

ITVLSI

Algorithm Coding

Group - 1

Ruchita V

Sathya Sravya V

Unnikrishnan R

Raja V



OUR PROJECT

Algorithm based transistor sizing of nanoscale digital circuits

(Determining values of W, L for which leakages and delays are minimum)

Genetic Algorithm

Definition and its Process

- Genetic algorithm is a stochastic optimisation technique that uses principles derived from evolutionary process in nature.
- In operations research, a genetic algorithm (GA) is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA). Genetic algorithms are commonly used to generate high-quality solutions to optimisation and search problems by relying on bio-inspired operators such as mutation, crossover and selection.
- In a genetic algorithm, a population of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem is evolved toward better solutions.
- Each candidate solution has a set of properties (its chromosomes or genotype) which can be mutated and altered; traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible.

Genetic_Algorithm

```
Genetic_algo()  
{  
    Initialize the generation;  
    Calculate the fitness function for all;  
    While (the number of generation > max)  
    {  
        Selection;(Random or Natural)  
        Crossover;  
        Mutation;  
        Calculate the fitness ;  
        Rank them on the basis of fitness by sorting;  
    }  
}
```

Motivation

For optimization using genetic algorithms

- 1) These are robust adaptive optimization techniques based on biological paradigm.
- 2) They perform efficient search on parameters along with maintaining ordered pool of them.
- 3) New parameters are produced from existing ones by applying genetic operators like crossover and mutation.
- 4) Helps in efficient use of information in the problem.

Motivation

contd..

5) These searches are not greatly influenced by local optima or discontinuous functions.

6) Microcode compaction can be modelled in the same way as these problems, which motivates us to use them in this field.

7) Also these algorithms were used in many famous problems such as Travelling salesperson and scheduling job shops.

Methodology of GA

ANALYSIS

The evolution usually starts from a population of randomly generated individuals, and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated

The fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation.



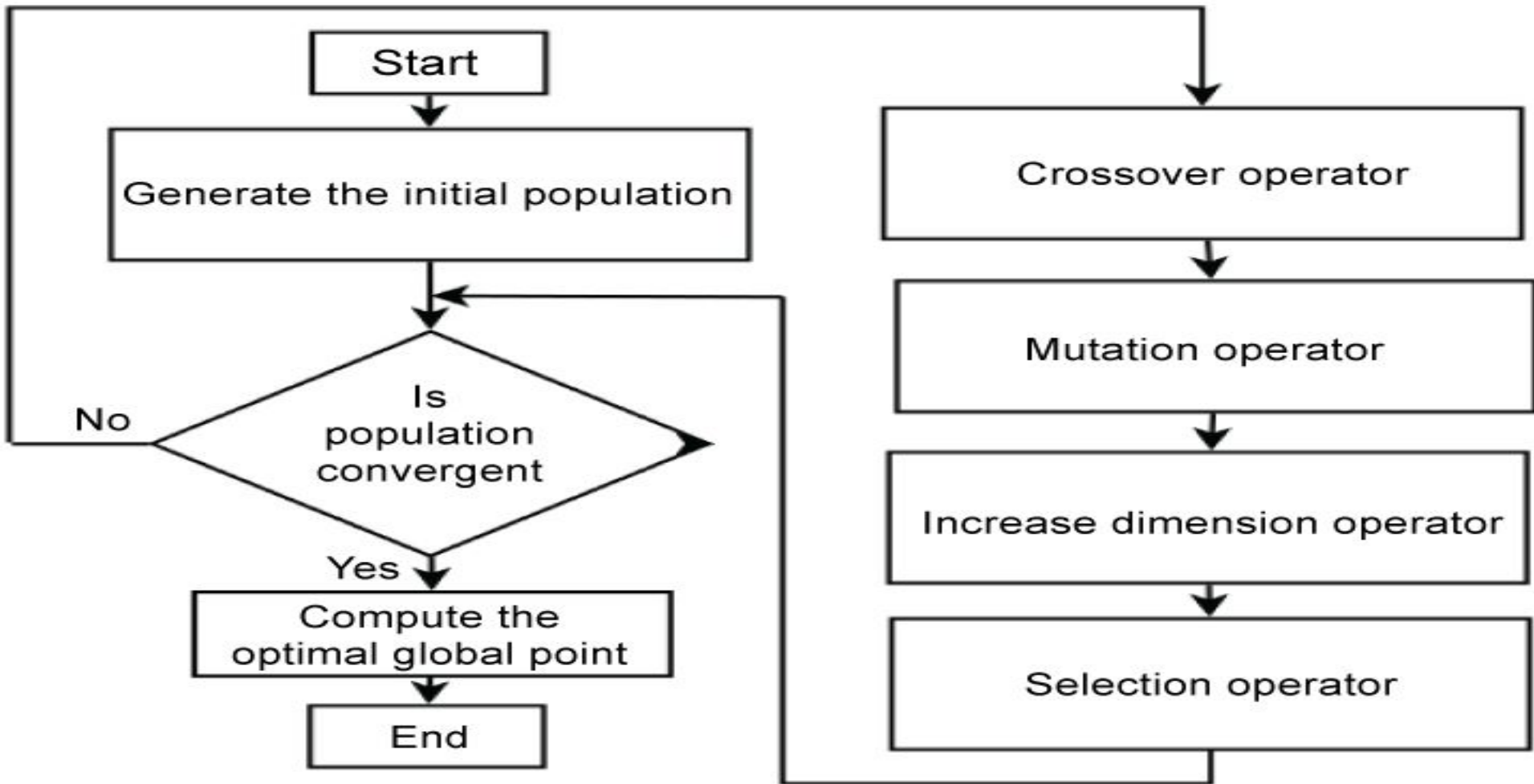


Figure 1: Flow diagram of the improved genetic algorithm.

Objective

- The function that is being minimized is called the objective.
- The inputs to the function that are allowed to vary in order to arrive at the optimum are called the parameters.
- There may be other inputs to the function which are fixed for a given optimization problem. W_p, W_n, L_p, L_n are the parameters that are varied in order to minimise the leakage power and keep the delay within certain bounds.

Individual encoding

- Genetic algorithms encode solutions to the given problem as chromosomal strings and operate on these encodings during the optimization process.
- This helps minimize the amount of problem specific information needed during the optimization process of a genetic algorithm.
- An encoding scheme that maps each chromosome string to a unique solution is preferred as the genetic algorithm will not waste time evaluating multiple encodings of the same solution.
- The chosen encoding scheme should also be easy to decode using minimal time and memory resources.

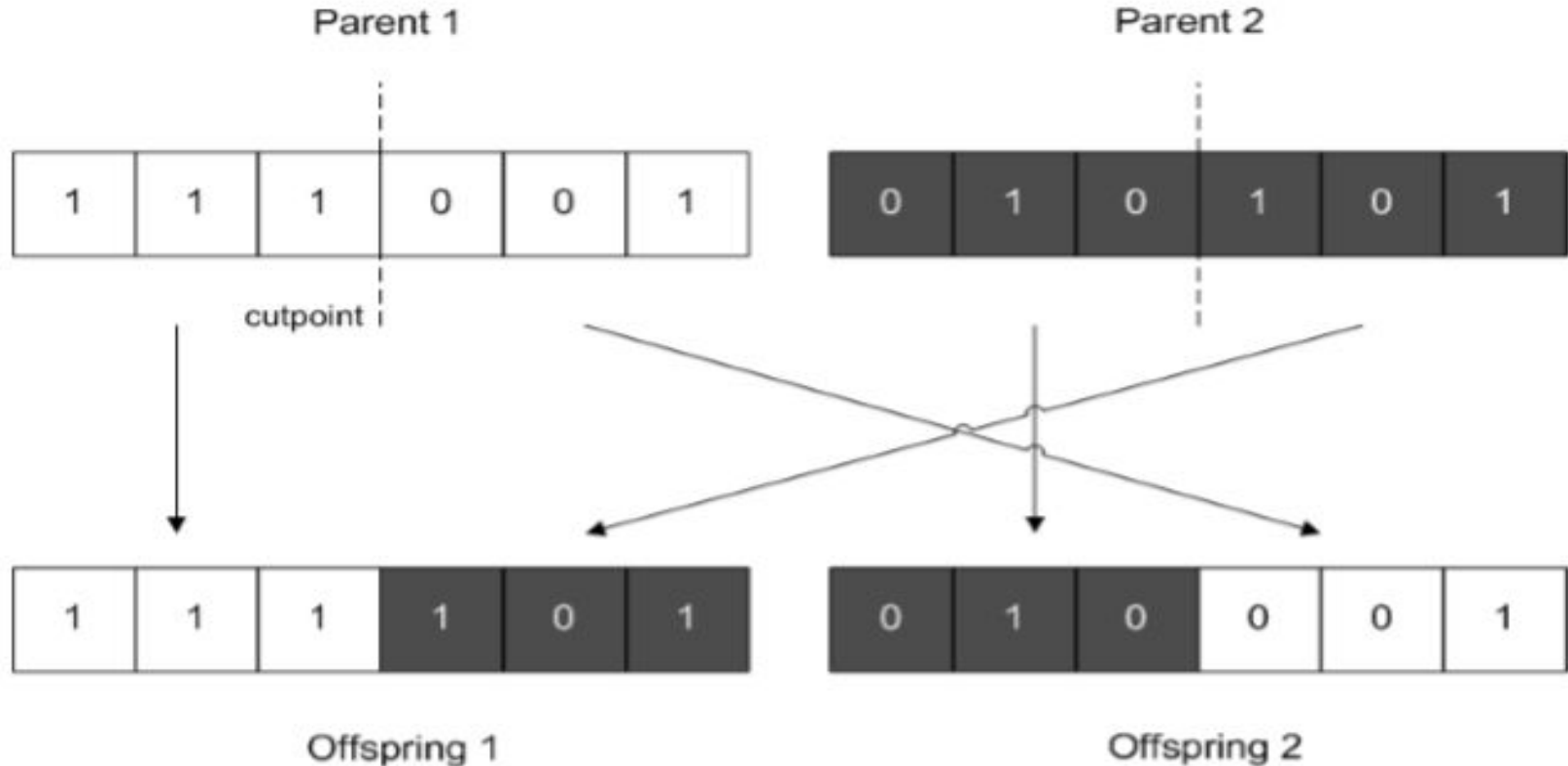
Crossover

- The crossover operator performs a probabilistic exchange of chromosomal information between two individuals to produce a new individual.
- The crossover operator selects two parent individuals from the population based on a selection scheme. It then produces an offspring individual by using certain information from the first parent and the rest of the information from the second parent.

Thus, the offspring individual inherits a subset of properties from both of its parents.

One point Crossover

12

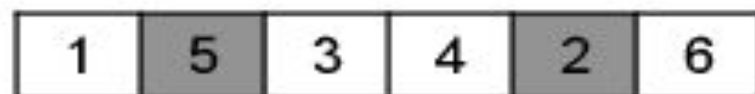
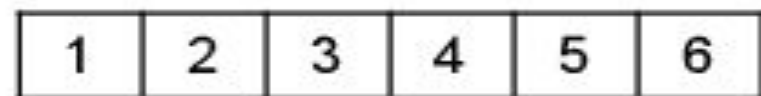


Mutation Operator

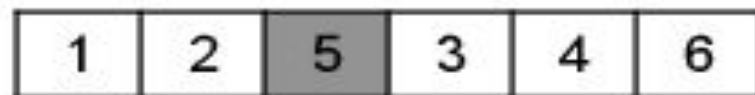
- The mutation operator typically picks a random individual from the population and performs an inversion or some other random operation on the individual chromosome. After a certain number of generations, the crossover operator tends to produce offspring that are very similar to the parent individuals.
- Then the mutation operator plays a critical role in restoring lost genetic material or providing diversity in the current population. Thus, the mutation operator helps prevent convergence to local optima. Mutation operator induces change in the solution so as to maintain diversity in population as well as prevent convergence.

Parent

Offspring



Swap mutation



Insertion mutation



Inversion mutation



Displacement mutation

Fitness measure

- In this part , we run hspice for different individual files , and get leakage and delay files.
- We extract the T_{hl} and T_{lh} from delay files and Leakage power when $V_{in}=0$ and $V_{in}=1$ from .ms0 files.
- We take average of both delays and both leakage powers and then assign a fitness measure.
- The fitness measure should reflect the quality of corresponding solution to the problem. It will also decide if the individuals(parameters) survive through generations.
- After applying genetic operations on individuals , the fittest (here the one with leakage power=0 , delay below a particular bound) will be considered for next generations.
- Here we adopted usage of a profit function to filter the ones with least leakage power and delays , we output the best 90% of given W,L s of mosfets for the CMOS inverter cell .

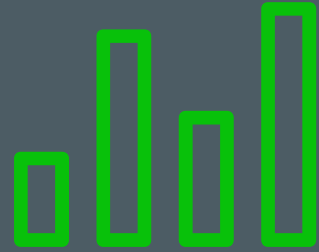
Termination

Thus we ended up with getting best values of W, L for which Leakage power and Delays are minimum .

We understand the quality of individuals from their fitness measures .

At the end we have a generation of individuals with their delays and leakages far less than the first generation .

IMPLEMENTATION RESULTS



This is a subtitle for your slide

FINDING OUT THE BEST W,L VALUES

FOR WHICH LEAKAGE AND DELAY ARE
MINIMUM

INITIAL VALUES :(FROM FILE WE TOOK)

Wp=360nm Wn=180nm

Lp=45nm Ln=45nm

Leakage powers: 1) 1.508e-7

2)1.693e-7

Delays :

1) 1.1e-11 2) 9.778e-12

FINAL VALUES (after processing initial ones ,
we got optimised leakage and delays)

Wp = 305.18nm Wn =279.75 nm

Lp = 46.53nm Ln= 45.45nm

Leakage powers: 1) 1.61e-07

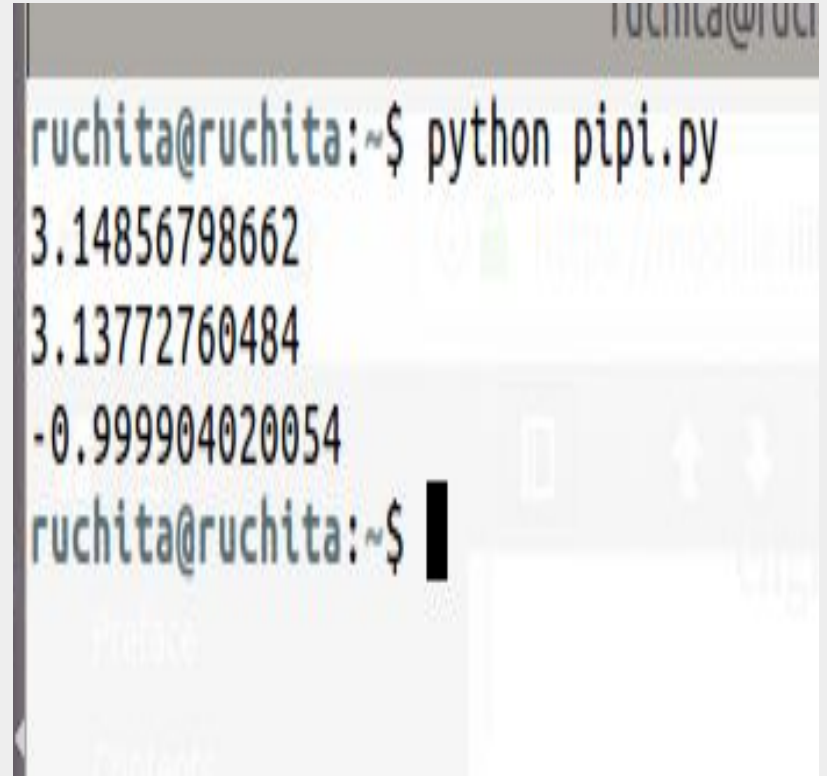
2)1.57e-07

Delays :

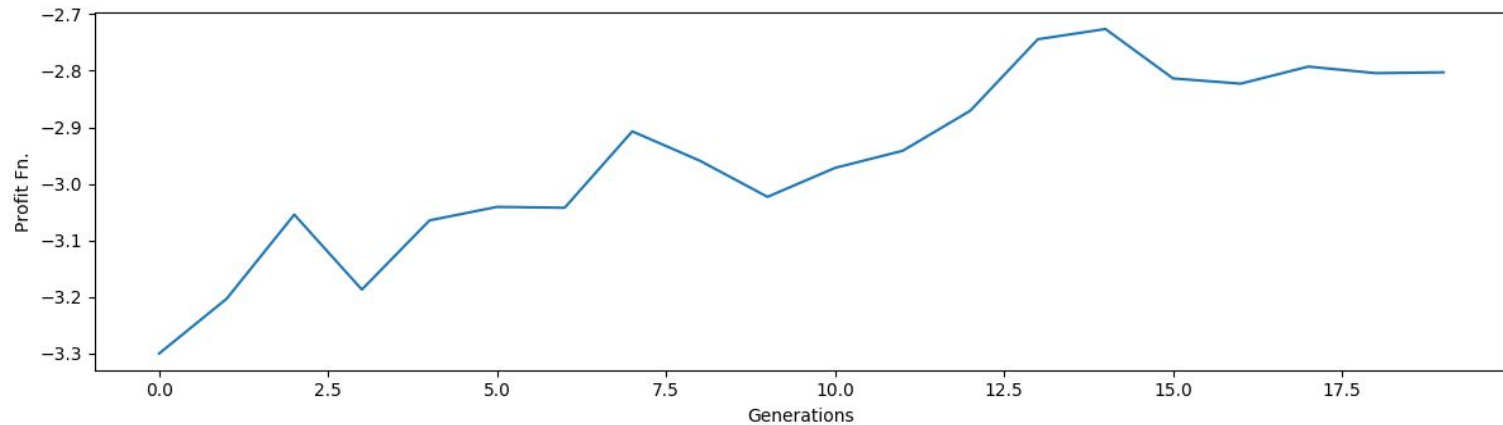
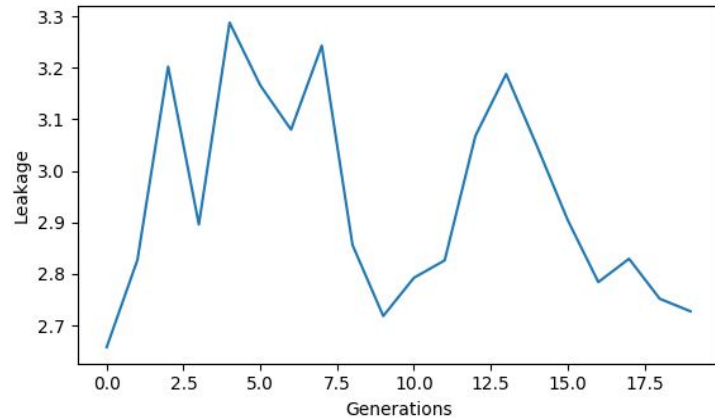
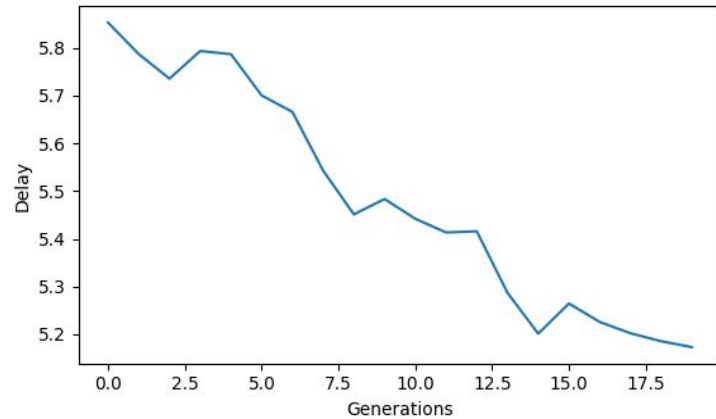
1) 7.65e-12 2) 4.15e-12

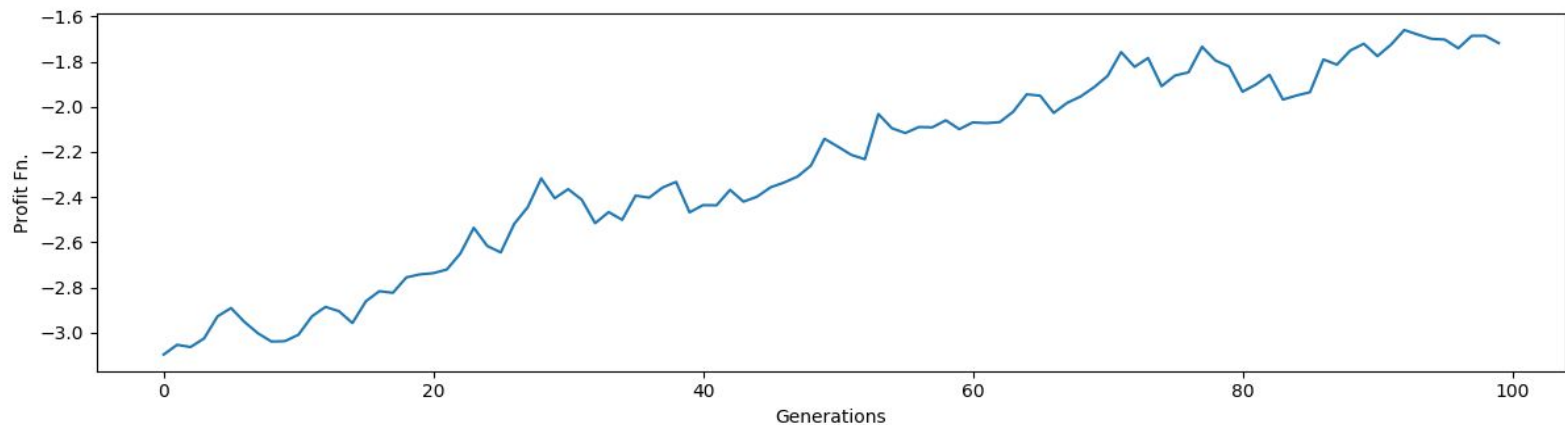
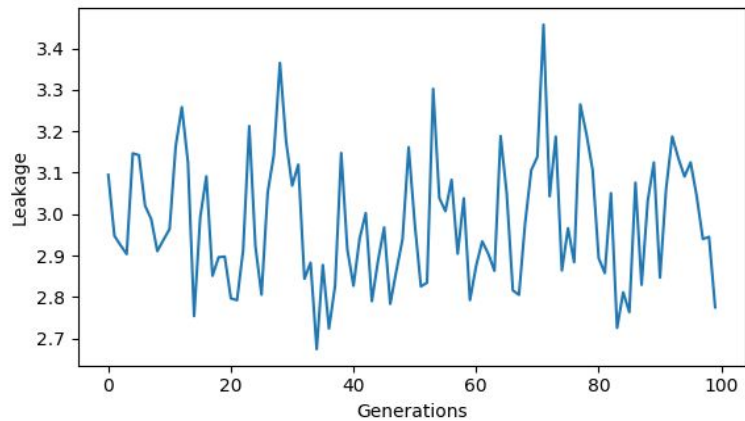
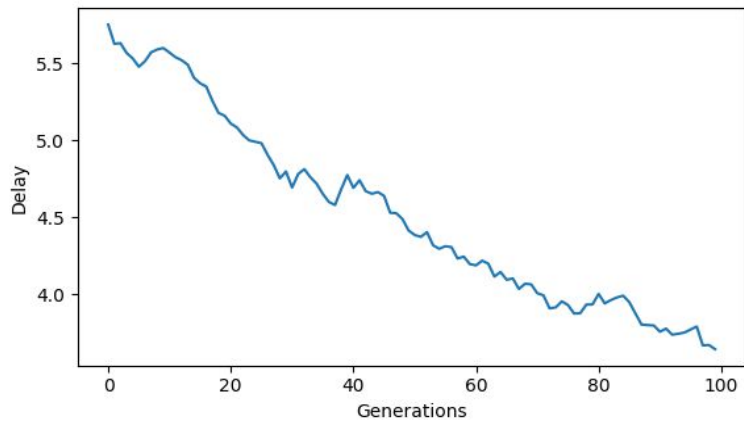
Easom function

Verification of algorithm's
working

A terminal window screenshot showing the execution of a script named 'pipi.py'. The script outputs three numerical values, each on a new line: 3.14856798662, 3.13772760484, and -0.999904020054. The prompt 'ruchita@ruchita:~\$' is visible at the top and bottom of the terminal output.

```
ruchita@ruchita:~$ python pipi.py  
3.14856798662  
3.13772760484  
-0.999904020054  
ruchita@ruchita:~$
```







OUR TEAM

Algorithm Coding Group-I

Contributions

Ruchita V

Sathya Sravya V

UnniKrishnan R

Raja V



CONCLUSION

Our Conclusions

- Genetic algorithms have been extensively researched and applied to a wide variety of single objective and multi-objective optimization applications.
- Here , the application of genetic algorithms to CMOS inverter cell is investigated.

Credits and References

Reference1

Lawrence Davis, Handbook of Genetic Algorithms, 1991.

Reference2

J. J. Grefenstette, R. Gopal, B. J. Rosmaita, and D. Van Gucht,
"Genetic Algorithms for the Traveling Salesman Problem,"
Proceedings of the 1st International Conference on Genetic
Algorithms, pp. 160-168, 1985.

ITVLSI
Group-1



THANK

YOU