Finding Vertex Cover: Acceleration via CUDA
Yang Liu, High Performance Research Computing, Texas A&M University
Jinbin Ju, Department of Electrical & Computer Engineering, Texas A&M University
Derek Rodriguez, Department of Electrical & Computer Engineering, Texas A&M University

Introduction

The Vertex Cover Problem

- Classical NP-complete problem (one of the twenty-one Karp’s NP-complete problems)

```
Figure 1 — Examples of Vertex Covers
```

Fixed-Parameter Tractability Algorithm (FPT)

- Parameter k (positive integer) and input size n
- Determines whether a vertex cover of at most k vertices exists or not in time f(k)n where f(k) is independent of n and f is polynomial of n

Our Approach

- Distribute and synchronize computation between CPU and GPU (graph decomposition)
- Synchronize threads in a block
- Apply reduction rules to vertices with degrees greater than k and vertices with degree one

Results

- Tested on graphs created from biological data
- Current implementation is up to 11 times faster than serial program

Purpose & Application

Vertex Cover

Given a graph G, a vertex cover of G is a vertex subset C such that every edge of G is incident to a vertex in C. Given a graph G and a positive integer k, determining whether a vertex cover of size k exists or not in time f(k)n where f(k) is independent of n and f is polynomial of n is known as the Vertex Cover Problem.

```
Given a graph G, a vertex cover of G is a vertex subset C such that every edge of G is incident to a vertex in C. Given a graph G and a positive integer k, determining whether a vertex cover of size k exists or not in time f(k)n where f(k) is independent of n and f is polynomial of n is known as the Vertex Cover Problem.
```

Techniques

Branching Process

- Pick a vertex v (has degree more than k)
- Two branches
  - |G| = |G’| + k
  - |G’| = |G| + k

```
Figure 2 — Branching Process
```

Distribution of Computation

- Input: G and k
- Output: Branch recursively until no such vertex covers exist

```
| Input: G and k | Output: Branch recursively until no such vertex covers exist |
```

Synchronization of Computation

- Copy subgraphs to GPU
- CUDA memory synchronizes copy on separate stream
- Concurrent kernel execution on separate stream
- Full GPU state
- Thread mapped memory
- Synchronization among threads in a block
- Branch memory

Results & Conclusions

```
Table 1 — Program Running Times
```

Future Research

```
Future research might focus on extending this work to other graph problems, exploring more efficient algorithms for solving the Vertex Cover Problem, and investigating the performance of different GPU architectures.
```

Acknowledgements & References

This research is based upon work supported by the National Science Foundation under Grant No. 1462734. This work was supported in part by the computing resources and technical support of High Performance Research Computing at Texas A&M University.

We would like to extend thanks to Michael Langston and Gary Rogers for providing us with the graph data used throughout the benchmarking process.

```
References


```