Hans Lienhop, 114926
Lucas Hübner, 116232

Exercise 2

Search Based Software Engineering
May 21, 2019

Task 1

a) What are $(1+1)$, $(1+\lambda)$, $(1,\lambda)$ and (μ,λ)?
They describe instances of evolutionary strategies.

b) Explain differences between them.

$(1+1)$: One parent produces one candidate solution and parent and child compete based on objective fitness for a position in the next generation.

$(1+\lambda)$: One parent produces λ candidate solutions and parent and children compete based on objective fitness for a position in the next generation.

$(1,\lambda)$: One parent produces λ candidate solutions and only the children compete based on objective fitness for a position in the next generation while the parent is disregarded.

(μ,λ): The μ fittest parents from randomly selected λ individuals from the population produce $\frac{\lambda}{\mu}$ candidate solutions each and these children form the new population.

Task 2

a) How does Line Recombination work? Explain in detail.
In genetic algorithms, children are bred by performing crossover operations using the copies of two parents and possibly mutating the results. The problem is that the children are limited by the hyperspace spanned by the parents. The best solution may be outside this hyperspace. Line recombination offers a method to explore solution outside this hyperspace. Line recombination can generate any point on the line defined by the two parents. First, a positive value p is set that defines, how far the hyperspace can be outreached. Next, two random values α and β between $-p$ and $1+p$ are generated. These values are used to create new values based on a combination of the parents. With two parent vectors $\overrightarrow{x}$ and $\overrightarrow{v}$ containing float values, all of their individual values x_i and v_i will be replaced with weighted recombinations. The new x_i value will be $\alpha x_i + (1-\alpha)v_i$ and the new v_i will be $\beta v_i + (1-\beta)x_i$. This results in two new points which are either on the line spanned by the to original points or exceeding it by at most p. This way, values outside of the given hyperspace can be reached.

b) How can Line Recombination be extended to get Intermediate Line Recombination?
It can be extended by choosing new α and β values for each new set of x_i and v_i. This way, children are not limited to the line vector, but can take values from the entire hypercube.

c) Implement Intermediate Line Recombination as a python function.
Check tasks/linearRecomb.py

Task 3

a) What is Fitness-Proportionate Selection (FPS) and how does it work?
Fitness Proportionate Selection is a way of parent selection in which every individual can become a parent with a probability which is proportional to its fitness. So an individual with a higher fitness has a higher probability of being picked. To achieve that, two vectors are necessary. One contains the individuals, the other their respective fitness values in the same order. The sum l of all fitness values is calculated. Thus, an individual is represented by a range. Then, a random number n between 0 and l is generated. The range to which the value n corresponds is found and the respecting individual returned to act as parent.

b) What is Stochastic Universal Sampling (SUS) and how does it work?
Stochastic Universal Sampling uses the same initialization as FPS but ensures that fitter-than-average individuals are chosen at least once. After initializing the vectors and calculating the sum s from n individuals, a random number between 0 and $\frac{s}{n}$ is generated, which serves as the first parent. This position is incremented by $\frac{s}{n}$ until n individuals are selected.

c) Implement SUS as a python function.
Check tasks/sus.py