**Report**

1. **Introduction**

The project is about implementing an algorithm in order to minimize the average waiting time for all requests. So in order to optimize the video serving streaming server to handle requests reliably and quickly.

1. **Requirement**

Function that takes a solution (2D array) and computes the fitness function:

if req.vid in \_c:

# check if endpoint eid is connected to cache

for \_cc in endpoints[req.eid].con:

if \_ci == \_cc[0]:

\_lats.append(\_cc[1])

Find the cost of downloading the same file from a cache server connected to the endpoint, multiply the difference by the number of requests. This gives the gain obtained when downloading from an available cache server for 1 request.

Hill Climbing and genetic algorithm used then generate all the neighboring solutions, by making simple ”moves” and then keep the best (fittest) solution.

Using hill climbing more than 50 solution generated but its slow technique as we vary crossover and mutation probabilities and changes the population size or food size in test for improvements

1. **Design**

The architecture is simple the IO file that is input and output read the input folder files and also they make array of dataset and variables list globally by making input read call

* total\_video
* total\_end
* total\_required
* total\_cached
* streamed\_cache

Compute\_solution\_score method check for cache video and end points   
then graph is built and cluster is identified for preprocessing method

Videos\_on\_cache retrieved by passing graph to solution file score and solution is printed

Overall **2565511** points.

498287 points, processed in 0.11 seconds tba algorithm also used there

**Conclusion**

The videos on server streaming can be made better and streaming can be improved and optimized using tba , hill climbing algorithms against every request the cache server connected to end point hence thus video streaming service is optimized to handle more and more request and respond quickly by putting cache server that store the copy of most popular video at end point and retrieved on request from server first and waiting time is minimized