Using GPU to Speed up Genetic Algorithms

Sohaib Faruqi

May 2, 2018

Genetic Algorithms

- 1. initalize population (of bitarrays)
  2. selection
  3. crossover
  4. mutation
  5. if not done, go back to step 2
- Members of population evaluated by fitness function
- Chose to do Vertex Cover and Maxone problems

Design Choices

- roulette selection
- two-point crossover
- For CUDA version
    - number of blocks = number of runs
    - number of threads/block = size of population

Results

- when only one run, CUDA faster once there's hundreds of members
- when doing multiple runs, CUDA faster after a couple of run
- over 23 times speedup when doing vertex cover experiment from Khuri paper
  - 100 vertices
  - graph density = 0.1
  - 200000 function evals
  - 100 runs

Graphs

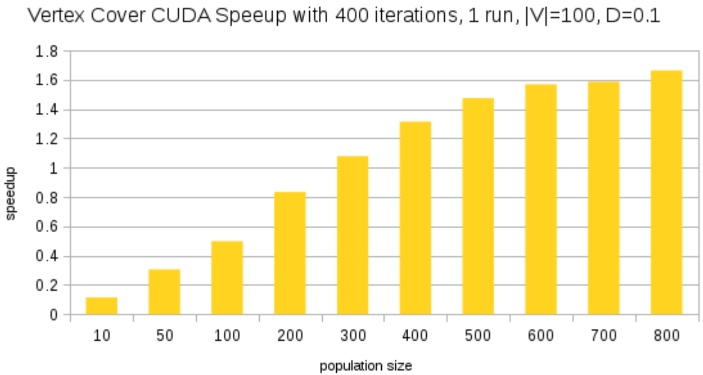

Figure: Speedup for Vertex Cover, varying population size

Graphs (2)

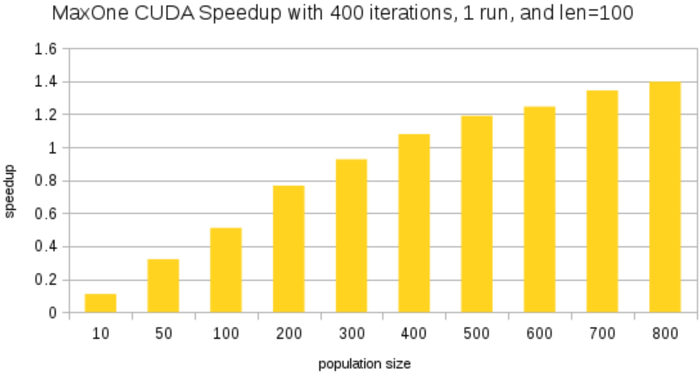

Figure: Speedup for MaxOne, varying population size

Graphs (3)

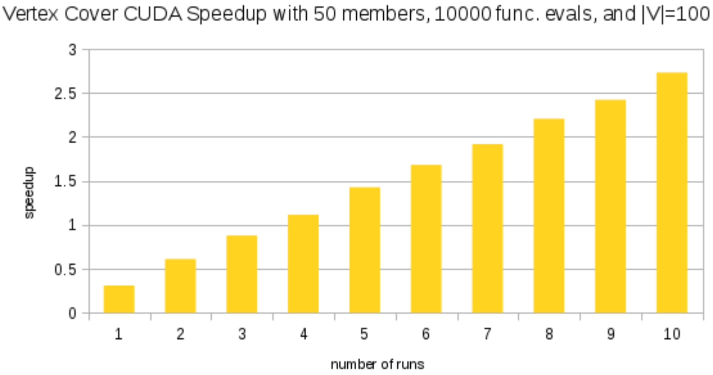

Figure: Speedup for Vertex Cover with graph density $= 0.1$, varying the number of runs

Graphs (4)

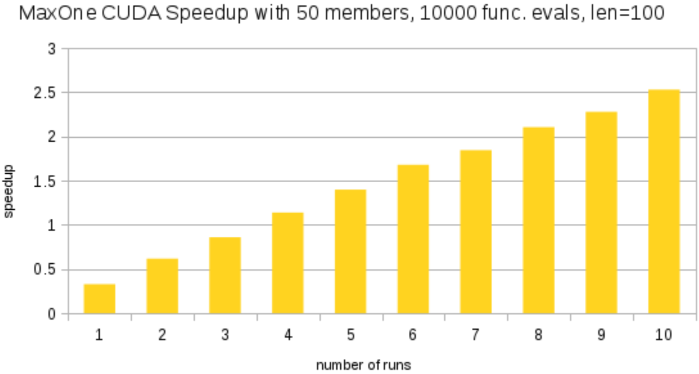

Figure: Speedup for MaxOne, varying the number of runs

Graphs (5)

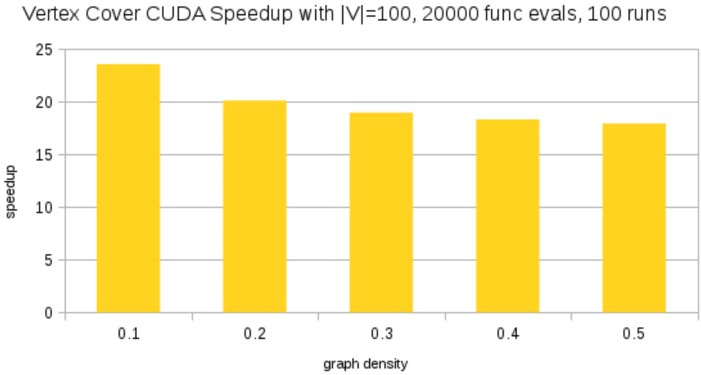

Figure: Speedup for Vertex Cover experiment in paper, with $|V| = 100$. Unable to do for $|V| = 200$ because sequential takes **very** long to finish