Project Approval

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

PES UNIVERSITY




Problem Statement

- To develop a highly efficient, usable and generic genetic algorithm framework in Python.

Existing libraries

- As of present, there is only one main framework for genetic algorithms in Python called DEAP.
- It allows users to implement GA involving different types of representations, basic parallelization, check pointing.
- However, it does not include large scale parallelization (eg: using Map Reduce in Hadoop).
- It also fails to implement the more recent research on genetic algorithm optimization.

Existing libraries (cont.)

deap

- creator
  - create()
- base
  - Toolbox
    - clone()
    - map()
    - register()
    - unregister()
  - Fitness
    - dominates()
- tools *
- algorithms
  - eaSimple()
  - eaMuPlusLambda()
  - eaMuCommaLambda()
  - eaGenerateUpdate()
  - Variations:
    - varAnd()
    - varOr()
- gp
  - PrimitiveTree
    - from_string()
  - PrimitiveSet
    - addEphemeralConstant()
    - addPrimitive()
    - addTerminal()
  - Primitive
  - Terminal
  - Ephemeral
    - compile()
    - compileADF()
    - graph()
  - PrimitiveSetTyped
    - addADF()
    - addEphemeralConstant()
    - addPrimitive()
    - addTerminal()

* → 1) Operators
↳ Multiple operators each for Initialization, Crossover, Mutation, Selection and Migration.

2) Statistics
- compile()
- register()

MultiStatistics
- compile()
- register()

3) Logbook
- pop()
- record()
- select()

4) HallOfFame
- update()
- insert()
- remove()
- clear()

ParetoFront
- update()

5) History
- update()
- getGenealogy()

Problems addressed

- **Need for Coding at different Abstraction Levels:** Our Project aims at developing a easy to use framework, generalizing all types of operations used in GAs. We also intend to provide users with ability to code at different abstraction levels as per need.

- **Large Scale Parallelization:** Our Project aims to use Hadoop Map Reduce in order to provide efficient parallelization when large amount of chromosomes are present in the population.

PES UNIVERSITY




Problems addressed (cont.)

- **Optimizations based on recent research on genetic algorithms:** Our Project aims to implement some optimisations techniques based on recent research on genetic algorithms. These include allowing usage of adaptive mutation probability, short/long term memory, etc

- **API to perform other Machine Learning Algorithms on data using Genetic Algorithm:** Our Project aims to provide users the option to perform other ML algorithms like ANN, KNN classification, etc using Genetic Algorithm.

Who will benefit

Students, teachers, 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

Literature Survey Overview

S.No	Paper	Author	Inference
1.	Improving genetic algorithms' efficiency using intelligent fitness functions	Jason Cooper, Chris Hinde	Improve efficiency using storage
2.	An Adaptive Genetic Algorithm based on Population Diversity strategy	Chen Lin	Adaptive mutation rate
3.	A Parallel Genetic Algorithm Based on Hadoop MapReduce for the Automatic Generation of JUnit Test Suites	Linda, Ferrucci, Alflonso, Saro	Basic Mapper and Reducer

Literature Survey Overview

S.No	Paper	Author	Inference
4.	An Enhancement of k-Nearest Neighbor Classification Using Genetic Algorithm	Anupam, Syed, Akram	How it can be modeled
5.	Evolve a neural network with a genetic algorithm	Matt Harvey	How it can be modeled

Literature survey (1)

1. *Improving genetic algorithms' efficiency using intelligent fitness functions(2003):*

This paper discuss how the usage of short term and long term memory can improve the time efficiency of genetic algorithms. Short term memory stores information about the current generation only while long term memory can store information from previous generations as well.

Literature survey (2)

2. *An Adaptive Genetic Algorithm based on Population Diversity strategy(2009):*

This paper discuss how the usage of a dynamic mutation probability based on the diversity of the current generation improves the efficiency of the genetic algorithm. The more diverse the population, the lesser the mutation probability should be while the less diverse the population, the more the mutation probability should be.

Literature survey (3)

3. *A Parallel Genetic Algorithm Based on Hadoop MapReduce for the Automatic Generation of JUnit Test Suites(2012):*

This paper discuss how the performance of genetic algorithms can be improved by exploiting the MapReduce parallelization paradigm on Hadoop. Basic Mappers and Reducers were also discussed in this paper.

Literature survey (4)

4. *An Enhancement of k-Nearest Neighbor Classification Using Genetic Algorithm (2005):*

Traditional KNN uses all attributes while trying to find nearest neighbours, but not all attributes are important and importance of attributes learnt from training set may not be constant. To factor in the importance of attributes and consider top most important ones Genetic algorithm can be used.

Literature survey (5)

5. *Evolve a neural network with a genetic algorithm (2010):*

Building a neural network and finding the right hyperparameters (parameters that can be controlled by designer such number of layers, units in each layer etc.) is usually done by trial and error. This can be time consuming and tedious process as an improvement over this brute force approach Genetic algorithm can be used.

Methodology to solve DEAP problems

- Provide users with different abstraction levels in the API – high level, low level
- Allow user to enable/disable different optimisation techniques as per need
- Web GUI to simulate different Genetic Algorithms online
- Create config from Web application, no need to have knowledge to functions of the API
- MapReduce/multiprocessing
- Simulate other ML algorithms from GA though API & online on our website.

PES UNIVERSITY




Features

- Simple easy to use framework for implementing genetic algorithms.
- Implements GA optimization techniques like adaptive mutation rate, intelligent fitness functions.
- Parallelization using Hadoop MapReduce/multiprocessing.

Proposed solution features (cont.)

- Checkpointing.
- Usage of a config file generated from our website to implement the genetic algorithm without the need to explicitly call our various framework functions.
- Framework allows users to implement Decision trees, k neighbours classifications, ANN using genetic algorithms.

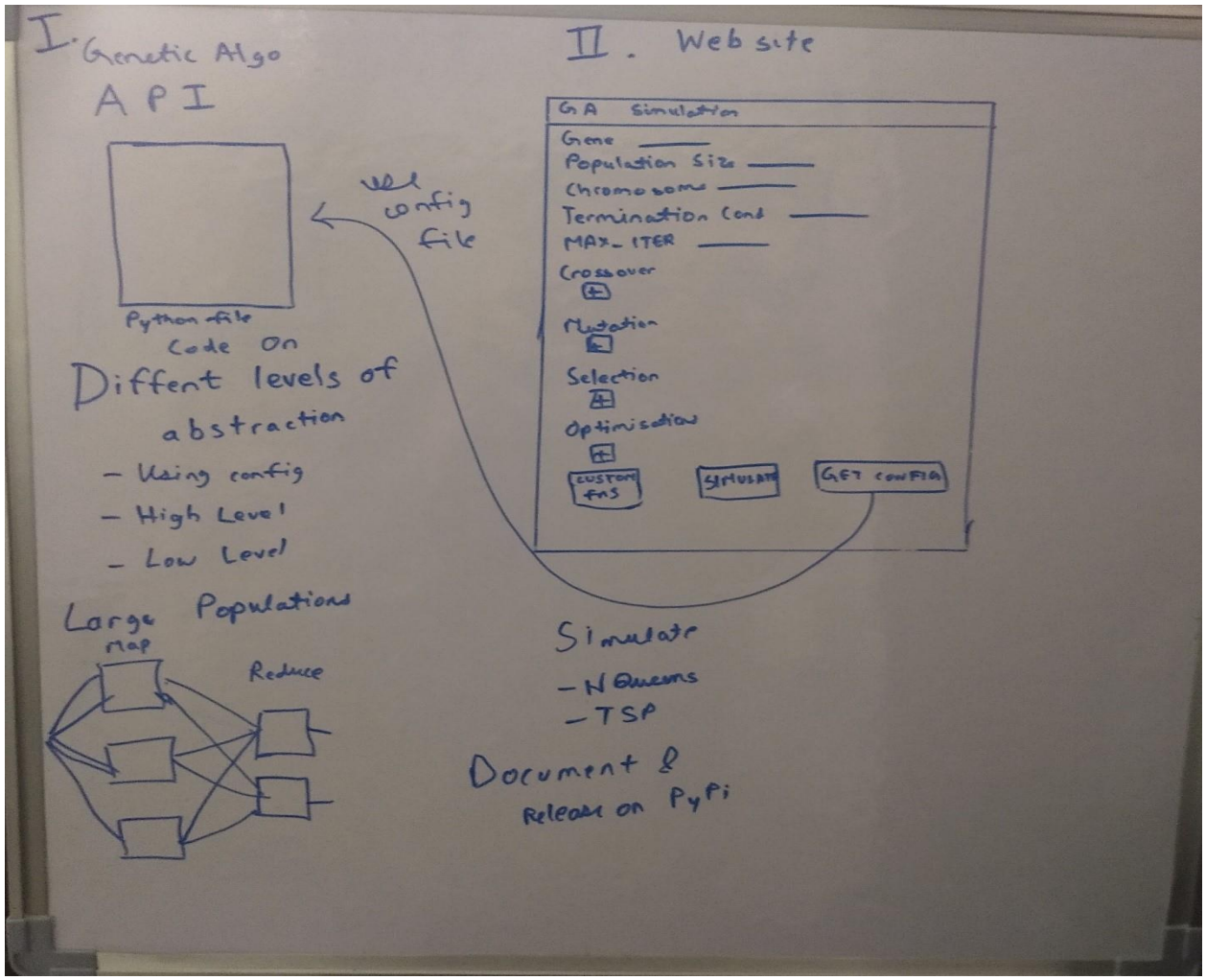
PES UNIVERSITY




Technologies to be used

- API: Python
- Website Backend: Flask
- Website Frontend: HTML, CSS, JavaScript
- Testing: pytest, Travis CI

Basic block diagram



Project Timelines & Plan

Week	Planned Activity
Week 1 – 4	Detailed Literature Survey, API Design Plan
Week 5 - 8	High Level Design, Basic API Coding
Week 9 - 12	API Enhancements Coding, MapReduce, Basic Testing, Basic Web App Front end & Backend using Flask.
Week 13 - 16	Further Enhancements, Advanced Testing, Complete Web Application, Report

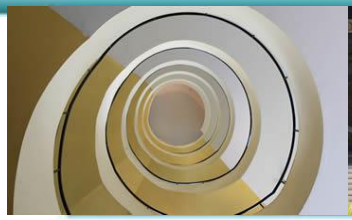
Project Effort Estimation

Week	Bharatraj	Daniel	Shreyas
Feb 1 – Feb 14	Literature Survey (10) API Class Design (25) Documentation (10)	Literature Survey (10) API Class Design (25) Documentation (10)	Literature Survey (10) API Class Design (25) Documentation (10)
Feb 15 – March 15	Coding classes of API (45) Basic class structure testing (35) Documentation (10)	Coding classes of API (45) Implementation of 1 optimisation (35) Documentation (10)	Implementation of 1 optimisation (45) Basic class structure testing (35) Documentation (10)
March 15 – April 17	Implement ANN (40) Implement KNN (40) Documentation (10)	Map Reduce (40) Flask (40) Documentation (10)	Map Reduce (40) GUI (40) Documentation (10)
April 18 – April 22	Final Testing (31)	Web app enhancements – config (31)	Checkpointing (31)

Project Effort Estimation

Activity	Hours/Person
Feasibility Study • Examining Existing Frameworks • Shortcomings of existing frameworks	20
Literature Survey • Papers on GA efficiency • ML algorithms using GA • Parallelization of GA	28
Requirement Specification	20
High Level Design	28
Coding & Implementation • Python API coding • Web application frontend and backend	90
Testing	40
API Documentation	20
Report	10
Total Effort / Person	256

PES UNIVERSITY






Thank You