Problem Statement and Goals
Agolearn

Team 1, Agonaught(s)
Yiding Li

Table 1: Revision History

Date	Developer(s)	Change
2024-01-15	Yiding Li	First Draft

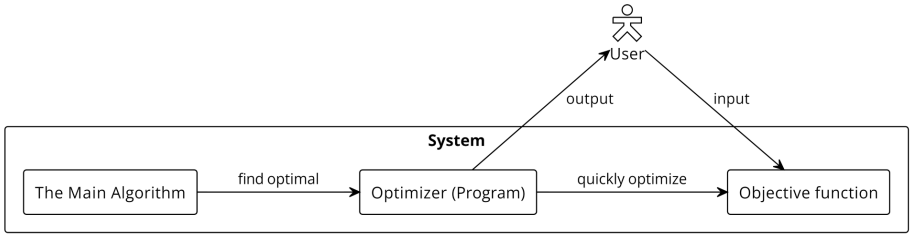
1 Problem Statement

1.1 Motivation

Evolutionary algorithms lend well to solving novel problems that are difficult to model or solve by traditional means. Genetic programming algorithms, in particular, are able to evolve programs that perform well against arbitrary objective functions.

While evolutionary methods lack the efficiency of traditonal optimization methods (such as simplex algorithms and gradient desgent algorithms), they work in more complex environments (such as moving constraints, moving optimization functions, partially observable environments, and objective functions that do not have well-defined derivatives). This project seeks to develop an optimizer that takes advantage of these features.

Figure 1: Input and output of a genetic programming algorithm



1.2 Inputs and Outputs

Input: TODO: Differentiate inputs and outputs for different objectives; if possible, add illustrations.

1. An objective function of either values (vectors of numbers) or functions of real numbers
2. Specifications for genetic programs, such as (a) node functions, (b) tree depth, and (c) node count
3. Parameters that control how the algorithm operates, such as (a) choices of evolutoinary operators, (b) number of episodes, and (c) the length of each episode or truncation conditions

Output: A value or genetic program that optimizes the given objective function

1.3 Stakeholders

TODO: Enrich this section. Computer scientistst that seek to optimize an objective function.

1.4 Environment

TODO: Seek advise and improve this section. A computer.

2 Goals

TODO: Enrich this section. Also, consider moving some from Strech Goals to Goals.

- Optimize against an objective function of values (vectors of real numbers)

3 Stretch Goals

- Optimize against an objective function of functions (genetic programs)
- Implement multi-processing to speed up computation (in reference to frameworks such as Deap)

4 Appendix

4.1 Evolutionary Algorithms

Evolutionary algorithms (**EA**) are optimization algorithms that draw on the evolutionary process. An EA begins with an initial population, then iteratively improves the population through generations by applying various evolutionary operators.

4.2 Evolutionary Operators

Evolutionary operators divide into **parent selectors**, **variators**, and **survivor selectors**. These operators emulate events in an evolutionary process:

- **Parent selectors** select from the population to form the parent pool.
- **Variators** act on the parent pool to produce a pool of offsprings. An offspring may inherit traits from parents (by crossover) or posess novel traits (by mutation).
- **Survivor selectors** select from the offspring pool to form the population for the next generation.

TODO: Figure is wrong and so is the label.

Figure 2: Evolutionary operators in

4.3 Genetic Algorithms

Genetic algorithms (**GA**) are evolutionary algorithms that work with programs. That is, such algorithms evolve *functions* against an objective function that receive functions. Genetic algorithms can evolve agents that behave well in a particular environment (e.g. a bipedal walker) or construct mathematical models (e.g. symbolic regression)

TODO: Tenative: change hyperlinks to numbered references.