Contents:

CTO Problem Statement

In this problem given a set of observers and targets, with each observer able to observe targets only in the feasible radius around it, objective is to maximize the average number of unique targets observed by all the observers.

Genetic Algorithm Applied to CTO:

Agents:- N Observers,M Targets.
Genes: Each observer has a movement direction (dx,dy) s.t after the time step observer position is updated as x=x+dx y=y+dy.
Fitness Evaluation: The total number of targets observed by each chromosome is its fitness value.
Selection :- Two strategies are tried out - 2-Tournament selection, 3-Tournament selection
Cross Over:- Two strategies are tried out- 1Point CrossOver and 2-Point Crossover.
Mutation:- Each gene is modified to a random direction with a probability of 0.05

Simulation Environment

The simulation is run for a total of 1500 time steps with observers count as 12 and targets count as 24 and radius changing from 5 to 25 in steps of 5.
Each such simulation is run 30 times to average out the results.

Execution Time Constraints

Original running time of one simulation was around 20 minutes with only 100 generations per time step.For a fixed observer count, target count, radius there are 8 types of simulations to be tried and 30 for each type of simulation.It would not be feasible to run such simulations

because of the heavy time for exectuion.So an improvement was tried to better the genetic algorithm by implementing parallelism in it.

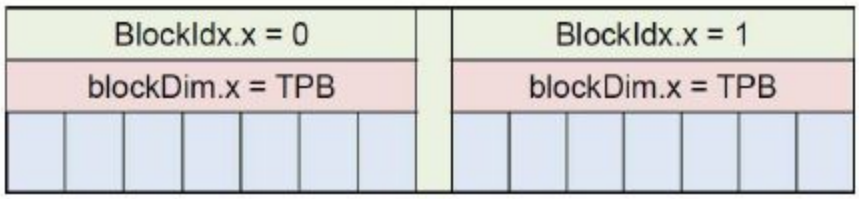
GPU Parallelism Approach - Parallel Genetic Algorithm

Genetic Algorithm is a type of evolutionary algorithm and in they are not parallel by nature.So there have been many approaches in trying to parallelise them by modifying the way they are implemented and parallelising sub-sections wherever possible.

I have read the paper “GPU-Accelerated Genetic Algorithms” attached in the references which parallelises each component of the genetic algorithm finally making the whole algorithm a lot faster.

There are two types of thread models which determines how the threads operate on the population.Suppose there are x population of chromosomes and each chromosome has y genes.Each thread layout operates as follows:

Thread Layout A



In this layout, each thread operates on a single chromosome(not the lowest level since each chromosome has genes). All the threads operate parallel.Selection,CrossOver Points, Fitness evaluation determination have been done in this layout.

Thread Layout B:

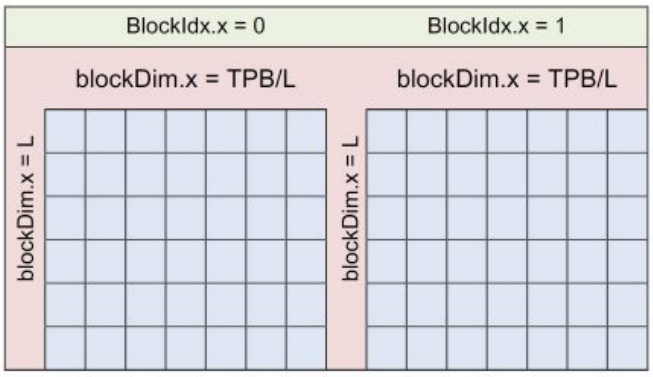
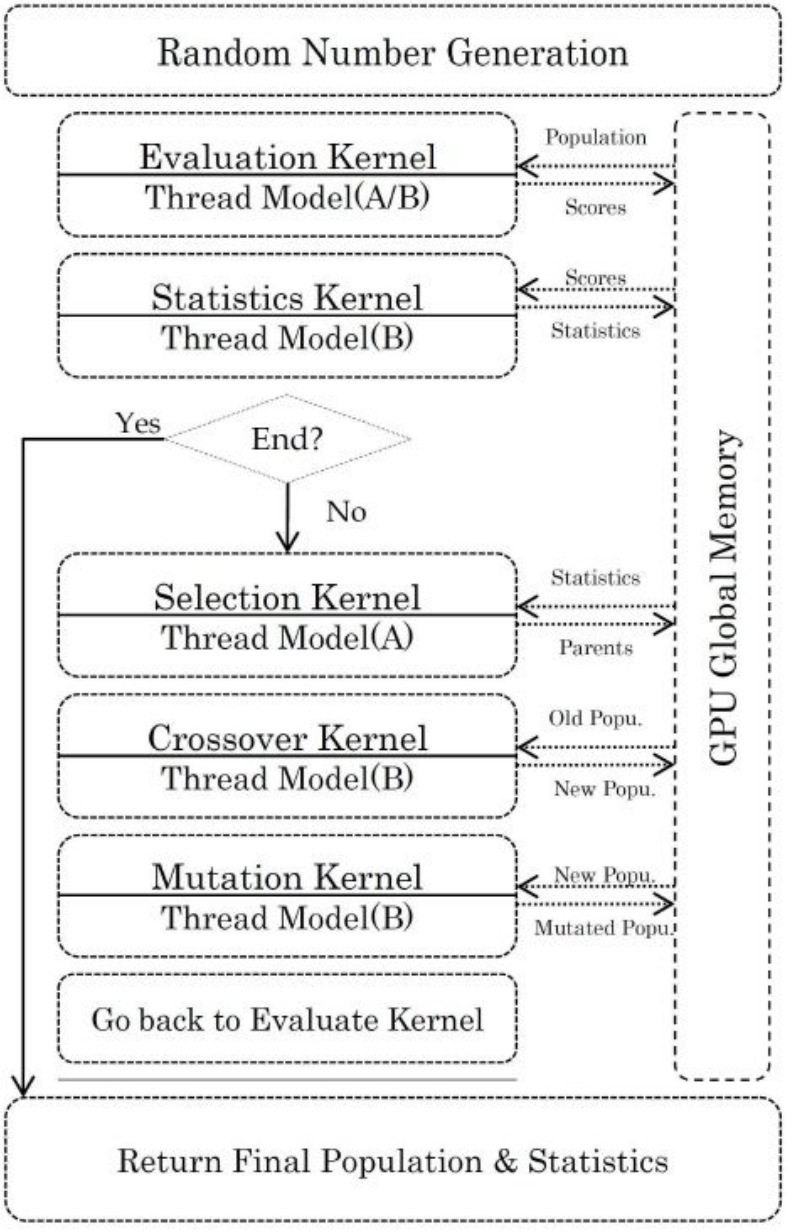


Figure 5: Thread Layout B

In this layout, each thread operates on a single gene.This is 2-Dimensional parallelism meaning each of the genes are operated simultaneously by different threads.CrossOver and Mutation have been employed using this thread model.

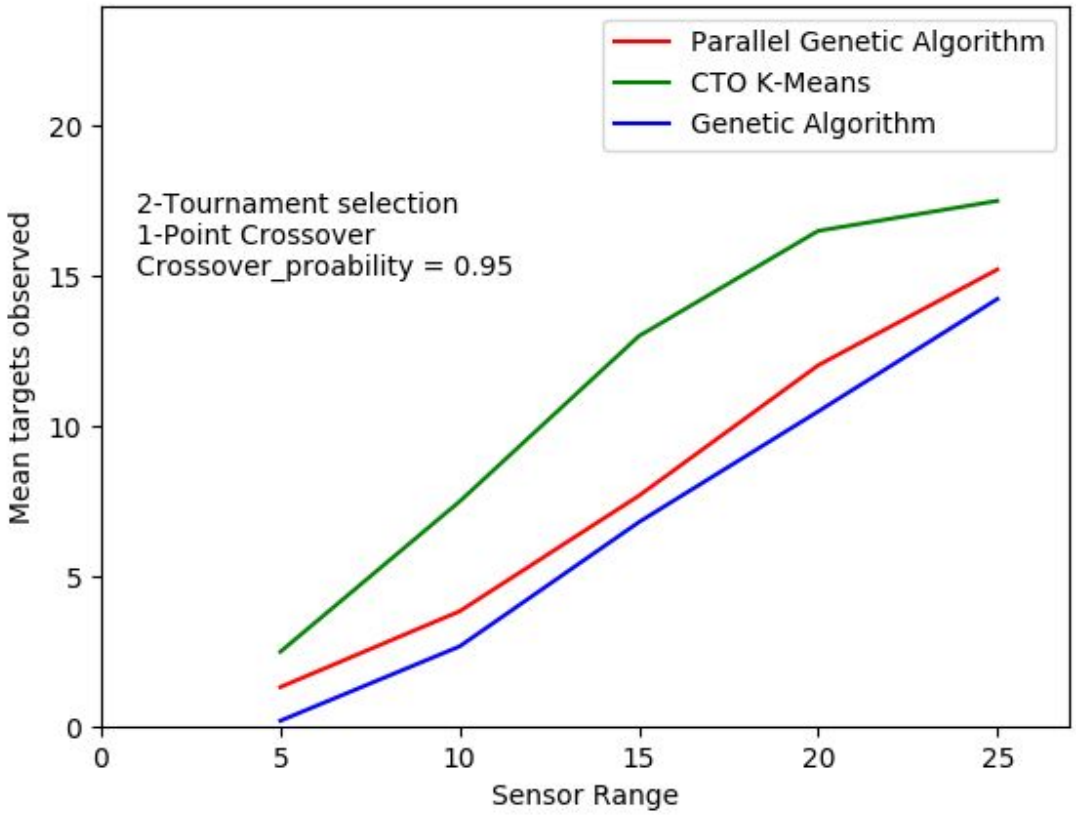
Overall Implementation is as follows:

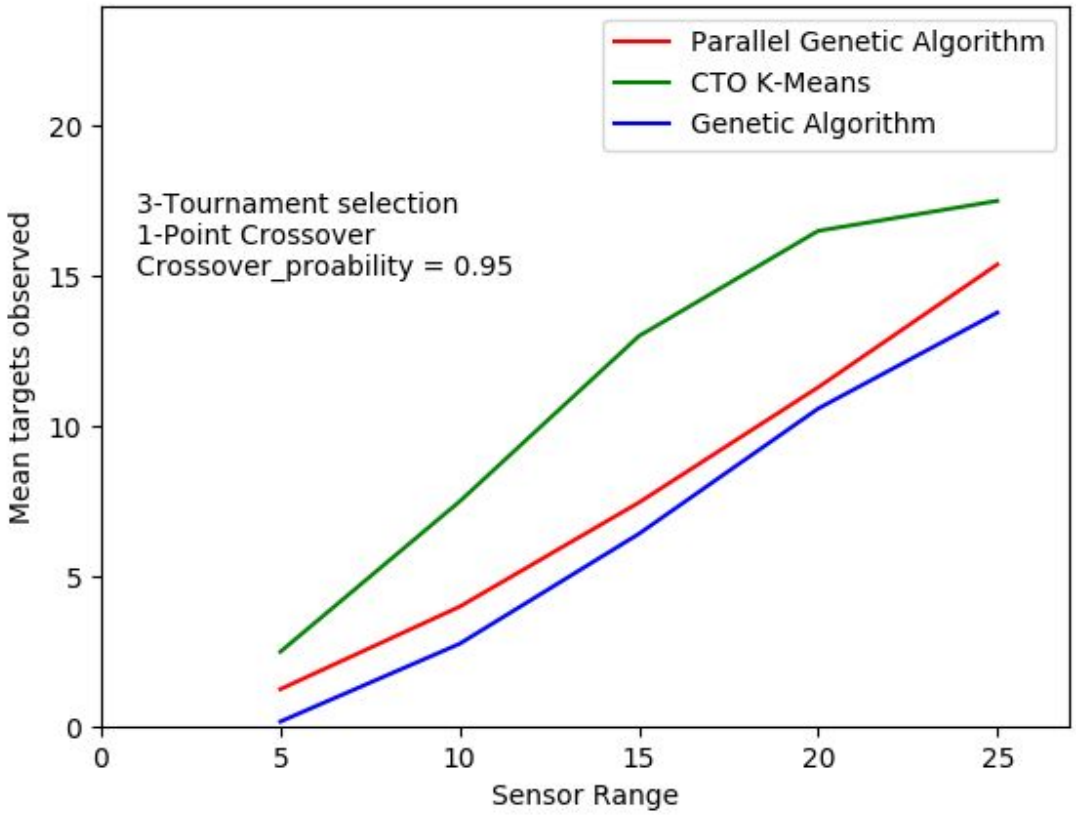


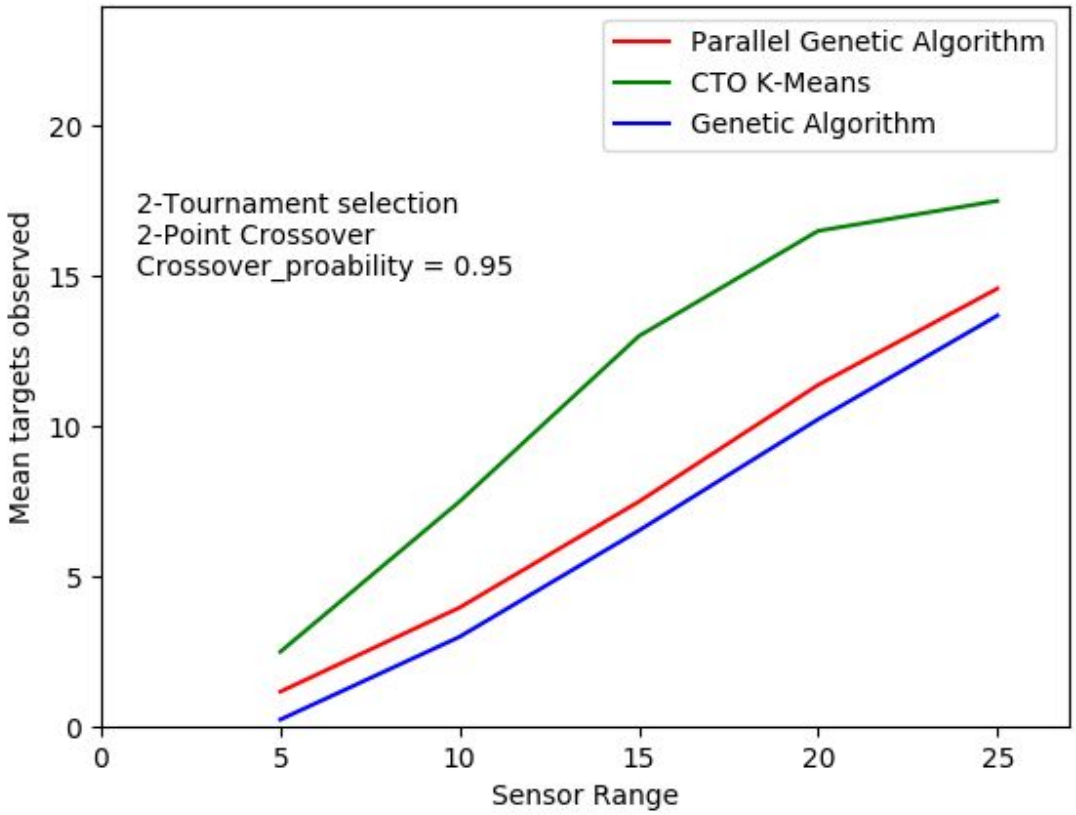
Execution Time Comparison

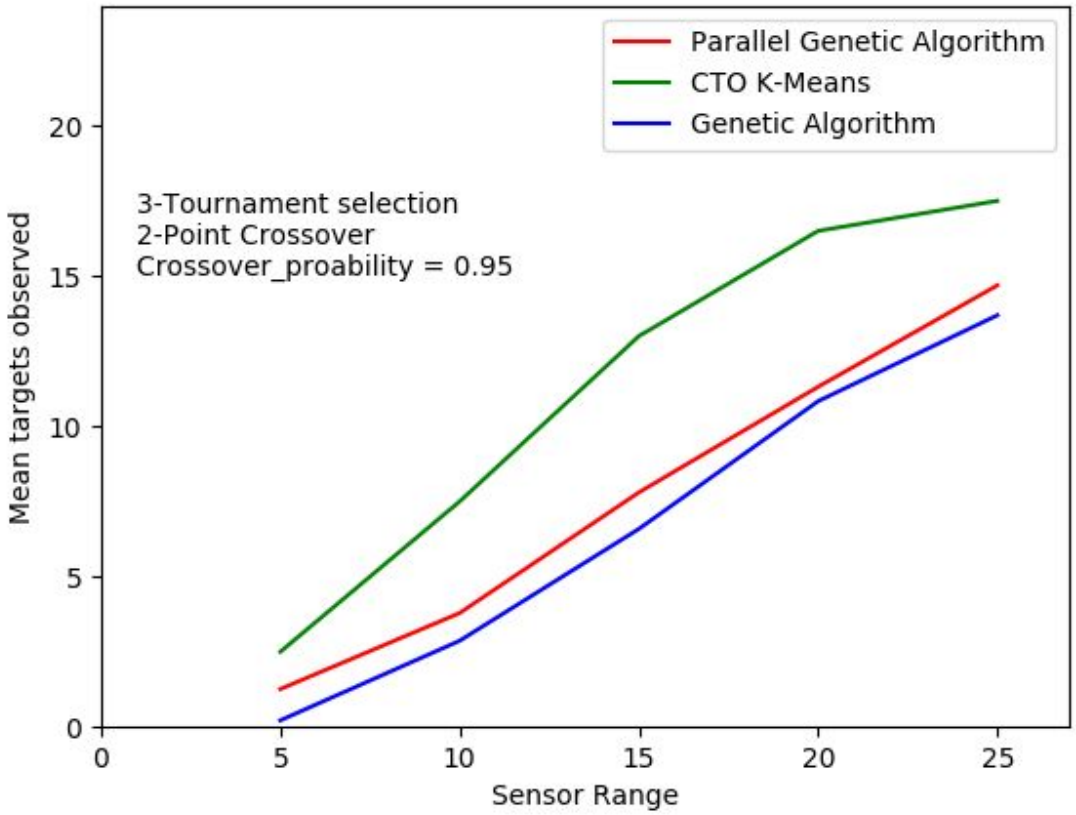
The execution using this parallel approach takes around 1 min for 300 generations per time step. Meaning it is nearly 60x faster than the original approach.**It is an improvement over the original ones since we are able to run more generations meaning we are getting an improvised result.**

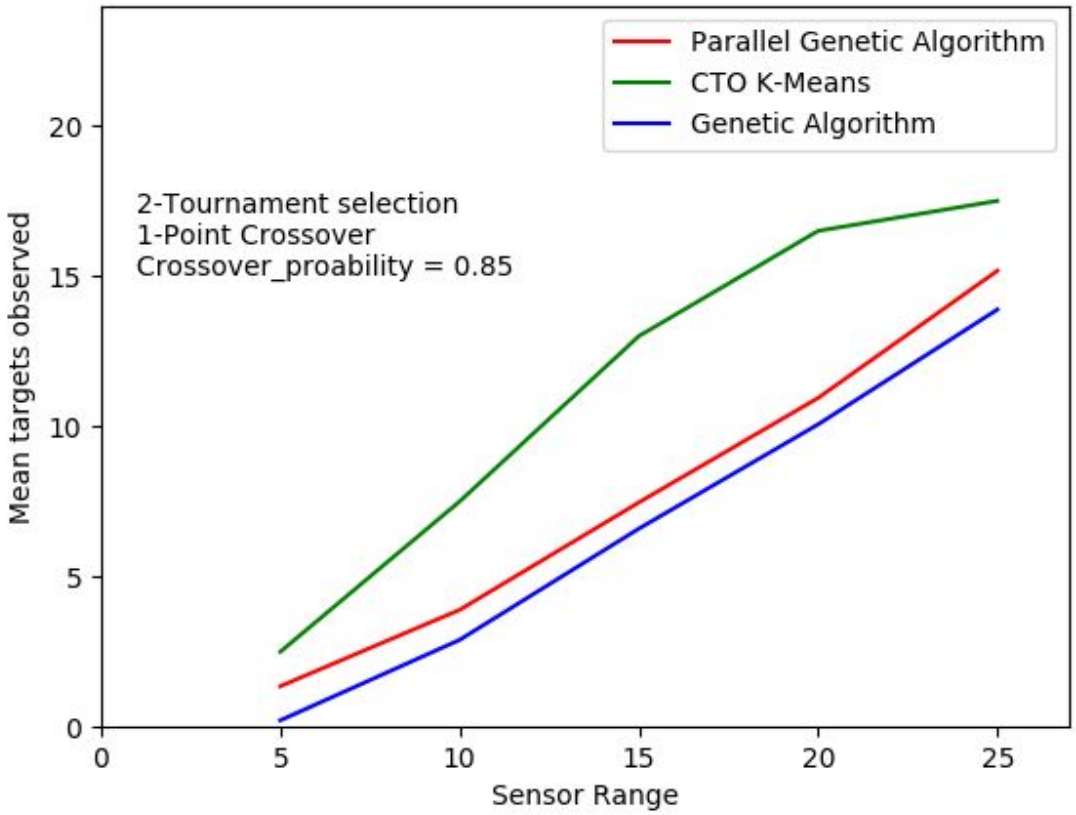
Classic CTO vs Genetic CTO vs Parallel Genetic CTO

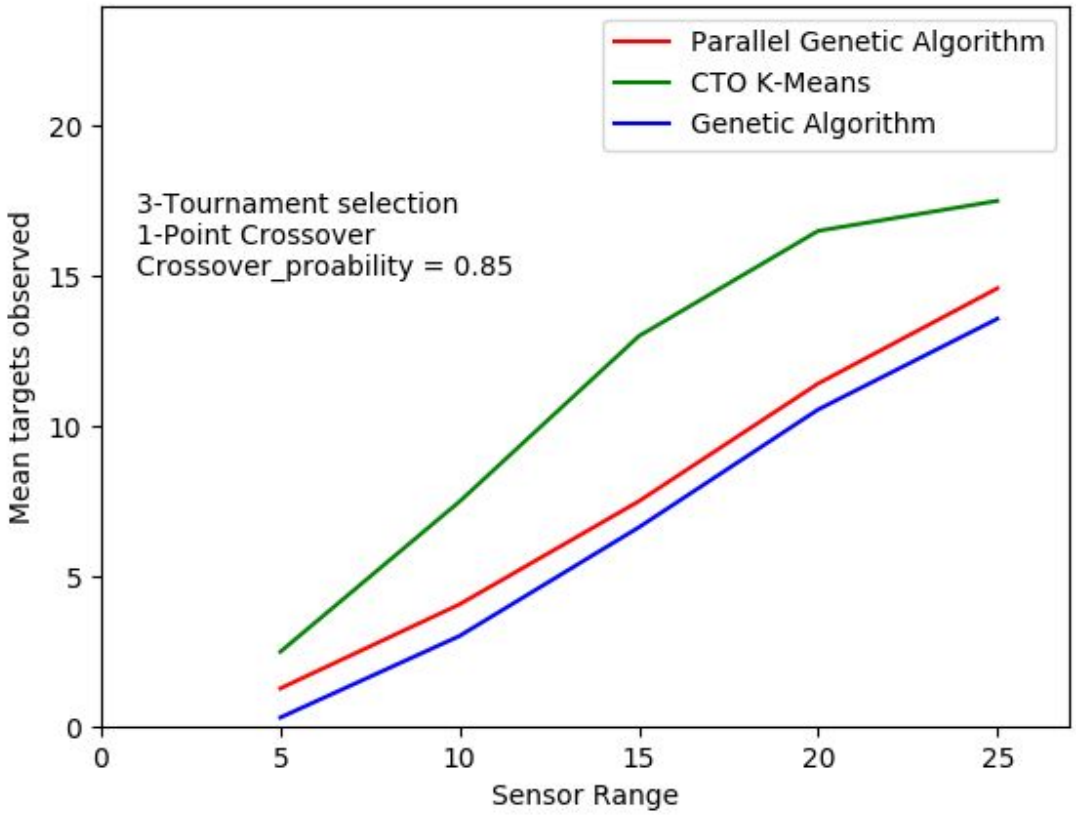

Parallel Genetic Algorithm
CTO K-Means
Genetic Algorithm
2-Tournament selection
1-Point Crossover
Crossover_proability = 0.95
Mean targets observed
Sensor Range
0
5
10
15
20
25

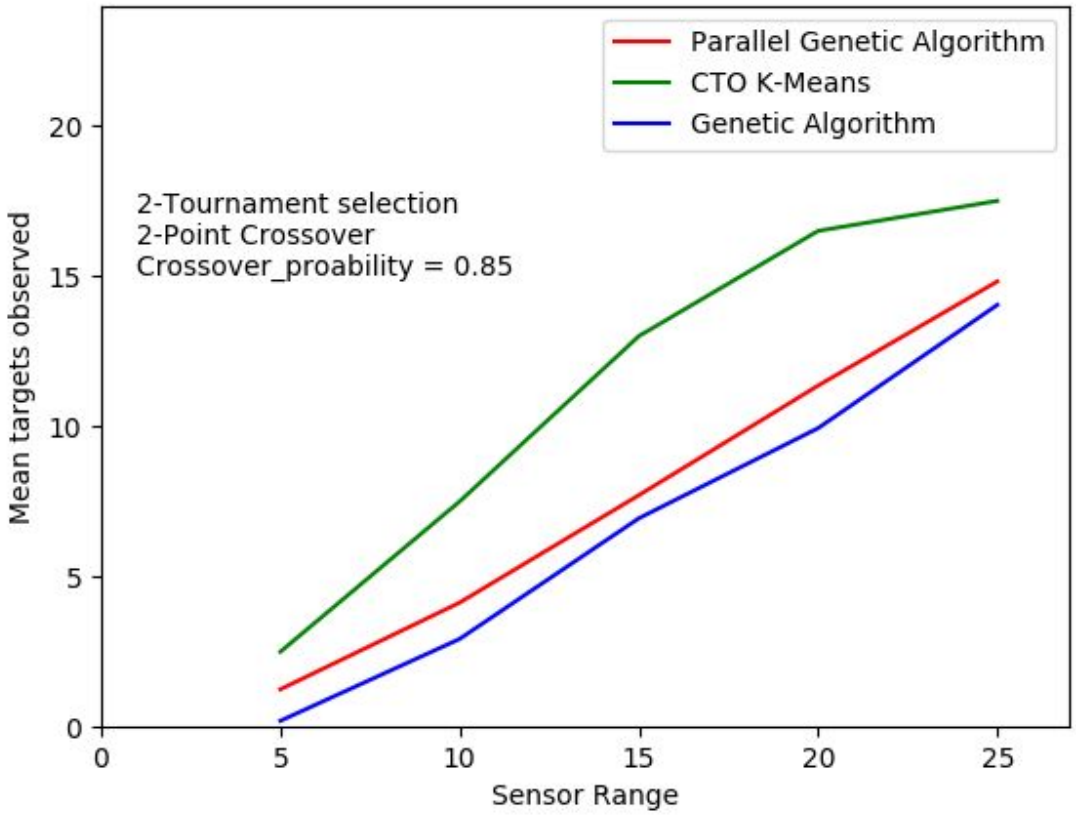

Parallel Genetic Algorithm
CTO K-Means
Genetic Algorithm
3-Tournament selection
1-Point Crossover
Crossover_proability = 0.95
Mean targets observed
Sensor Range
0
5
10
15
20
25


Parallel Genetic Algorithm
CTO K-Means
Genetic Algorithm
2-Tournament selection
2-Point Crossover
Crossover_proability = 0.95
Mean targets observed
Sensor Range
0
5
10
15
20
25


Parallel Genetic Algorithm
CTO K-Means
Genetic Algorithm
3-Tournament selection
2-Point Crossover
Crossover_proability = 0.95
Mean targets observed
Sensor Range
0
5
10
15
20
25


Parallel Genetic Algorithm
CTO K-Means
Genetic Algorithm
2-Tournament selection
1-Point Crossover
Crossover_proability = 0.85
Mean targets observed
Sensor Range
0
5
10
15
20
25


Parallel Genetic Algorithm
CTO K-Means
Genetic Algorithm
3-Tournament selection
1-Point Crossover
Crossover_proability = 0.85
Mean targets observed
Sensor Range
0
5
10
15
20
25


Parallel Genetic Algorithm
CTO K-Means
Genetic Algorithm
2-Tournament selection
2-Point Crossover
Crossover_proability = 0.85
Mean targets observed
Sensor Range
0
5
10
15
20
25

References:

- https://cs.gmu.edu/~sean/papers/luke05tunable.pdf
- http://cvit.iiit.ac.in/images/ConferencePapers/2010/Rajvi10GPU.pdf
- http://neo.lcc.uma.es/Articles/WRH98.pdf
- http://mll.iiit.ac.in/images/ConferencePublications/2017/improving_Cooperative-_target.pdf
- https://memics.cz/2010/pres/palava/Saturday/pospichal.pdf
- http://courses.mai.liu.se/FU/MAI0083/Adrian_Horga-Metaheuristics_report.pdf
- https://pdfs.semanticscholar.org/d08b/89ba5c2fea38c225118d1de1438f1710ca6a.pdf
- https://devtalk.nvidia.com/default/topic/400823/cuda-programming-and-performance/genetic-algorithm/
- https://www.nvidia.com/docs/IO/116711/sc11-cuda-c-basics.pdf
- https://devblogs.nvidia.com/even-easier-introduction-cuda/