PES UNIVERSITY




Project Progress Review #2

(Customer Requirement Specifications)

Project Title : Efficient Python Genetic Algorithm Framework
Project ID : PW19CGM01
Project Guide : Ms. Chitra G.M.
Project Team : Bharatraj S Telkar (01FB15ECS066)
Daniel I (01FB15ECS086)
Shreyas Vivek Patil (01FB15ECS286)

Project Abstract and Scope

Our Genetic Algorithm Framework is proposed to be a very efficient, generic Framework where users can easily simulate all variations of Genetic Algorithms very easily. It is a usable way to **explore the problem solving ability of Genetic Algorithms.**

Usage of Spark to explore improvements in GA by using large populations

Project Abstract and Scope

In addition to the API, we would provide a **nice frontend web service** for online GA execution where

Users can enter various options of the genetic algorithm and simulate them. It also allows users to enter custom code for various operations and simulate them

Scope

- Generic GA Framework
- GA Optimisations
- Parallelization using pySpark
- Other ML algo (ANN) using our framework
- Checkpoints
- Statistics
- Website Service GUI

Further Literature Survey

S.No	Paper	Author	Inference
1.	*Scaling genetic algorithms using map reduce*	Verma, Abhishek, Xavier	Basic GA using Map Reduce
2.	*Evolve a neural network with a genetic algorithm*	Matt Harvey	Choosing efficient ANN hyperparameters

User Characteristics

Students, teachers, programmers, researchers, company employees / entrepreneurs can all use our genetic algorithm framework while experimenting with different Machine Learning Algorithms and observing performance. They can also play around and simulate different Genetic Algorithms online on our website.

Dependencies / Assumptions / Risks

Dependencies

1. SKULPT – allows to execute python code in the browser itself
2. Spark – to implement parallelisation
3. Flask – backend
4. Python’s multiprocessing library
5. Psql - database

Dependencies / Assumptions / Risks

Assumptions

- Problem being solved using GA follows the usual process of evaluation -> selection - > crossovers -> mutations. It supports basic changes to this. But any major changes is infeasible to do using our API
- User knows how to best solve the problem using GA

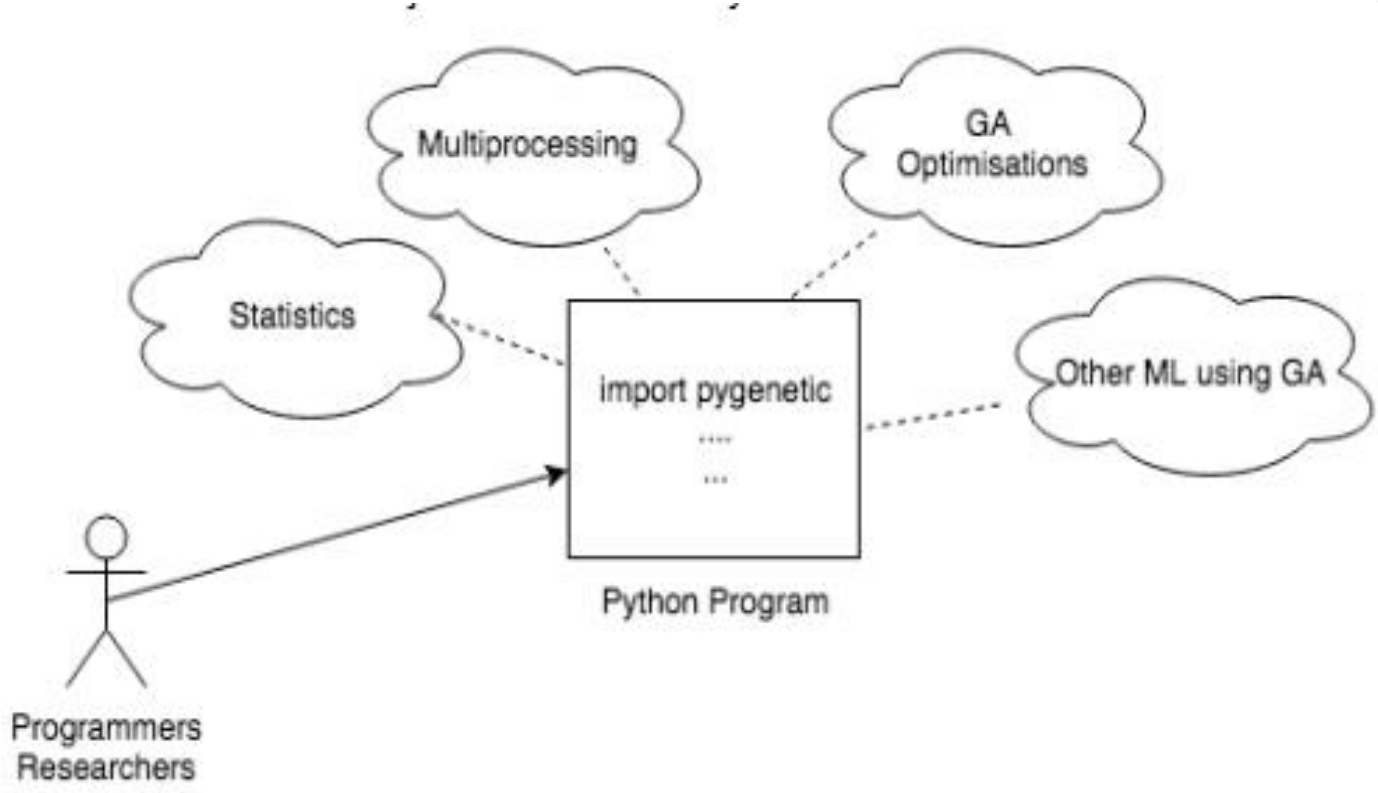
Dependencies / Assumptions / Risks

Risks

- Difficulty in parallelizing the operations for different kinds of problems to be solved using GA.
- Execution of malicious code from the UI

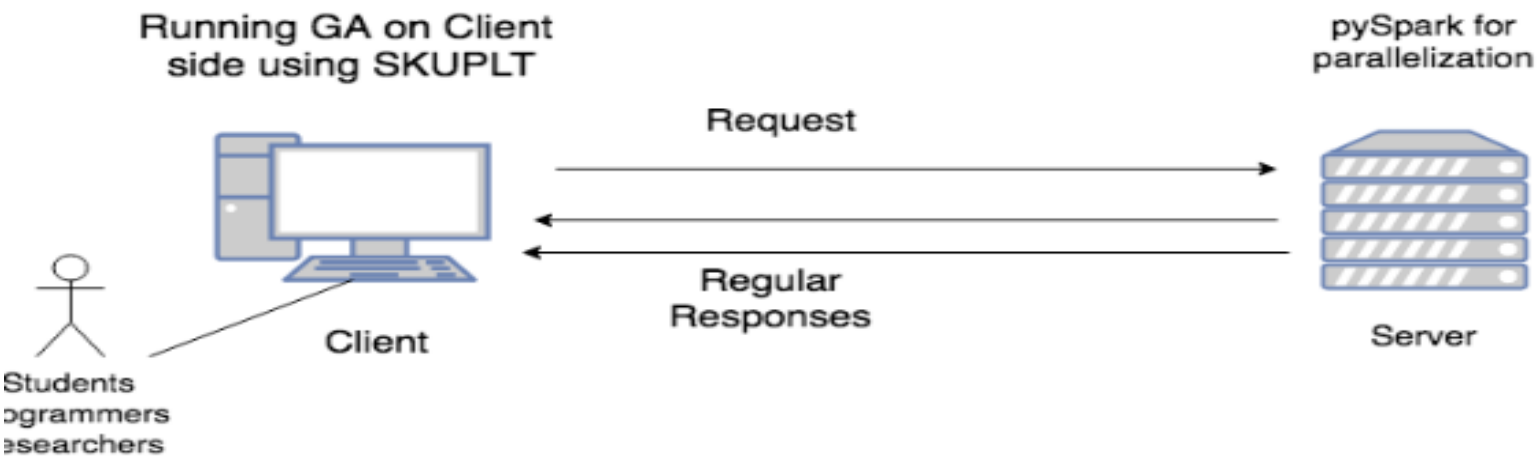

PES

Our API


Multiprocessing
GA
Optimisations
Statistics
import pygenetic
....
...
Other ML using GA
Python Program
Programmers
Researchers

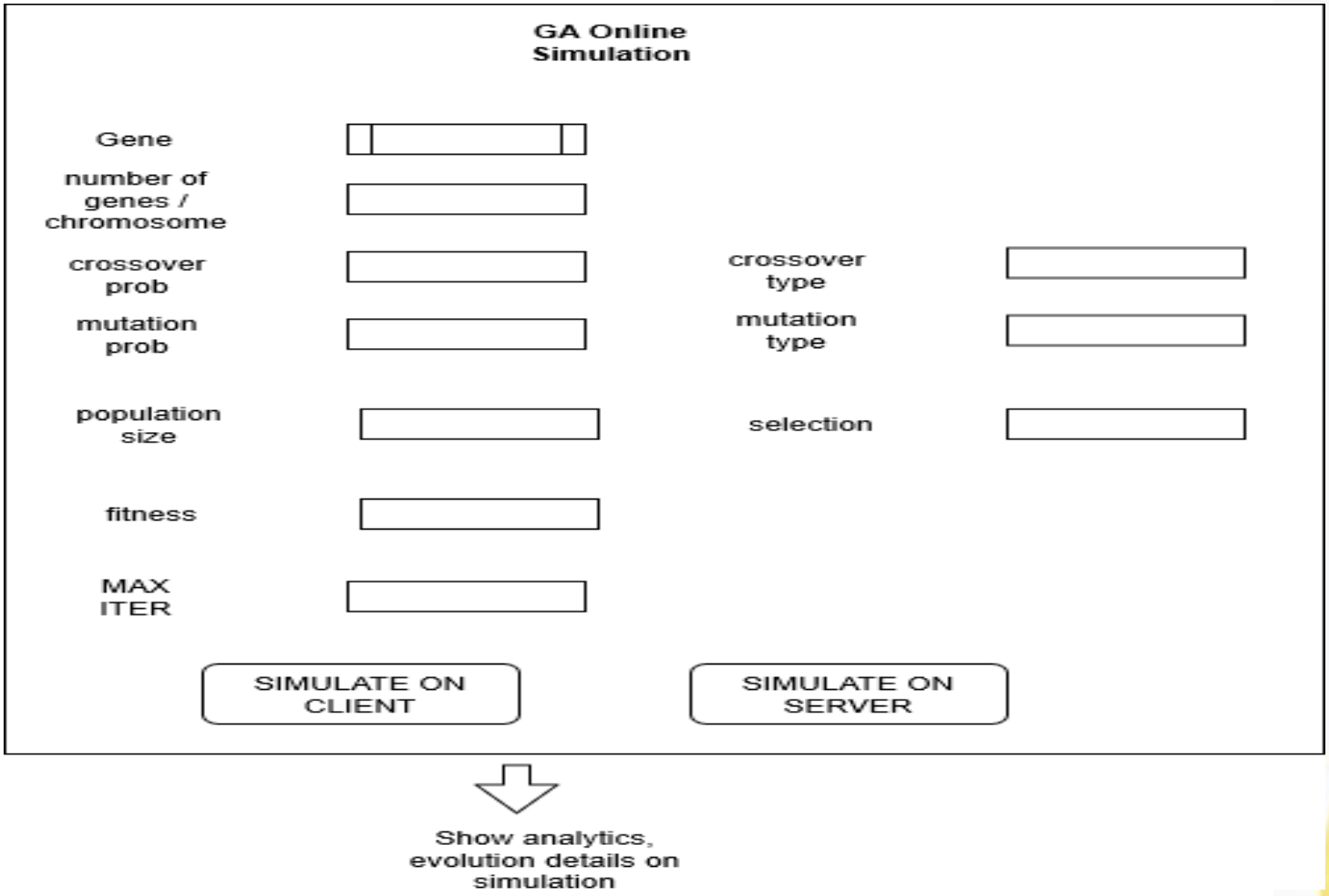
System Architecture – Frontend Web Service

Step 1: Take users inputs about the GA from the UI
Step 2: Convert the user inputs into python code which uses our GA API
Step 3: Run the code and observe GA execution and results



Choice: User can choose the run the code on the client side or the server side.
Client:no need to contact server for every GA generation update, no db support, no parallelization
Server:need for regular updates, db support, parallelization pySpark, "Genetic Algorithm as a Service"

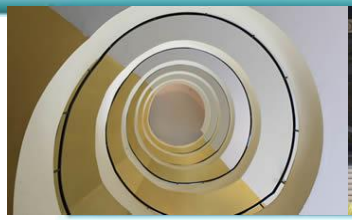
User Interface



Technologies Used

- Python - 3.0 and above
- SKULPT – allows to execute python code in the browser itself
- Flask - backend
- Pyspark - for parallelisation of operations in algo
- Pytest - for testing purposes.
- Travis CI – for continuous integration

PES UNIVERSITY






Thank You