# Methods and materials

## Manipulator arms

### Definition

Each manipulator arm configuration () consists of three features: joint, joint axis and link length. The joints are divided into three types:

* Roll – Revolute joint about Z-axis
* Pitch – Revolute joint about Y-axis
* Prismatic – Linear sliding along Z-axis

The Joint axis is defined relative to the previous coordinate system, therefore according to the previous joint axis.

The link length is the length of each link along the Z-axis of its coordinate system.

Figure 1 gives an example for Roll\_Z\_0.1 🡪Pitch\_Y\_0.4 🡪 Prismatic\_Y\_0.7.

X

Y

Z

X

Y

Z

Y

X

Z

Figure 1- Configuration Example

### Parameters

For the construction of each configuration the following parameters have been set:

* All the links are cylindrical where at radius of 0.05m, 0.045m, 0.04m, 0.035m, 0.03m, 0.025m for 1,2,3,4,5,6 DOF, respectively.
* For Revolute joints (Roll and Pitch) the joint range is [-PI, PI] and for prismatic joints, the range is [0, 2\*link length]
* All the configuration will be between 3-6 DOF
* The length of the links will be 0.1, 0.4, or 0.7 meters
* Success: the arm need to reach the middle and the lower points and one of the top points (Figure \*\*\*)

This gives about 34,636,800 theoretic configurations.

### Assumptions

In order to reduce the number of configurations and to fit the specific problem some assumptions have been made:

* The first joint is rotational along Z ax (Roll\_Z)
* The first link length = 0.1m
* The total length of all the links > 1m
* 2 adjacent prismatic joints must be perpendiculars
* No more than 3 prismatic joints
* After a Roll joint, the next joint will not be Roll\Pitch joint in the Z-axis
* After a Roll joint, the next joint will not be in the X-axis
* After a Pris Z joint that its previous joint is a Roll joint, the next joint will not be in the X-axis

those assumptions reduced the available manipulators to 1,695,044 configurations.

### Indices of manipulator performance

In order to make comparisons between the configurations and to be set as the optimization objectives several manipulators Indices and other indices have been checked:

All the indices that been checked are local indices, meaning indices that only depend on a specific position of the arm or the arm structure. Because the indices are calculated for several points the point with the lowest grade was selected as the grade of the configuration.

The chosen indices from the literature presented in Table 1.

Since the upper limit of the Mid-Range Proximity Index is different for each configuration and in order to regard the differences between the type of joints, the value of the joint current position was normalized by the joint range, which bounds the index between [0 – 0.5].

Table - indices to examine



## Simulation

### Simulator

Gazebo simulator and MoveIt have been used for motion planning, inverse kinematics, control, and collision checking, in order to simulate different configurations of robotic arms. Robot Operating System (ROS) has been used to connect between those platforms and simplify the process.

Gazebo is a simulator tool that simulates the real world and the environment where the selected manipulator arm will work. In the simulation, the plant is modeled as a cylinder at a height of 0.75 meters and a radius of 0.5 meters (figure ???).

Moveit is a tool to calculate motion planning and control the movement of each configuration in the Gazebo simulator. the motion planning algorithm that selected for each configuration is rapidly exploring random tree (RRT) algorithm, which can handle problems with obstacles and differential constraints. RRT isn't finding the optimal path but find a valid path in a short time. In the simulation, the simulation running time is very important and the RRT has been limited to 2 seconds to find a possible path.

The dynamic and kinematic structure that is used in ROS and Gazebo, to present arms, is Universal Robotics Description Format (URDF). Because the URDF also considers the dynamic part of the arm there is necessary to calculate the weight of each link. The weight of each link is calculated by the ratio between the accumulated length to the accumulated weight of the link- the ratio was calculated according to 2 different types of manipulators: UR5 and MOTOMAN YR-MH005LN (figure \*\*\*)

In the simulator, 4 points have been chosen to be reached by the arm. All the points are at a distance of 30 cm from the plant and in a different orientation.

\*\*\*\* Add the points + plant image with coordinate frame \*\*\*

Since the manipulator will be mounted on a mobile platform that will drive through the greenhouse, a prismatic joint, parallel to the plant for each configuration, was added in the base of the arm. The range of this prismatic joint is 1.5 meters so it will be able to move before in after the plant and the mobile platform position won’t be the reason that the configuration didn’t reach the desired point.

On the arm end effector, a multi-spectral camera will be mounted in order to conduct early detection of biotic and abiotic stresses in specialty crops, for this purpose, the image orientation is not important, therefore, a rotation joint on Z-axis was added to the end of each arm.

## Optimization

### the set-based concept approach

As mentioned earlier there are 1,701,647 configurations. with an average simulation run time of 15 seconds per configuration, examining all configuration will take about 300 days on a Virtual Machine with the following specification:

Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz, 8GB RAM and 90GB of storage.

Therefore, an optimization method was developed in order to find the optimal configuration in a reasonable time. In this case, the set-based concept approach, which combined configurations with the same properties in one design concept. In this specific problem, there are almost 800 different concepts and it will be time-consuming to make a deep search of each concept. therefore, a multi-objective evolutionary search for satisficing concepts based on dynamic window-of-interest was developed. the aim is to reveal which of the concepts have at least one solution with a performance vector within a dynamically changed window-of-interest. In the first part, a predefined computing time allocated to find satisficing concepts that would be explored more deeply in the second part of the optimization.

In the second part, the concepts that satisficed in the previous part are going to be explored more widely and to find the Pareto front of the problem by using a multi-objective evolutionary algorithm.

### Objectives

The Objectives of the optimization problem selected from the indices of the previous section and are the following:

* Maximum manipulability [0-1] (f1)
* Minimum Z (Mid-Range Proximity) [0 -0.5] (f2)
* Minimum degrees of Freedom [4 - 6] (f3)

This optimization problem is mixed with minimum and maximum objectives, to change the problem to be minimum problem f1 objective has been changed as follow:

Min (1 – manipulability) [0-1] (f1)

### Independent variables:

The independent variables of the optimization problem are:

* X1: Joints Types: array [Roll, Pitch, Prismatic]
* X2: Previous ax: array [X, Y, Z]
* X3: Links Lengths: array [0.1 ,0.4, 0.7] (meters)
* X4: Number Degrees of Freedom: Int [3, 4, 5, 6]

### Constrains

The constraints of the optimization problem are derived from the assumptions of the simulator section and are as follows

1. The first joint is rotational along with Z-axis:

X1[0] = Roll, X2[0] = Z

1. First link length = 0.1m :

X3[0]=0.1

1. The total length of all the links > 1m :

Sum (X3) > 1

1. 2 adjacent prismatic joints must be perpendiculars:

If X1[i-1]==Prismatic X1[i]==Prismatic than X2[i]!=Z

1. No more than 3 prismatic joints: \*\*\*\*\*\*
2. After Roll joint can’t be Roll\Pitch joint in the Z-axis:

If X1[i]==Roll and (X1[i+1]==Roll or X1[i+1]==Pitch) than X2[i+1]!=Z

1. After a Roll joint, the next joint will not be in the X-axis

if (X1[i]==Roll) than X2[i+1]!=X

1. After a Pris Z joint that its previous joint is a Roll joint, the next joint will not be in the X-axis

if ( (X1[i-1]==Roll and X1[i]== Pris and X2[i] == Z)) than X2[i+1]!=X

1. Joints limits: for Roll\ Pitch [0-360°] and for Pris [0 – 2\*link length]
2. Number of detection points: 4
3. Success: the arm need to reach to one of the 2 top points and to the middle and the lower points and one of the top points

### Concepts

Each concept is defined from the following variables:

1. DOF – Number degrees of freedom of the configuration
2. Pitch joints – Number of pitch joint in the configuration
3. Long Link - Number of long links (0.7m) in the configuration
4. Acc length – Accumulated length of all the links of the configuration. This variable is defined as a range of distances
5. # of Parallel axes Y – Number of parallel axes along world Y ax in a row
6. Longest Link – What is the longest link in the configuration
7. P/R ratio – The ratio between Prismatic joints to Revolute joints (Roll\Pitch)

All the variables and their possible values are presented in Table 1.

Not all the combinations between the variables are possible. For example, it's not possible to concept to be with 4 DOF and 5 Pitch joint.

The total amount of concepts that received is 794. The number of configurations in each concept varies from 1 to 68520.

Table - Concepts



## An interface between the simulator and optimization algorithms

In order the simulator will be able to communicate with an optimization algorithm, an interface was built in Python. The interface is built from two parts, the configuration builder and indices calculator. In Figure 3 it can be seen a scheme of the interaction between the interface and the simulator and the optimization method.

### Configuration builder

The configuration builder gets from the optimization algorithm the selected variables by the algorithm to be simulated (DOF, Joints Types, Joints axes, Links Lengths) and creates from this data a URDF file which contains the kinematic and dynamic representation of the manipulator. After creating the URDF file the interface enters it with the predefined detection points into the simulator.

### Indices Calculator

The indices calculator gets in return, from the simulator, if the configuration succeeded to reach the desired points and if it’s succeeded what was the Jacobian and joints position at every point. The indices calculator uses the Jacobian in order to calculate the manipulability index and the joints position to calculate the mid proximity joint index.

After calculating the manipulability index and the mid proximity joint index, the indices calculator returns those indices to the optimization algorithm in order to evaluate this configuration.

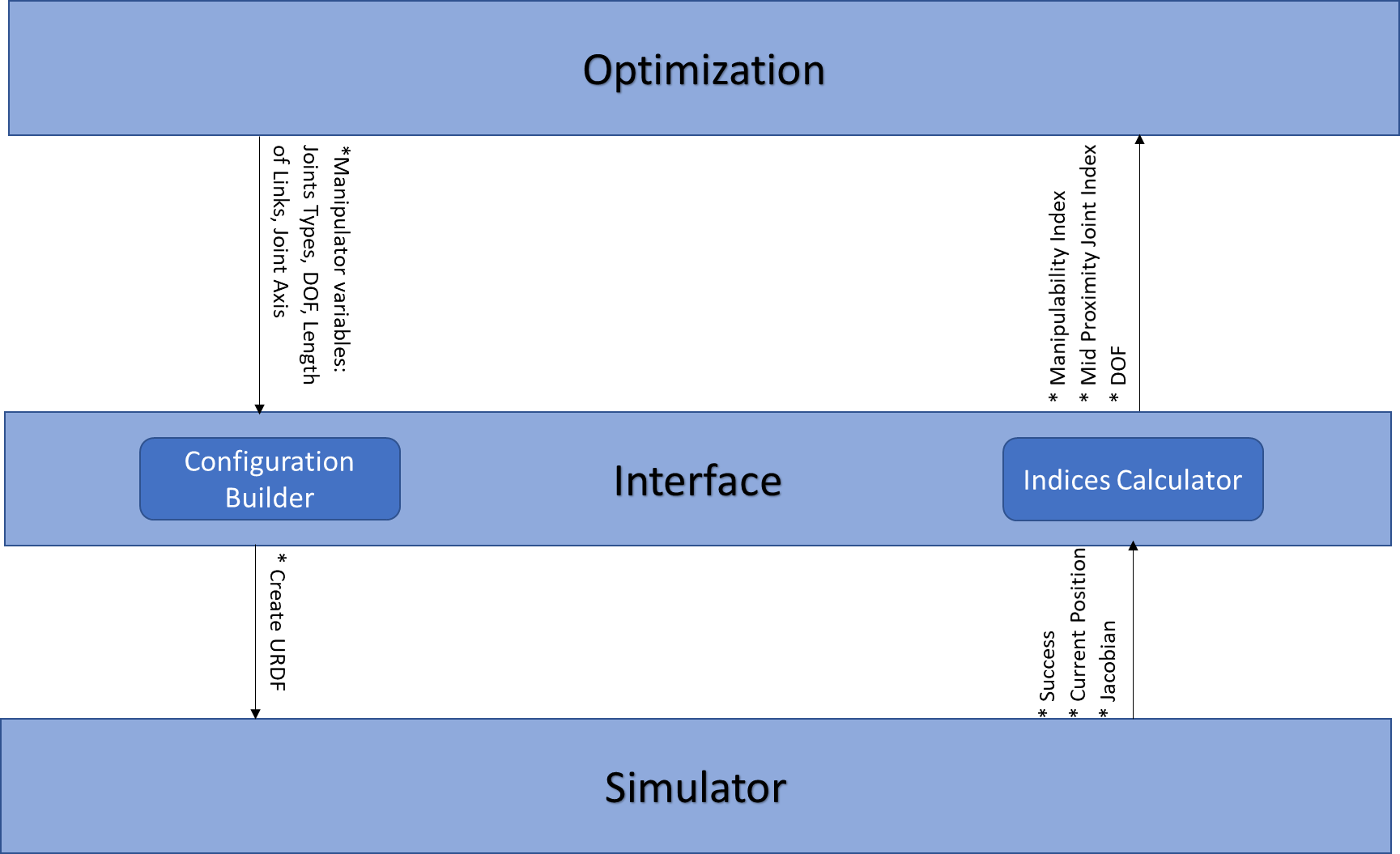


Figure - Interface between optimization to the simulator

## Build a preliminary Window of Interest

As described before, the optimization is conducted in 2 steps: the first step is to find several concepts from the 794 concepts that satisficing The concept that satisficing is a concept that is at the Window Of Interest (WOI). In order to start this step in reasonable WOI, random configurations were simulated.

### Configuration selection

The configurations were selected from all the concepts of inconceivable selection. The selection is done as follows:

* Simulation of 60000 selected configurations - One week of computer time (there were 1.5 available computers) – each configuration takes about 15 seconds to simulate
* From concepts with less than 25 configurations, all the configurations selected
* From concepts with more than 25 configurations, selected 25 configurations or 3% of the configuration, what higher
* All the selected configurations were tested in the simulator

### Pareto Front

After all the selected configurations were simulated a Pareto front is calculated.

The Pareto Front was calculated as follow:

* All the configurations are plotted in a 3D space according to their results when the axes are the 3 objectives: Manipulability Index, Mid-Range Proximity Index, and DOF.
* All the ‘Non-Dominated’ configurations building the Pareto front

The calculated Pareto front can be built from several concepts, mustn't be from one concept, and will be set as the WOI of the next part.

## Evolutionary optimization with Dynamic-Window of interest

In this type of algorithms, the WOI is dynamic (DWOI), meanings that WOI updated during the processes and continues to approach the origin. The way of calculation the initial WOI described in the previous section. The mating is done only inside each concept, isn’t done between concepts. Concepts with a small number of configurations the selection will be randomly and concepts with a large number of configurations the selection will be done by the genetic algorithm.

### Genetic Algorithm

In this part, a Genetic Algorithm (GA) is used to find the concepts that satisficing the DWOI, and will be good enough to continue to the next part that will find the optimal configuration from the satisficing concepts.

The GA runs only inside each concept and has no effect on other concepts except for changing the DWOI.

The GA that used in this case is designed as following (figure 5):

In order to speed up the genetic algorithm, all the concepts with up to 750 configurations were simulated before, this action takes 14 days of computation time and all the concepts with 5dof simulated before as well, which takes 5 more days of computation and helps to handle the resources.

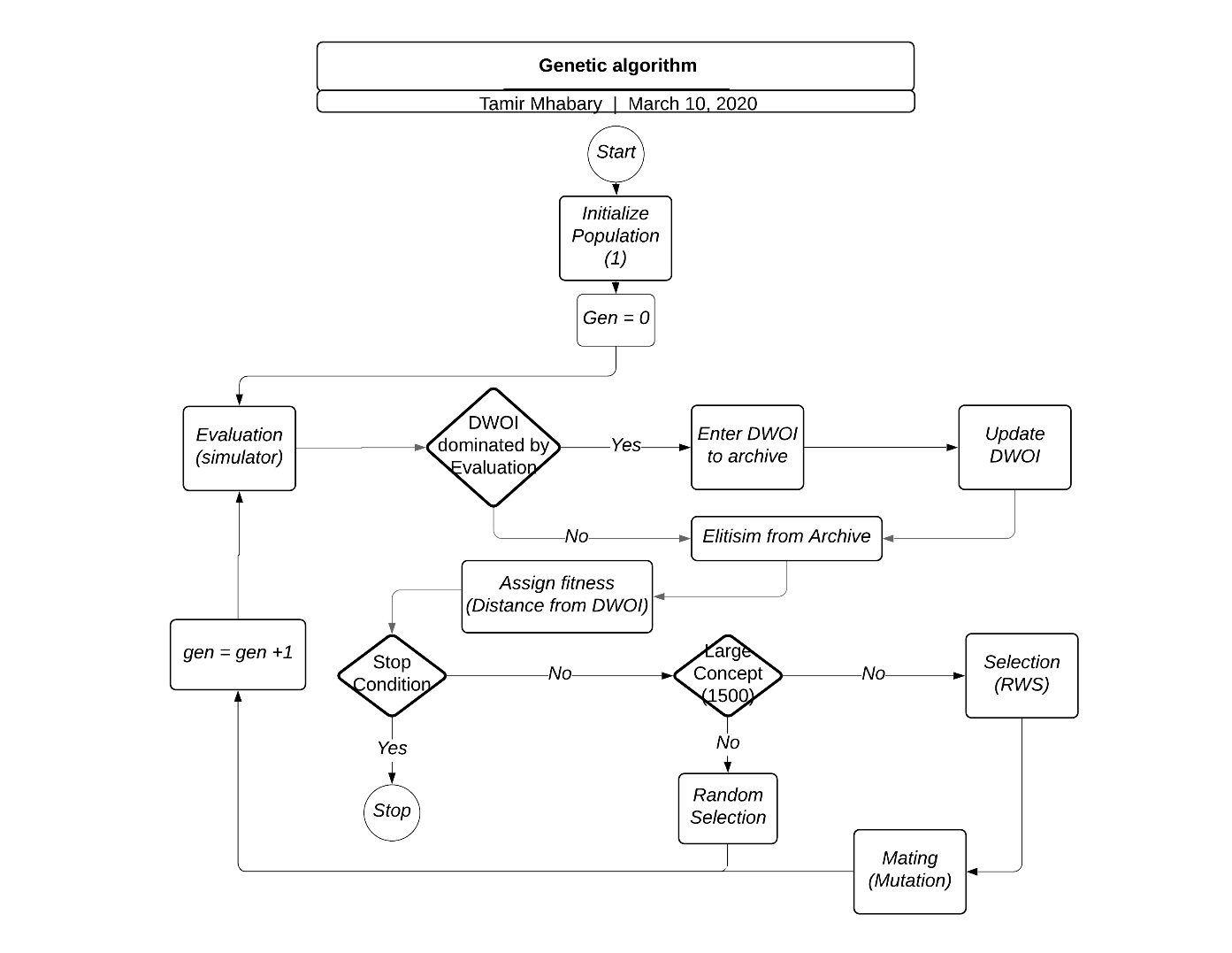


Figure - Genetic Algorithm

#### Initializing & Evaluation

Each concept starts with a random population of one. The evaluation is done by simulation of the selected configurations and calculation of their indices.

#### DWOI

The calculated indices are checked, and if one of the configurations is dominating one of the configurations in the DWOI it replaces it. After the update of the DWOI, the old DWOI is entered into the archive. After the domination check and DWOI update (if needed) a fitness will be assigned to each configuration.

#### Elitism

Before Assign Fitness elitism is performed. The elitism is done by using an archive with previous Non-dominated results and it guaranty that only parents with good genes will enter into the mating pool.

#### Assign Fitness

The fitness is assigned by calculating Euclidean distance for each configuration from the DWOI when the configurations with the smallest distance are getting the higher fitness. The fitness is calculated as follow:

When is the fitness of the i-th element and d is the distance from the DWOI.

#### Stop Condition

After the assign fitness step, there is a stop condition step. In this step, it's checked if one of the following conditions are fulfilled:

1. All the configurations in the concepts are evaluated
2. The mating doesn’t succeed to generate more offsprings
3. ~~Spreading of the Non-dominated results: if the \*\*\*~~
4. Arrived at Gen = X
5. If the predefined time of \*\*\*\* is passed.

Conditions 1-3 are local stop conditions, it means that they are stopping only the concept and not all the process as conditions 4-5.

If the stop conditions aren’t fulfilled the algorithm continues for creating new configurations.

#### Selection

If the concept is a small concept (a concept with less than 1500 configurations) than the new configurations are selected randomly from the concept’s configurations. But if the concept is a large concept, then the configurations' fitnesses are entered into the selection, which performed by Roulette Wheel Selection (RWS).

#### Mating

The mating step builds only from mutation.

Each configuration disassembled to parts in the same number as it DOF.

Each part of the configuration contains Joint, Axes and Link length. For example, Roll z 0.1.

As shown in fig 4, each part gets a suited number between 1-29 (except 10 & 20), and this number will be its id number.

After each part gets its id, two types of neighbors were calculated. The 1st type is first-order neighbors when changing only one feature in the part (axes or link length) gives us another part which doesn’t give configuration outside the concept, as if the Joint type changed or link length is 0.7 do. The 2nd type is second-order neighbors when two features are changed to get part from another part.

For example, id 1- Roll z 0.1 the first-order neighbors are id 2- Roll z 0.4, id4- Roll y 0.1 and id7 – Roll x 0.1. The second order for id1- are id 5- Roll y 0.4 and id 8- Roll x 0.4.

The mutation is performed as a following:

1) Select Random Number between 1-5 ( arm index)

2) Select Random neighbor from the first neighbors

3) In the selected arm index replace the link with the selected neighbor

4) Check the new configuration is inside the concept and not simulated before

5) If 4 is true continue

6) Else return to 1

\* In advanced generations step 2 include also the second neighbors \*\*\*\*\*

Table - Parts ID



Table - ID and its Neighbors



### Resource Allocation

In order to get the best results 2 types of resource allocation heuristics been tested: Fair and Greedy. The two heuristics were tested on the same GA, given the same time of computation and run on the same hardware, in order to prevent bias to one of the heuristics.

In both heuristics concepts that inside the DWOI will not get resources as long as they inside the DWOI.

#### Fair Heuristic

Each concept gets the same resources regardless of its population size and how it progresses. The rationale behind this heuristic is all the concepts need to be treated in the same way.

#### Greedy Heuristic

In this heuristic, the resources are allocated during the running of the GA according to the convergence rate of the concepts. Concepts with a high convergence rate will get more resources than concepts with a low convergence rate.

Concept Convergence Rate( ) - for i-th concept its distance from the origin of the axis in generation 0 () and in generation X () calculated and their difference will be divided by the number of generations X.

Convergence Rate of the i-th concept where i=1,…, Number of concepts.

The distance from the origin is calculated as follows: from the i-th concept all the Non-dominated results are taken and the distance from each point is calculated from the origin. The minimum distance is the concept distance from the origin.

Greedy heuristic algorithm:

1. In the first 10(% or generations) all the concepts get the same resources
2. After the first 10(% or generations) Concept Convergence Rate are calculated for each concept -
3. Assigning in 3 sets: when each set is sorted descending order

set 1:

set 2:

set 3:

1. The best 90% of the concepts will continue as follow: all the concepts in set 1 (even if they are more than 90% of the concepts). If all the concepts in set 1 are less than 90% of the concepts than the remaining concepts will come from set 2 and if still not enough from set 3.
2. The concepts that continue will get the same resources
3. The concepts that not continue their will be saved and in the next resource allocation step can get resource again if their will be better than another concept, if they are belonging to set 2. If they are belonging to set 3 they will be eliminated and won’t get more resources.
4. If none of the globals stop conditions occurred go back to step 2

Where (high threshold) and (low threshold)

in step 4 there is a problem of getting a very small number or even 0 concepts, therefore the minimum number of concepts is 10% of the total number of concepts.

## Results analysis

\* Analysis the front

\* select concepts to continue with

\* Select indices for the last part: HV, IGD, etc.

## Find Pareto front using a multi-objective evolutionary algorithm

In this part, the concepts selected in the previous part entered into an Evolutionary Algorithm for multi-object optimization. The algorithm which selected is \*\*\*\*\* because of \*\*\*\*\*.

\* Select algorithm

\* Compute Indices