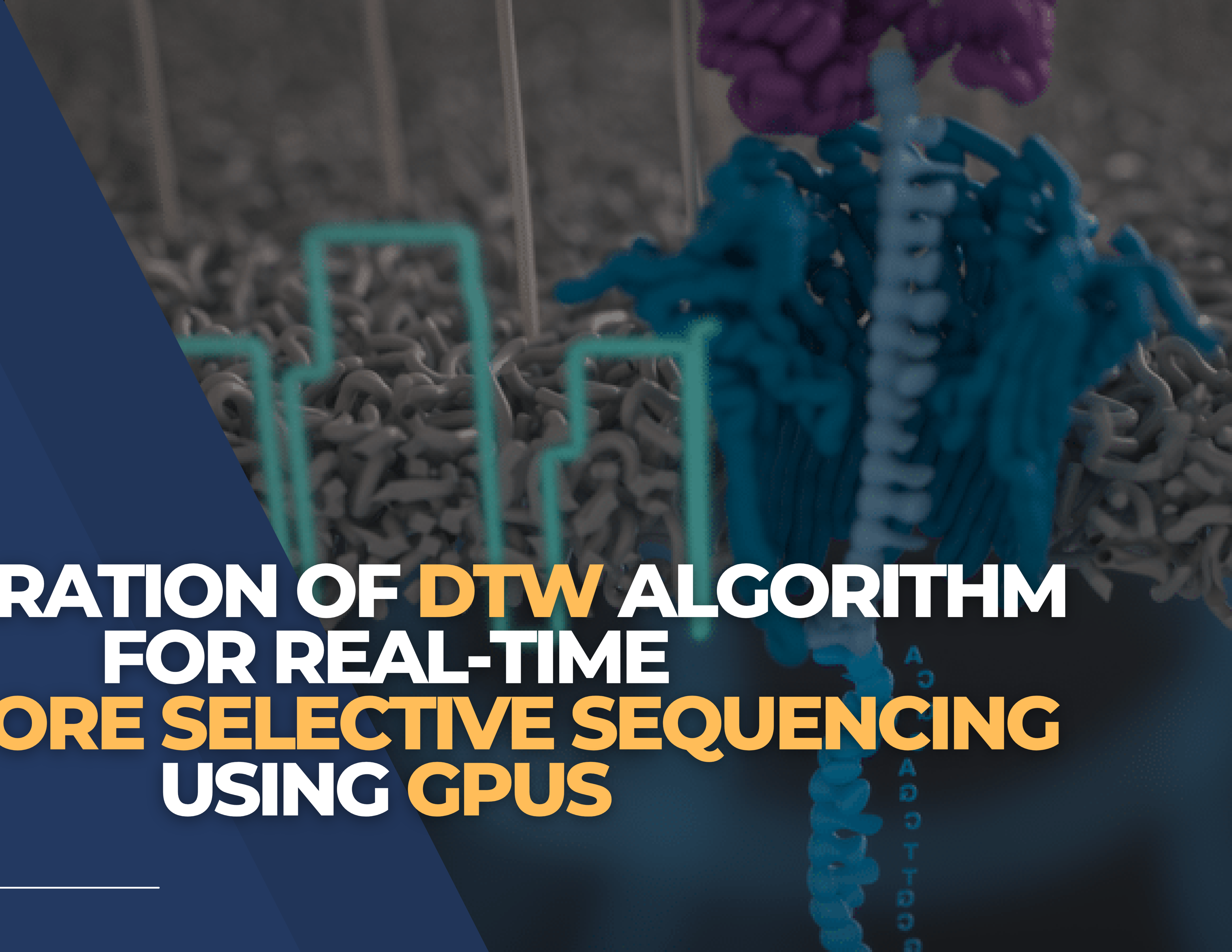
GROUP 18



ACCELERATION OF DTW ALGORITHM FOR REAL-TIME NANOPORE SELECTIVE SEQUENCING USING GPUS

TEAM MEMBERS

E/16/089
Denuke Dissanayake

E/16/313
Maneesha Randeniya

E/16/360
Nipun Dewanarayana

SUPERVISORS

Prof. Roshan
Ragel

Dr. Hasindu
Gamaarachchi

BACKGROUND

- **Nanopore sequencing** is a unique, scalable technology that enables direct, real-time analysis of **long DNA or RNA** fragments.

- Monitoring changes to an electrical current as nucleic acids are passed through a protein nanopore.

- The resulting Electrical signal is decoded to provide the specific DNA or RNA sequence.

- Modern nanopore sequencers offer **selective sequencing** capability.

BACKGROUND

- **DTW ALGORITHM**
    - measures the **optimal alignment** between signals
    - Time and Space complexity - **$O(n^2)$**
    - starts by building the **distance matrix.**
    - Next, **finds the alignment path** that runs through the cost matrix's low-cost areas

PROBLEM DEFINITION

- DTW algorithm has a computational complexity of **O(n^2)** - **High Computational Demand**.
- Because of the high computational demand, portable MinION sequencers **must connect to a large server** to do the analyses.
- Consequently, it will **reduce** the widespread adaptation of selective sequencing in a **portable setting**.
- It will require a **higher time** for the calculation

SOLUTION

- **Implement** the DTW algorithm using GPUs
- **Optimise** the DTW algorithm to Reduce the runtime

EXISTING CODE

- All calculations are done in the **CPU**
- Calculations are done in **sequentially**
- CPU DTW run time
    - Intel i7 10th Gen 16GB RAM - **~27 sec**
    - Kepler Workstation (CE Department) - **~23 sec**

PROPOSED METHOD

- Identifying the critical places requires parallelism.
- Use CUDA for GPU programming
- Use threads in GPU to parallelise the existing code

MILESTONES

- **Identify** the places that need to be **parallelized**.
- Implement a **parallel** mechanism using **CUDA**
- Apply **Optimization** techniques to algorithm

IDENTIFY THE PLACES TO PARALLELIZE

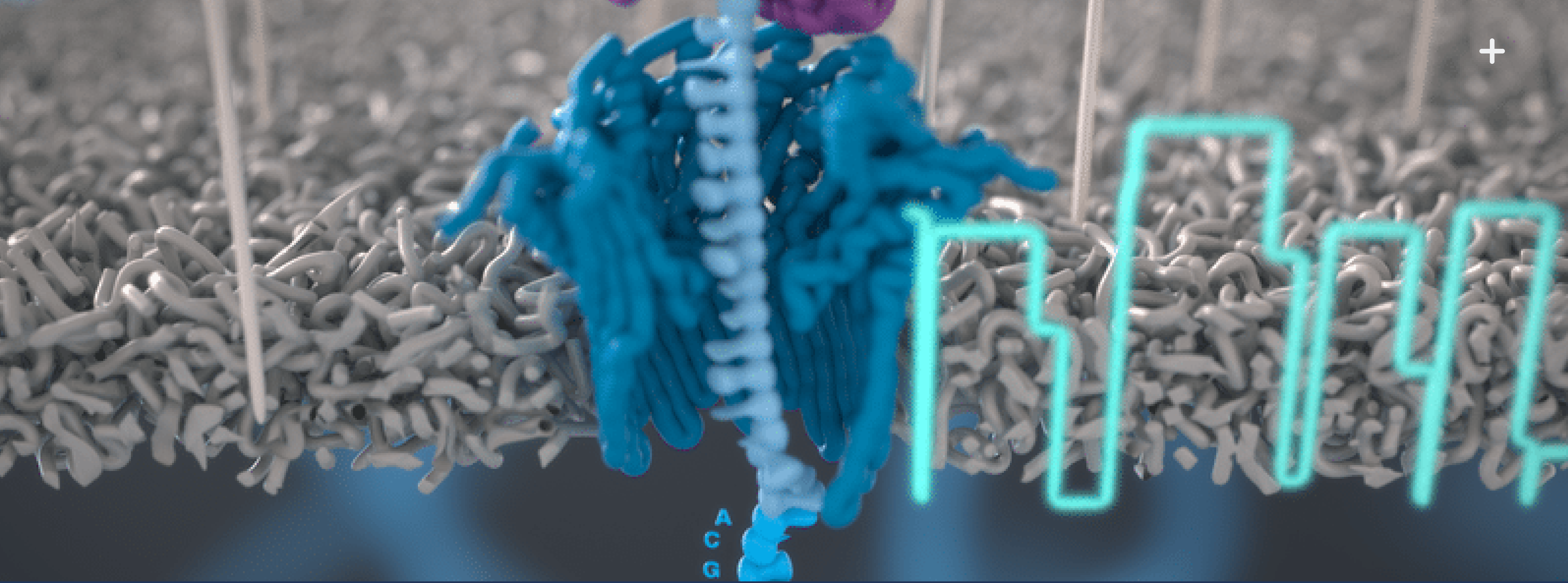
1. **Calling the GPU Kernel**
    - Execute the DTW calculation for multiple samples signals
2. **Matching multiple references**
    - Each sample compare with multiple reference signals
3. **Calculate cost matrix**
    - Calculation of the Cost matrix in the DTW algorithm

2. IMPLEMENT PARALLEL MECHANISM IN CUDA

- Get the data values to arrays
- Allocate the memory in GPU
- Copy arrays to GPU memory
- Execute the Kernel
    - One thread for one signal
- Copy the result back to CPU memory

CHALLENGES

- **Mallocs** inside GPU functions are not working (Do cudaMallocs separately and pass them to the functions).
- **Debugging tools** not working properly in the GPU kernel
- **Restructureing** the existing code.
- **fprintf** not supported inside the GPU kernel



THANK YOU