MST USING PRIM'S ALGORITHM IN CUDA

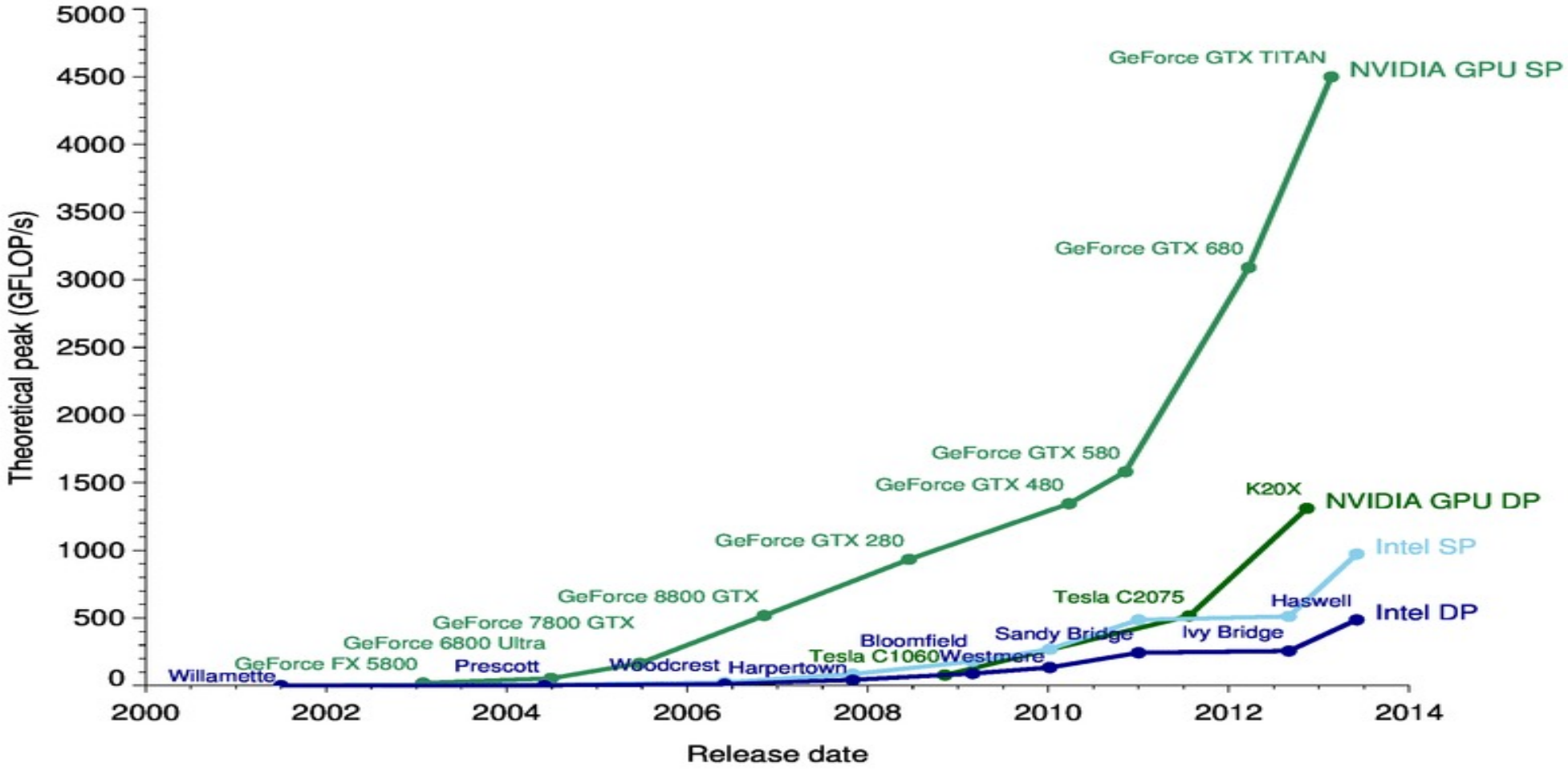
Presented by: Suhas Reddy

University at Buffalo The State University of New York

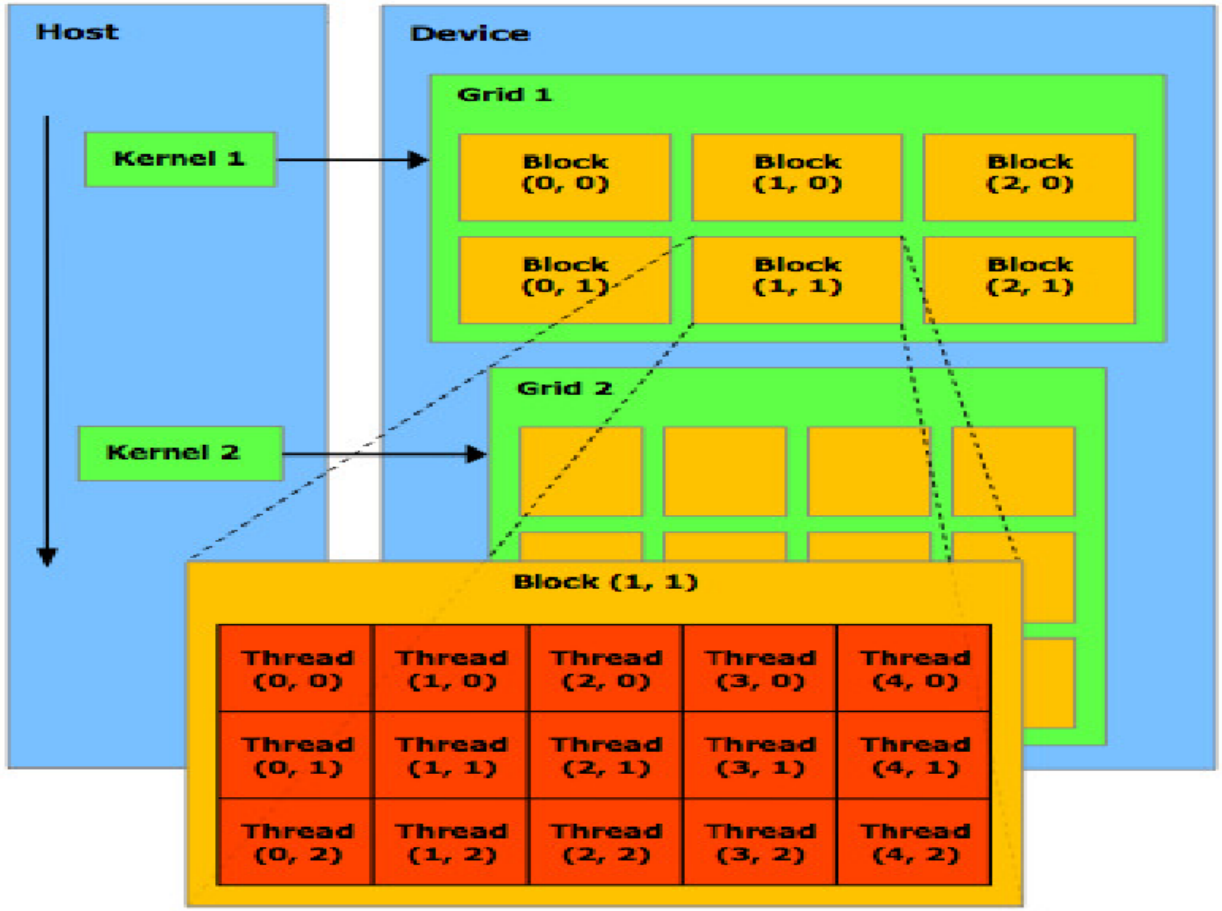
Outline

- CPU vs GPU Evolution
- Problem Statement
- Applications
- Sequential Implementation
- Parallel Implementation
- Results

CPU vs GPU Evolution

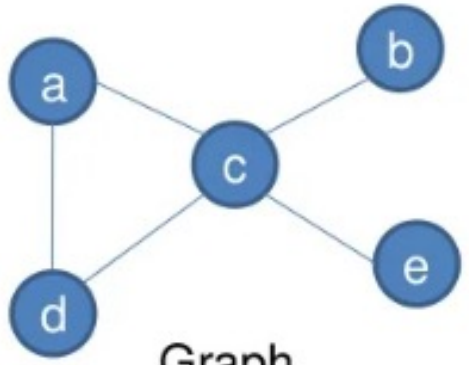


GPGPU Architecture Implementation

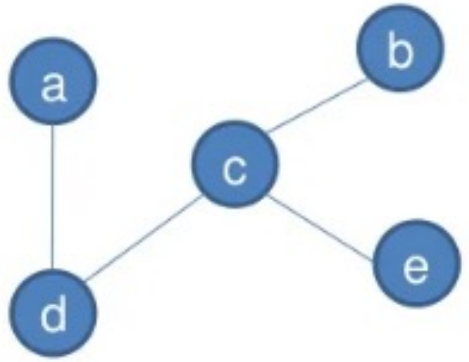


Problem Statement

- Prim's algorithm is a greedy algorithm that finds a minimum spanning tree for a weighted undirected graph.
- Developed in 1930 by Czech mathematician Vojtech Jarnik and later rediscovered and republished by computer scientists Robert C. Prim in 1957 and E.W Dijkstra in 1959.



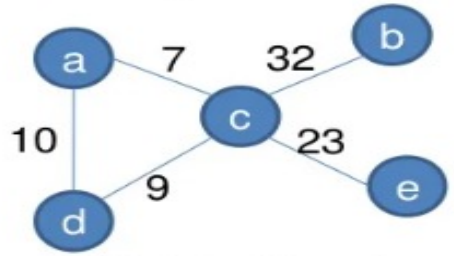
Graph



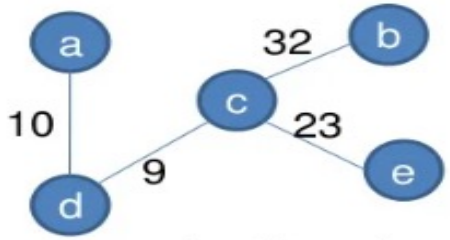
Spanning Tree 1

Problem Statement

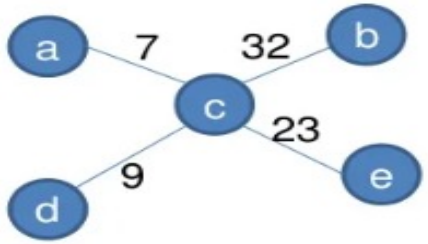
Minimum Spanning Tree in an undirected connected weighted graph is a spanning tree of minimum weight. Example:



Weighted Graph

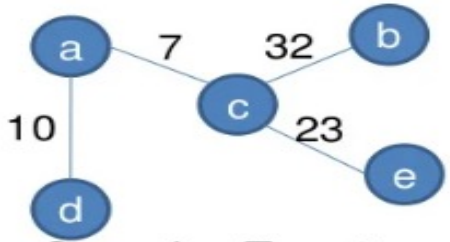


Spanning Tree 1, w=74



Spanning Tree 2, w=71

(Minimum Spanning Tree)



Spanning Tree 3, w=72

Applications

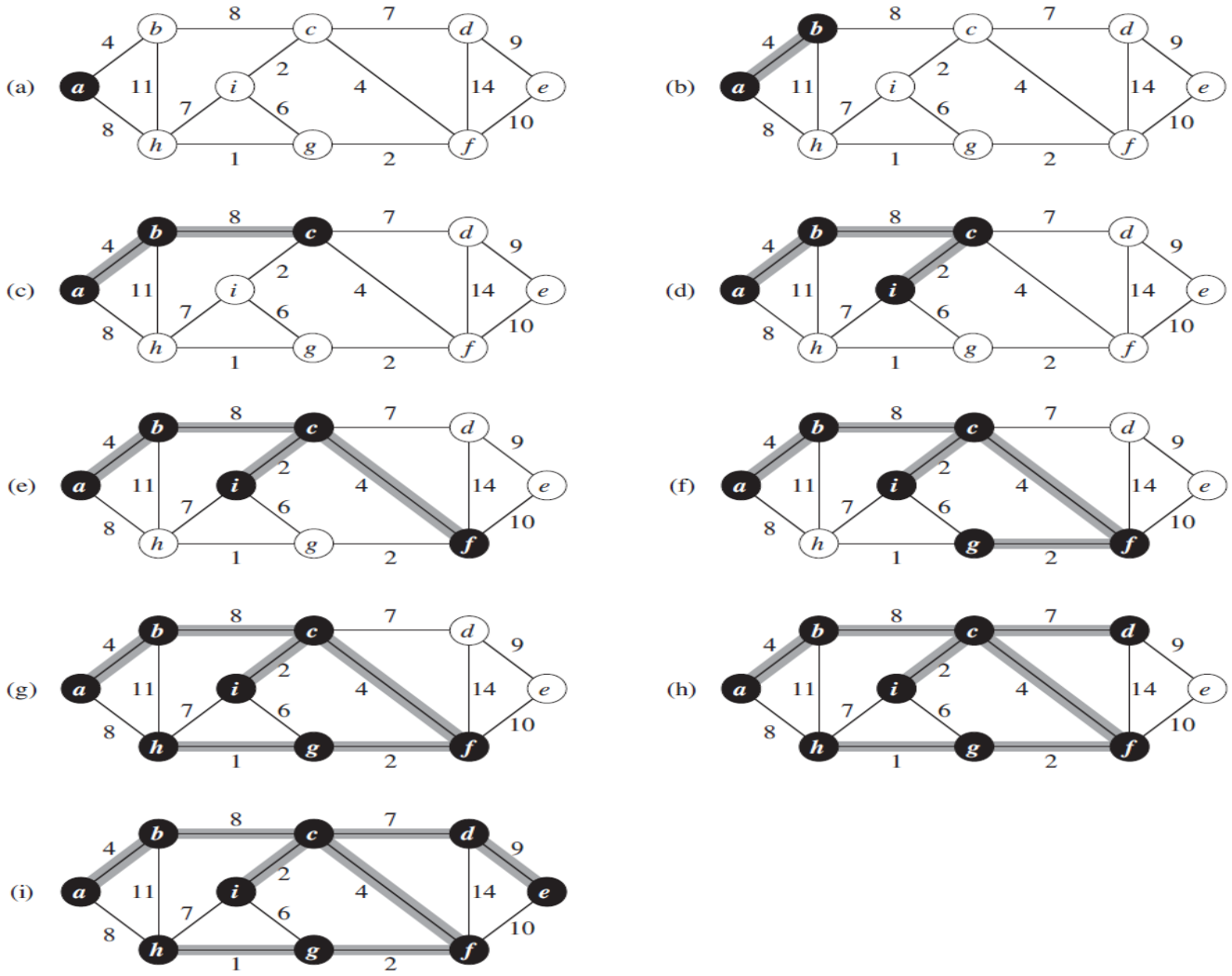
- Routing algorithms
- Clustering
- Travelling Salesman Problem
- Finding airline routes

Prim's Algorithm

MST-Prim(G, w)

```
1:  s ← arbitrary vertex in G
2:  S ← ∅, d(s) ← 0 and d[v] ← ∞ for every v ∈ V \ {s}
3:  while S ≠ V do
4:      u ← vertex in V \ S with the minimum d[u]
5:      S ← S ∪ {u}
6:      for each v ∈ V \ S such that (u, v) ∈ E do
7:          if w(u, v) < d[v] then
8:              d[v] ← w(u, v)
9:              π[v] ← u
10: return {(u, π[u]) | u ∈ V \ {s}}
```

Prim's Algorithm



Prim's Implementation in CUDA Approach

- One straight forward approach is to introduce CUDA directly to the above algorithm.
- We can bring parallelism here exploring the unvisited vertices in parallel leveraging the multi-core processors.
- Parallelly explore all the unvisited edges of the vertices already visited and add the unexplored vertex of the corresponding min edge.

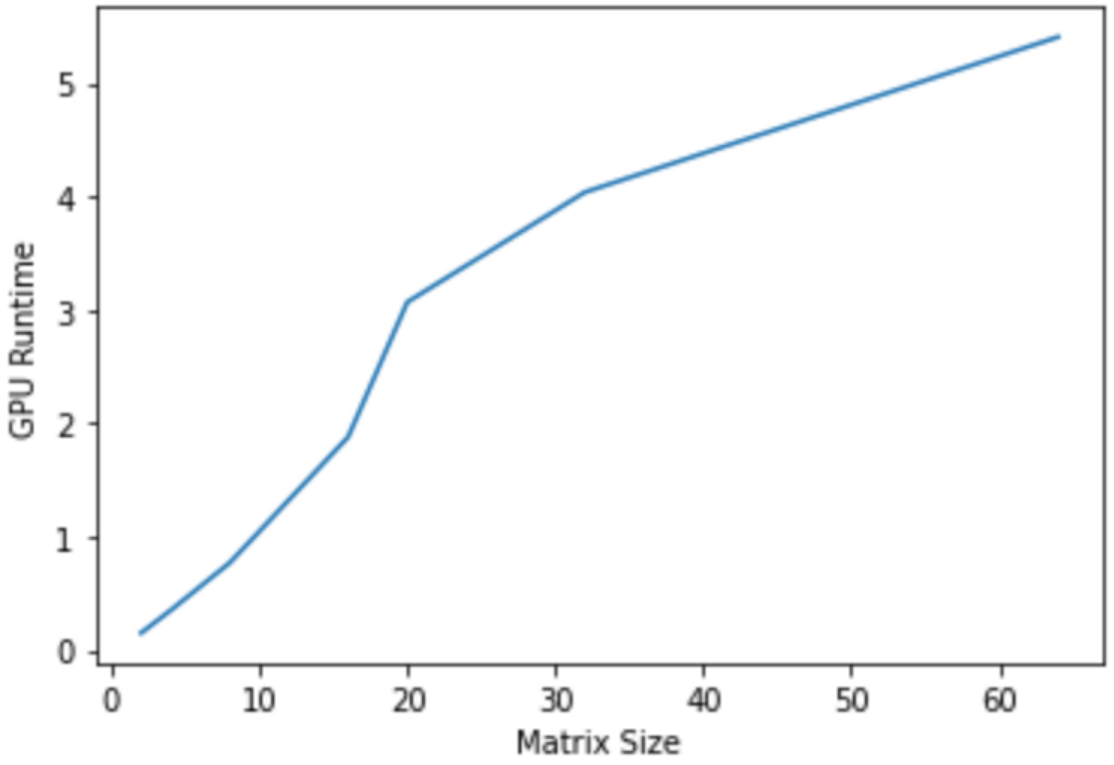
MST-Prim(G, w)

1: $s \leftarrow$ arbitrary vertex in G
2: $S \leftarrow \emptyset, d(s) \leftarrow 0$ and $d[v] \leftarrow \infty$ for every $v \in V \setminus \{s\}$
3: **while** $S \neq V$ **do**
4: $\quad u \leftarrow$ vertex in $V \setminus S$ with the minimum $d[u]$
5: $\quad S \leftarrow S \cup \{u\}$
6: $\quad$ **for** each $v \in V \setminus S$ such that $(u, v) \in E$ **do**
7: $\quad\quad$ **if** $w(u, v) < d[v]$ **then**
8: $\quad\quad\quad d[v] \leftarrow w(u, v)$
9: $\quad\quad\quad \pi[v] \leftarrow u$
10: **return** $\{(u, \pi[u]) | u \in V \setminus \{s\}\}$

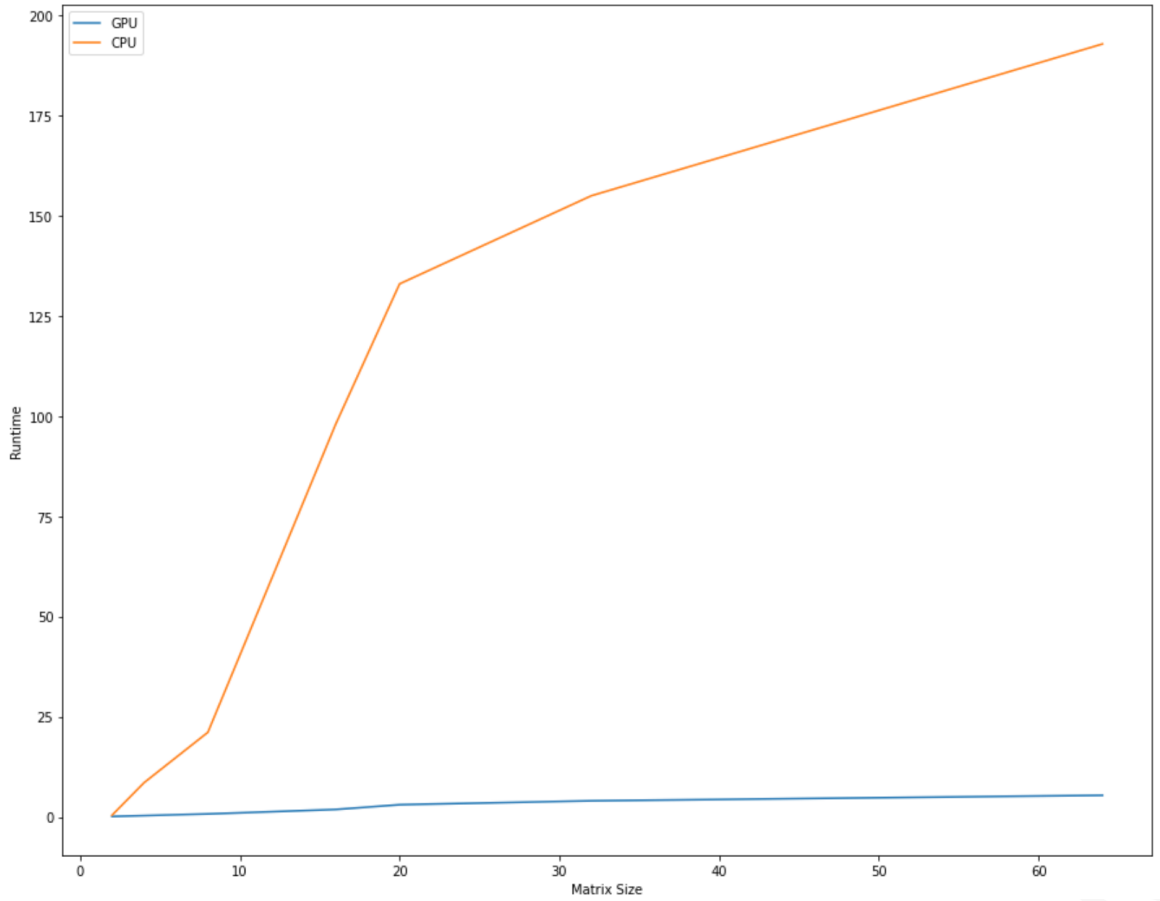
Results

Graph Size (1000 Nodes)	CPU Runtime (s)	GPU Runtime
2	0.45	0.15
4	8.55	0.35
8	21.14	0.77
16	98.01	1.88
20	133.02	3.07
32	155	4.04
64	192.88	5.41

GPU Runtime



Runtime Plot Comparison CPU vs GPU



References

- https://www.mecs-press.org/ijmecs/ijmecs-v3-n4/IJMECS-V3-N4-8.pdf
- https://www.geeksforgeeks.org/prims-minimum-spanning-tree-mst-greedy-algo-5/
- https://developer.nvidia.com/cuda-toolkit
- Prim's Algorithm
- Evolution_GPU_vs_CPU

University at Buffalo The State University of New York

THANK YOU