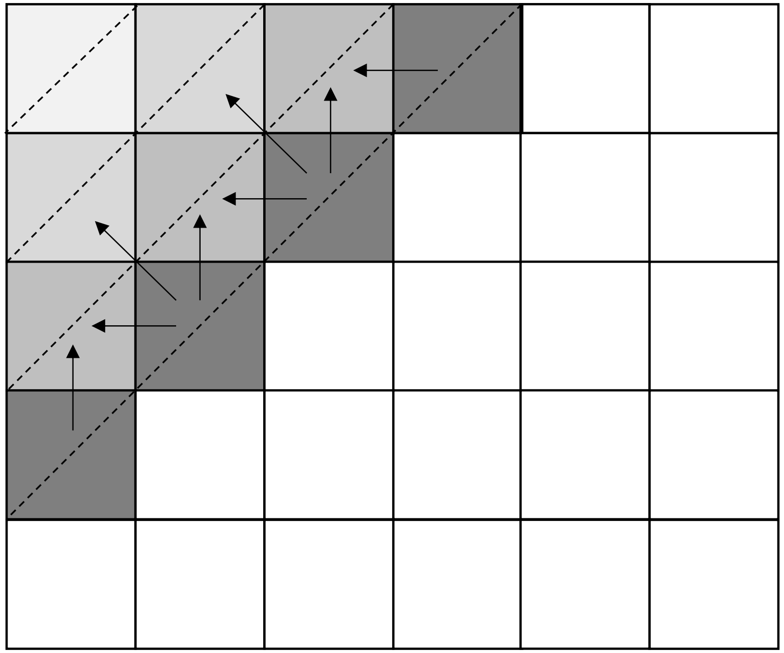
COMP 5112 Tutorial 2

Assignment 1 solution and Pthreads setup

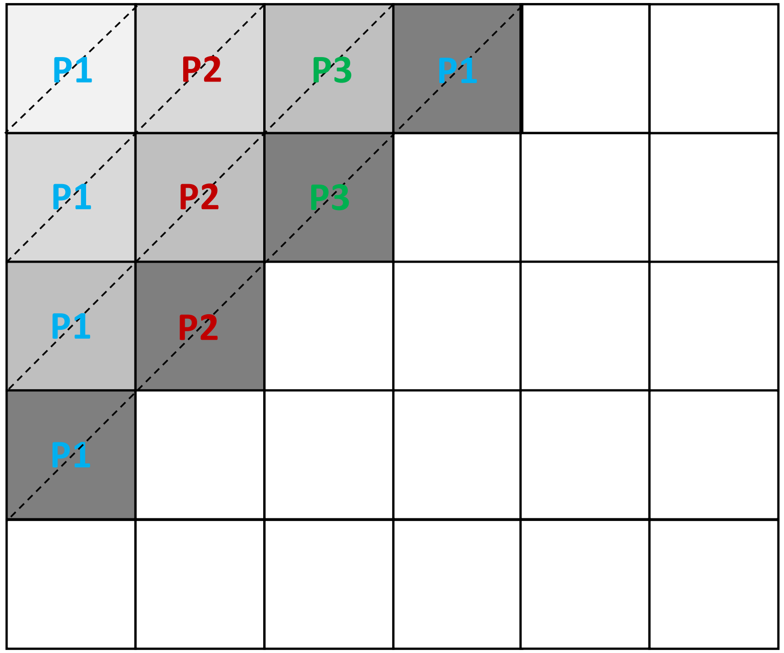
MPI Smith-Waterman: Naïve Solution

- Cells on the same anti-diagonal can be computed in parallel
- Idea: Iterate anti-diagonals, processes compute different parts of the anti-diagonal



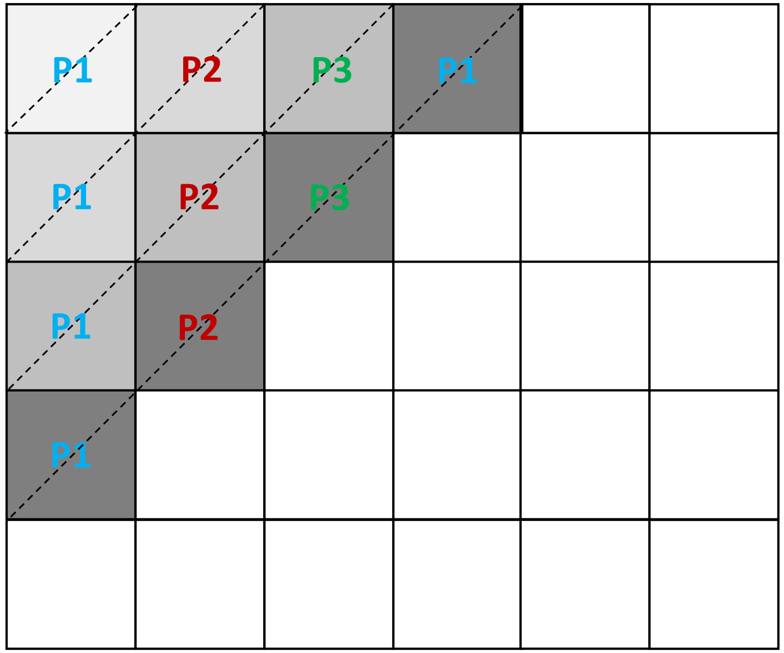
MPI Smith-Waterman: Naïve Solution

- Allocate process-local score matrix
- For each anti-diagonal:
  - Compute with multiple processes
  - Sync changed values (Send/Recv, Broadcast)
- Return max score



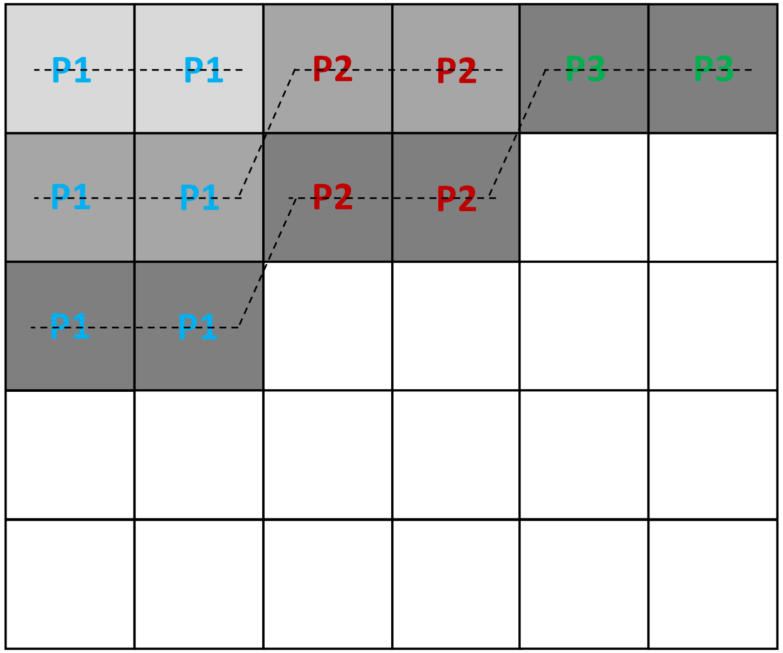
Performance Issues

- High communication cost
  - The entire anti-diagonal needs to be updated in each iteration
- Low memory locality
  - Memory writes are not sequential
  - Low cache hit rate



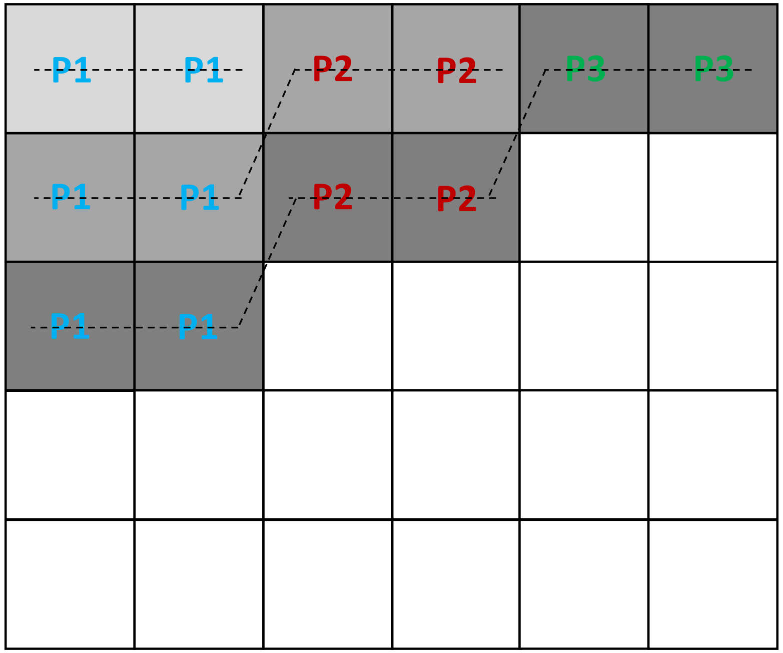
MPI Smith-Waterman: Optimized Solution

- Idea: Divide each row into p (# processes) blocks
- In each iteration, one process computes one block



MPI Smith-Waterman: Optimized Solution

- Low communication cost
  - Only need to send/recv less than p cells in each iteration
- High memory locality
  - Each process is doing sequential write



Tips

- Make sure to test your program on CS Lab 2 machines before submission. Compile error -> 0 points.
- Correctness is the requirement for efficiency.

Pthreads

```
# Compile

g++ -std=c++11 -lpthread main.cpp pthreads_smith_waterman_skeleton.cpp -o pthreads_smith_waterman

# Run

./pthreads_smith_waterman <input file> <number of threads>

# Example

./pthreads_smith_waterman sample.in 4
```