**Helwan University Faculty of Computers & Artificial Intelligence – General Programme Computer Science Department – Module: CS361 Artificial Intelligence – Fall “Semester 1” 2022-2023**

**13) Solving the Knapsack Problem using both Genetic Algorithms & Differential Evolution (Solve both the 0-1 Knapsack Problem and the Unbounded Knapsack Problem).**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Name** | **ID** | **Level** | **div** |
| **1** | **محمد حمدى سعد عبد الراضى** | **202000746** | **3** | **is/**cs |
| **2** | **أروى سامح عبد الغفار محمد** | **202000104** | **3** | **is/**cs |
| **3** | **كريم على السيد محمد** | **202000667** | **3** | **is/**cs |
| **4** | **خلود جمال الدين صبحى** | **202000288** | **3** | **is/**cs |
| **5** | **سما محسن حسين محمد** | **202000401** | **3** | **is/**cs |
| **6** | **ضحى محمد رمضان احمد** | **202000473** | **3** | **is/**cs |

**Shared folder link: https://github.com/KarimALI88/final\_knapsack**

**>>Project idea and overview**

**The idea of the project is a knapsack problem, knapsack problem is a problem in combinatorial optimization: Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible.**

**>>Applications**

**Knapsack problem (KP) has broad applications in different fields such as machine scheduling, space allocation, and asset optimization.**

**• One application of this algorithm is int download managers (e.g.: Internet Download Manager).**

**->The data is broken into chunks. As per the maximum size of data that can be retrieved in one go, the server uses this algorithm and packs the chunks so as to utilize the full size limit.**

**This algorithm is one of the many algorithms that download managers use apart from compressing, encrypting etc.**

**• Home Energy Management.**

**• Cognitive Radio Networks.**

**• Resource management in software.**

**• Large-scale multi-period precedence constrained knapsack problem: A mining application.**

**• relay selection in secure cooperative wireless communication.**

**• power allocation management**

**>>Resources**

**1…this paper proposes an innovative model based on dynamic expectation efficiency and establishes a new optimization algorithm of 0-1 knapsack problem after analysis and research.**

**The algorithm description is:**

* **Those objects of higher price-performance ratio are loaded into backpack in advance, the others do expectation efficiency operation**
* **These objects that have higher expectation efficiency value are loaded backpack in advance, and the lower is abandoned**

**The link of the paper:**

[**https://www.researchgate.net/publication/276494351\_An\_Algorithm\_of\_0-1\_Knapsack\_Problem\_Based\_on\_Economic\_Model**](https://www.researchgate.net/publication/276494351_An_Algorithm_of_0-1_Knapsack_Problem_Based_on_Economic_Model)

**2...Knapsack (multi objective) is usually used in multi-objective engineering optimization which has a lot of non-measurable objectives that must be achieved, known as MOP in contrast to single solution which is known as SOP. Instead of finding one individual solution, it is used to find multiple solutions, and these solutions are known as Pareto. The decision maker provides the appropriate conditions to choose the appropriate solution, the highest and best among all solutions, and uses EAs algorithms to find Pareto solutions. One of the most important problems in multi objective is that a relatively large number of solutions must be evaluated to find the best results, and this requires a large number of things that must be population. This problem is called moga, and the alternative solution to it is called parallel-izing the MOGA.This makes the processor look for different solutions instead of following a similar goal. There are many models that are used to implement parallel-izing the MOGA, and the measurement is made according to the quality of convergence and time. The knapsack problem is of great scientific importance. The knapsack 0/1 problem consists of a set of elements, the weight and profits associated with each element, and the upper limit of the capacity of the bag. The task is to find a subset of items that maximize the total profit in the subset, yet all of the specified items t are in the bag, that is, the total weight does not exceed the specified capacity This single objective problem can be directly extended to a multi-purpose issue By allowing an arbitrary number of bags.**

**The link of the paper:**

[**https://www.scirp.org/html/5071.html**](https://www.scirp.org/html/5071.html)

**3…Knapsack problem in the field of combinatorial optimization is a typical, easy-to-describe but difficult to deal with NP-complete problems. Knapsack problem for large-scale, are not yet very effective way, is widely used in engineering practice.**

**The problem of the backpack, KP also noted is a combinatorial optimization problem. RSA is considered the first true asymmetric encryption algorithm for this problem. The problem of the backpack is one of 21 NP-complete problems by Richard Karp. outlined in his 1972 paper. NP-hard version of this problem has been used in primitive and cryptographic protocols, such as Merkle-Hellman cryptosystem or Chor-Rivest cryptosystem. The problem of the backpack is a classic example of misunderstanding regarding the relationship between the NP-completeness and cryptography. Total weight of the items in the backpack is open, is open all possible items, but items in the backpack is confidential. Asymmetric encryption algorithms based on the backpack were all broken so far, the latest being that of Chor-Rivest. Knapsack problem is not well-known computational problems, backpack system with its encryption, decryption, fast and eye-catching.**

**The link of the paper:**

[**https://www.sciencedirect.com/science/article/pii/S1878029611008486**](https://www.sciencedirect.com/science/article/pii/S1878029611008486)

**4…The Knapsack Problem is an example of a combinatorial optimization problem, which seeks to maximize the benefit of objects in a knapsack without exceeding its capacity.**

**In the program, we implemented two selection functions, roulette-wheel and group selection. The results from both of them differed depending on whether we used elitism or not. Elitism significantly improved the performance of the roulette-wheel function.**

**we tested the program with different crossover ratios and single and double crossover points but the results given were not that different.**

**we use Genetic Algorithms to solve the 0-1Knapsack problem where one has to maximize the benefit of objects in a knapsack without exceeding its capacity.**

**Genetic Algorithms definitely rule them all and prove to be the best approach in obtaining solutions to problems traditionally thought of as computationally infeasible such as the knapsack problem**

**The link of the paper:**

[**http://www.sc.ehu.es/ccwbayes/docencia/kzmm/files/AG-knapsack.pdf**](http://www.sc.ehu.es/ccwbayes/docencia/kzmm/files/AG-knapsack.pdf)

**5…In this paper, an artificial glowworm swarm optimization algorithm for solving 0-1 knapsack problem is proposed, and the detailed realization of the algorithm is illustrated.**

**Knapsack problem (KP) is a classic combinatorial optimization NP-complete problem, The research focuses on knapsack problem has higher theory and practical value,selection, resource allocation and loading, etc in financial and industrial fields and combinatorial mathematics, computation complexity theory, cryptography and applied mathematics Recently**

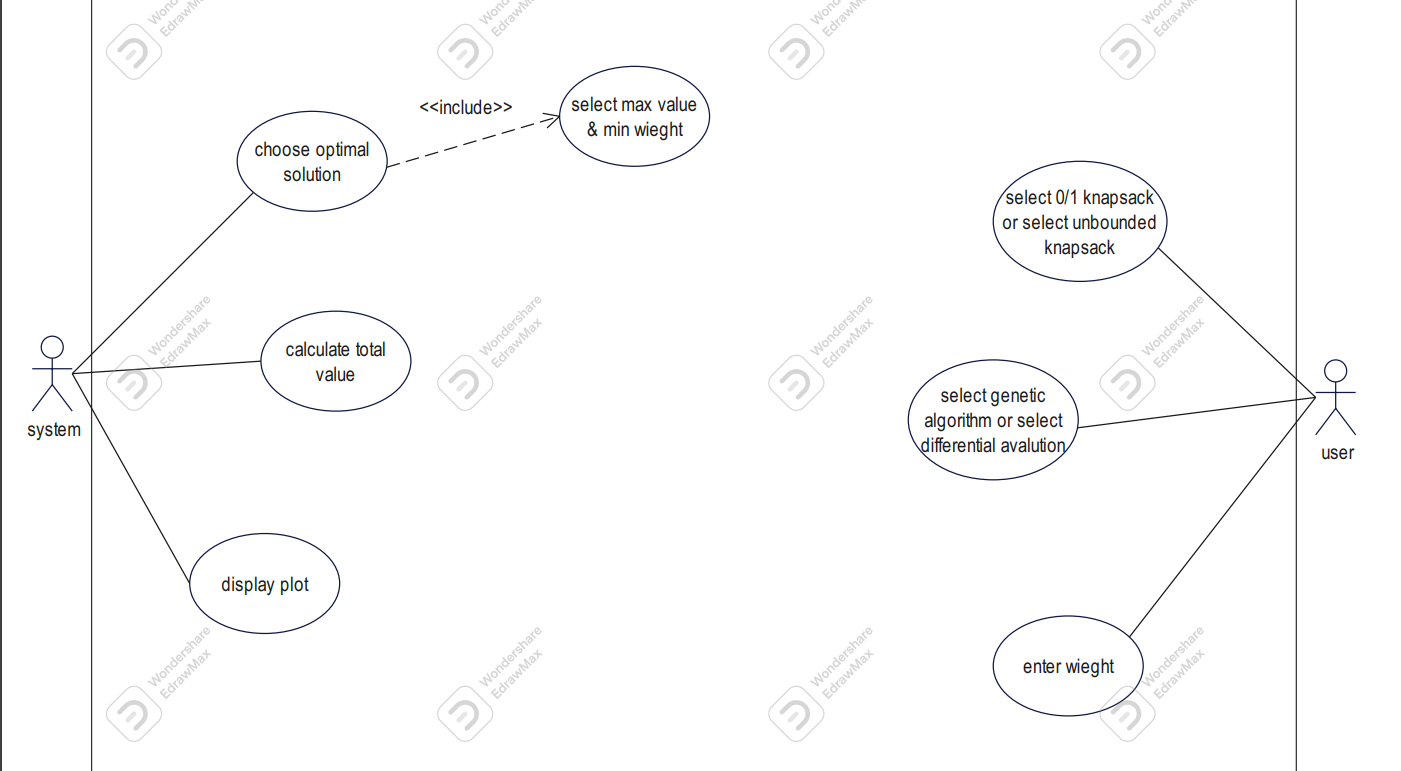
**The process of the research shows AGSO is more superior than particle swarm optimization algorithm. in multi-modal functions optimization.**

**The link of the paper:**

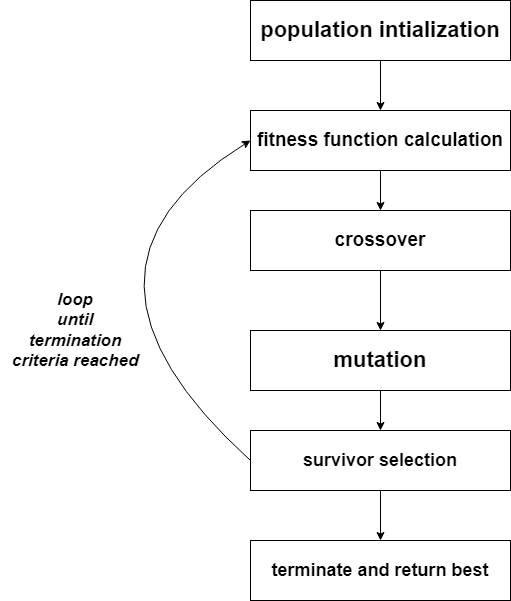
[**https://images.app.goo.gl/aaLxcvxAbGFCHZwy6**](https://images.app.goo.gl/aaLxcvxAbGFCHZwy6)

**…………………………………………………………………………………………………………..**

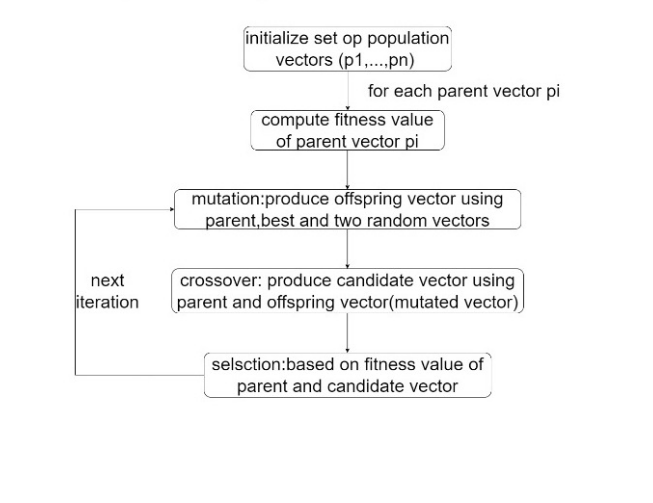
**Use-case diagram:**



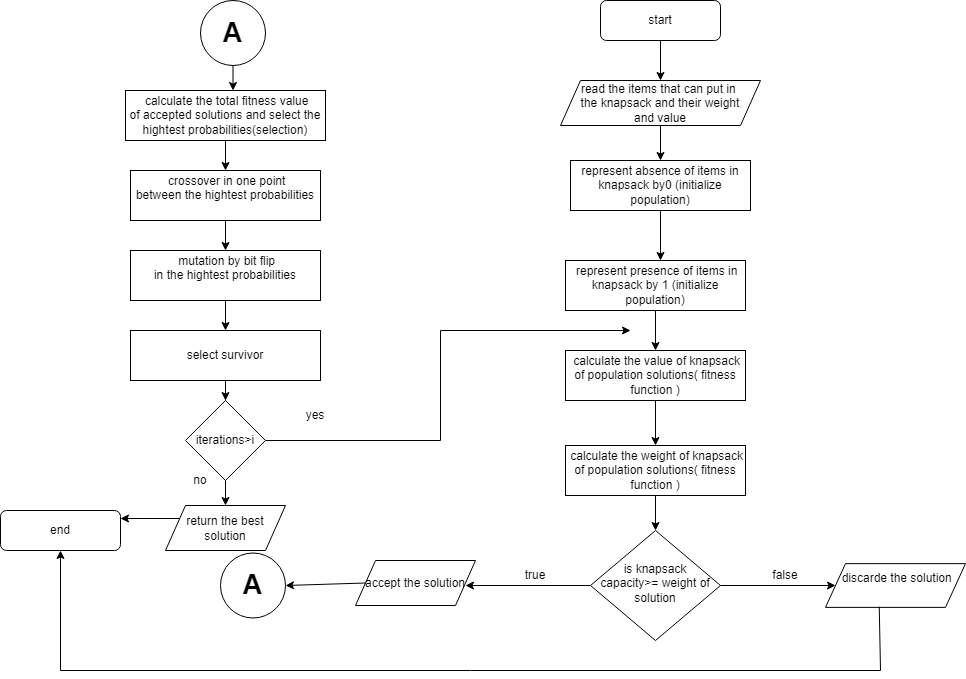
**Block diagram genetic algorithms:**



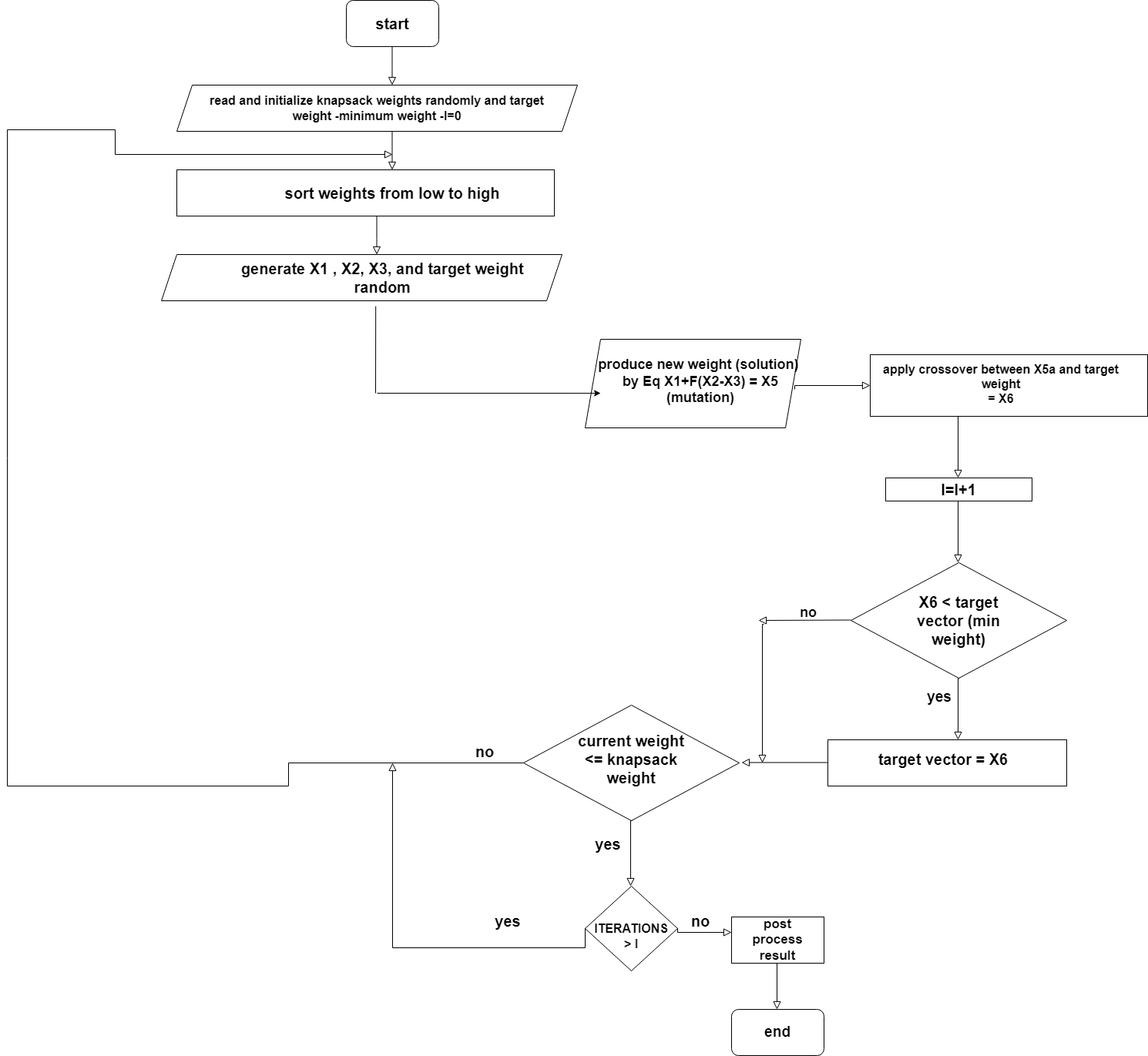
**Block diagram Differential Evolution algorithms:**



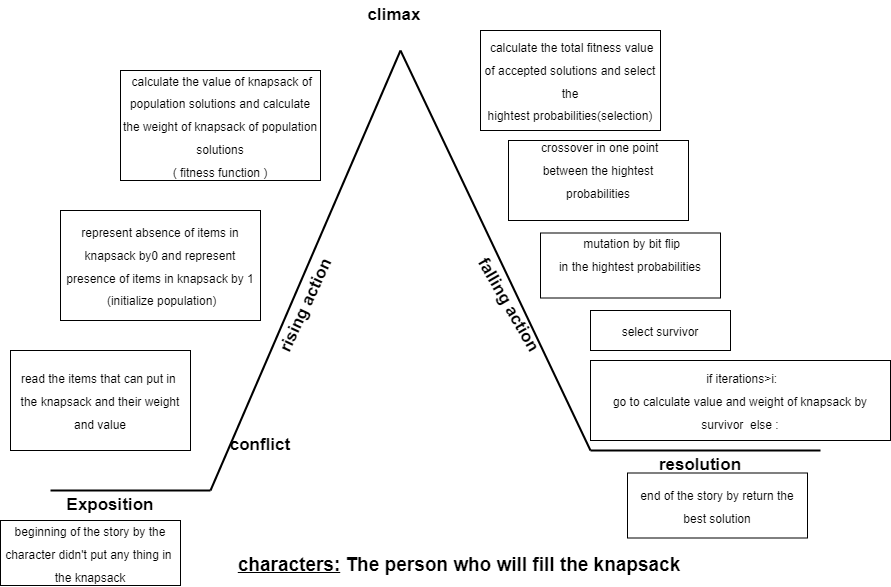
**Flowchart diagram genetic algorithms:**



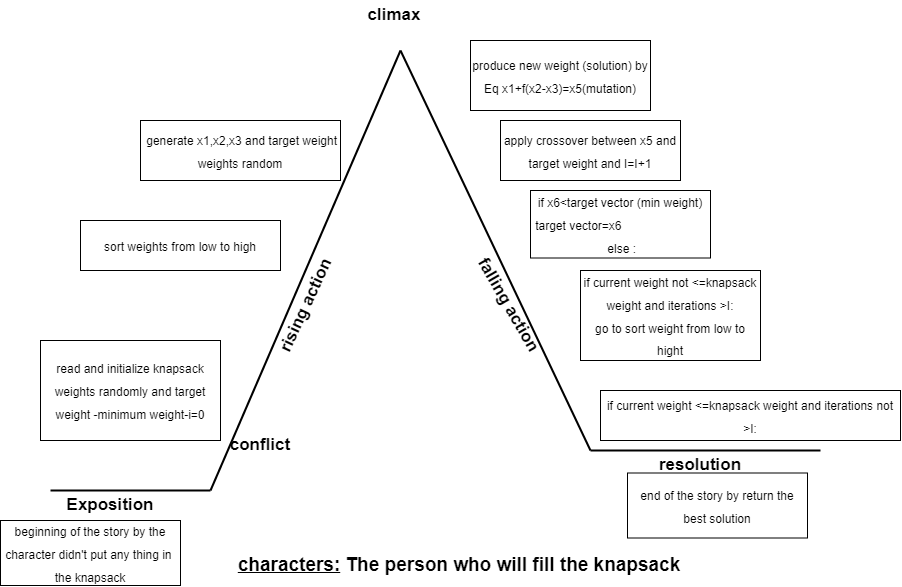
**Flowchart diagram Differential Evolution algorithms:**



**plot diagram genetic algorithms:**



**plot diagram Differential Evolution algorithms:**



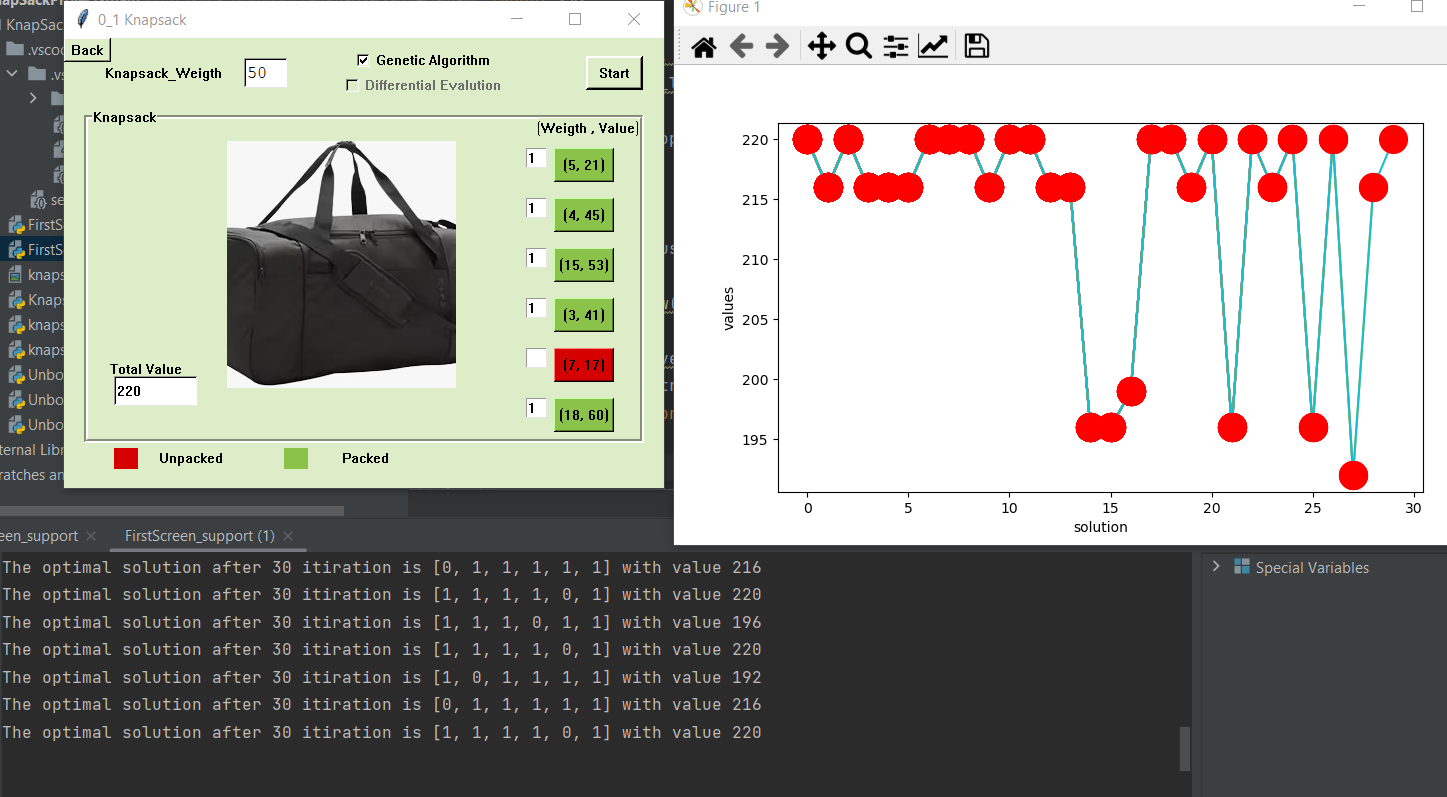
**Experiments & Results**

**After several attempts of solving experiments that differ from one algorithm to another we reached the results that give us the largest value that can be found inside the knapsack for a certain weight as shown in the results of the codes.**

**Analysis of the results**

**For example: in 0/1 knapsack (genetic algorithm/** **Differential Evolution)**

**It give us many result but get the optimal solution after 30 steps nearly**



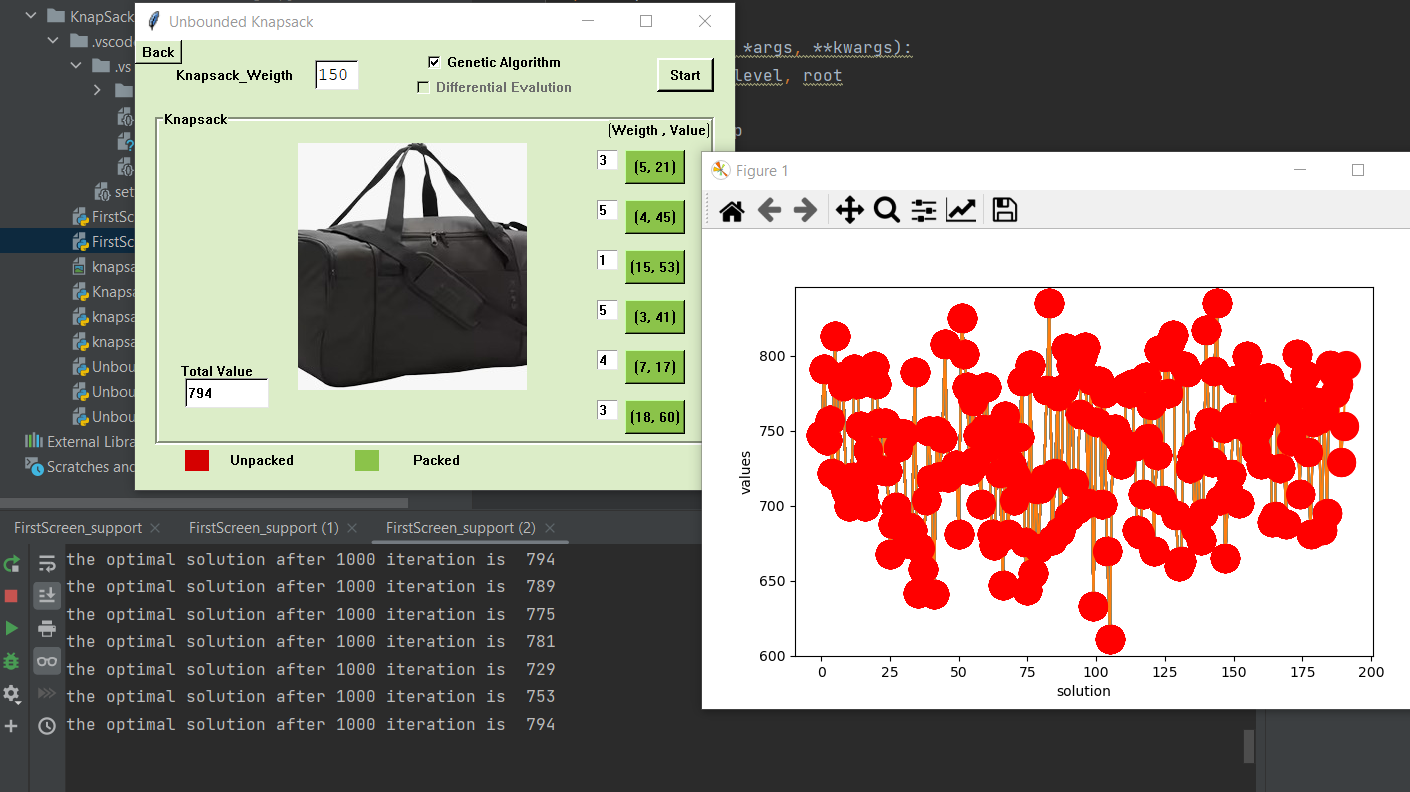
**But its problems when increasing the weight of the knapsack with the same values ​​and weights for the things that can enter the knapsack , it gives us all the existing values ​​and becomes the the optimal solution.**

**It means that the problem of (0/1) is that the object either enters the knapsack or does not enter the knapsack .....Let's assume that the weight of the things available to us is inserted into the knapsack=52.** **If the weight of the knapsack = 52 or more, we will get the same optimal solution**

***\*\*genetic algorithm Reach the optimal solutions faster than differential***

**in unbounded knapsack (genetic algorithm/ Differential Evolution)**

**It give us many result but get the optimal solution after 1000 steps nearly**



**One of its advantages is that it fills the bag with more than one item of the same type**

**It means that the (unbounded ) is that the object either enters the knapsack or does not enter the knapsack or take a lot of it .....Let's assume that the weight of the things available to us is inserted into the knapsack=52. If the weight of the knapsack = 125 or more, we will get the optimal solution**

**…………………………………………………………………………………………………………**

**We test the optimal solution by get the hightest value** **things in the knapsack and the weight of things =< weight of knapsack and the algorithm will exclude the things that will not be achieved that.**

**…………………………………………………………………………………………………..**

**the future modifications i’d like to try when solving this problem:**

**...I will try to minimize attempts to reach the optimal solution, especially in unbounded knapsack**

**…Increase the number of things that can be added in knapsack**