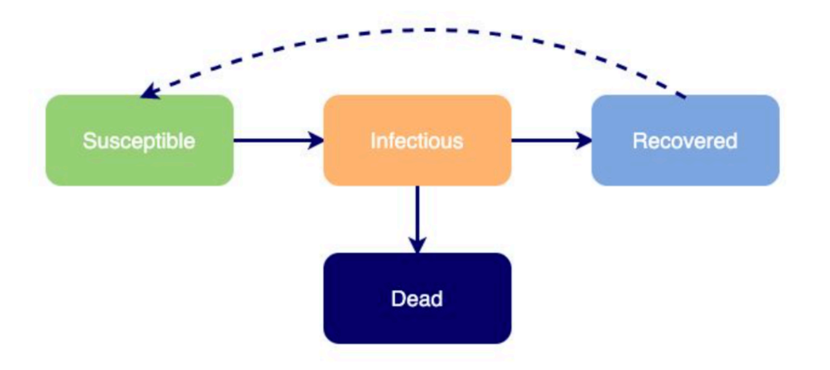
Parallelize Massively Transmission Simulations of COVID-19 Diseases

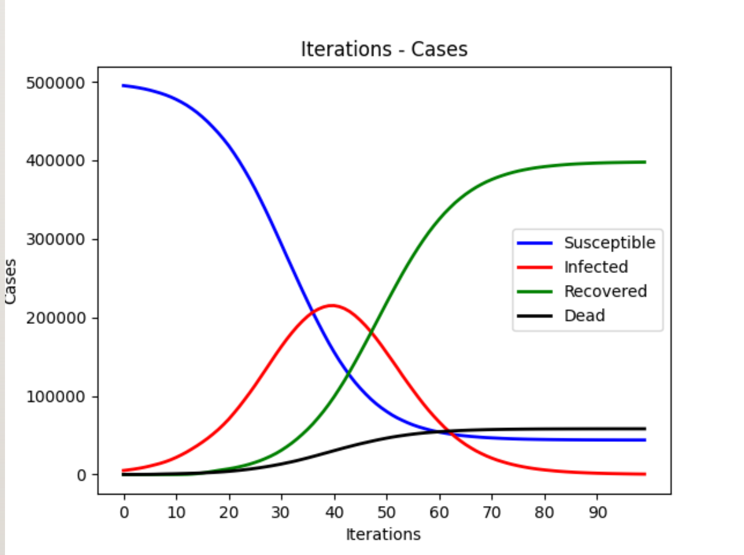
TEAM 12 106062307 林奕鑫 106062322 江岷錡

Preface & Motivation

SIR Model (**S**usceptiable, **I**nfectious, **R**ecovered)

- Epidemiological Model
- Compute # of the infected in a closed population





Challenges & Limitation

```
For iterations:
  For node1 in allNodes:
    CheckWhetherInfectious(node1)
    For node2 in allNodes:
      CheckWhetherSusceptible(node2)
      // Euclidean Distance
      distance = CalculateDistance(node1, node2)
      if (distance < infectious_radius) {
        prob = CalculateInfectiousRate(node1, node2)
      }
    UpdateInfectedNode()

  For node in allNodes:
    move(node)
```

$$F(x) = a * exp(-b * x) \quad (0 <= x <= r)$$

$$a, b : infectious\_param \quad r : infectious\_radius$$

Computation-Intensive Workload

Computation overhead dreasitically increases when # of node increase

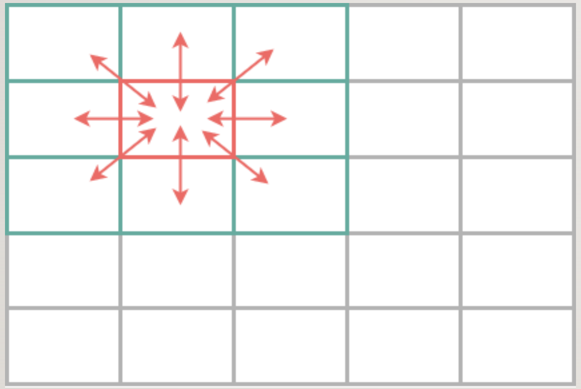
Iteration Dependency

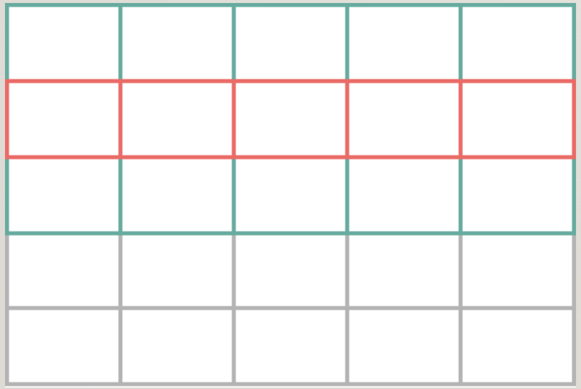
Hard to parallize between phases and load balancing

Randomly Distribution of Nodes

No common data split method to parallelize

Blocked-Version Parallelism with OpenMP





Blocked Version

- Split map into blocks
- Make width/height of block = infectious radius

  => Only need to consider nearby 9 blocks for each block
- Only parallelize within iteration
- One thread maps to one blocks
- Lock Awareness? Synchronization?

Row-Regioned Version

- Split map into rows
- One thread maps to one rows

Optimization

Blocked-Version Parallelism with OpenMP

Data Computation Parallelism

Use OpenMP Thread library

Collapse 2 layers for-loop in blocked version

Prevent synchronization with "nowait"

Both Spatial & Temporal Locality

Reduce the mem-copy time

Increase the cache hit

Lock-Free with Backward State Assignment

Reduce the lock contention & fully parallelize

Utilize row-based split method

Reduce # of times that thread context switches

CUDA VERSION IMPLEMENTATION

Distance & Infection Rate Calculation

Pair-Wise Calculation of Distance & Infection Rate

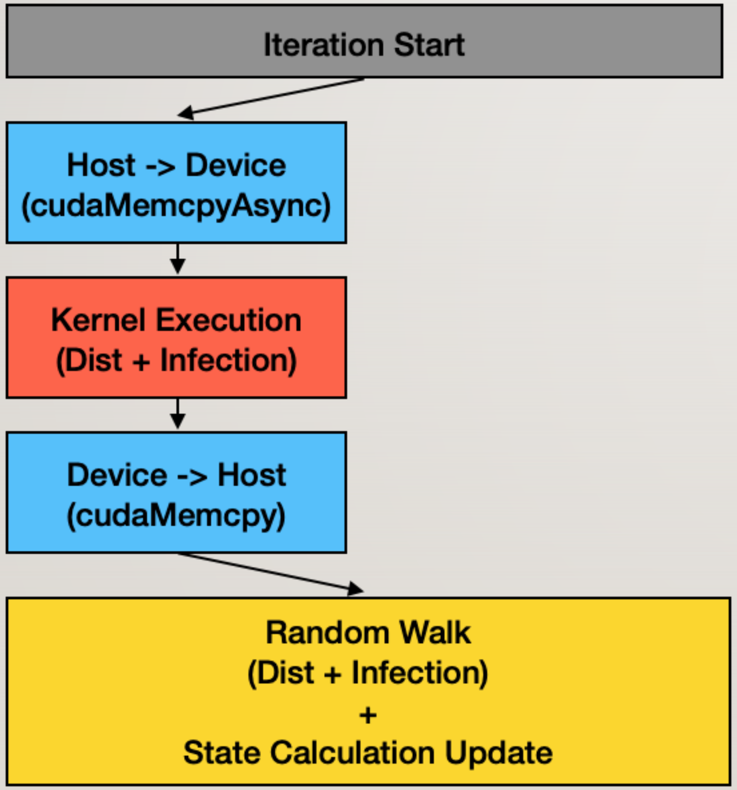
Memory Concern

Tradeoff Between Memory and Speed

GPU Related Optimization

Specific Technique about Cuda Implementation

Blocked-Version on GPU



Pro:

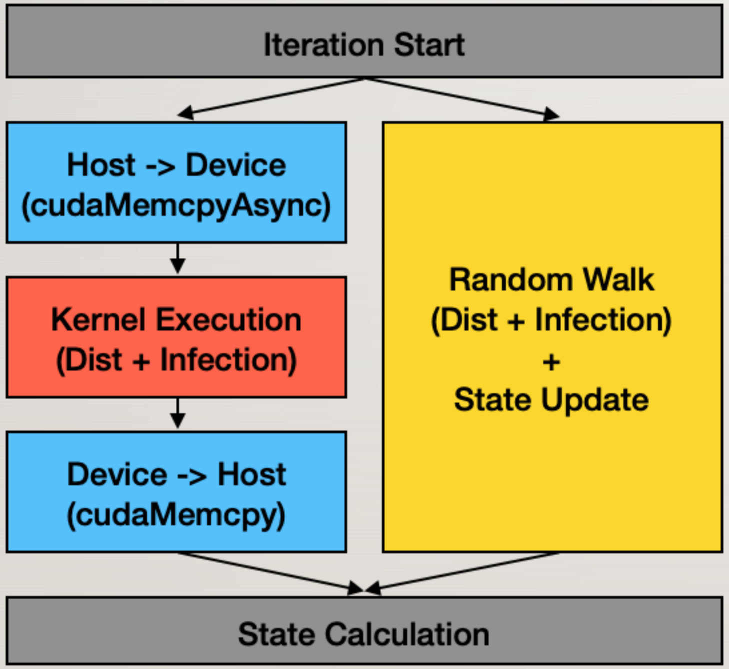
- Row Region Version – Reduced Caculation Workload
- Minimize Memory Space Utilization

Con:

- Complex Data Structure
- Moving Logic Add More Burden On CPU Calculation
- Map Data Structure Should be Locked to Avoid Race Condition
- Can't Overlap Computation Between CPU & GPU

Implementation – Version 2.0

Adjacency Matrix on GPU



Pro:

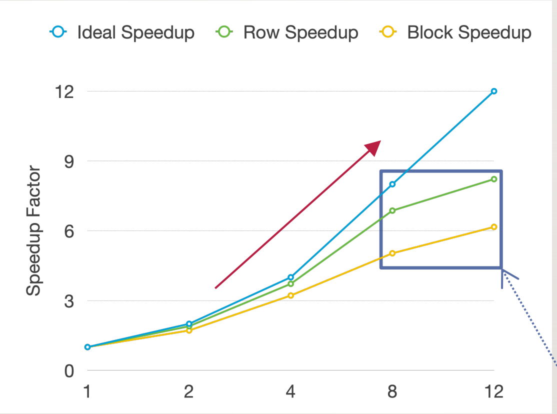
- Calculation Overlapping Between CPU & GPU
- OpenMP Integration in CPU Part
- Less Branch Diverge on Kernel Execution
- Coalesced Memory Access Pattern

Con:

- Memory-Consuming – Can Only Simulate Roughly 35000 People on GTX1660 Super (6G)
- Unnecessary Calculation – Each Pair of Node Will be Calculate Twice

Experiment

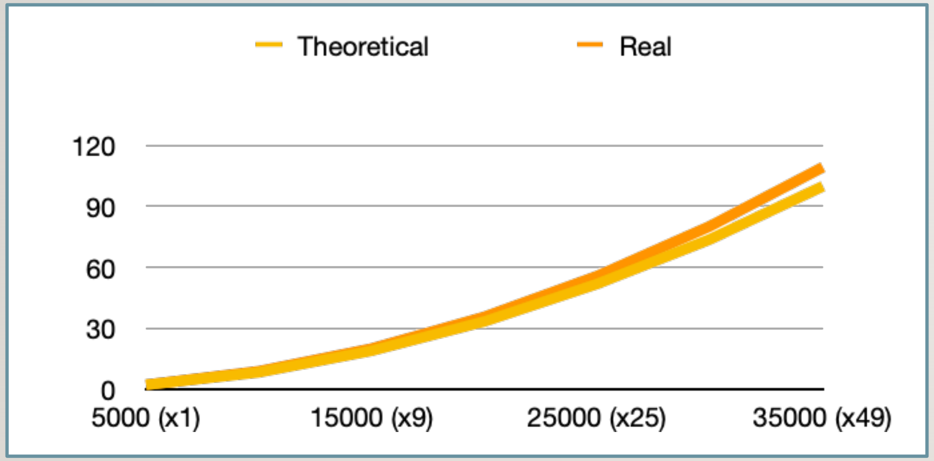
OpenMP Parallelism for SIR Model



- Larger data block > smaller data block when data randomly distributes
  - Greater load balancing when data block is bigger
  - But when data is too intensive, the straggler thread would emerge
  - Needs to find balancing way to get optimal load balancing result
- Gradually converge when # threads > 8

Experiment

CUDA Profiling & Scalability



Time Complexity:

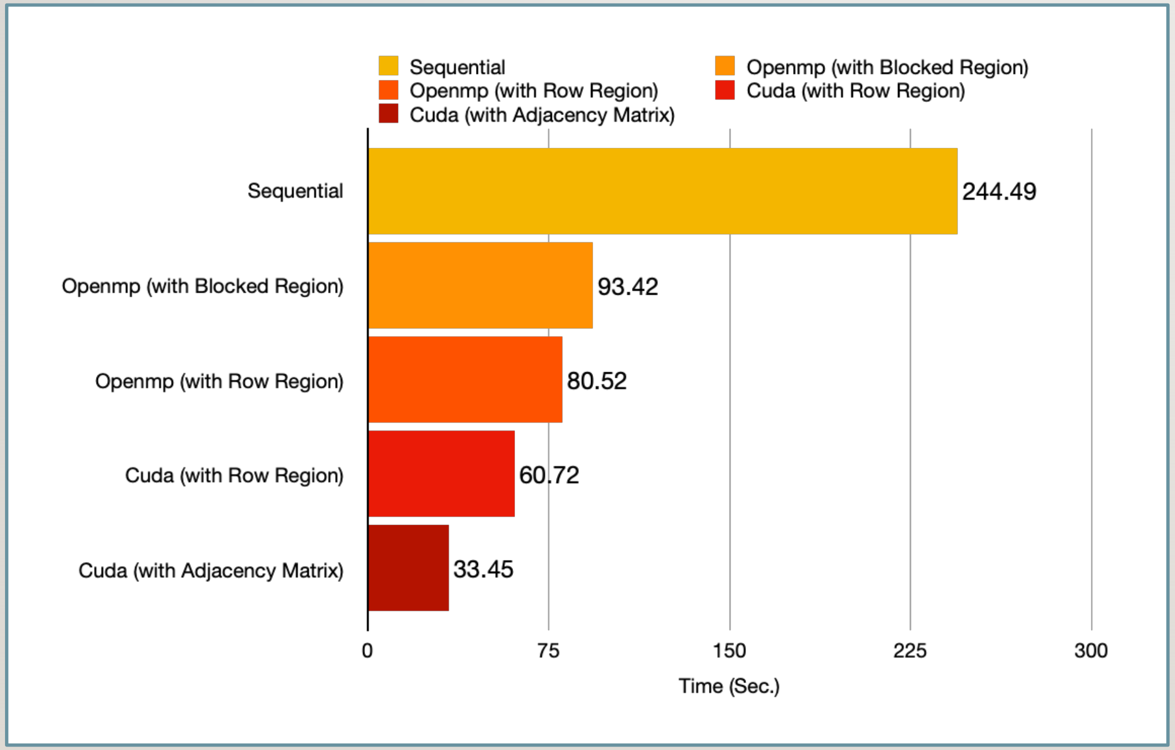
- O(V^2)
- Some Overhead:
  - Memory Copy – Between Device & Host
  - Data Structure Implementation
  - Code Structure

Profiling Result

- Global Load Throughput: 107.63GB/s
- Achieved Occupancy: 0.775715
- SM Efficiency: 99.98%
- Global Store Throughput: 344.41GB/s

Experiment

Cuda + OpenMP Performance



Overall Performance:

- Node Count: 25000
- GTX 1660 SUPER (1408 cuda cores)
- AMD Ryzen 7 3700X 8-Core Processor (16 Thread)
- clock_gettime with MONOTONIC

Reference

- How simulation modelling can help reduce the impact of COVID-19[J]. Journal of Simulation. Currie C S M, Fowler J W, Kotiadis K, et al. 2020: 1-15.
- A Time-dependent SIR model for COVID-19 with Undetectable Infected Persons. Yi-Cheng Chen, Ping-En Lu, Cheng-Shang Chang, Tzu-Hsuan Liu. 2020
- Mathematical models of SIR disease spread with combined non-sexual and sexual transmission routes. Joel C.Miller. 2017
- Simulations for epidemiology and public health education[M]//Operational Research for Emergency Planning in Healthcare: Volume 2. Huang C Y, Tsai Y S, Wen T H. 2016
- A simulation model of the epidemiology of urban dengue fever: literature analysis, model development, preliminary validation, and samples of simulation results[J]. Focks D A, Daniels E, Haile D G, et al. The American journal of tropical medicine and hygiene, 1995, 53(5): 489-506.
- Directed-graph epidemiological models of computer viruses[M]//Computation: the micro and the macro view. Kephart J O, White S R. 1992: 71-102.