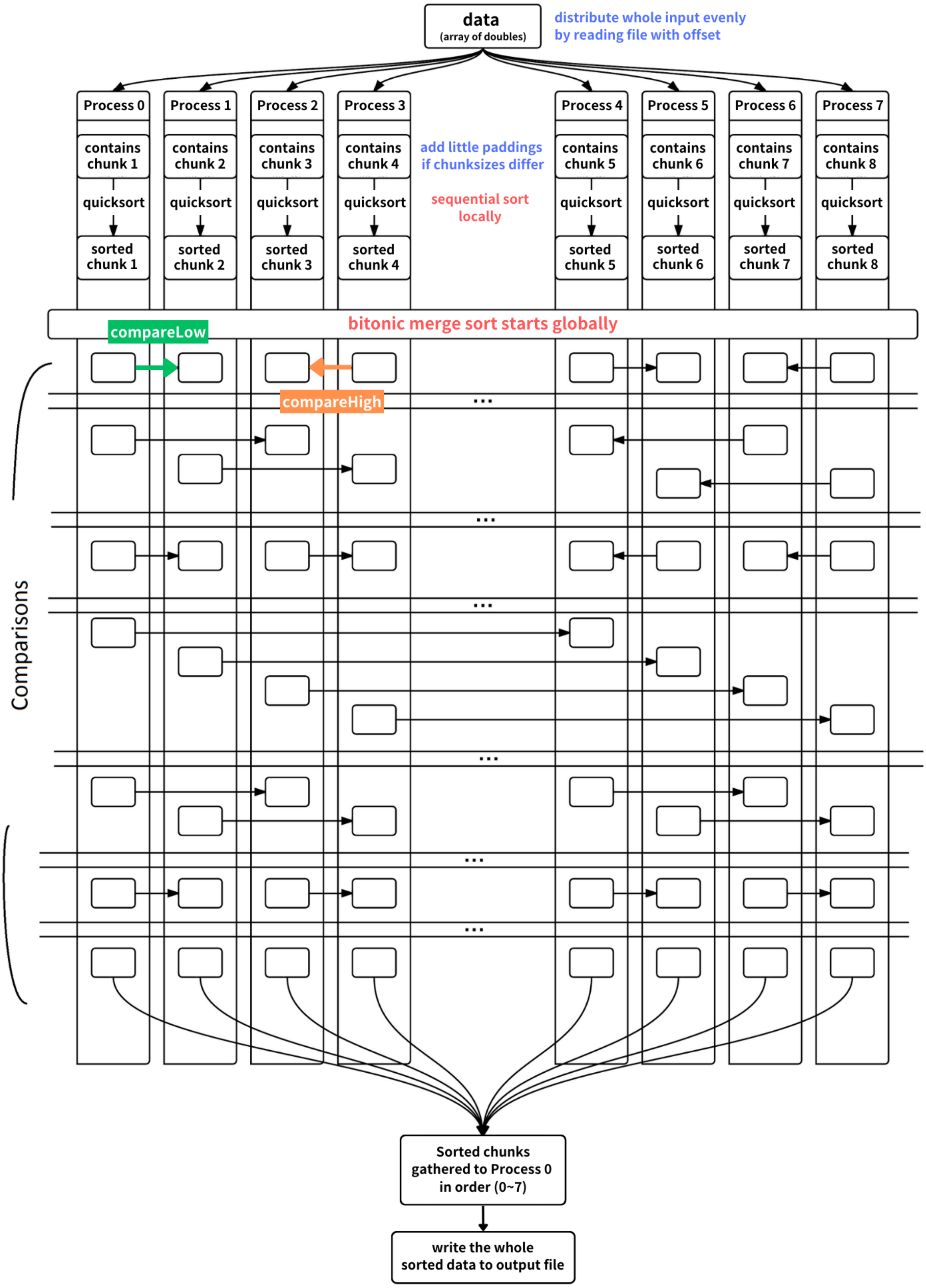
2. 接著是IO平行化，計算出每個process的間隔(display)就可以用`MPI_File_read_at` 讀取各自負責的區段。

    整個sorting流程(local+bitonic)結束後，所有chunks在rank 0 ~ rank np應該已經完成排序，可以用`MPI_Gather`依序集合回rank 0。接著用自己寫的`writeFile`函式過濾掉一開始的padding elements(DBL_MAX)，並以`MPI_File_write_at`將`filtered_chunk`間隔`filtered_size`寫成output檔案。

Sorting logic

吃進來的array被分配到processes之後，每個process裡的chunk先各自完成local的quicksort，接著進入bitonic merge sort，processes之間會比較並交換chunks。因為前面的quicksort一律都是排序成ascending，所以我在compare階段寫了兩個函式compareLow和compareHigh，主要差異界是

1.
2.



Explain your implementation

- Essential
    - **How do you sort in your program?**
    - Present your sorting network in a 8 processes scenario with a diagram

    - How do you handle the input/output items?
- Optional
    - Any tips to improve the performance?

      Experiment & Analysis

Experiment & Analysis

System, Environment Spec

- queue: ct160
- nodes = 1, ppn = 40

  測資最多只用到32個processes，所以一個node就夠了 (台衫一號每個節點有40核心)，跨node計算會花更多時間。
- library: openmpi-4.0.5

Performance Metrics

先用`MPI_Wtime()`計算前後時間差異，再以`MPI_Reduce`得到所有processes的時間總和：

- computing time

  主要考慮local進行的`quicksort`、如果需要padding也會增加時間。
- communication time

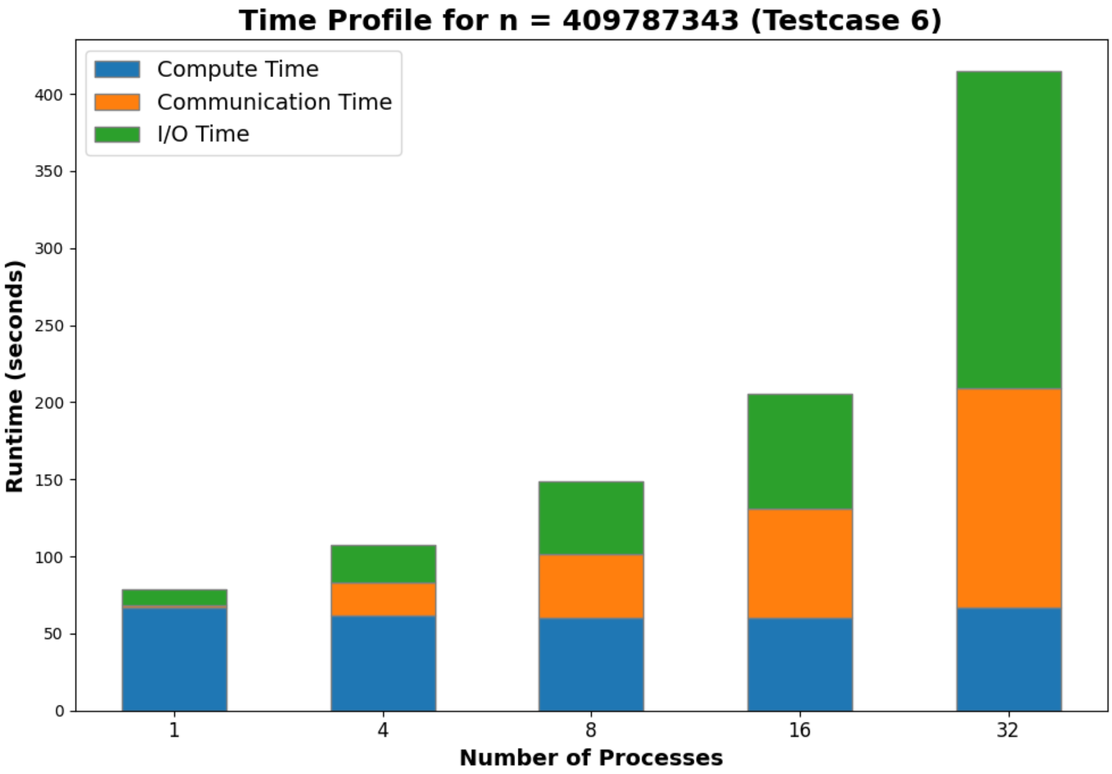
  包含`MPI_Gather`、`MPI_Bcast`，還有整個bitonic sort階段，因為`compareLow`、`compareHigh`有用`MPI_Sendrecv`在processes之間交換資料。
- I/O time

  `MPI_File_open`、`MPI_File_read_at`、`MPI_File_close`、`MPI_File_write_at`
- How do you compute the values in the plots?
- sequential time?

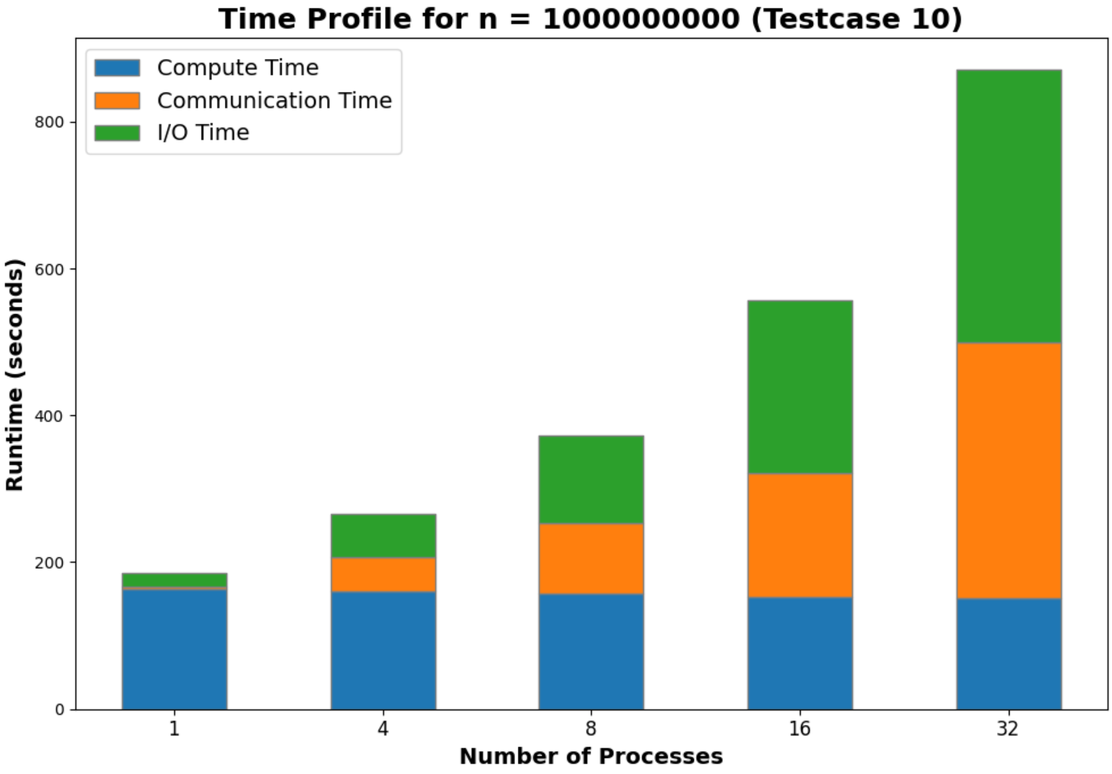
Speedup Factor & Profile

Time profile

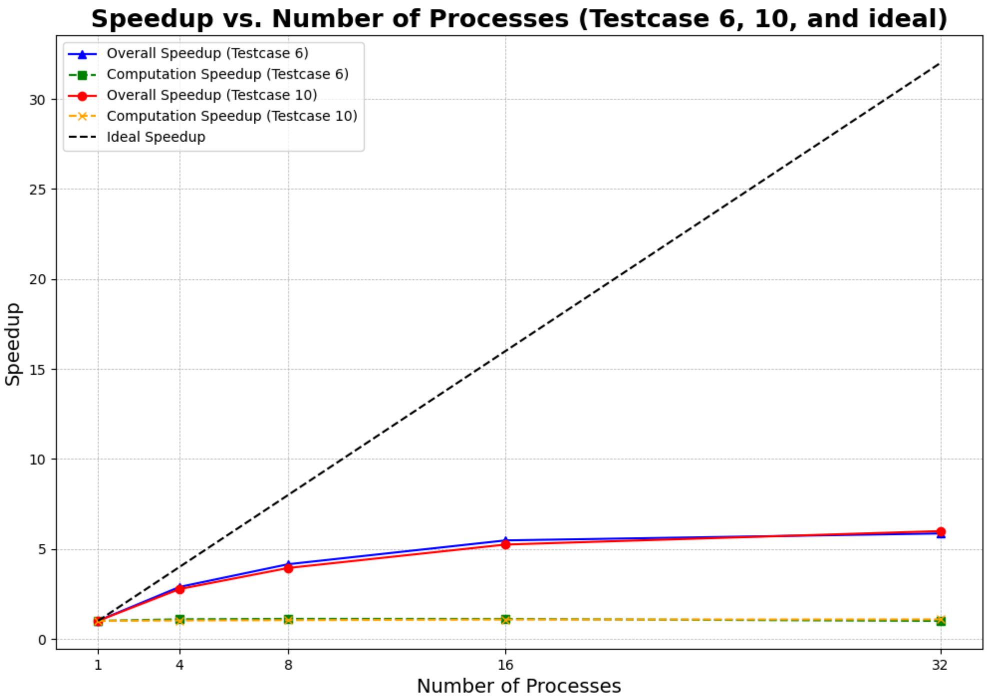
- n = 409789343 (medium size array)



- n = 1000000000 (large size array)



Speedup



- Make sure your plots are properly labeled and formatted
- You can generate your own testcase if meaningful

Experiences & Conclusion

- non-blocking
- exchange pointer
- What have you learned from this assignment?
- What difficulties did you encounter in this assignment?
- Feedbacks
- Others
    - You are encouraged to conduct more experiments and analysis