PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

# A SELF-LEARNING METAHEURISTIC FRAMEWORK BASED ON REINFORCEMENT LEARNING FOR COMBINATORIAL PROBLEM

**MARCELO ORLANDO BECERRA ROZAS**

PROYECTO DE TESIS
PROGRAMA DE DOCTORADO EN INGENIERÍA INFORMÁTICA

Noviembre, 2021

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

# A SELF-LEARNING METAHEURISTIC FRAMEWORK BASED ON REINFORCEMENT LEARNING FOR COMBINATORIAL PROBLEM

**MARCELO ORLANDO BECERRA ROZAS**

**Broderick Crawford Labrín**
Supervisor de Tesis

Noviembre, 2021

## Abstract

It is common to encounter combinatorial problems in binary domains when facing real problems using computational resources. As there are a large number of possible candidate solutions, this complicates the use of classical and common algorithmic techniques to address them. When this happens, it can become a problem associated with the high cost of resources they generate, so great importance is given to solving these problems efficiently. A study can be made on the existing methods to deal with this problem, such as Metaheuristics. There are metaheuristics that allow operating in discrete search spaces, however, in the case of continuous swarm intelligence metaheuristics, it is necessary to adapt them to operate in these domains. To carry out this adaptation, it is essential to use a binary scheme, in order to take advantage of the original moves of metaheuristics designed for continuous problems. In this work we propose to solve several combinatorial problems by applying different reinforcement learning techniques, machine learning area, on swarm-based metaheuristics for the selection of binarization schemes. Testing within these techniques, different elements such as transfer functions and binarization techniques. Apriori we can notice from several studies that implementations of swarm-based metaheuristics with reinforcement learning techniques affect the exploration-exploitation trade-off, however, much more results and experimentation are required, to be able to state that this hybridization works on several problems and not only on the ones raised in these studies. Therefore, to demonstrate the performance of the proposal, different parameterizations will be thoroughly evaluated by means of the respective statistical tests. Different benchmank problems will also be evaluated against other swarm-based metaheuristics that use these reinforcement learning techniques for the selection of binarization schemes.



**Keywords:** Metaheuristics, binarization, machine learning, swarm intelligence, reinforcement learning.

# Contents

# List of Figures

# 1 Introduction

Nowadays, optimization problems related to industry and the scientific world have grown to a large extent, causing more and more metaheuristics (MH) to emerge trying to solve NP-hard combinatorial optimization problems [1]. According to the premise exposed by the *"No Free Lunch Theorem"* [2, 3], it "compels" our conscience to want to develop more and more robust optimization algorithms that present feasible solutions, of quality and in reasonable computation times, capable of solving NP-Hard problems.

In order to develop more robust algorithms, we can distinguish different variations of techniques used in MH; in the first instance there is the hybridization of mathematical programming with MH or also known as *"Matheuristics"* [4]. There are methods that interrelate MH with problem simulation, known as *"Simheuristics"* [5]. There are also hybridization methods between MH techniques combining their exploration-exploitation components [6], currently, the area that is in constant development and on which this research is focused, is the interaction between MH and Machine Learning (ML) techniques, where ML techniques support MH operators in various ways in order to improve the performance [7–9].

With the continuous advances in digitization and the incorporation of artificial intelligence in different processes, it is necessary to be able to solve highly complex problems in binary domains in reduced computation times. There are multiple MH that allow solving problems in binary domains that do not need to vary their structure, however, MH designed to work on continuous domain problems have demonstrated and presented better performance and throughput in benchmark problems due to the fact that they allow controlling the exploration-exploitation balance [10].

Of this particular case, the exploration-exploitation balance, there are countless comments that the performance of the MH is greatly affected by the trade-off between exploration and exploitation. However, these comments are based solely on analysis of suboptimal results as they cannot encompass the large number of tests or comparison with other techniques (which also have suboptimal results) preventing them from achieving the true optimum. There is also a lot of literature on this topic, but unfortunately, it is not clear what the specific reasons are, or it is not explained when and why a performance is poor or good. Finally, there are very few papers that consider metrics to measure and control this balance, thus leav-

ing exploration and exploitation ratios totally randoms.

Therefore, the objective of this work is to present a framework of different implementations of RL in swarm-based MH for the resolution of combinatorial problems by selecting binarization schemes in an autonomous and fully on-line manner. Testing within these techniques, different elements such as transfer functions or the binarization technique. Apriori we can note due to several studies that swarm-based MH implementations with RL techniques affect the exploration-exploitation balance [11–14], however, more results and experiments are required, to be able to affirm that this MH-RL hybridization works in several problems and not only in those raised by these studies. This is why, in order to demonstrate the performance of the proposal, different parameterizations will be evaluated in an exhaustive manner by means of the respective necessary statistical tests. Also, different benchmark problems will be evaluated against other swarm-based MH that use these RL techniques for the selection of binarization schemes. It is necessary to mention that with "performance" we refer to the results obtained when measuring the fitness, that is, the one that obtains a lower average fitness in a minimization problem or a higher average fitness in a maximization problem will have a better "performance".

## 1.1 Problem statement

The following subsections state the problem to be addressed through the research questions and respective hypotheses, together with the main and specific objectives, as well as the research methodology to be used.

## 1.2 Research Questions

The research questions that arise that drive the motivation for this work are as follows:

- How to implement reinforcement learning techniques for binarization scheme selection in swarm-based metaheuristics?
- Is it possible that reinforcement learning techniques for binarization scheme selection improve the performance of swarm-based metaheuristics solving combinatorial problems?
- Is it possible that new transfer functions and binarization techniques of choice in the literature improve the performance of swarm-based metaheuristics in solving combinatorial problems?

## 1.3 Hypothesis

The hypotheses of this work are presented in this section. We propose four hypotheses with their respective null hypotheses.

- The use of reinforcement learning techniques for the selection of binarization schemes improves the performance of swarm-based metaheuristics when solving combinatorial problems.
- ($H_0$) The use of reinforcement learning techniques for the selection of binarization schemes does NOT improve the performance of swarm-based metaheuristics when solving combinatorial problems.
- The use of reinforcement learning techniques for the selection of binarization schemes affects the exploration-exploitation balance of swarm-based metaheuristics when solving combinatorial problems.
- ($H_0$) The use of reinforcement learning techniques for the selection of binarization schemes NOT affects the exploration-exploitation balance of swarm-based metaheuristics when solving combinatorial problems.
- Other metrics besides fitness influence the performance of exploration and exploitation balancing in swarm-based metaheuristics solving combinatorial problems.
- ($H_0$) Other metrics than fitness do NOT influence the performance of exploration and exploitation balancing in swarm-based metaheuristics solving combinatorial problems.

## 1.4 Objectives

The objectives of the alleged work are presented in this section. Identifying general and specific.

### 1.4.1 Main Objective

To develop a framework that compiles various reinforcement learning techniques in swarm-based metaheuristics by selecting binarization schemes to solve combinatorial problems. Performing in turn, different adjustments, such as reward functions, operators and metrics.

### 1.4.2 Specific Objectives

- Generate a collection of techniques to improve the performance of swarm-based metaheuristics.
- Implement various reinforcement learning techniques in swarm-based metaheuristics that allow autonomous selection of binarization schemes.
- Compare the results of the different techniques implemented and demonstrate the performance of the different techniques.

## 1.5 Research Methodology

The proposed work has a quantitative approach with an explanatory scope, since a comparison is made between several proposed and existing techniques. It is also an exploratory study since it is necessary to study which similar techniques and implementations have better performance.

# 2 Related Work

This section will present the general concepts necessary to understand the framework that compiles several reinforcement learning techniques in swarm-based metaheuristics.

## 2.1 Metaheuristics

First we must know where MH come from in order to understand what they are and how they work. According to the work of Talbi in [15], we can divide optimization methods into two large groups: exact methods, methods by which we can obtain better solutions in a given search space; and approximate methods, which generate high quality solutions in reasonable computation times for practical use. Among the approximate methods we find the approximate and heuristic algorithms, in the latter, we find the MH, algorithms that have two subcategories. Those based on a single solution and those based on population. The figure (1) details graphically what has been expressed above, making clearer both the categories and subcategories of the optimization methods presented by Talbi.

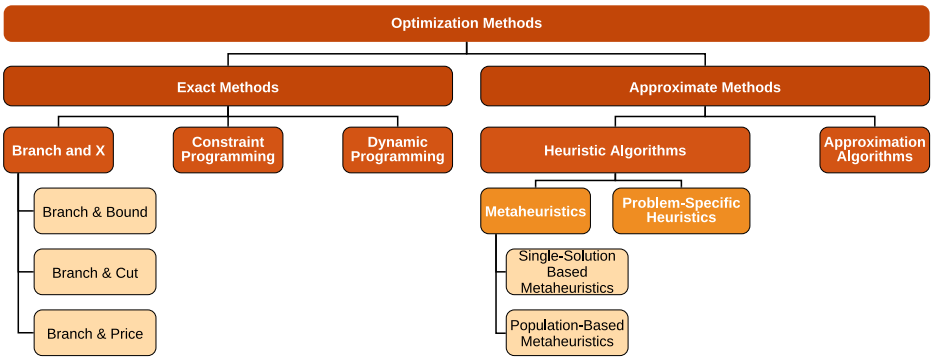


Figure 1: Categories and Subcategories of Optimization Methods

MH is a branch of optimization in computer science and applied mathematics, related to algorithms and computational complexity theory. Where an MH is defined as an iterative method which seeks to solve a problem, using parameters given by the same. Guiding a subordinate heuristic in order to find solutions close to the optimal one, creating a union between exploitation and exploration in an ideal way in the search space..

Over the past 20 years, numerous MH have been developed in various disciplines that lie at the intersection of several fields, such as artificial intelligence, computational intelligence, soft computing, mathematical programming, and operations research. Most MH mimic natural metaphors to solve complex optimization problems [15]. This is why different types of MH have emerged, focusing on different types of problems and showing different search behaviors to reach the solution. Therefore they can be classified according to their inspiration, it is known that the movements and ways to build solutions are based on various behaviors [16], such as those based on Evolution [17], based on swarm intelligence [1], based on physics [18] and based on human behavior and societies [19].

### 2.1.1 Taxonomy of Metaheuristics

There are different approaches to classify and describe an MH algorithm, depending on the characteristics to be emphasized. Therefore, we will describe the most important ways of classification.

- **Nature Inspired vs. Non-Nature Inspired:** It is based on comparing the origin of the algorithm. There are some that find their origin in the behavior of some species in nature such as *Grey Wolf Optimization*, *Whale Optimization Algorithm* or *Ant Algorithms*; and unlike others that are not inspired by nature, such as *Tabu Search* and *Iterated Local Search*.

- **Population-Based vs. Single Point Search:** The number of solutions used at the same time determines whether the algorithm works on a population, or on a single solution. The latter are called trajectory methods, and are formed by MH based on local search *Tabu Search*, *Iterated Local Search* and *Variable Neighborhood Search*. Where each one shares the characteristic of describing a trajectory in the search space, while it is being performed. On the other hand, population-based MH describe the evolution of a set of points in the search space.

- **Dynamic vs. Static Objective Function:** They can also be compared depending on how the objective function is used. While some maintain the function proposed in the representation of the problem, others modify it during its search, such as the *Guided Local Search* algorithm. The purpose of this is that the function manages to incorporate the information gathered during the search process.

- **One vs. Various Neighborhood Structures:** Most metaheuristic algorithms work with a single neighborhood structure. In other words, the topology of the fitness landscape does not change over the course of the algorithm. Other MH, such as the Variable Neighborhood Search, use a set of neighborhood structures that offer the possibility to diversify the search by switching between different landscapes.

- **Memory Usage vs. Memory-Less Methods:** Another feature is the memory usage given to each MH. Where some use an adaptive memory for their search history, in contrast to others that use a static memory as in the Branch & Bound algorithm.

## 2.2 Whale Optimization Algorithm

The Whale Optimization Algorithm (WOA) is inspired by the hunting behavior of humpback whales, specifically, how they make use of a strategy known as "bubble netting". This strategy consists of locating the prey and, by means of moving in spiral turns that are similar to a "9", enclosing in on

the prey. This algorithm was invented by Mirjalili and Lewis in 2015 [17].

The WOA metaheuristic starts with a set of random solutions. At each iteration, the search agents update their positions with respect to a randomly chosen search agent or the best solution obtained so far. There is a parameter $a$ that is reduced from 2 to 0 to provide changes between exploration and exploitation. When the equation vector (1) has value: $\mid \overrightarrow{A} \mid \geq 1$, a new random search agent is chosen. On the other hand, when $\mid \overrightarrow{A} \mid < 1$, the best solution is selected; the point of this is to be able to update the position of the search agents.

On the other hand, the value of the parameter $p$ (random number between 0 and 1) allows the algorithm to switch between a spiral or circular motion. In order to assimilate this, there are three movements that are crucial when working with the metaheuristic:

1. **Searching for prey ($p < 0.5$ and $|A| \geq 1$):** The whales search for prey randomly based on the position of each prey. When the algorithm determines that $\mid \overrightarrow{A} \mid \geq 1$, then we can say that it is exploring and allows WOA to perform a global search. We represent this first move with the following mathematical model:

$$\begin{aligned} \overrightarrow{X}_i^{t+1} &= \overrightarrow{X_{rand}^t} - \overrightarrow{A} \cdot \overrightarrow{D} \\ \overrightarrow{D} &= |\overrightarrow{C} \cdot \overrightarrow{X_{rand}^t} - \overrightarrow{X}_i^t| \end{aligned} \tag{1}$$

   where $t$ denotes the current iteration, $\overrightarrow{A}$ and $\overrightarrow{C}$ are coefficient vectors and $\overrightarrow{X_{rand}}$ is a random position vector (i.e., a random whale) chosen from the current population. The vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ can be computed according to the following Equation (2):

$$\begin{aligned} \overrightarrow{A} &= 2\overrightarrow{a} \cdot \overrightarrow{r} - \overrightarrow{a} \\ \overrightarrow{C} &= 2 \cdot \overrightarrow{r} \end{aligned} \tag{2}$$

   where, $\overrightarrow{a}$ decreases linearly from 2 to 0 over iterations (both in the exploration and exploitation phases) and $\overrightarrow{r}$ corresponds to a random vector of values between $[0, 1]$.

2. **Encircling the prey ($p < 0.5$ and $|A| < 1$):** Once the whales have found and recognized their prey, they begin to encircle them. Since the

position of the optimal design in the search space is not known in the first instance, the metaheuristic assumes that the current best solution is the target prey or is close to the optimum. Therefore, once the best search agent is defined, the other agents will attempt to update their positions toward the best search agent. Mathematically, it is modeled in Equation (3):

$$\begin{aligned} \overrightarrow{X}_i^{t+1} &= \overrightarrow{X^*}_i^t - \overrightarrow{A} \cdot \overrightarrow{D} \\ \overrightarrow{D} &= |\overrightarrow{C} \cdot \overrightarrow{X^*}_i^t - \overrightarrow{X}_i^t| \end{aligned} \tag{3}$$

where $\overrightarrow{X^*}$ is the position vector of the best solution obtained so far and $\overrightarrow{X}$ is the position vector. The vector $\overrightarrow{A}$ and $\overrightarrow{C}$ are calculated in Equation (2). It is worth mentioning that $\overrightarrow{X}$ must be updated at each iteration if a better solution exists.

3. **Bubble net attack ($p \geq 0.5$):** For this attack, the "shrinking net mechanism" is presented and this behavior is achieved by decreasing the value of $a$ in the Equation (2). Thus, as the whale spirals, it shrinks the bubble net until it finally catches the prey. This motion is modeled with the following Equation (4):

$$\begin{aligned} \overrightarrow{X}_i^{t+1} &= \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*}_i^t \\ \overrightarrow{D'} &= |\overrightarrow{X^*}_i^t - \overrightarrow{X}_i^t| \end{aligned} \tag{4}$$

where $\overrightarrow{D'}$ is the distance of the i-th whale from the prey (the best solution obtained so far), $b$ is a constant for defining the shape of the logarithmic spiral and $l$ is a random number between $[-1, 1]$.

It is worth mentioning that humpback whales simultaneously swim around the prey within a shrinking circle and along a spiral trajectory. In order to model this simultaneous behavior, there is a 50% probability of choosing between the encircling prey mechanism (2) or the spiral model (3) to update the position of the whales during optimization. The mathematical model is as follows: .

$$\overrightarrow{X}_i^{t+1} = \begin{cases} \overrightarrow{X^*}_i^t - \overrightarrow{A} \cdot \overrightarrow{D} & \text{If } p < 0.5 \\ \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*}_i^t & \text{If } p \geq 0.5 \end{cases} \tag{5}$$

The pseudocode (1) of the metaheuristic [20] is included for a better understanding of what has been previously stated:

**Algorithm 1** Whale Optimization Algorithm

Initialize the whale population $X_i$ $(i = 1, 2, \ldots, n)$
Calculate the fitness of each search agent
$X^*$ = The best search agent
**while** $t \leq$ Maximum number of iterations **do**
    **for** each search agent **do**
        Update $a, A, C, l$ and $p$
        **if** $(p < 0,5)$ **then**
            **if** $(|A| < 1)$ **then**
                Update the position of the current search agent using of Equation (3).
            **else**$(|A| \geq 1)$
                Select a random search agent $(X_{Rand})$
                Update the position of the current search agent using Equation (1).
            **end if**
        **else**$(p \geq 0,5)$
            update the position of the current search agent using Equation (4).
        **end if**
    **end for**
    Check if any search agent goes beyond the search space and we modify it.
    Calculate the fitness of each search agent
    Update $X^*$ if there is a better solution
    $t \leftarrow t_{+1}$
**end while**
Return $(X^*)$

## 2.3 Grey Wolf Optimization

The Grey Wolf Optimizer (GWO) is inspired by and mimics the leadership hierarchy and hunting mechanism of grey wolves in the wild. At the top of the pack are the alpha wolves, responsible for decision making, followed by the beta and delta wolves. The rest of the pack are called or referred to as omegas [21]. The prey location is the optimal solution and the wolves represent potential solutions in the search space. The wolves closest to the prey are the alpha, beta and delta wolves, wolves corresponding to the best, second and third best solutions found so far respectively. The representation

of these leading wolves in the search space is formed as: $\alpha$, $\beta$ and $\delta$. The position of the rest of the pack, previously referred to as *omegas* are updated in the search space based on their positions relative to that of the leaders. For prey hunting, a set of steps must be applied: encircle, stalk, attack, and search for prey.

1. **Encircling the prey:** The pack surrounds the prey by repositioning agents or wolves depending on the location of the prey, this type of action can be modeled as follows:

$$\overrightarrow{X}(t+1) = \overrightarrow{X_p}(t) - \overrightarrow{A} \cdot \overrightarrow{D} \tag{6}$$

   Where, $t$ is the iteration, $\overrightarrow{X_p}$ is the position of the prey, $\overrightarrow{X}$ is the position of the wolf and $\overrightarrow{D}$ can be defined as follows:

$$\overrightarrow{D} = \mid \overrightarrow{C} \cdot \overrightarrow{X_p}(t) - \overrightarrow{X}(t) \mid \tag{7}$$

   On the other hand, the coefficient vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ of equations (6) and (7) respectively can be calculated as:

$$\begin{aligned} \overrightarrow{A} &= 2a \cdot \overrightarrow{r_1} - a \\ \overrightarrow{C} &= 2\overrightarrow{r_2} \end{aligned} \tag{8}$$

   Where, $a$ decreases linearly from 2 to 0 as the iterative process progresses. Finally $\overrightarrow{r_1}$ and $\overrightarrow{r_2}$ correspond to uniform random vectors of value between 0 and 1.

2. **Stalking the prey until it stops:** This action is performed by the entire pack based on information from the $\alpha$, $\beta$ and $\delta$ wolves, which are expected to know the location of the prey they are stalking, this can be mathematically modeled as:

$$\overrightarrow{X}(t+1) = \frac{\overrightarrow{X_1} + \overrightarrow{X_3} + \overrightarrow{X_3}}{3} \tag{9}$$

Where $\overrightarrow{X_1}$, $\overrightarrow{X_3}$ and $\overrightarrow{X_3}$ are defined as:

$$\begin{aligned}\overrightarrow{X_1} &= \mid \overrightarrow{X_\alpha} - \overrightarrow{A_1} \cdot \overrightarrow{D_\alpha} \mid \\ \overrightarrow{X_2} &= \mid \overrightarrow{X_\beta} - \overrightarrow{A_2} \cdot \overrightarrow{D_\beta} \mid \\ \overrightarrow{X_3} &= \mid \overrightarrow{X_\delta} - \overrightarrow{A_3} \cdot \overrightarrow{D_\delta} \mid\end{aligned} \tag{10}$$

$\overrightarrow{X_\alpha}$, $\overrightarrow{X_\beta}$ and $\overrightarrow{X_\delta}$ are the first three best solutions at a given iteration t, $\overrightarrow{A_1}$, $\overrightarrow{A_2}$ y $\overrightarrow{A_3}$ we define them as in the equation (8) for the vector $\overrightarrow{A}$ and finally $\overrightarrow{D_\alpha}$, $\overrightarrow{D_\beta}$ and $\overrightarrow{D_\delta}$ are defined using the following:

$$\begin{aligned}\overrightarrow{D_\alpha} &= \mid \overrightarrow{C_1} \cdot \overrightarrow{X_\alpha} - \overrightarrow{X} \mid \\ \overrightarrow{D_\beta} &= \mid \overrightarrow{C_2} \cdot \overrightarrow{X_\beta} - \overrightarrow{X} \mid \\ \overrightarrow{D_\delta} &= \mid \overrightarrow{C_3} \cdot \overrightarrow{X_\delta} - \overrightarrow{X} \mid\end{aligned} \tag{11}$$

Where, $\overrightarrow{C_1}$, $\overrightarrow{C_2}$ and $\overrightarrow{C_3}$ are defined in the equation (8) for the vector $\overrightarrow{C}$

3. **Attack the prey:** Agents approach the prey, which is achieved by decreasing the scan rate $a$. The parameter $a$ is updated linearly at each iteration to go from 2 to 0 as follows:

$$a = 2 - t\frac{2}{T} \tag{12}$$

Where, $t$ corresponds to the number of the current iteration and T is the number of total iterations. According to Mirjalili in [21], due to the values that $a$ can take, it allows to transit smoothly between exploration and exploitation. It indicates that half of the iterations are devoted to exploration and the other half is allocated to exploitation. This is interpreted as the wolves move or change their position to any random position between their current position and the position of the prey.

4. **Search for prey:** To search for prey the wolves separate from each other. This behavior is modeled by setting large values of the parameter $a$ in the equation (12) to allow exploration of the search space. Therefore, wolves diverge from each other to better explore the search

space and then converge again to attack when they find better prey. It is necessary to mention, that any wolf can find a better (optimal) prey. If a wolf gets close to the prey, it will become the new alpha and the other wolves will be divided into beta, delta and omegas according to their distance from the prey.

The parameter $a$ gives random weights to the dam and shows the impact of the dam as in the equations (6) and (7). This allows GWO to exhibit more randomic behavior, thus favoring exploration and evasion of local optima. It is worth noting, that $a$ provides random values at all times keeping in mind the goal of emphasizing exploration not only at the beginning of the optimization process but until its end as well.

As was presented in the whale algorithm (section 2.2), for a better understanding of all the movements of the metaheuristic, pseudocode (2) indicating how it operates algorithmically is made available to the reader:

**Algorithm 2** Grey Wolf Optimization

initialize a population of n random gray wolf positions
Find $\alpha$, $\beta$ and $\delta$ as the first three best fitness solutions
Initialize $t = 0$
**while** $t \leq$ Maximum number of iterations **do**
    **for** each wolf in the pack **do**
        Update the current position of the wolf according to Eq. 9
    **end for**
**end while**
Update $a$, $A$ and $C$ according to Eqs. (12) y (8)
Evaluate the positions of the individual wolves
Update the positions of $\alpha$, $\beta$ and $\delta$ three best solutions of the current population
$t \leftarrow t_{+1}$
Return the wolf to its optimum position

## 2.4 Sine-Cosine Algorithm

This algorithm based on the trigonometric functions sine and cosine, was created by Mirjalili in 2016 [22]. Like all optimization techniques based on iterations this one starts with a random population, then through the equations of motion (13) and (14).

$$X_i^{t+1} = X_i^t + r_1 \cdot sin(r_2) \cdot |r_3 P_i^t - X_i^t| \tag{13}$$

$$X_i^{t+1} = X_i^t + r_1 \cdot cos(r_2) \cdot |r_3 P_i^t - X_i^t| \tag{14}$$

Therefore the general equation of the metaheuristic Sine Cosine Algorithm is the one presented in Eq. (15).

$$X_i^{t+1} = \begin{cases} X_i^{t+1} = X_i^t + r_1 \cdot sin(r_2) \cdot |r_3 P_i^t - X_i^t|, & r_4 < 0.5 \\ X_i^{t+1} = X_i^t + r_1 \cdot cos(r_2) \cdot |r_3 P_i^t - X_i^t| & r_4 \geq 0.5 \end{cases} \tag{15}$$

Where:

$$r_1 = 2 \cdot \pi \cdot Rand() \tag{16}$$
$$r_2 = 2 \cdot Rand() \tag{17}$$
$$r_3 = Rand() \tag{18}$$
$$r_4 = 2 \cdot \pi \cdot Rand() \tag{19}$$

The four parameters of the equations of motion are $r_1, r_2, r_3, r_4$. Where $r_1$ determines the direction of the motion, i.e. towards or away from the best known solution (Eq. 16). $r_2$ defines the magnitude of the motion (eq. 17). $r_3$ integrates how random the motion will be, when $r_3 > 1$ the motion will be more stochastic (Eq. 18). $r_4$ determines whether the motion will be performed with the sine or cosine function in the same proportion (Eq. 19).

The metaheuristic procedure, as with the whale and grey wolf, is explained in the pseudocode (3):

## 2.5 Binarization Techniques

Within the continuous MH, there are binarization techniques, which consist of transferring the values of a continuous domain to a binary one. This transfer of values is carried out with the aim of preserving the quality movements of continuous MH and thus generate quality binary solutions.

In several NP-Hard combinatorial problems, continuous MH operating with a binarization scheme have presented a great performance, which has

**Algorithm 3** Sine-Cosine Algorithm

Initialize a set of search agents (Solutions)
Initialize $t = 0$
**while** $t \leq$ Maximum number of iterations **do**
    Evaluate each of the search agents by objective function
    Update best solution obtained so far ($X_{Best}$)
    Update $r_1$, $r_2$, $r_3$ and $r_4$
    Update the position of the search agent using Eq. 15
**end while**
$t \leftarrow t_{+1}$
Return $X_{Best}$

attracted the interest of the scientific community, some examples of the previously mentioned are: Binary Magnetic Optimization Algorithm [23], Binary Bat Algorithm [24], Binary Dragonfly [25], among others [26–33].

Among the most commonly used binarization schemes in recent times [10], the two-step techniques stand out because they do not alter the functioning of the other elements of the MH.

### 2.5.1 Two-Step Binarization Scheme

For various problems, two-step binarization schemes have shown great relevance [34]. As the name says, this binarization scheme is composed of two steps, the first one being the transfer function. This function allows transferring the values generated by the continuous MH to a continuous interval between $[0, 1]$. Once the values are in a continuous interval, the second step is performed; the binarization, which consists of transferring this interval to a binary value. With the Figure (2) we can appreciate the above described in a better way.

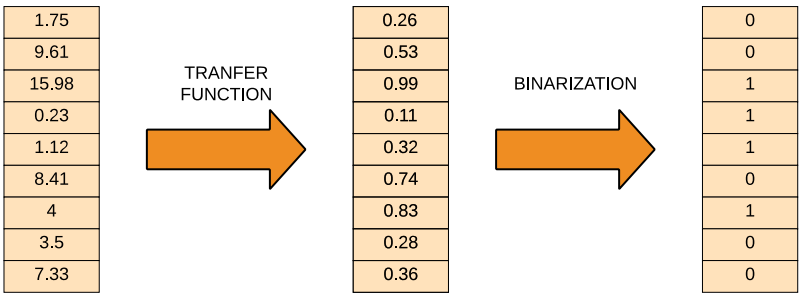


Figure 2: Classic Binarization Scheme

### 2.5.2 First Step: Transfer Functions

Kennedy et al. in 1997 [35] introduced transfer functions. Their main advantage is the delivery of a probability between 0 and 1 at a low computational cost. In the literature there are several innovative transfer functions other than the classical S and V, such as: O-types, X-types, Z-types, U-types [36–45], and even variations of the classical ones that could be called as "time-varying" [25, 46–48]. They can be represented graphically in the figures (3 - 21). These functions have four variations each, as shown in Table (1).

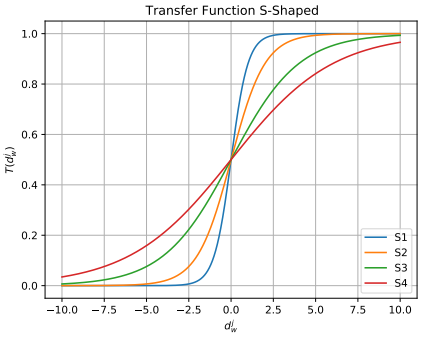


Figure 3: S-Shaped Transfer Functions

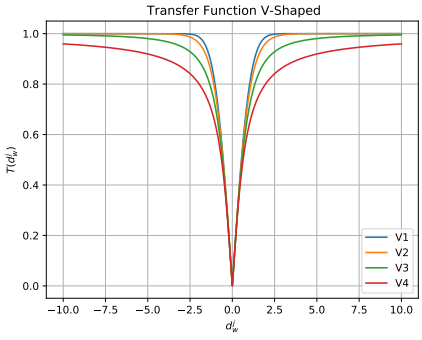


Figure 4: S-Shaped Transfer Functions

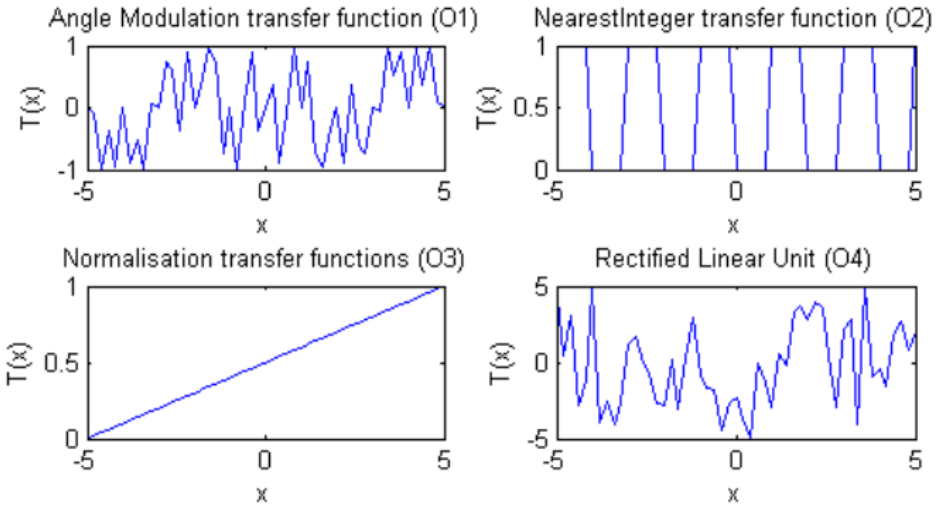


Figure 5: O-Shaped Transfer Functions

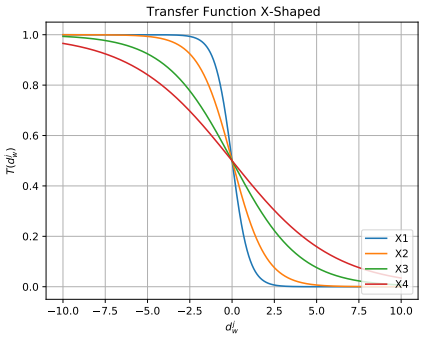


Figure 6: X-Shaped Transfer Functions

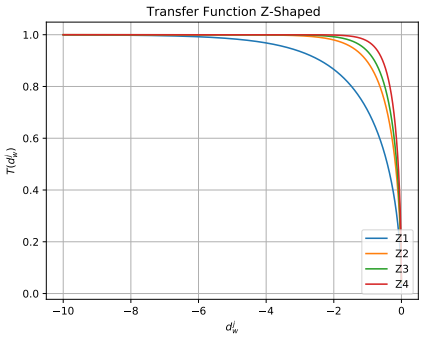


Figure 7: Z-Shaped Transfer Functions

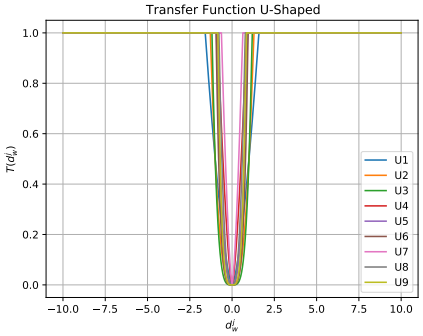


Figure 8: U-Shaped Transfer Functions

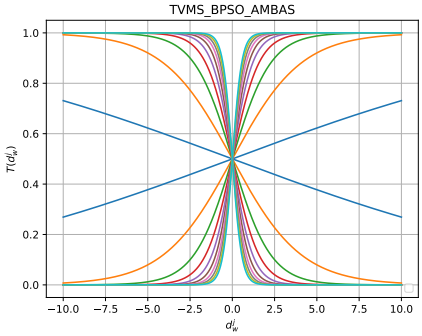


Figure 9: X-Shaped Time Variable Transfer Functions

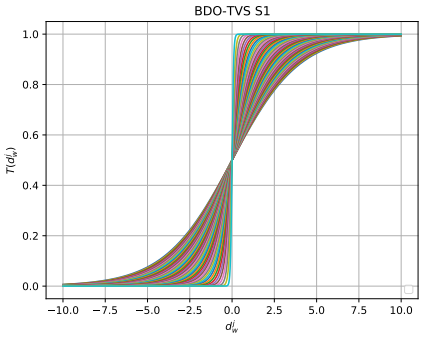


Figure 10: S1-Shaped Time Variable Transfer Functions

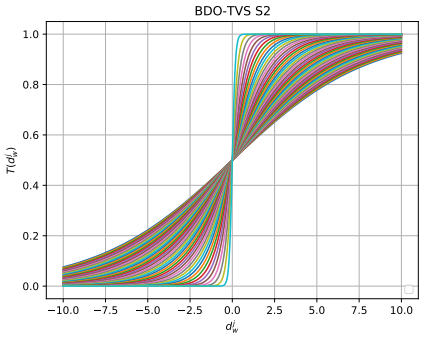


Figure 11: S2-Shaped Time Variable Transfer Functions

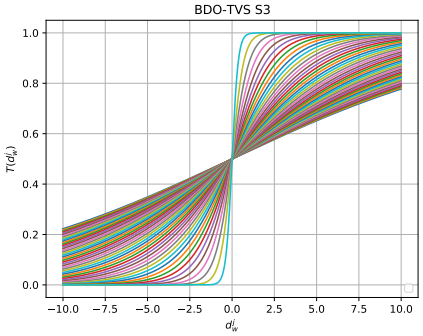


Figure 12: S3-Shaped Time Variable Transfer Functions

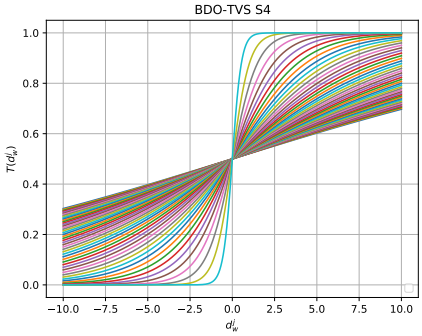


Figure 13: S4-Shaped Time Variable Transfer Functions

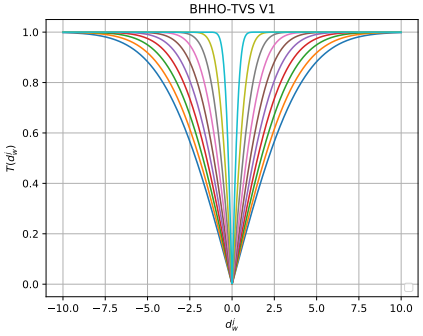


Figure 14: V1-Shaped Time Variable Transfer Functions

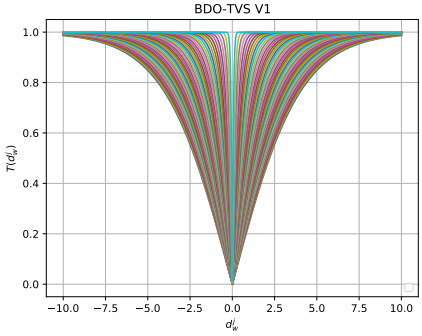


Figure 15: V1-Shaped Time Variable Transfer Functions

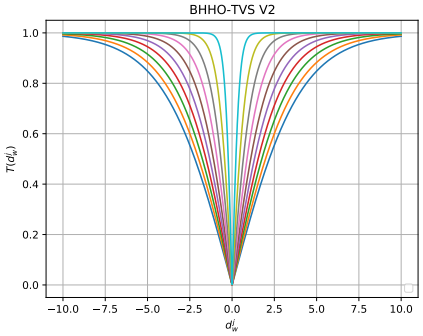


Figure 16: V2-Shaped Time Variable Transfer Functions

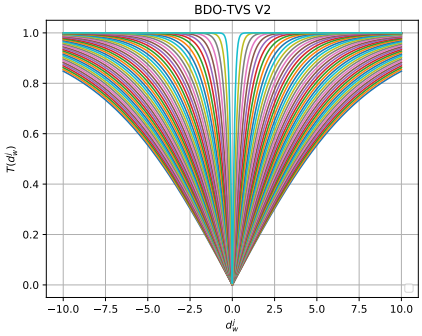


Figure 17: V2-Shaped Time Variable Transfer Functions

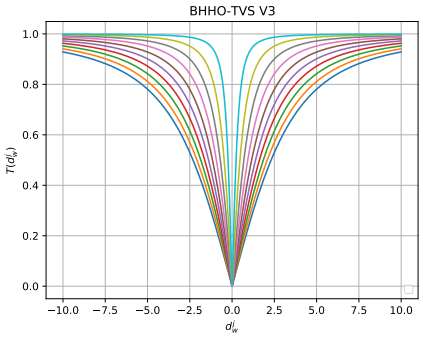


Figure 18: V3-Shaped Time Variable Transfer Functions

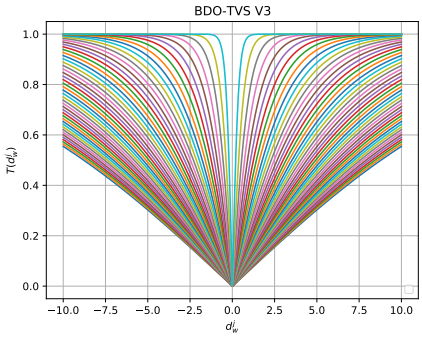


Figure 19: V3-Shaped Time Variable Transfer Functions

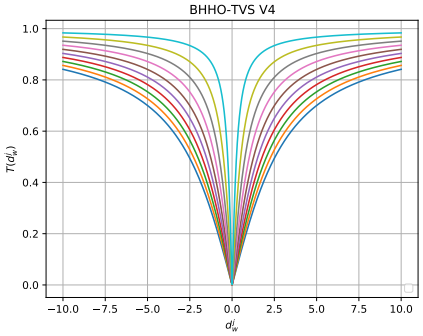


Figure 20: V4-Shaped Time Variable Transfer Functions

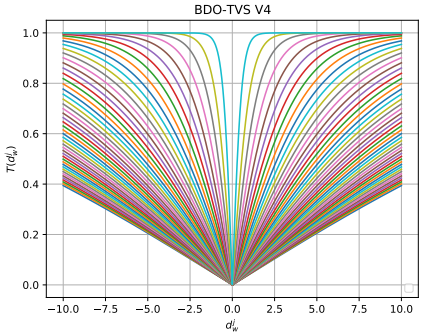


Figure 21: V4-Shaped Time Variable Transfer Functions

Table 1: Transfer Functions

| **Type** | **Transfer Function** | |
|---|---|---|
| S1 [29, 31] | $\mathrm{T}(\mathrm{d}_w^j) = \frac{1}{1+e^{-2d_w^j}}$ | (20) |
| S2 [29, 49] | $\mathrm{T}(\mathrm{d}_w^j) = \frac{1}{1+e^{-d_w^j}}$ | (21) |
| S3 [29, 31] | $\mathrm{T}(\mathrm{d}_w^j) = \frac{1}{1+e^{\frac{-d_w^j}{2}}}$ | (22) |
| S4 [29, 31] | $\mathrm{T}(\mathrm{d}_w^j) = \frac{1}{1+e^{\frac{-d_w^j}{3}}}$ | (23) |
| V1 [29, 50] | $\mathrm{T}(\mathrm{d}_w^j) = \left\lvert erf\left(\frac{\sqrt{\pi}}{2}d_w^j\right)\right\rvert$ | (24) |
| V2 [29, 50] | $\mathrm{T}(\mathrm{d}_w^j) = \left\lvert tanh(d_w^j)\right\rvert$ | (25) |
| V3 [29, 31] | $\mathrm{T}(\mathrm{d}_w^j) = \left\lvert \frac{d_w^j}{\sqrt{1+(d_w^j)^2}}\right\rvert$ | (26) |
| V4 [29, 31] | $\mathrm{T}(\mathrm{d}_w^j) = \left\lvert \frac{2}{\pi}arctan\left(\frac{\pi}{2}d_w^j\right)\right\rvert$ | (27) |
| O1 [36, 37] | $T(d_w^j) = sin(2\cdot\pi(x-a)\cdot b\cdot cos(2\cdot\pi(x-a)\cdot c)) + d$ <br> $a = 0, b = 1, c = 1, d = 0$ | (28) |
| O2 [36, 38] | $\mathrm{T}(\mathrm{d}_w^j) = \lfloor \lvert x mod 2\rvert \rfloor$ | (29) |
| O3 [36, 39] | $\mathrm{T}(\mathrm{d}_w^j) = \frac{(d_w^j + d_{wmin}^j)}{(\lvert d_{wmin}^j\rvert + d_{wmax}^j}) \quad (d_{wmin}^j \le d_w^j \le d_{wmax}^j)$ | (30) |
| O4 [36, 40] | $\mathrm{T}(\mathrm{d}_w^j) = d_w^j$ | (31) |
| X1 [41, 42] | $\mathrm{T}(\mathrm{d}_w^j) = \frac{1}{1+e^{2d_w^j}}$ | (32) |
| X2 [41, 42] | $\mathrm{T}(\mathrm{d}_w^j) = \frac{1}{1+e^{d_w^j}}$ | (33) |
| X3 [41, 42] | $\mathrm{T}(\mathrm{d}_w^j) = \frac{1}{1+e^{\frac{d_w^j}{2}}}$ | (34) |
| X4 [41, 42] | $\mathrm{T}(\mathrm{d}_w^j) = \frac{1}{1+e^{\frac{d_w^j}{3}}}$ | (35) |
| Z1 [43, 44] | $\mathrm{T}(\mathrm{d}_w^j) = \sqrt{1-2^{d_w^j}}$ | (36) |
| Z2 [43, 44] | $\mathrm{T}(\mathrm{d}_w^j) = \sqrt{1-5^{d_w^j}}$ | (37) |
| Z3 [43, 44] | $\mathrm{T}(\mathrm{d}_w^j) = \sqrt{1-8^{d_w^j}}$ | (38) |
| Z4 [43, 44] | $\mathrm{T}(\mathrm{d}_w^j) = \sqrt{1-20^{d_w^j}}$ | (39) |

### 2.5.3 Second Step: Binarization

The function of binarization is to discretize the probability obtained from the transfer function, delivering a binary value. For this step there are different techniques in the literature such as those exemplified in the Table (2).

Table 2: Techniques of Binarization

| **Type** | **Binarization** |
|---|---|
| Standard | $\mathrm{X}_{new}^{j} = \begin{cases} 1 & if\ rand \leq T(d_w^j) \\ 0 & else. \end{cases}$ (40) |
| Complement | $\mathrm{X}_{new}^{j} = \begin{cases} X_w^j & if\ rand \leq T(d_w^j) \\ 0 & else. \end{cases}$ (41) |
| Static Probability | $\mathrm{X}_{new}^{j} = \begin{cases} 0 & if\ T(d_w^j) \leq \alpha \\ X_w^j & if\ \alpha < T(d_w^j) \leq \frac{1}{2}(1+\alpha) \\ 1 & if\ T(d_w^j) \geq \frac{1}{2}(1+\alpha) \end{cases}$ (42) |
| Elitist | $\mathrm{X}_{new}^{j} = \begin{cases} X_{Best}^j & if\ rand < T(d_w^j) \\ 0 & else. \end{cases}$ (43) |
| Roulette Elitist | $\mathrm{X}_{new}^{j} = \begin{cases} P[X_{new}^j = \zeta_j] = \frac{f(\zeta)}{\sum_{\delta \in Q_g} f(\delta)} & \text{if rand } \leq T(d_w^j) \\ P[X_{new}^j = 0] = 1 & else. \end{cases}$ (44) |

## 2.6 Machine Learning

ML is a subfield of computer science and artificial intelligence, which encompasses a series of algorithms that learn from a data set and are capable of making predictions [9]. ML techniques have been gaining popularity in recent times and various applications can be found, ranging from everyday applications, research and even industrial use [51–55]. According to the literature, the classification of the different techniques that ML has are presented in different ways, however, all agree that the most common classification divides the algorithms according to their learning paradigm, in other words, if it is supervised or not, or if we are reinforcing learning. These paradigms

are classified in Figure (22).

- **Supervised Learning:**As the name indicates, these algorithms depend on the supervision of a user for their correct operation, in other words, the algorithms need a training phase [56]. The classic example of supervised learning would be image classification, the algorithm is trained by giving it a set of images with their respective labels. In this way it learns from the information contained in the images, being able to generate image-label relationships. As a result of this relationship and training, if the algorithm receives a new image that does not belong to the training set, it will be able to identify the corresponding label.

- **Unsupervised Learning:** In this type of algorithm, the training phase is not necessary as in the supervised algorithms, since they detect patterns and characteristics among the data provided, which allows them to fulfill certain tasks [57]. An example could be clustering techniques, where you want to group data containing similar characteristics without having to define what these characteristics are within the dataset. K-means [58] as a direct example takes care of identifying centroids within the dimensions of the data given to the algorithm and thus identifies similar features among the data.

- **Semi-supervised Learning:** These mixed algorithms are an intermediate classification to the supervised and unsupervised paradigms, since they share characteristics of both. An example could be image classification again, however, not all labels are known for a given data set. So this mixed strategy between algorithms that use the information from the image-label relationship (supervised), together with unsupervised algorithms to complete or use the information obtained from the features and patterns of the image-label relationships (unsupervised) must be used [59–61].

- **Reinforcement Learning:** Reinforcement learning [62], has been growing in popularity due to its ease of access to large-scale computing, as well as the great performance that has been achieved. A possible example to this, is the work done by Mnih et al. [63], where they use reinforcement learning techniques called Deep Q-network, where it overcomes 49 Atari 2600 console games. The algorithm receives as input the pixels and the score of the game, and through a deep network manages to learn the sequence of moves to be performed to maximize the score.

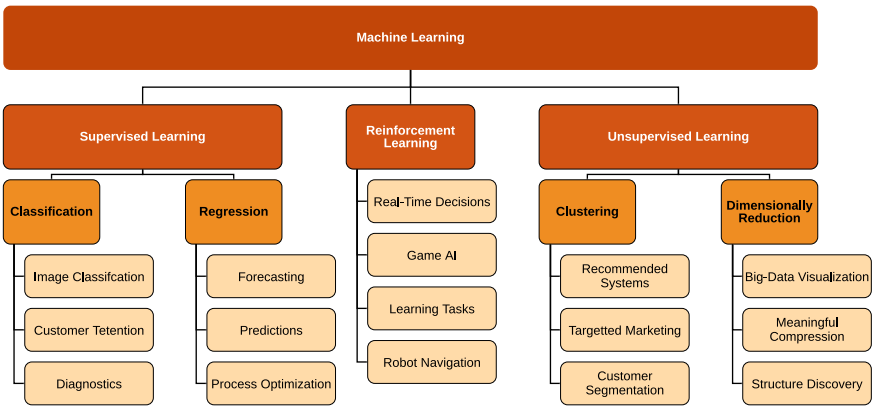


Figure 22: Machine Learning Classification

## 2.7 Hybrids in Metaheuristics

A hybrid MH is described as the combination of a metaheuristic algorithm and a different learning algorithm, including Matheuristics, Constraint Programming, ML [6], among others. For this work we will focus on the hybrids generated with ML, where we find two groups: ML supporting MH, or MH supporting ML.

Focusing on the first group mentioned above, in the work of García et al. [64], two lines of research are shown. First, we find the integration of ML as the replacement of an operator, such as the handling of a population, local search and parameter tuning. Second, is to use ML as a selector of a set of MH, choosing the most appropriate one depending on the problem to be addressed.

When using ML as a selector, we can divide this category into three groups, the first of which is algorithm selection that chooses from a set of techniques for the problem in order to obtain better performance for a set of similar instances [65]. Secondly we find the hyperheuristic strategies, where their goal is to use the MH to cover a set of problems. And finally we find cooperative strategies, which combine algorithms sequentially, with the objective of improving the robustness of the solution. For this work, we will

investigate the hybridization between the use of MH in conjunction with reinforcement learning techniques, techniques that belong to the ML area.

## 2.8 Reinforcement Learning Fundamentals

Based on Reinforcement Learning (RL), an agent consists of four sub-elements: a policy, a value function, a reward function and optionally, an environment model [66].

Then, as a definition:

- A policy defines the agent's behavior at each instant of time, i.e., it is a mapping from the set of perceived states to the set of actions to be performed when the agent is in those states.
- The value function allows the agent to aim at maximizing the sum of total rewards in the long-term. It calculates the value of a state-action pair as the total amount of rewards that the agent can expect to accumulate in the future, starting from the state he is in. Thus, the agent selects the action based on value judgments. Indeed, while reward determines the immediate and intrinsic desirability of a state-action pair, value indicates the long-term desirability of a state-action pair considering likely future state-action pairs and their rewards.
- A reward function represents the agent's goal, i.e., it translates each perceived state-action pair into a single number. In other words, a reward indicates intrinsic desirability of that state-action pair. It is a way of communicating to the agent what the agent wants to obtain, but not how to achieve it.
- A model of the environment is intended to reproduce the behavior of the environment, i.e. the model that directs the agent to the next state and the next reward on the basis of the current state-action pair. The model of the environment is not always available, that is why it is one of the optional elements.

Then, at each of the discrete time steps, i.e., $t = 0, 1, 2, ..., N$, the agent and the environment interact with each other. Thus, the former receives from the latter a representation of it at time $t$, $s_t \in S$, where $S$ is the set of available states, which summarizes all the information regarding the environment. Based on this, an action $a_t \in A(s_t)$ is selected by the agent,

where $A(s_t)$ is the set of available actions in state $s_t$. In the next step, i.e., $t+1$, the agent is in a new state $(s_t+1)$ and receives a reward $r_{t+1} \in R$ in response to the performed action $a_t$.

Since in RL the agent's objective is the maximization of the value function, at time $t$ the agent chooses the action that maximizes the expected value of the total rewards that can be obtained in the future from the state in which the agent is. The expected reward $R_t$, is generally defined as a function of the current and discounted future rewards, the equation representing this is as follows:

$$R_t = \sum_{j=0}^{n} \gamma^j \cdot r_{t+j+1} \tag{45}$$

Where, $\gamma \in [0, 1]$ is the discount factor. Therefore, the agent's goal is to learn a policy capable of maximizing long-term rewards by interacting with the environment based on one's own experience. To do so, at each step $t$, starting from the state $s_t$, the agent has to compute as follows the value of the action-value function $Q^\pi(s, a)$ for each possible action on the basis of the policy $\pi$. The policy $\pi$, can be defined as:

$$Q^\pi(s, a) = P_\pi\{R_t \mid s_t = s, a_t = a\} \tag{46}$$

A fundamental property of the action-value function is the fulfillment of the following recursive relation, known as Bellman's equation for $Q^\pi(s, a)$:

$$Q^\pi(s, a) = P_\pi\{r_{t+1} + \gamma \cdot Q^\pi(s_{t+1}, a_{t+1}) \mid s_t = s, a_t = a\} \tag{47}$$

To learn a policy capable of maximizing long-run rewards, the agent has to select the action that fulfills a relation, called a greedy policy.

$$Q^*(s, a) = max Q^\pi(s, a) \tag{48}$$

Since $Q^*(s, a)$ is a value function that is under the policy $\pi$, then, Bellman's equation is satisfied, and it is called Bellman's optimality equation. Which, expresses that the value of a state following an optimal policy must be equal to the expected return for the best action selected in that state, in mathematical language, $Q^*(s, a) = max_{a \in A(S)} Q^\pi(s, a)$

Due to the iterative method, it is possible to compute the action-value function $Q^\pi(s, a)$. It is called policy evaluation and starts with an arbitrary

initialization of $Q_0^\pi(s,a)$. At each computational iteration, the action-value function is approximated using the Bellman equation as an update rule:

$$Q_{k+1}^\pi(s,a) = P_\pi\{r_{t+1} + \gamma \cdot Q_k^\pi(s_{t+1}, a_{t+1}) \mid s_t = s, a_t = a\} \tag{49}$$

In addition, in each state it is useful to check whether there is another policy that, if followed, would be able to perform better than the one currently being followed. This process is known as *policy improvement*. The action that seems the best according to the largest action-value function ($Q^\pi(s,a)$) is selected.

## 2.9 Reinforcement Learning Supporting Metaheuristics

Search methods face a number of challenges in solving combinatorial optimization problems, among them are: Avoiding a local optimum, being applicable to different problems of varying sizes and at the same time being able to produce good quality solutions in a short time [67]. Where MH and hyperheuristics seek to address these challenges [15, 68]. To better address them, hybrids between RL and MH algorithms have been created [69], in which they aim to support and improve MH.

The following are some reasons, which we consider noteworthy, why hybridization is advantageous:

- It causes the MH to be adaptive, which allows the algorithm to be applicable to different problems.
- It does not require complete information about the problem, as RL models learn by gathering experience [70].
- By using independent learning agents [71], it allows in some cases, the computational cost to be lower. Since it uses a single update formula at each step.
- If general features are used, the information learned by the RL can be used in other parts of the same problem [72].
- The behavior of various RL methods end in optimal state-action pairs [62], which can be exploited. As an example of this, one can see how the policy choosing the next action evolves at each step.

## 2.10 Temporal Difference Learning

Temporal Difference (TD) is an incremental learning procedure, specialized in problems where it is necessary to predict the solution through experience or previously obtained data [71]. TD methods, unlike conventional learning methods, learning occurs when a change in prediction occurs over time. That is, just like the Monte-Carlo (MC) method, it learns directly from experience without waiting for the final result [62].

Because both methods use experience as mentioned above, we can state that given an experience $\pi$ updates the current estimate $V(S_t)$ (Eq. 50) for non-terminal states $S_t$. And unlike the MC method, TD methods only need the next time step $t+1$ to obtain a feedback and update the reward $R_{t+1}$. So as to obtain the estimate $V(S_{t+1})$, as exemplified by the following equation:

$$V(S_t) \leftarrow V(S_t) + \alpha \left[ R_{t+1} + \gamma V(S_{t+1} - V(S_t)) \right] \tag{50}$$

## 2.11 Q-Learning: Off-policy TD Control

Among the TD algorithms, we find the Q-Learning (QL) algorithm [73], which provides agents with the ability to learn to act optimally without the need to construct domain maps. Having several possible $s$ states, where the "environment" is the current $s$ in which the agent interacts and performs decisions. The agent has a set of possible actions, which affect the reward and the next state. Once an action is performed, the state changes. When changing state, the agent receives a reward for the decision made. Where the rewards received by the agent consequently generate learning in the agent. To solve the problem, the agent learns the best course of actions it can take, which has a maximum cumulative reward. The sequence of actions from the first state to the terminal state is called an episode. The transition of states is given by the equation (51).

$$Q_{new}(s_t, a_t) = (1 - \alpha) \cdot Q_{old}(s_t, a_t) + \alpha \cdot [r_n + \gamma \cdot maxQ(s_{t+1}, a_{t+1})] \tag{51}$$

Where $Q_{new}(s_t, a_t)$ is nominating the reward of the action taken in state $s_t$ and $r_n$ is the reward received when action $a_t$ is taken, $maxQ(s_{t+1}, a_{t+1})$ is the maximum value of the action for the next state, the value of $\alpha$ must be $0 < \alpha \leq 1$ and corresponds to the learning factor. On the other hand, the value of $\gamma$ must be $0 \leq \gamma \leq 1$ and corresponds to the discount factor. If $\gamma$ reaches the value of 0, only the immediate reward will be considered, while as

it approaches the value 1 the future reward receives greater emphasis relative to the immediate reward. QL is algorithmically presented with the following pseudo-code:

**Algorithm 4** Q-Learning: Off-Policy TD Control

Algorithm parameters: $\alpha \, \epsilon \, [0, 1]$
Initialize $Q(s, a)$
**while** $t \leq$ Maximum number of iterations **do**
    Inicializamos $s$
    **while** $s \neq s_{terminal}$ **do**
        Select action $a$
        Observe $r$
        Create $s'$
        Choose from $s'$ using the policy obtained from $Q$
        $Q(s, a) \longleftarrow (1 - \alpha) \cdot Q(s, a) + \alpha \cdot [r + \gamma \cdot maxQ(s', a')]$
    **end while**
    $t \leftarrow t_{+1}$
**end while**

## 2.12 SARSA: On-policy TD Control

It is a learning reinforcement method, belonging to the TD learning prediction methods. It uses the generalized policy iteration pattern, which consists of two processes that are performed simultaneously and interact with each other. Where one performs the value function, with the current policy, and at the same time the other one improves the current policy. These two processes complement each other and in each iteration, but it is not necessary that each one is completed before the next one begins.

The SARSA algorithm is divided into on-policy and off-policy approaches. In the present work we will focus on its on-policy variable, where a learning agent learns the current value function derived from the policy currently in use. To understand how it works, the first step is to learn an action-value function instead of a state-value function. In particular, for the on-policy method we must estimate $Q_\pi(s, a)$ for the current behavioral policy $\pi$ and for all states $s$ and actions $a$. It is important to emphasize that an episode consists of a sequence of states and state-action pairs.

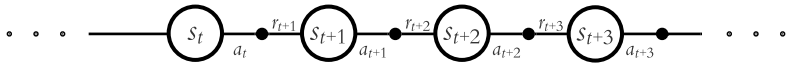


Figure 23: SARSA Algorithm Sequence

In Figure 23 the state-to-state transitions are considered and the values of each have been learned. To understand the algorithm, let us consider the transitions as a pair of values, state-action to state-action, where the values of the state-action pairs are learned. Formally these cases are identical: both are Markov chains with a reward process. The theorems that ensure convergence of state values under TD also apply to the corresponding algorithm for action values, with the following equation:

$$Q(s_t, a_t) \longleftarrow Q(s_t, a_t) + \alpha \cdot [r_{t+1} + \gamma \cdot Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \qquad (52)$$

After each transition the state is updated, until a terminal state is reached. When a state $s_{t+1}$ is terminal then $Q(s_{t+1},\ a_{t+1})$ is defined as zero. Each transition process is composed of five events: $s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1}$ (State-Action-Reward-State-Action); giving name to the SARSA algorithm. The pseudo-code of the algorithm is shown below:.

**Algorithm 5** SARSA: On-Policy TD Control

Algorithm parameters: $\alpha \in [0, 1]$
Initialize $Q(s, a)$
**while** $t \leq$ Maximum number of iterations **do**
    Initialize $s$
    Choose $a$ of $s$ using the policy obtained from $Q$
    **while** $s \neq s_{terminal}$ **do**
        Select action $a$
        Observe $r$
        Create $s'$
        Choose $a$ of $s$ using the policy obtained from $Q$
        $Q(s, a) \longleftarrow Q(s, a) + \alpha \cdot [r + \gamma \cdot Q(s', a') - Q(s, a)]$
    **end while**
    $t \leftarrow t_{+1}$
**end while**

## 2.13 Difference between Q-Learning and SARSA

The two algorithms; QL and SARSA, learn the value of the optimal state-action pairs, $Q^*(s, a)$, through the transitions from state-action pair to state-action pair. These methods are both online control algorithms. They are online algorithms in that they perform the updates of the action-value function estimate at the end of each step without waiting for the term condition. In both methods, the updated estimate of $Q^*(s, a)$ is already available for use in the next state. They are control algorithms because they perform actions to achieve their purpose, which is the estimation of the optimal state-action value function $Q^*(s, a)$.

The difference between these two RL techniques is the evaluation of the $\pi$ policy (Eq. 46).

- QL is an *off-policy* algorithm, i.e., the agent learns the value of the state-action pair independently of the action performed, since its updates are performed independently of the current action, but with respect to the action that maximizes the value of the next state-action pair.
- While SARSA is a *on-policy* algorithm, i.e., the agent learns the value of the state-action pair based on the action performed. In this way it evaluates the current policy, unlike QL which performs one policy and

evaluates another.

The Q-table formation procedure is the same for both algorithms. The only difference between them is the update rule that is followed at each step, equations (51) and (52). The different update rule allows SARSA to learn faster than QL. However, this makes SARSA a more conservative algorithm while QL is more likely to find the most optimal policy.

## 2.14 Backward Q-Learning

Backward Q-Learning is another RL technique. Although its name is similar to QL, it is not an ordinary QL updating function, this time, a backward update is performed, hence the name Backward Q-Learning.

In this structure, the action is directly affected, while the policy is indirectly affected. As the agent increases the interaction with the environment, the precise knowledge of the agent also increases. The agent thanks to its structure can improve the speed of learning, balance the explore-exploit dilemma and converge to the global minimum using those previous states, actions and information in an episode. Then, it is recorded that agents went through states, chose actions and acquired rewards in an episode, and then this information will be used to update the QL function again.

When the agent reaches the goal state in the current episode, it will utilize the data that had occurred to backward update the QL function. For example, the $s_{t-n}$ state is defined as an initial state, and the $s_{t+1}$ state is defined as a terminal state. The agent updates the QL function N times from the initial state $s_{t-n}$ to the terminal state $s_{t+1}$ in one episode. And the agent simultaneously records each element of the four events "s, a, r, s," then the agent will use the information to update backward the QL function N times. Therefore, we redefine the one-step QL function as:

$$Q(s_t^i, a_t^i) \longleftarrow Q(s_t^i, a_t^i) + \alpha \cdot [r_{t+1}^i + \gamma \cdot maxQ(s_{t+1}^i, a_{t+1}^i) - Q(s_t^i, a_t^i)] \quad (53)$$

for $i = 1, 2, ..., N$ where $i$ is the number of times for update the QL function in current episode. And the agent simultaneously records the four events in $M^i$, that is:

$$M^i \longleftarrow s_t^i, a_t^i, r_t^i, s_{t+1}^i \quad (54)$$

when the agent reaches the goal state, the agent will backward update the QL function based on the information of $M^i$ as follows:

$$Q(s_t^j, a_t^j) \longleftarrow Q(s_t^j, a_t^j) + \alpha \cdot [r_{t+1}^j + \gamma \cdot maxQ(s_{t+1}^j, a_{t+1}^j) - Q(s_t^j, a_t^j)] \quad (55)$$

where $j = N, N-1, N-2, ..., 1$. Hence, the agent can utilize the previous information to backward update the QL function. A pseudo-code is added for a better understanding of the above:

**Algorithm 6** Backward Q-Learning

Initialize arbitrarily all $Q(s, a)$, M and set $\alpha$ and $\gamma$
**for** each episode **do**
    Choose a random state $s_t$ or initialize state $s_t$
    Choose an action $a_t$ from state $s_t$
    **while** $N \geq 1$ **do**
        **for** each step in the episode **do**
            Execute the select action $a_t^i$ to the environment, then receive immediate reward $r_{t+1}$ and observe the new state $s_{t+1}^i$
            Choose an action $a_{t+1}^i$ from state $s_{t+1}^i$
            Record the four events in $M^i \longleftarrow s_t^i, a_t^i, r_t^i, s_{t+1}^i$
            Update $Q(s_t^j, a_t^j)$ using the Eq. 53
            Update state and action: $s_t^i \longleftarrow s_{t+1}^i$ and $a_t^i \longleftarrow a_{t+1}^i$
            $i \leftarrow i_{+1}$
        **end for**
    **end while**
    **for** $j = N$ to 1 **do**
        Backward update $Q(s_t^j, a_t^j)$ using Eq. 55
    **end for**
    Initialize all $M$ values
**end for**

## 2.15 Set Covering Problem

The SCP is a classical covering problem that consists into find a subset of columns in a zero-one matrix such that they can cover all the rows of that matrix at a minimum cost. Let $A = (a_{ij})$ be a $m \times n$ binary matrix with $I = \{1, \ldots, m\}$ and $J = \{1, \ldots, n\}$ being the row and column sets respectively. We say that a column $j$ can cover a row $i$ if $a_{ij} = 1$. The cost of selecting the column $j$ is represented by $c_j$, a non-negative value, and $x_j$ is a decision variable to indicate if the column $j$ is selected ($x_j = 1$) or not ($x_j = 0$).

$$Minimize \sum_{j=1}^{n} c_j x_j \tag{56}$$

Subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad \forall i \in I \tag{57}$$

$$x_j \in \{0, 1\} \quad \forall j \in J \tag{58}$$

One of the many practical applications of this problem is the location of fire stations. Lets consider a city divided in a finite number of areas which need to locate and build fire stations. Each one of this areas need to be covered by at least one station, but a single fire station can only bring coverage to its own area and the adjacent ones; also, the problem requires that the number of stations to build needs to be the minimum.

Intentionally, we have selected an instance of SCP with $m = 11$ and $n = 11$ to represent it graphically in figures (24), (25) and by Eq. (59 - 70). When a SCP formulation has a constant cost (a value of 1 in this case), we will refer to it as an *Unicost* SCP instance.

$$Minimize \sum_{j=1}^{11} c_j x_j \tag{59}$$

$$
\begin{aligned}
AREA_1: &\quad x_1 + x_2 + x_3 + x_4 && \geq 1 && (60)\\
AREA_2: &\quad x_1 + x_2 + x_3 + x_5 && \geq 1 && (61)\\
AREA_3: &\quad x_1 + x_2 + x_3 + x_4 + x_5 + x_6 && \geq 1 && (62)\\
AREA_4: &\quad x_1 + x_3 + x_4 + x_6 + x_7 && \geq 1 && (63)\\
AREA_5: &\quad x_2 + x_3 + x_5 + x_6 + x_8 + x_9 && \geq 1 && (64)\\
AREA_6: &\quad x_3 + x_4 + x_5 + x_6 + x_7 + x_8 && \geq 1 && (65)\\
AREA_7: &\quad x_4 + x_6 + x_7 + x_8 && \geq 1 && (66)\\
AREA_8: &\quad x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} && \geq 1 && (67)\\
AREA_9: &\quad x_5 + x_8 + x_9 + x_{10} + x_{11} && \geq 1 && (68)\\
AREA_{10}: &\quad x_8 + x_9 + x_{10} + x_{11} && \geq 1 && (69)\\
AREA_{11}: &\quad x_9 + x_{10} + x_{11} && \geq 1 && (70)
\end{aligned}
$$

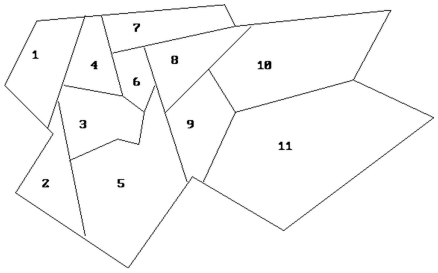


Figure 24: An example of SCP

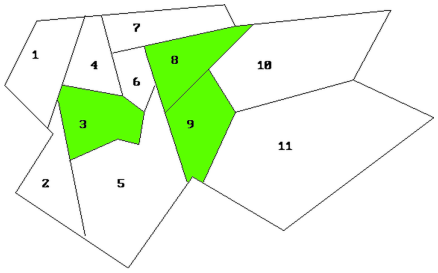


Figure 25: Solution to the practical example of SCP

## 2.16 Knapsack Problem

The Knapsack Problem (KP) is categorized as a *NP-hard* problem. It assumes a knapsack with a maximum capacity $C$ and a set of objects $N$. Where each object element has a value $p_i$ and a weight $w_i$. The objective is to select a subset of objects belonging to $N$ that the sum maximizes the value that can be stored inside the knapsack, without exceeding the maximum weight of the knapsack [74, 75]. Mathematically it is defined as:

$$
Maximize \quad \sum_{i=1}^{n} p_i \tag{71}
$$

Subject to:

$$\sum_{i=1}^{n} w_i \leq C \tag{72}$$

# 3 Proposal

This proposal focuses on a framework that compiles several RL techniques to support the selection of binarization schemes in swarm-based metaheuristics when solving combinatorial problems. Several adjustments will be made to their original versions, such as transfer functions. In this way, giving our intelligent selector an even larger repertoire to choose from.

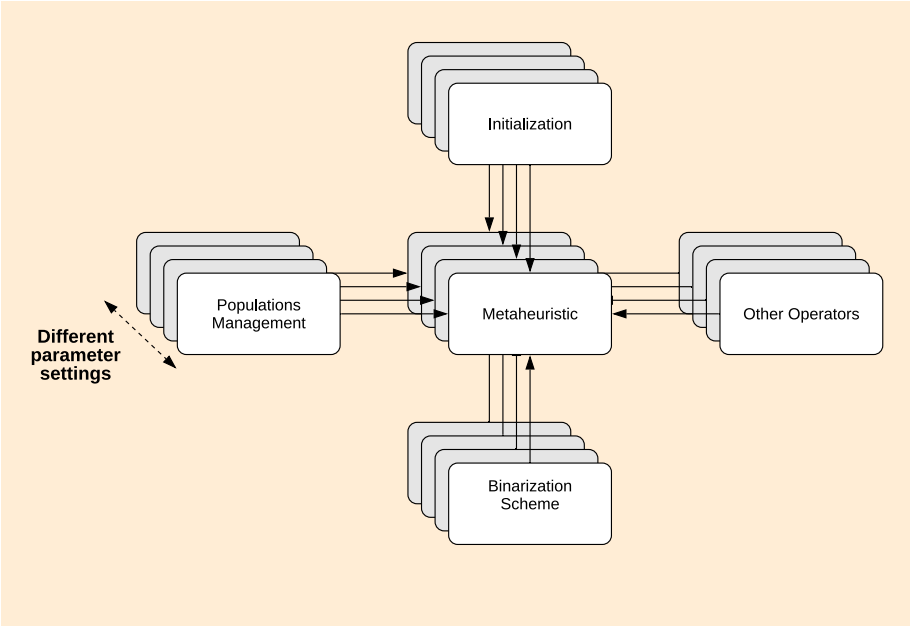


Figure 26: Metaheuristic Scheme

The Figure exemplifies the multiple configurations that a MH can have, understanding these as a set of operators, which have parameters to be configured according to the problem to be solved.

## 3.1 Binarization Scheme Selector

As previously stated, the aim of this work is to realize a framework in which different MHs are hybridized with RL techniques. Part of the work consists of testing these RL techniques as intelligent selectors that allow us to autonomously choose the binarization to be used in each of the iterations (Fig. 27) in an online way, thus affecting the exploration-exploitation tradeoff and thus managing to evade local optima, recall that when we refer to online, it means that our selector makes a decision after each iteration without the need to wait for the term condition. The other part of this work is to introduce other components to the scheme, for example, more transfer functions and other binarization techniques. There are studies that have previously performed RL techniques, namely QL for the selection of binarization techniques [11–14].

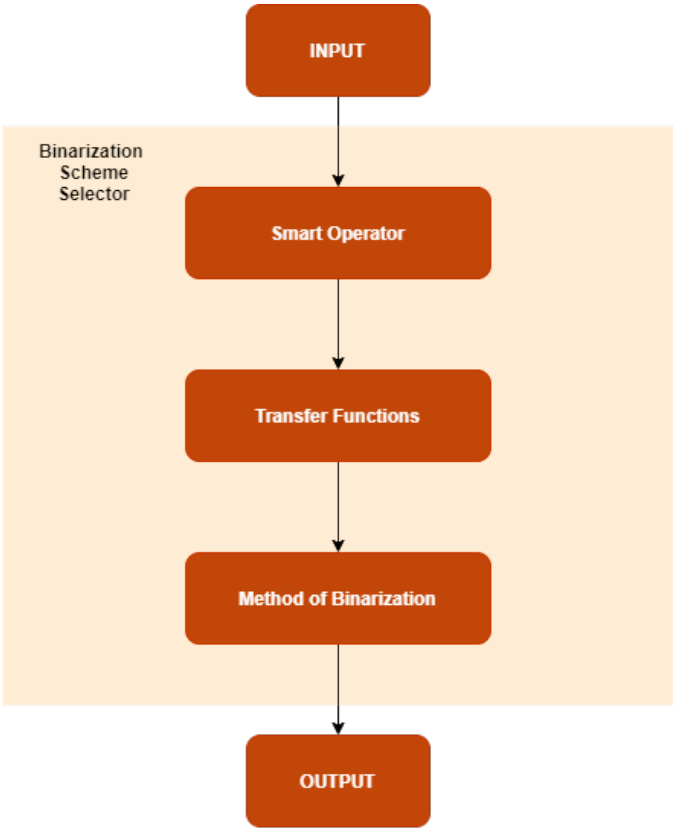


Figure 27: General Proposal Scheme

One of the strengths of this research is the choice of different RL techniques, currently two techniques from the list of considered ones are implemented: QL and SARSA. Both techniques are implemented as an intelligent operator, which chooses the binarization technique to be used based on a reward system, with which it learns deterministically.

In the first instance, the choice has been limited to the V4 transfer function, in addition to using the elitist and complement binarization method, as it is one of the most widely used [10, 29, 34, 76] to later compare them with our versions. These two static, i.e., independent versions of our proposal can be seen in Table (3). The structure of the implemented proposal is exemplified in Fig. (28).

Table 3: Recommended binarization schemes in the literature.

| Cite | Binarization | Transfer Function | Name |
|---|---|---|---|
| [76] | Elitist (Eq. 43) | V4 (Eq. 27) | BCL |
| [29] | Complement (Eq. 41) | V4 (Eq. 27) | MIR |

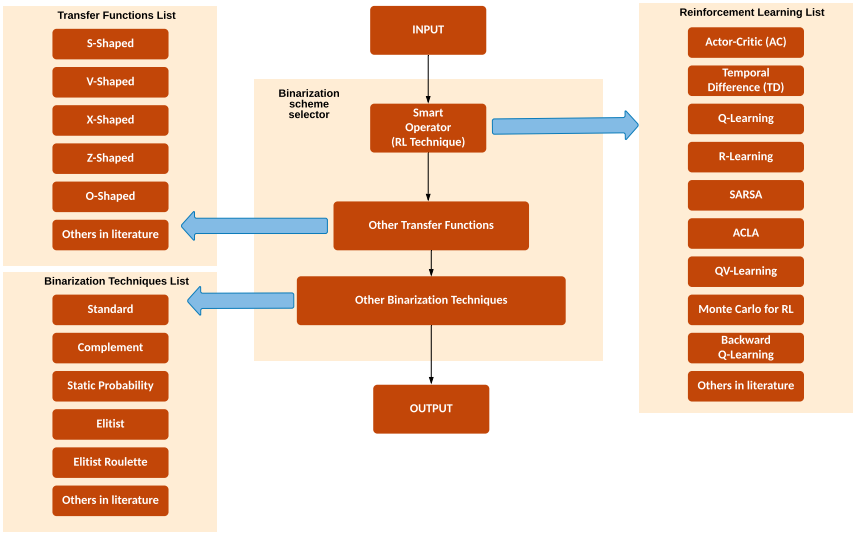


Figure 28: Proposal Scheme with the combinations

## 3.2 Rewards

The rewards in the RL algorithms is fundamental for a correct performance, that is why in the literature there are several ways to calculate the rewards [77–80]. The type of reward from the chosen metric determines the $R_t$ value for the general equations of the proposed RL techniques.

As a first instance in our proposal, the following forms of rewards will be considered. Among the simplest in the literature is the first, used in the work of Xu et al. [77] and Choong et al. [78], where it is increased by a fixed value for the action that generated an improvement in overall fitness, and a decrease or punishment of the same fixed value in case no improvement was achieved. The second type of reward is a variation of the previous one used in Abed work [79], where there is no penalty for the action taken. Finally, we have three more types of rewards collected by Nareyek in [80]. All these rewards are detailed in the Table (4):

Table 4: Reward Types

| **Name of Reward** | **Mathematical Formula** | |
|---|---|---|
| With Penalty | $\mathrm{r}_n = \begin{cases} +1, & \text{If fitness improves} \\ -1, & \text{Otherwise} \end{cases}$ | (73) |
| Without Penalty | $\mathrm{r}_n = \begin{cases} +1, & \text{If fitness improves} \\ 0, & \text{Otherwise} \end{cases}$ | (74) |
| Global Best | $\mathrm{r}_n = \begin{cases} \frac{W}{BestFitness}, & \text{If fitness improves} \\ 0, & \text{Otherwise} \end{cases}$ | (75) |
| Root Adaption | $\mathrm{r}_n = \begin{cases} \sqrt{BestFitness}, & \text{If fitness improves} \\ 0, & \text{Otherwise} \end{cases}$ | (76) |
| Scalating Adaption | $\mathrm{r}_n = \begin{cases} W \cdot BestFitness, & \text{If fitness improves} \\ 0, & \text{Otherwise} \end{cases}$ | (77) |

## 3.3 Metrics

As previously stated, there are a variety of rewards whose metric to be evaluated is fitness improvement. This means that if there is an improvement in overall fitness, a reward will be given, while if it is maintained and does not

undergo improvement, the RL technique will give a penalty if so defined in the reward type. Recently the work of Chen et al. [81] looks at other metrics that vary from fitness or different from fitness, such as population diversity, average fitness of the population and the fitness of the best individual. Another of the motivations of this work is to venture, like Chen, to search and experiment with other metrics to evaluate in the metaheuristic process and not to focus so much on the classical ones, in this way to be able to have results and compare.

### 3.4 Determination of States

Thanks to the work of Choi et al. [82], there are different ways to determine diversity. In the first instance the determination of diversity is calculated based on the work of Hussain et al. [83], represented mathematically as:

$$Div = \frac{1}{l \cdot n} \sum_{d=1}^{l} \sum_{i=1}^{n} |\bar{x}^d - x_i^d|, \tag{78}$$

Where $Div$ is the diversity status determination, $\bar{x}^d$ denotes the mean of the individuals in dimension $d$, $x_i^d$ is the value of the $i$-th individual of the $d$-th dimension, $n$ is the number of individuals in the population, and $l$ is the size of the dimension of the individuals.

Let $XPL\%$ and $XPT\%$ be the exploration and exploitation percentages, respectively. The values of $XPL\%$ and $XPT\%$ are calculated from the work of Morales-Castañeda et al. [84] as follows:

$$XPL\% = \frac{Div}{Div_{max}} \cdot 100, \tag{79}$$

$$XPT\% = \frac{|Div - Div_{max}|}{Div_{max}} \cdot 100. \tag{80}$$

Where from both equations, $Div$ is the diversity state determination given by the equation (78) and $Div_{max}$ denotes the maximum value of the diversity state found in the whole optimization problem.

### 3.5 Repair of Infeasible Solutions

For the KP problem, the decision was made to use two widely known repair strategies [85], *greedy add* (Algorithm 7) and *greedy drop* (Algorithm 8); which are adapted depending on whether the proposed solution in each

state is feasible or infeasible respectively. First the operator *greedy add*, is used to repair the feasible solutions and thus improve the already proposed solution. Then the operator *greedy drop* is used at the moment of receiving an infeasible solution, so it is necessary to remove items inside the knapsack. The metric used by these operators is the density, which is the ratio between the profit ($p_i$) and the weight ($w_i$) of an item. Using the metric, the operators analyze each item found in the solution vector $V$ in increasing order and change the value of the item to 0 or 1, depending on whether the item $x_i$ meets the constraint or not.

**Algorithm 7** Greedy add operator

```
1: Input values: V, w_total, C
2: for i = 0 to n-1 do
3:     if V[i] = 0 and w_total ≤ C then
4:         V[i] = 1
5:         w_total = w_total + w_i
6:     end if
7: end for
```

**Algorithm 8** Greedy drop operator

```
1: Input values: V, w_total, C
2: for i = n-1 to 0 do
3:     if V[i] = 1 and w_total > C then
4:         V[i] = 0
5:         w_total = w_total + w_i
6:     end if
7: end for
```

In the SCP problem, the repair operator is used on the solutions delivered by K-means and the operator [86]. The repair process starts with a solution vector $S_{input}$, after analysis of the operator the vector $S_{output}$ is obtained (Algorithm 9). The repair strategy starts by iteratively using the heuristic operator indicating the column to be added. Once all the rows are covered, the columns that have all their rows covered by other columns are removed.

**Algorithm 9** SCP Repair

**Input values:** $S_{input}$
$S = S_{input}$
**while** $repair(S) ==$ True **do**
    $S.append(Heuristic(S))$
**end while**
$S \leftarrow itemRepeated(S)$
$S_{output} = S$

# 4 Preliminary Results

Since the present work consists of the compilation of multiple RL techniques, in this installment we have reached the point of implementing three techniques that we have already mentioned and explained; QL, SARSA and Backward Q-Learning. In Table (5) are the implemented configurations.

Table 5: Details of the implemented configuration

| Reinforcement Learning Technique | Transfer Functions Used | Total of Rewards Used | Metaheuristic Used | Problem Solved | Total Instances Used |
|---|---|---|---|---|---|
| Q-Learning | S and V | 5 | WOA GWO SCA | SCP | 45 |
| | S,V,X,Z and O1 | 1 | | | |
| SARSA | S and V | 5 | | | |
| | S,V,X,Z,O1 | 1 | | | |
| Backward Q-Learning | S and V | 1 | | | |
| | S,V,X,Z and O1 | 1 | | | |
| Q-Learning | S and V | 5 | | 0-1KP | 19 |
| SARSA | | | | | |

It should be noted, that in order to facilitate the reading of the tables to come, different acronyms or identifiers have been assigned, as shown in Table (6).

Table 6: SARSA and Q-Learning implementations

| Reward Types | Name |
|---|---|
| With Penalty (Eq. 73) | SA1 |
| Without Penalty (Eq. 74) | SA2 |
| Global Best (Eq. 75) | SA3 |
| Root Adaption (Eq. 76) | SA4 |
| Scalating Adaption. (Eq. 77) | SA5 |
| With Penalty (Eq. 73) | QL1 |
| Without Penalty (Eq. 74) | QL2 |
| Global Best (Eq. 75) | QL3 |
| Root Adaption (Eq. 76) | QL4 |
| Scalating Adaption. (Eq. 77) | QL5 |
| With Penalty (Eq. 73) | BQSA1 |

### 4.1 Statistical Tests

To determine which of the implemented MHs performs best with our smart selector in solving the problems, we employ statistical methodologies that provide a viable option to compare our results.

To compare two MH implementations, we utilize the Wilcoxon-Mann-Whitney statistical test, a non-parametric test that is employed when data are not normally distributed and are independent of each other. When calculating and evaluating the p-value, we cannot infer that the results of our implementation A are worse than the results of B if the value is less than 0.05. This assessment is connected to the hypothesis that is the subject of the statistical test:

$H_0$ = Algorithm A $\geq$ Algorithm B
$H_1$ = Algorithm A $<$ Algorithm B

### 4.2 Experimental Results

In this work, experiments have been performed on two problems, Set Covering Problem and 0-1 Knapsack Problem. The experiments solving the SCP with the Beasley OR Library instances add up to 45 instances [87]. For the

Table 7: Configuration details from SCP instances employed in this work

| Instance set | m | n | Cost range |
|---|---|---|---|
| 4 | 200 | 1000 | [1,100] |
| 5 | 200 | 2000 | [1,100] |
| 6 | 200 | 1000 | [1,100] |
| A | 300 | 3000 | [1,100] |
| B | 300 | 3000 | [1,100] |
| C | 400 | 4000 | [1,100] |
| D | 400 | 4000 | [1,100] |

experiments solving the 0-1KP we consider 19 of the 31 instances recorded in the literature [88, 89].

In this work, experiments have been performed on two problems, Set Covering Problem and 0-1 Knapsack Problem. The experiments solving the SCP with the Beasley OR Library instances add up to 45 instances [87]. For experiments solving the 0-1KP we consider 19 of the 31 instances recorded in the literature and used in the [88, 89], these instances can be downloaded at: `http://artemisa.unicauca.edu.co/~johnyortega/instances_01_KP/`. These instances can be categorized into two groups:

- Low-dimensional, Nºs 1-10.
- Large-dimensional, Nºs 11-19.

Tables (7) and (8) contain descriptions of the datasets used in both problems.

The instances for the SCP problem were run with a total of 40 population and 1000 iterations, having a total of 40,000 calls to the objective function, as used in the work of Lanza et al. [76]. The instances for the 0-1KP problem were run with a total of 20 population and 5000 iterations, having a total of 100,000 calls to the objective function, as used in the work of Abdel-Basset et al. [88]. The implementation was developed in Python 3.8.5 and processed using the free Google Colaboraty service [90]. The parameter settings of the RL technique algorithms are presented in Table (9).

Table 8: 0-1KP Datasets Descriptions

| **No** | **Dataset** | **Capacity** | **Dimension** |
|---|---|---|---|
| 1 | KP1 | 269 | 10 |
| 2 | KP2 | 878 | 20 |
| 3 | KP3 | 20 | 4 |
| 4 | KP4 | 11 | 4 |
| 5 | KP5 | 375 | 15 |
| 6 | KP6 | 60 | 10 |
| 7 | KP7 | 50 | 7 |
| 8 | KP8 | 10.000 | 23 |
| 9 | KP9 | 80 | 5 |
| 10 | KP10 | 879 | 20 |
| ***Uncorrelated data instances*** | | | |
| 11 | KP1 100 | 100 | 1.000 |
| 12 | KP1 200 | 200 | 1.000 |
| 13 | KP1 500 | 500 | 1.000 |
| ***Weakly correlated instances*** | | | |
| 14 | KP2 100 | 100 | 1.000 |
| 15 | KP2 200 | 200 | 1.000 |
| 16 | KP2 500 | 500 | 1.000 |
| ***Strongly correlated instances*** | | | |
| 17 | KP3 100 | 100 | 1.000 |
| 18 | KP3 200 | 200 | 1.000 |
| 19 | KP3 500 | 500 | 1.000 |

Table 9: Parameters Settings of the Reinforcement Learning Techniques

| **Reinforcement Learning Technique** | $\alpha$ | $\gamma$ |
|---|---|---|
| Q-Learning<br>SARSA<br>Backward Q-Learning | 0.1 | 0.4 |

Summary tables (10-12) are presented for ease of reading, in these tables the results of the three MH with QL, SARSA and Backward Q-Learning solving SCP and 0-1KP can observed. In these tables we have highlighted the best values, and we have also underlined the RPD of the best solutions obtained. More details can be seen in the tables (13-24) where all instances and each of their values are shown, not only the last row of the table. The complete tables are composed as follows: the first column (Inst.) presents the name of each solved instance, in the second column (Opt.), the optimal value of each instance. While the next 3 columns present the best results (Best), the average results (Avg) and the relative percentage deviation calculated according to Eq. (81), all for each of the implemented versions. Finally, the last row is the sum of each column, presented above in the summary tables.

$$\text{RPD} = \frac{100 \cdot (Best - Opt)}{Opt}. \tag{81}$$

Table 10: Summary Table of SCP with 85 Actions

| | **WOA** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | BCL | | MIR | | QL1 | | SA1 | |
| | Best | RPD | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | 490.31 | 178.89 | 1195.82 | 787.64 | 264.84 | 4.37 | **263.84** | 4.02 |
| | **GWO** | | | | | | | |
| | BCL | | MIR | | QL1 | | SA1 | |
| | Best | RPD | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | **260.11** | 2.42 | 272.22 | 11.75 | 266.91 | 5.45 | 265.04 | 4.83 |
| | **SCA** | | | | | | | |
| | BCL | | MIR | | QL1 | | SA1 | |
| | Best | RPD | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | 305.56 | 20.45 | 1123.22 | 728.24 | 266.27 | 4.9 | **265.33** | 4.59 |

Table 11: Summary Table of SCP with 40 Actions

| | **WOA** | | | | | |
|---|---|---|---|---|---|---|
| | BCL | | MIR | | BQSA1 | |
| | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | 490.31 | 178.89 | 1195.82 | 787.64 | **266.71** | 5.08 |
| | **GWO** | | | | | |
| | BCL | | MIR | | BQSA1 | |
| | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | **260.11** | 2.42 | 272.22 | 11.75 | 268.6 | 6.19 |
| | **SCA** | | | | | |
| | BCL | | MIR | | BQSA1 | |
| | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | 305.56 | 20.45 | 1123.22 | 728.24 | **268.02** | 5.99 |

Table 12: Summary Table of 01-KP

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **WOA** | | | | | | | | | |
| | **SA1** | | **SA2** | | **SA3** | | **SA4** | | **SA5** | |
| | Best | RPD | Best | RPD | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | 2347.0 | 17.7 | 2329.06 | 17.81 | 2282.16 | 18.16 | 2324.79 | 18.35 | 2330.64 | 17.73 |
| | **QL1** | | **QL2** | | **QL3** | | **QL4** | | **QL5** | |
| | Best | RPD | Best | RPD | Best | RPD | Best | Avg | Best | RPD |
| $\sum$ | **2410.42** | 17.22 | 2390.95 | 17.46 | 2343.85 | 18.04 | 2304.11 | 18.18 | 2381.16 | 17.54 |
| | **GWO** | | | | | | | | | |
| | **SA1** | | **SA2** | | **SA3** | | **SA4** | | **SA5** | |
| | Best | RPD | Best | RPD | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | **2570.37** | 15.21 | 2567.32 | 15.16 | 2566.32 | 15.15 | 2538.74 | 15.39 | 2545.42 | 15.35 |
| | **QL1** | | **QL2** | | **QL3** | | **QL4** | | **QL5** | |
| | Best | RPD | Best | RPD | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | 2560.74 | 14.98 | 2477.0 | 16.11 | 2488.53 | 16.22 | 2502.0 | 16.0 | 2486.64 | 16.05 |
| | **SCA** | | | | | | | | | |
| | **SA1** | | **SA2** | | **SA3** | | **SA4** | | **SA5** | |
| | Best | RPD | Best | RPD | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | **2417.0** | 17.17 | 2323.79 | 17.79 | 2315.58 | 17.98 | 2348.42 | 17.77 | 2326.37 | 17.95 |
| | **QL1** | | **QL2** | | **QL3** | | **QL4** | | **QL5** | |
| | Best | RPD | Best | RPD | Best | RPD | Best | RPD | Best | RPD |
| $\sum$ | 2384.06 | 17.31 | 2344.0 | 17.85 | 2344.74 | 17.61 | 2335.27 | 17.87 | 2343.74 | 17.97 |

Table 13: SCP - WOA - 85 ACTIONS

| | | BCL | | | MIR | | | QL1 | | | SA1 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | Opt. | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| 41 | 429 | 542.0 | 583.7 | 26.34 | 683.0 | 713.4 | 59.21 | 438.0 | 438.0 | 2.1 | 434.0 | 434.0 | 1.17 |
| 42 | 512 | 756.0 | 809.9 | 47.66 | 1114.0 | 1163.8 | 117.58 | 543.0 | 543.0 | 6.05 | 538.0 | 538.0 | 5.08 |
| 43 | 516 | 739.0 | 846.9 | 43.22 | 1230.0 | 1264.7 | 138.37 | 539.0 | 539.0 | 4.46 | 538.0 | 538.0 | 4.26 |
| 44 | 494 | 596.0 | 739.8 | 20.65 | 954.0 | 1032.5 | 93.12 | 523.0 | 523.0 | 5.87 | 518.0 | 518.0 | 4.86 |
| 45 | 512 | 755.0 | 821.0 | 47.46 | 1121.0 | 1192.6 | 118.95 | 537.0 | 537.0 | 4.88 | 530.0 | 530.0 | 3.52 |
| 46 | 560 | 815.0 | 913.0 | 45.54 | 1391.0 | 1464.0 | 148.39 | 572.0 | 572.0 | 2.14 | 571.0 | 571.0 | 1.96 |
| 47 | 430 | 597.0 | 652.2 | 38.84 | 854.0 | 912.2 | 98.6 | 440.0 | 440.0 | 2.33 | 441.0 | 441.0 | 2.56 |
| 48 | 492 | 757.0 | 844.3 | 53.86 | 1148.0 | 1208.4 | 133.33 | 504.0 | 504.0 | 2.44 | 500.0 | 500.0 | 1.63 |
| 49 | 641 | 922.0 | 1031.1 | 43.84 | 1586.0 | 1678.7 | 147.43 | 680.0 | 680.0 | 6.08 | 676.0 | 676.0 | 5.46 |
| 410 | 514 | 696.0 | 787.2 | 35.41 | 1090.0 | 1153.4 | 112.06 | 528.0 | 528.0 | 2.72 | 527.0 | 527.0 | 2.53 |
| 51 | 253 | 378.0 | 398.4 | 49.41 | 582.0 | 607.0 | 130.04 | 260.0 | 260.0 | 2.77 | 262.0 | 262.0 | 3.56 |
| 52 | 302 | 502.0 | 542.2 | 66.23 | 804.0 | 879.0 | 166.23 | 330.0 | 330.0 | 9.27 | 328.0 | 328.0 | 8.61 |
| 53 | 226 | 320.0 | 363.3 | 41.59 | 506.0 | 547.0 | 123.89 | 233.0 | 233.0 | 3.1 | 231.0 | 231.0 | 2.21 |
| 54 | 242 | 318.0 | 366.5 | 31.4 | 540.0 | 566.6 | 123.14 | 251.0 | 251.0 | 3.72 | 251.0 | 251.0 | 3.72 |
| 55 | 211 | 305.0 | 327.7 | 44.55 | 397.0 | 417.5 | 88.15 | 217.0 | 217.0 | 2.84 | 217.0 | 217.0 | 2.84 |
| 56 | 213 | 358.0 | 379.3 | 68.08 | 531.0 | 540.8 | 149.3 | 226.0 | 227.0 | 6.1 | 224.0 | 224.0 | 5.16 |
| 57 | 293 | 387.0 | 475.7 | 32.08 | 642.0 | 703.8 | 119.11 | 306.0 | 306.0 | 4.44 | 305.0 | 305.0 | 4.1 |
| 58 | 288 | 422.0 | 468.7 | 46.53 | 673.0 | 732.6 | 133.68 | 297.0 | 297.0 | 3.12 | 296.0 | 296.0 | 2.78 |
| 59 | 279 | 430.0 | 495.5 | 54.12 | 698.0 | 728.8 | 150.18 | 284.0 | 284.0 | 1.79 | 286.0 | 286.0 | 2.51 |
| 510 | 265 | 429.0 | 455.2 | 61.89 | 594.0 | 655.2 | 124.15 | 273.0 | 273.0 | 3.02 | 273.0 | 273.0 | 3.02 |
| 61 | 138 | 285.0 | 356.5 | 106.52 | 752.0 | 818.9 | 444.93 | 144.0 | 144.0 | 4.35 | 144.0 | 144.0 | 4.35 |
| 62 | 146 | 413.0 | 509.5 | 182.88 | 1100.0 | 1175.9 | 653.42 | 152.0 | 152.0 | 4.11 | 153.0 | 153.0 | 4.79 |
| 63 | 145 | 337.0 | 450.9 | 132.41 | 1030.0 | 1099.7 | 610.34 | 149.0 | 149.0 | 2.76 | 147.0 | 147.0 | 1.38 |
| 64 | 131 | 286.0 | 327.3 | 118.32 | 655.0 | 704.3 | 400.0 | 134.0 | 134.0 | 2.29 | 133.0 | 133.0 | 1.53 |
| 65 | 161 | 357.0 | 429.1 | 121.74 | 1050.0 | 1124.6 | 552.17 | 176.0 | 176.0 | 9.32 | 172.0 | 172.0 | 6.83 |
| a1 | 253 | 479.0 | 577.4 | 89.33 | 1243.0 | 1323.1 | 391.3 | 266.0 | 266.0 | 5.14 | 264.0 | 264.0 | 4.35 |
| a2 | 252 | 452.0 | 613.1 | 79.37 | 1150.0 | 1211.2 | 356.35 | 267.0 | 267.0 | 5.95 | 266.0 | 266.0 | 5.56 |
| a3 | 232 | 436.0 | 526.0 | 87.93 | 1117.0 | 1174.2 | 381.47 | 243.0 | 243.0 | 4.74 | 243.0 | 243.0 | 4.74 |
| a4 | 234 | 469.0 | 558.6 | 100.43 | 1080.0 | 1136.9 | 361.54 | 242.0 | 242.0 | 3.42 | 247.0 | 247.0 | 5.56 |
| a5 | 236 | 447.0 | 576.1 | 89.41 | 1139.0 | 1168.0 | 382.63 | 247.0 | 247.0 | 4.66 | 245.0 | 245.0 | 3.81 |
| b1 | 69 | 380.0 | 509.0 | 450.72 | 1353.0 | 1407.2 | 1860.87 | 71.0 | 71.0 | 2.9 | 71.0 | 71.0 | 2.9 |
| b2 | 76 | 374.0 | 508.2 | 392.11 | 1265.0 | 1372.7 | 1564.47 | 78.0 | 78.0 | 2.63 | 78.0 | 78.0 | 2.63 |
| b3 | 80 | 468.0 | 574.8 | 485.0 | 1753.0 | 1808.5 | 2091.25 | 81.0 | 81.0 | 1.25 | 82.0 | 82.0 | 2.5 |
| b4 | 79 | 372.0 | 589.4 | 370.89 | 1536.0 | 1616.1 | 1844.3 | 83.0 | 83.0 | 5.06 | 83.0 | 83.0 | 5.06 |
| b5 | 72 | 376.0 | 440.5 | 422.22 | 1372.0 | 1441.0 | 1805.56 | 74.0 | 74.0 | 2.78 | 72.0 | 73.0 | 0.0 |
| c1 | 227 | 523.0 | 671.9 | 130.4 | 1488.0 | 1581.9 | 555.51 | 242.0 | 242.0 | 6.61 | 243.0 | 243.0 | 7.05 |
| c2 | 219 | 507.0 | 629.8 | 131.51 | 1654.0 | 1741.8 | 655.25 | 237.0 | 237.0 | 8.22 | 232.0 | 232.0 | 5.94 |
| c3 | 243 | 629.0 | 759.5 | 158.85 | 2059.0 | 2123.0 | 747.33 | 256.0 | 256.0 | 5.35 | 258.0 | 258.0 | 6.17 |
| c4 | 219 | 570.0 | 716.0 | 160.27 | 1645.0 | 1721.1 | 651.14 | 232.0 | 232.0 | 5.94 | 231.0 | 231.0 | 5.48 |
| c5 | 215 | 496.0 | 636.9 | 130.7 | 1619.0 | 1684.6 | 653.02 | 226.0 | 226.0 | 5.12 | 227.0 | 227.0 | 5.58 |
| d1 | 60 | 419.0 | 620.8 | 598.33 | 1950.0 | 2057.5 | 3150.0 | 64.0 | 64.0 | 6.67 | 64.0 | 64.0 | 6.67 |
| d2 | 66 | 480.0 | 700.6 | 627.27 | 2264.0 | 2314.1 | 3330.3 | 69.0 | 69.0 | 4.55 | 68.0 | 68.0 | 3.03 |
| d3 | 72 | 548.0 | 735.3 | 661.11 | 2445.0 | 2526.3 | 3295.83 | 77.0 | 77.0 | 6.94 | 77.0 | 77.0 | 6.94 |
| d4 | 62 | 476.0 | 720.4 | 667.74 | 2025.0 | 2079.9 | 3166.13 | 63.0 | 63.0 | 1.61 | 63.0 | 63.0 | 1.61 |
| d5 | 61 | 461.0 | 749.8 | 655.74 | 1930.0 | 2055.9 | 3063.93 | 64.0 | 64.0 | 4.92 | 64.0 | 64.0 | 4.92 |
| | | 490.31 | 595.31 | 178.89 | 1195.82 | 1258.45 | 787.64 | 264.84 | 264.87 | 4.37 | **263.84** | 263.87 | 4.02 |

Table 14: SCP - WOA - 40 ACTIONS

| | | BCL | | | MIR | | | BQSA1 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | Opt. | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| 41 | 429 | 542.0 | 583.7 | 26.34 | 683.0 | 713.4 | 59.21 | 441.0 | 441.0 | 2.8 |
| 42 | 512 | 756.0 | 809.9 | 47.66 | 1114.0 | 1163.8 | 117.58 | 548.0 | 548.0 | 7.03 |
| 43 | 516 | 739.0 | 846.9 | 43.22 | 1230.0 | 1264.7 | 138.37 | 541.0 | 541.0 | 4.84 |
| 44 | 494 | 596.0 | 739.8 | 20.65 | 954.0 | 1032.5 | 93.12 | 520.0 | 520.0 | 5.26 |
| 45 | 512 | 755.0 | 821.0 | 47.46 | 1121.0 | 1192.6 | 118.95 | 540.0 | 540.0 | 5.47 |
| 46 | 560 | 815.0 | 913.0 | 45.54 | 1391.0 | 1464.0 | 148.39 | 586.0 | 586.0 | 4.64 |
| 47 | 430 | 597.0 | 652.2 | 38.84 | 854.0 | 912.2 | 98.6 | 445.0 | 445.0 | 3.49 |
| 48 | 492 | 757.0 | 844.3 | 53.86 | 1148.0 | 1208.4 | 133.33 | 510.0 | 510.0 | 3.66 |
| 49 | 641 | 922.0 | 1031.1 | 43.84 | 1586.0 | 1678.7 | 147.43 | 689.0 | 689.0 | 7.49 |
| 410 | 514 | 696.0 | 787.2 | 35.41 | 1090.0 | 1153.4 | 112.06 | 522.0 | 525.0 | 1.56 |
| 51 | 253 | 378.0 | 398.4 | 49.41 | 582.0 | 607.0 | 130.04 | 265.0 | 266.0 | 4.74 |
| 52 | 302 | 502.0 | 542.2 | 66.23 | 804.0 | 879.0 | 166.23 | 328.0 | 328.0 | 8.61 |
| 53 | 226 | 320.0 | 363.3 | 41.59 | 506.0 | 547.0 | 123.89 | 234.0 | 234.0 | 3.54 |
| 54 | 242 | 318.0 | 366.5 | 31.4 | 540.0 | 566.6 | 123.14 | 252.0 | 252.0 | 4.13 |
| 55 | 211 | 305.0 | 327.7 | 44.55 | 397.0 | 417.5 | 88.15 | 217.0 | 217.0 | 2.84 |
| 56 | 213 | 358.0 | 379.3 | 68.08 | 531.0 | 540.8 | 149.3 | 224.0 | 224.0 | 5.16 |
| 57 | 293 | 387.0 | 475.7 | 32.08 | 642.0 | 703.8 | 119.11 | 310.0 | 310.0 | 5.8 |
| 58 | 288 | 422.0 | 468.7 | 46.53 | 673.0 | 732.6 | 133.68 | 297.0 | 297.0 | 3.12 |
| 59 | 279 | 430.0 | 495.5 | 54.12 | 698.0 | 728.8 | 150.18 | 281.0 | 281.0 | 0.72 |
| 510 | 265 | 429.0 | 455.2 | 61.89 | 594.0 | 655.2 | 124.15 | 277.0 | 277.0 | 4.53 |
| 61 | 138 | 285.0 | 356.5 | 106.52 | 752.0 | 818.9 | 444.93 | 147.0 | 147.0 | 6.52 |
| 62 | 146 | 413.0 | 509.5 | 182.88 | 1100.0 | 1175.9 | 653.42 | 155.0 | 155.0 | 6.16 |
| 63 | 145 | 337.0 | 450.9 | 132.41 | 1030.0 | 1099.7 | 610.34 | 150.0 | 150.0 | 3.45 |
| 64 | 131 | 286.0 | 327.3 | 118.32 | 655.0 | 704.3 | 400.0 | 135.0 | 135.0 | 3.05 |
| 65 | 161 | 357.0 | 429.1 | 121.74 | 1050.0 | 1124.6 | 552.17 | 174.0 | 176.5 | 8.07 |
| a1 | 253 | 479.0 | 577.4 | 89.33 | 1243.0 | 1323.1 | 391.3 | 267.0 | 267.0 | 5.53 |
| a2 | 252 | 452.0 | 613.1 | 79.37 | 1150.0 | 1211.2 | 356.35 | 271.0 | 271.0 | 7.54 |
| a3 | 232 | 436.0 | 526.0 | 87.93 | 1117.0 | 1174.2 | 381.47 | 247.0 | 247.0 | 6.47 |
| a4 | 234 | 469.0 | 558.6 | 100.43 | 1080.0 | 1136.9 | 361.54 | 248.0 | 248.0 | 5.98 |
| a5 | 236 | 447.0 | 576.1 | 89.41 | 1139.0 | 1168.0 | 382.63 | 248.0 | 248.0 | 5.08 |
| b1 | 69 | 380.0 | 509.0 | 450.72 | 1353.0 | 1407.2 | 1860.87 | 72.0 | 72.0 | 4.35 |
| b2 | 76 | 374.0 | 508.2 | 392.11 | 1265.0 | 1372.7 | 1564.47 | 79.0 | 79.0 | 3.95 |
| b3 | 80 | 468.0 | 574.8 | 485.0 | 1753.0 | 1808.5 | 2091.25 | 82.0 | 82.0 | 2.5 |
| b4 | 79 | 372.0 | 589.4 | 370.89 | 1536.0 | 1616.1 | 1844.3 | 83.0 | 83.0 | 5.06 |
| b5 | 72 | 376.0 | 440.5 | 422.22 | 1372.0 | 1441.0 | 1805.56 | 74.0 | 74.0 | 2.78 |
| c1 | 227 | 523.0 | 671.9 | 130.4 | 1488.0 | 1581.9 | 555.51 | 248.0 | 248.0 | 9.25 |
| c2 | 219 | 507.0 | 629.8 | 131.51 | 1654.0 | 1741.8 | 655.25 | 236.0 | 236.0 | 7.76 |
| c3 | 243 | 629.0 | 759.5 | 158.85 | 2059.0 | 2123.0 | 747.33 | 259.0 | 259.0 | 6.58 |
| c4 | 219 | 570.0 | 716.0 | 160.27 | 1645.0 | 1721.1 | 651.14 | 234.0 | 234.0 | 6.85 |
| c5 | 215 | 496.0 | 636.9 | 130.7 | 1619.0 | 1684.6 | 653.02 | 228.0 | 229.5 | 6.05 |
| d1 | 60 | 419.0 | 620.8 | 598.33 | 1950.0 | 2057.5 | 3150.0 | 65.0 | 65.5 | 8.33 |
| d2 | 66 | 480.0 | 700.6 | 627.27 | 2264.0 | 2314.1 | 3330.3 | 69.0 | 69.0 | 4.55 |
| d3 | 72 | 548.0 | 735.3 | 661.11 | 2445.0 | 2526.3 | 3295.83 | 77.0 | 77.0 | 6.94 |
| d4 | 62 | 476.0 | 720.4 | 667.74 | 2025.0 | 2079.9 | 3166.13 | 63.0 | 63.0 | 1.61 |
| d5 | 61 | 461.0 | 749.8 | 655.74 | 1930.0 | 2055.9 | 3063.93 | 64.0 | 64.0 | 4.92 |
| | | 490.31 | 595.31 | 178.89 | 1195.83 | 1258.45 | 787.64 | **266.71** | 266.9 | 5.08 |

Table 15: SCP - GWO - 85 ACTIONS

| | | BCL | | | MIR | | | QL1 | | | SA1 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | Opt. | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| 41 | 429 | 432.0 | 433.5 | 0.7 | 431.0 | 434.4 | 0.47 | 433.0 | 433.0 | 0.93 | 434.0 | 434.0 | 1.17 |
| 42 | 512 | 533.0 | 539.1 | 4.1 | 543.0 | 547.8 | 6.05 | 538.0 | 538.0 | 5.08 | 537.0 | 537.0 | 4.88 |
| 43 | 516 | 527.0 | 532.5 | 2.13 | 541.0 | 546.3 | 4.84 | 544.0 | 544.0 | 5.43 | 532.0 | 532.0 | 3.1 |
| 44 | 494 | 507.0 | 514.3 | 2.63 | 513.0 | 520.5 | 3.85 | 516.0 | 516.0 | 4.45 | 512.0 | 512.0 | 3.64 |
| 45 | 512 | 523.0 | 528.2 | 2.15 | 540.0 | 549.6 | 5.47 | 543.0 | 543.0 | 6.05 | 542.0 | 542.0 | 5.86 |
| 46 | 560 | 570.0 | 579.1 | 1.79 | 579.0 | 583.2 | 3.39 | 577.0 | 577.0 | 3.04 | 573.0 | 573.0 | 2.32 |
| 47 | 430 | 434.0 | 437.8 | 0.93 | 442.0 | 445.7 | 2.79 | 443.0 | 443.0 | 3.02 | 439.0 | 439.0 | 2.09 |
| 48 | 492 | 499.0 | 506.3 | 1.42 | 509.0 | 512.6 | 3.46 | 507.0 | 507.0 | 3.05 | 503.0 | 503.0 | 2.24 |
| 49 | 641 | 656.0 | 671.4 | 2.34 | 685.0 | 690.3 | 6.86 | 682.0 | 682.0 | 6.4 | 681.0 | 681.0 | 6.24 |
| 410 | 514 | 520.0 | 525.7 | 1.17 | 528.0 | 531.2 | 2.72 | 528.0 | 528.0 | 2.72 | 524.0 | 524.0 | 1.95 |
| 51 | 253 | 260.0 | 265.8 | 2.77 | 264.0 | 266.5 | 4.35 | 266.0 | 266.0 | 5.14 | 263.0 | 263.0 | 3.95 |
| 52 | 302 | 322.0 | 328.3 | 6.62 | 326.0 | 333.7 | 7.95 | 329.0 | 329.0 | 8.94 | 326.0 | 326.0 | 7.95 |
| 53 | 226 | 230.0 | 233.3 | 1.77 | 232.0 | 234.2 | 2.65 | 234.0 | 234.0 | 3.54 | 233.0 | 233.0 | 3.1 |
| 54 | 242 | 247.0 | 249.8 | 2.07 | 250.0 | 253.1 | 3.31 | 252.0 | 252.0 | 4.13 | 248.0 | 248.0 | 2.48 |
| 55 | 211 | 212.0 | 213.8 | 0.47 | 216.0 | 217.9 | 2.37 | 218.0 | 218.0 | 3.32 | 217.0 | 217.0 | 2.84 |
| 56 | 213 | 216.0 | 223.8 | 1.41 | 222.0 | 228.2 | 4.23 | 227.0 | 227.0 | 6.57 | 222.0 | 224.0 | 4.23 |
| 57 | 293 | 299.0 | 306.8 | 2.05 | 309.0 | 311.8 | 5.46 | 311.0 | 311.0 | 6.14 | 308.0 | 308.0 | 5.12 |
| 58 | 288 | 290.0 | 296.2 | 0.69 | 297.0 | 300.7 | 3.12 | 298.0 | 298.0 | 3.47 | 296.0 | 296.0 | 2.78 |
| 59 | 279 | 282.0 | 285.0 | 1.08 | 285.0 | 293.0 | 2.15 | 289.0 | 289.0 | 3.58 | 285.0 | 285.0 | 2.15 |
| 510 | 265 | 269.0 | 275.0 | 1.51 | 277.0 | 280.7 | 4.53 | 278.0 | 278.0 | 4.91 | 275.0 | 275.0 | 3.77 |
| 61 | 138 | 140.0 | 145.2 | 1.45 | 144.0 | 148.1 | 4.35 | 146.0 | 146.0 | 5.8 | 145.0 | 145.0 | 5.07 |
| 62 | 146 | 147.0 | 152.3 | 0.68 | 155.0 | 158.9 | 6.16 | 155.0 | 155.0 | 6.16 | 154.0 | 154.0 | 5.48 |
| 63 | 145 | 147.0 | 149.8 | 1.38 | 151.0 | 151.5 | 4.14 | 150.0 | 150.0 | 3.45 | 150.0 | 150.0 | 3.45 |
| 64 | 131 | 131.0 | 134.3 | 0.0 | 134.0 | 135.3 | 2.29 | 134.0 | 134.0 | 2.29 | 134.0 | 134.0 | 2.29 |
| 65 | 161 | 167.0 | 172.3 | 3.73 | 175.0 | 183.7 | 8.7 | 173.0 | 173.0 | 7.45 | 174.0 | 174.0 | 8.07 |
| a1 | 253 | 262.0 | 263.2 | 3.56 | 271.0 | 277.6 | 7.11 | 266.0 | 266.0 | 5.14 | 266.0 | 266.0 | 5.14 |
| a2 | 252 | 263.0 | 268.1 | 4.37 | 275.0 | 279.2 | 9.13 | 269.0 | 269.0 | 6.75 | 270.0 | 270.0 | 7.14 |
| a3 | 232 | 241.0 | 245.3 | 3.88 | 250.0 | 256.1 | 7.76 | 248.0 | 248.0 | 6.9 | 246.0 | 246.0 | 6.03 |
| a4 | 234 | 244.0 | 247.4 | 4.27 | 250.0 | 257.8 | 6.84 | 252.0 | 252.0 | 7.69 | 243.0 | 243.0 | 3.85 |
| a5 | 236 | 244.0 | 246.5 | 3.39 | 255.0 | 259.0 | 8.05 | 249.0 | 249.0 | 5.51 | 247.0 | 247.0 | 4.66 |
| b1 | 69 | 70.0 | 71.1 | 1.45 | 75.0 | 81.5 | 8.7 | 70.0 | 70.0 | 1.45 | 71.0 | 71.0 | 2.9 |
| b2 | 76 | 76.0 | 78.3 | 0.0 | 86.0 | 91.3 | 13.16 | 81.0 | 81.0 | 6.58 | 80.0 | 80.0 | 5.26 |
| b3 | 80 | 81.0 | 82.2 | 1.25 | 89.0 | 98.1 | 11.25 | 84.0 | 84.0 | 5.0 | 82.0 | 82.0 | 2.5 |
| b4 | 79 | 82.0 | 82.9 | 3.8 | 95.0 | 100.0 | 20.25 | 81.0 | 81.0 | 2.53 | 84.0 | 84.0 | 6.33 |
| b5 | 72 | 73.0 | 73.5 | 1.39 | 81.0 | 90.5 | 12.5 | 74.0 | 74.0 | 2.78 | 74.0 | 74.0 | 2.78 |
| c1 | 227 | 239.0 | 248.1 | 5.29 | 263.0 | 272.7 | 15.86 | 249.0 | 249.0 | 9.69 | 250.0 | 250.0 | 10.13 |
| c2 | 219 | 233.0 | 238.1 | 6.39 | 260.0 | 265.0 | 18.72 | 240.0 | 240.0 | 9.59 | 240.0 | 240.0 | 9.59 |
| c3 | 243 | 252.0 | 257.3 | 3.7 | 279.0 | 289.1 | 14.81 | 265.0 | 265.0 | 9.05 | 261.0 | 261.0 | 7.41 |
| c4 | 219 | 233.0 | 235.7 | 6.39 | 250.0 | 257.9 | 14.16 | 235.0 | 235.0 | 7.31 | 232.0 | 232.0 | 5.94 |
| c5 | 215 | 226.0 | 229.4 | 5.12 | 248.0 | 256.3 | 15.35 | 233.0 | 234.0 | 8.37 | 231.0 | 231.0 | 7.44 |
| d1 | 60 | 62.0 | 63.5 | 3.33 | 83.0 | 98.2 | 38.33 | 65.0 | 65.0 | 8.33 | 65.0 | 65.0 | 8.33 |
| d2 | 66 | 66.0 | 68.3 | 0.0 | 97.0 | 113.7 | 46.97 | 71.0 | 71.0 | 7.58 | 70.0 | 70.0 | 6.06 |
| d3 | 72 | 75.0 | 76.4 | 4.17 | 108.0 | 121.6 | 50.0 | 77.0 | 77.0 | 6.94 | 78.0 | 78.0 | 8.33 |
| d4 | 62 | 62.0 | 63.5 | 0.0 | 90.0 | 101.9 | 45.16 | 65.0 | 65.0 | 4.84 | 64.0 | 64.0 | 3.23 |
| d5 | 61 | 61.0 | 64.0 | 0.0 | 97.0 | 106.4 | 59.02 | 66.0 | 66.5 | 8.2 | 66.0 | 66.0 | 8.2 |
| | | **260.11** | 264.5 | 2.42 | 272.22 | 278.51 | 11.75 | 266.91 | 266.94 | 5.45 | 265.04 | 265.09 | 4.83 |

Table 16: SCP - GWO - 40 ACTIONS

| | | BCL | | | MIR | | | BQSA1 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | Opt. | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| 41 | 429 | 432.0 | 433.5 | 0.7 | 431.0 | 434.4 | 0.47 | 439.0 | 439.0 | 2.33 |
| 42 | 512 | 533.0 | 539.1 | 4.1 | 543.0 | 547.8 | 6.05 | 543.0 | 543.0 | 6.05 |
| 43 | 516 | 527.0 | 532.5 | 2.13 | 541.0 | 546.3 | 4.84 | 534.0 | 534.0 | 3.49 |
| 44 | 494 | 507.0 | 514.3 | 2.63 | 513.0 | 520.5 | 3.85 | 523.0 | 523.0 | 5.87 |
| 45 | 512 | 523.0 | 528.2 | 2.15 | 540.0 | 549.6 | 5.47 | 547.0 | 547.0 | 6.84 |
| 46 | 560 | 570.0 | 579.1 | 1.79 | 579.0 | 583.2 | 3.39 | 584.0 | 584.0 | 4.29 |
| 47 | 430 | 434.0 | 437.8 | 0.93 | 442.0 | 445.7 | 2.79 | 448.0 | 448.0 | 4.19 |
| 48 | 492 | 499.0 | 506.3 | 1.42 | 509.0 | 512.6 | 3.46 | 510.0 | 510.5 | 3.66 |
| 49 | 641 | 656.0 | 671.4 | 2.34 | 685.0 | 690.3 | 6.86 | 694.0 | 694.0 | 8.27 |
| 410 | 514 | 520.0 | 525.7 | 1.17 | 528.0 | 531.2 | 2.72 | 530.0 | 530.0 | 3.11 |
| 51 | 253 | 260.0 | 265.8 | 2.77 | 264.0 | 266.5 | 4.35 | 269.0 | 269.0 | 6.32 |
| 52 | 302 | 322.0 | 328.3 | 6.62 | 326.0 | 333.7 | 7.95 | 331.0 | 331.0 | 9.6 |
| 53 | 226 | 230.0 | 233.3 | 1.77 | 232.0 | 234.2 | 2.65 | 231.0 | 232.5 | 2.21 |
| 54 | 242 | 247.0 | 249.8 | 2.07 | 250.0 | 253.1 | 3.31 | 252.0 | 252.0 | 4.13 |
| 55 | 211 | 212.0 | 213.8 | 0.47 | 216.0 | 217.9 | 2.37 | 219.0 | 219.0 | 3.79 |
| 56 | 213 | 216.0 | 223.8 | 1.41 | 222.0 | 228.2 | 4.23 | 230.0 | 230.0 | 7.98 |
| 57 | 293 | 299.0 | 306.8 | 2.05 | 309.0 | 311.8 | 5.46 | 312.0 | 312.0 | 6.48 |
| 58 | 288 | 290.0 | 296.2 | 0.69 | 297.0 | 300.7 | 3.12 | 300.0 | 300.0 | 4.17 |
| 59 | 279 | 282.0 | 285.0 | 1.08 | 285.0 | 293.0 | 2.15 | 290.0 | 290.0 | 3.94 |
| 510 | 265 | 269.0 | 275.0 | 1.51 | 277.0 | 280.7 | 4.53 | 277.0 | 277.0 | 4.53 |
| 61 | 138 | 140.0 | 145.2 | 1.45 | 144.0 | 148.1 | 4.35 | 147.0 | 147.0 | 6.52 |
| 62 | 146 | 147.0 | 152.3 | 0.68 | 155.0 | 158.9 | 6.16 | 157.0 | 157.0 | 7.53 |
| 63 | 145 | 147.0 | 149.8 | 1.38 | 151.0 | 151.5 | 4.14 | 151.0 | 151.0 | 4.14 |
| 64 | 131 | 131.0 | 134.3 | 0.0 | 134.0 | 135.3 | 2.29 | 132.0 | 132.0 | 0.76 |
| 65 | 161 | 167.0 | 172.3 | 3.73 | 175.0 | 183.7 | 8.7 | 179.0 | 180.0 | 11.18 |
| a1 | 253 | 262.0 | 263.2 | 3.56 | 271.0 | 277.6 | 7.11 | 269.0 | 269.0 | 6.32 |
| a2 | 252 | 263.0 | 268.1 | 4.37 | 275.0 | 279.2 | 9.13 | 273.0 | 273.0 | 8.33 |
| a3 | 232 | 241.0 | 245.3 | 3.88 | 250.0 | 256.1 | 7.76 | 248.0 | 248.0 | 6.9 |
| a4 | 234 | 244.0 | 247.4 | 4.27 | 250.0 | 257.8 | 6.84 | 248.0 | 248.0 | 5.98 |
| a5 | 236 | 244.0 | 246.5 | 3.39 | 255.0 | 259.0 | 8.05 | 249.0 | 249.0 | 5.51 |
| b1 | 69 | 70.0 | 71.1 | 1.45 | 75.0 | 81.5 | 8.7 | 73.0 | 73.0 | 5.8 |
| b2 | 76 | 76.0 | 78.3 | 0.0 | 86.0 | 91.3 | 13.16 | 80.0 | 80.0 | 5.26 |
| b3 | 80 | 81.0 | 82.2 | 1.25 | 89.0 | 98.1 | 11.25 | 84.0 | 84.0 | 5.0 |
| b4 | 79 | 82.0 | 82.9 | 3.8 | 95.0 | 100.0 | 20.25 | 85.0 | 85.0 | 7.59 |
| b5 | 72 | 73.0 | 73.5 | 1.39 | 81.0 | 90.5 | 12.5 | 75.0 | 75.0 | 4.17 |
| c1 | 227 | 239.0 | 248.1 | 5.29 | 263.0 | 272.7 | 15.86 | 252.0 | 252.5 | 11.01 |
| c2 | 219 | 233.0 | 238.1 | 6.39 | 260.0 | 265.0 | 18.72 | 239.0 | 239.0 | 9.13 |
| c3 | 243 | 252.0 | 257.3 | 3.7 | 279.0 | 289.1 | 14.81 | 264.0 | 264.0 | 8.64 |
| c4 | 219 | 233.0 | 235.7 | 6.39 | 250.0 | 257.9 | 14.16 | 238.0 | 238.0 | 8.68 |
| c5 | 215 | 226.0 | 229.4 | 5.12 | 248.0 | 256.3 | 15.35 | 234.0 | 234.0 | 8.84 |
| d1 | 60 | 62.0 | 63.5 | 3.33 | 83.0 | 98.2 | 38.33 | 66.0 | 66.0 | 10.0 |
| d2 | 66 | 66.0 | 68.3 | 0.0 | 97.0 | 113.7 | 46.97 | 71.0 | 71.0 | 7.58 |
| d3 | 72 | 75.0 | 76.4 | 4.17 | 108.0 | 121.6 | 50.0 | 80.0 | 80.0 | 11.11 |
| d4 | 62 | 62.0 | 63.5 | 0.0 | 90.0 | 101.9 | 45.16 | 64.0 | 65.0 | 3.23 |
| d5 | 61 | 61.0 | 64.0 | 0.0 | 97.0 | 106.4 | 59.02 | 66.0 | 66.0 | 8.2 |
| | | **260.11** | 264.5 | 2.42 | 272.22 | 278.51 | 11.75 | 268.6 | 268.7 | 6.19 |

Table 17: SCP - SCA - 85 ACTIONS

| | | BCL | | | MIR | | | QL1 | | | SA1 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | Opt. | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| 41 | 429 | 526.0 | 577.8 | 22.61 | 689.0 | 707.2 | 60.61 | 440.0 | 440.0 | 2.56 | 435.0 | 435.0 | 1.4 |
| 42 | 512 | 600.0 | 720.5 | 17.19 | 1060.0 | 1165.8 | 107.03 | 547.0 | 547.0 | 6.84 | 540.0 | 540.0 | 5.47 |
| 43 | 516 | 589.0 | 741.6 | 14.15 | 1185.0 | 1269.0 | 129.65 | 541.0 | 541.0 | 4.84 | 538.0 | 538.0 | 4.26 |
| 44 | 494 | 580.0 | 709.2 | 17.41 | 986.0 | 1020.8 | 99.6 | 523.0 | 523.0 | 5.87 | 516.0 | 516.0 | 4.45 |
| 45 | 512 | 700.0 | 810.1 | 36.72 | 1126.0 | 1184.3 | 119.92 | 531.0 | 531.0 | 3.71 | 535.0 | 535.0 | 4.49 |
| 46 | 560 | 692.0 | 793.2 | 23.57 | 1421.0 | 1475.5 | 153.75 | 581.0 | 581.0 | 3.75 | 579.0 | 579.0 | 3.39 |
| 47 | 430 | 498.0 | 606.0 | 15.81 | 886.0 | 920.3 | 106.05 | 438.0 | 438.0 | 1.86 | 440.0 | 440.0 | 2.33 |
| 48 | 492 | 577.0 | 719.3 | 17.28 | 1119.0 | 1226.7 | 127.44 | 506.0 | 506.0 | 2.85 | 502.0 | 502.0 | 2.03 |
| 49 | 641 | 779.0 | 918.0 | 21.53 | 1535.0 | 1662.8 | 139.47 | 686.0 | 686.0 | 7.02 | 684.0 | 684.0 | 6.71 |
| 410 | 514 | 576.0 | 722.9 | 12.06 | 1143.0 | 1198.6 | 122.37 | 530.0 | 530.0 | 3.11 | 529.0 | 529.0 | 2.92 |
| 51 | 253 | 302.0 | 356.2 | 19.37 | 568.0 | 612.1 | 124.51 | 268.0 | 268.0 | 5.93 | 266.0 | 266.0 | 5.14 |
| 52 | 302 | 356.0 | 420.9 | 17.88 | 831.0 | 889.9 | 175.17 | 328.0 | 328.0 | 8.61 | 328.0 | 328.0 | 8.61 |
| 53 | 226 | 265.0 | 308.4 | 17.26 | 498.0 | 537.8 | 120.35 | 232.0 | 232.0 | 2.65 | 233.0 | 233.0 | 3.1 |
| 54 | 242 | 307.0 | 335.0 | 26.86 | 553.0 | 577.6 | 128.51 | 252.0 | 252.0 | 4.13 | 251.0 | 251.0 | 3.72 |
| 55 | 211 | 257.0 | 307.9 | 21.8 | 411.0 | 431.3 | 94.79 | 217.0 | 217.0 | 2.84 | 219.0 | 219.0 | 3.79 |
| 56 | 213 | 263.0 | 317.5 | 23.47 | 472.0 | 534.9 | 121.6 | 229.0 | 229.0 | 7.51 | 223.0 | 223.0 | 4.69 |
| 57 | 293 | 345.0 | 422.9 | 17.75 | 643.0 | 704.9 | 119.45 | 309.0 | 309.0 | 5.46 | 310.0 | 310.0 | 5.8 |
| 58 | 288 | 359.0 | 390.6 | 24.65 | 707.0 | 729.2 | 145.49 | 296.0 | 296.0 | 2.78 | 297.0 | 297.0 | 3.12 |
| 59 | 279 | 372.0 | 414.8 | 33.33 | 719.0 | 739.6 | 157.71 | 289.0 | 289.0 | 3.58 | 284.0 | 284.0 | 1.79 |
| 510 | 265 | 317.0 | 390.6 | 19.62 | 621.0 | 656.5 | 134.34 | 279.0 | 279.0 | 5.28 | 277.0 | 277.0 | 4.53 |
| 61 | 138 | 171.0 | 235.4 | 23.91 | 755.0 | 830.7 | 447.1 | 146.0 | 146.0 | 5.8 | 146.0 | 146.0 | 5.8 |
| 62 | 146 | 188.0 | 244.2 | 28.77 | 1092.0 | 1155.8 | 647.95 | 155.0 | 155.0 | 6.16 | 154.0 | 154.0 | 5.48 |
| 63 | 145 | 195.0 | 293.4 | 34.48 | 1091.0 | 1132.9 | 652.41 | 150.0 | 150.0 | 3.45 | 149.0 | 149.0 | 2.76 |
| 64 | 131 | 166.0 | 242.6 | 26.72 | 623.0 | 697.7 | 375.57 | 132.0 | 132.0 | 0.76 | 134.0 | 134.0 | 2.29 |
| 65 | 161 | 209.0 | 244.1 | 29.81 | 1000.0 | 1123.1 | 521.12 | 171.0 | 171.0 | 6.21 | 176.0 | 176.0 | 9.32 |
| a1 | 253 | 286.0 | 308.1 | 13.04 | 1320.0 | 1343.3 | 421.74 | 267.0 | 267.0 | 5.53 | 266.0 | 266.0 | 5.14 |
| a2 | 252 | 290.0 | 335.6 | 15.08 | 1175.0 | 1210.1 | 366.27 | 271.0 | 271.0 | 7.54 | 270.0 | 270.0 | 7.14 |
| a3 | 232 | 265.0 | 289.9 | 14.22 | 1067.0 | 1170.2 | 359.91 | 242.0 | 242.0 | 4.31 | 243.0 | 243.0 | 4.74 |
| a4 | 234 | 274.0 | 326.3 | 17.09 | 1095.0 | 1126.9 | 367.95 | 247.0 | 247.0 | 5.56 | 247.0 | 247.0 | 5.56 |
| a5 | 236 | 276.0 | 319.3 | 16.95 | 1100.0 | 1165.1 | 366.1 | 249.0 | 249.0 | 5.51 | 246.0 | 246.0 | 4.24 |
| b1 | 69 | 79.0 | 103.1 | 14.49 | 1341.0 | 1413.9 | 1843.48 | 71.0 | 71.0 | 2.9 | 71.0 | 71.0 | 2.9 |
| b2 | 76 | 88.0 | 110.2 | 15.79 | 1364.0 | 1402.3 | 1694.74 | 78.0 | 78.0 | 2.63 | 78.0 | 78.0 | 2.63 |
| b3 | 80 | 87.0 | 106.9 | 8.75 | 1670.0 | 1830.4 | 1987.5 | 82.0 | 82.0 | 2.5 | 82.0 | 82.0 | 2.5 |
| b4 | 79 | 88.0 | 108.2 | 11.39 | 88.0 | 1461.8 | 11.39 | 83.0 | 83.0 | 5.06 | 82.0 | 82.0 | 3.8 |
| b5 | 72 | 80.0 | 118.6 | 11.11 | 1404.0 | 1467.7 | 1850.0 | 74.0 | 74.0 | 2.78 | 74.0 | 74.0 | 2.78 |
| c1 | 227 | 291.0 | 320.0 | 28.19 | 1508.0 | 1559.6 | 564.32 | 244.0 | 245.0 | 7.49 | 244.0 | 244.0 | 7.49 |
| c2 | 219 | 271.0 | 287.0 | 23.74 | 1716.0 | 1770.6 | 683.56 | 239.0 | 239.0 | 9.13 | 238.0 | 238.0 | 8.68 |
| c3 | 243 | 280.0 | 334.0 | 15.23 | 268.0 | 1880.5 | 10.29 | 259.0 | 259.0 | 6.58 | 257.0 | 257.0 | 5.76 |
| c4 | 219 | 243.0 | 276.6 | 10.96 | 1657.0 | 1737.7 | 656.62 | 232.0 | 232.0 | 5.94 | 232.0 | 232.0 | 5.94 |
| c5 | 215 | 268.0 | 287.8 | 24.65 | 1537.0 | 1649.7 | 614.88 | 230.0 | 230.0 | 6.98 | 229.0 | 229.0 | 6.51 |
| d1 | 60 | 81.0 | 102.5 | 35.0 | 1979.0 | 2046.9 | 3198.33 | 65.0 | 65.0 | 8.33 | 64.0 | 64.0 | 6.67 |
| d2 | 66 | 77.0 | 106.1 | 16.67 | 2201.0 | 2328.6 | 3234.85 | 69.0 | 69.0 | 4.55 | 69.0 | 69.0 | 4.55 |
| d3 | 72 | 92.0 | 110.7 | 27.78 | 2413.0 | 2523.6 | 3251.39 | 76.0 | 76.0 | 5.56 | 76.0 | 76.0 | 5.56 |
| d4 | 62 | 68.0 | 91.7 | 9.68 | 1929.0 | 2091.9 | 3011.29 | 63.0 | 63.0 | 1.61 | 63.0 | 63.0 | 1.61 |
| d5 | 61 | 77.0 | 97.7 | 26.23 | 1979.0 | 2086.6 | 3144.26 | 65.0 | 65.0 | 6.56 | 64.0 | 64.0 | 4.92 |
| | | 305.56 | 364.08 | 20.45 | 1123.22 | 1254.5 | 728.24 | 266.27 | 266.29 | 4.9 | **265.3**3 | 265.33 | 4.59 |

Table 18: SCP - SCA - 40 ACTIONS

| | | BCL | | | MIR | | | BQSA1 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | Opt. | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| 41 | 429 | 526.0 | 577.8 | 22.61 | 689.0 | 707.2 | 60.61 | 439.0 | 440.0 | 2.33 |
| 42 | 512 | 600.0 | 720.5 | 17.19 | 1060.0 | 1165.8 | 107.03 | 551.0 | 551.0 | 7.62 |
| 43 | 516 | 589.0 | 741.6 | 14.15 | 1185.0 | 1269.0 | 129.65 | 546.0 | 546.0 | 5.81 |
| 44 | 494 | 580.0 | 709.2 | 17.41 | 986.0 | 1020.8 | 99.6 | 530.0 | 530.0 | 7.29 |
| 45 | 512 | 700.0 | 810.1 | 36.72 | 1126.0 | 1184.3 | 119.92 | 546.0 | 546.0 | 6.64 |
| 46 | 560 | 692.0 | 793.2 | 23.57 | 1421.0 | 1475.5 | 153.75 | 585.0 | 585.0 | 4.46 |
| 47 | 430 | 498.0 | 606.0 | 15.81 | 886.0 | 920.3 | 106.05 | 448.0 | 448.0 | 4.19 |
| 48 | 492 | 577.0 | 719.3 | 17.28 | 1119.0 | 1226.7 | 127.44 | 515.0 | 515.0 | 4.67 |
| 49 | 641 | 779.0 | 918.0 | 21.53 | 1535.0 | 1662.8 | 139.47 | 691.0 | 691.0 | 7.8 |
| 410 | 514 | 576.0 | 722.9 | 12.06 | 1143.0 | 1198.6 | 122.37 | 529.0 | 529.0 | 2.92 |
| 51 | 253 | 302.0 | 356.2 | 19.37 | 568.0 | 612.1 | 124.51 | 269.0 | 269.0 | 6.32 |
| 52 | 302 | 356.0 | 420.9 | 17.88 | 831.0 | 889.9 | 175.17 | 331.0 | 331.0 | 9.6 |
| 53 | 226 | 265.0 | 308.4 | 17.26 | 498.0 | 537.8 | 120.35 | 234.0 | 234.0 | 3.54 |
| 54 | 242 | 307.0 | 335.0 | 26.86 | 553.0 | 577.6 | 128.51 | 252.0 | 252.0 | 4.13 |
| 55 | 211 | 257.0 | 307.9 | 21.8 | 411.0 | 431.3 | 94.79 | 219.0 | 219.0 | 3.79 |
| 56 | 213 | 263.0 | 317.5 | 23.47 | 472.0 | 534.9 | 121.6 | 232.0 | 232.0 | 8.92 |
| 57 | 293 | 345.0 | 422.9 | 17.75 | 643.0 | 704.9 | 119.45 | 315.0 | 315.0 | 7.51 |
| 58 | 288 | 359.0 | 390.6 | 24.65 | 707.0 | 729.2 | 145.49 | 300.0 | 300.0 | 4.17 |
| 59 | 279 | 372.0 | 414.8 | 33.33 | 719.0 | 739.6 | 157.71 | 290.0 | 290.0 | 3.94 |
| 510 | 265 | 317.0 | 390.6 | 19.62 | 621.0 | 656.5 | 134.34 | 280.0 | 280.0 | 5.66 |
| 61 | 138 | 171.0 | 235.4 | 23.91 | 755.0 | 830.7 | 447.1 | 144.0 | 144.0 | 4.35 |
| 62 | 146 | 188.0 | 244.2 | 28.77 | 1092.0 | 1155.8 | 647.95 | 158.0 | 158.0 | 8.22 |
| 63 | 145 | 195.0 | 293.4 | 34.48 | 1091.0 | 1132.9 | 652.41 | 149.0 | 150.0 | 2.76 |
| 64 | 131 | 166.0 | 242.6 | 26.72 | 623.0 | 697.7 | 375.57 | 135.0 | 135.0 | 3.05 |
| 65 | 161 | 209.0 | 244.1 | 29.81 | 1000.0 | 1123.1 | 521.12 | 183.0 | 183.0 | 13.66 |
| a1 | 253 | 286.0 | 308.1 | 13.04 | 1320.0 | 1343.3 | 421.74 | 265.0 | 265.0 | 4.74 |
| a2 | 252 | 290.0 | 335.6 | 15.08 | 1175.0 | 1210.1 | 366.27 | 273.0 | 273.0 | 8.33 |
| a3 | 232 | 265.0 | 289.9 | 14.22 | 1067.0 | 1170.2 | 359.91 | 245.0 | 245.0 | 5.6 |
| a4 | 234 | 274.0 | 326.3 | 17.09 | 1095.0 | 1126.9 | 367.95 | 254.0 | 254.0 | 8.55 |
| a5 | 236 | 276.0 | 319.3 | 16.95 | 1100.0 | 1165.1 | 366.1 | 252.0 | 252.0 | 6.78 |
| b1 | 69 | 79.0 | 103.1 | 14.49 | 1341.0 | 1413.9 | 1843.48 | 72.0 | 72.0 | 4.35 |
| b2 | 76 | 88.0 | 110.2 | 15.79 | 1364.0 | 1402.3 | 1694.74 | 80.0 | 80.0 | 5.26 |
| b3 | 80 | 87.0 | 106.9 | 8.75 | 1670.0 | 1830.4 | 1987.5 | 82.0 | 82.0 | 2.5 |
| b4 | 79 | 88.0 | 108.2 | 11.39 | 88.0 | 1461.8 | 11.39 | 84.0 | 84.0 | 6.33 |
| b5 | 72 | 80.0 | 118.6 | 11.11 | 1404.0 | 1467.7 | 1850.0 | 74.0 | 74.0 | 2.78 |
| c1 | 227 | 291.0 | 320.0 | 28.19 | 1508.0 | 1559.6 | 564.32 | 245.0 | 245.0 | 7.93 |
| c2 | 219 | 271.0 | 287.0 | 23.74 | 1716.0 | 1770.6 | 683.56 | 239.0 | 239.5 | 9.13 |
| c3 | 243 | 280.0 | 334.0 | 15.23 | 268.0 | 1880.5 | 10.29 | 261.0 | 261.0 | 7.41 |
| c4 | 219 | 243.0 | 276.6 | 10.96 | 1657.0 | 1737.7 | 656.62 | 236.0 | 236.0 | 7.76 |
| c5 | 215 | 268.0 | 287.8 | 24.65 | 1537.0 | 1649.7 | 614.88 | 232.0 | 232.0 | 7.91 |
| d1 | 60 | 81.0 | 102.5 | 35.0 | 1979.0 | 2046.9 | 3198.33 | 64.0 | 64.0 | 6.67 |
| d2 | 66 | 77.0 | 106.1 | 16.67 | 2201.0 | 2328.6 | 3234.85 | 69.0 | 69.0 | 4.55 |
| d3 | 72 | 92.0 | 110.7 | 27.78 | 2413.0 | 2523.6 | 3251.39 | 78.0 | 78.0 | 8.33 |
| d4 | 62 | 68.0 | 91.7 | 9.68 | 1929.0 | 2091.9 | 3011.29 | 64.0 | 64.0 | 3.23 |
| d5 | 61 | 77.0 | 97.7 | 26.23 | 1979.0 | 2086.6 | 3144.26 | 66.0 | 66.0 | 8.2 |
| | | 305.56 | 364.08 | 20.45 | 1123.22 | 1254.5 | 728.24 | **269.02** | 269.08 | 5.99 |

Table 19: 0-1KP - WOA - Q-Learning

| | | QL1 | | | QL2 | | | QL3 | | | QL4 | | | QL5 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | Opt. | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| KP1 | 295 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 |
| KP2 | 1024 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 |
| KP3 | 35 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 |
| KP4 | 23 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 |
| KP5 | 481.0694 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 |
| KP6 | 52 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 |
| KP7 | 107 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 |
| KP8 | 9767 | 9762.0 | 9765.16 | 0.05 | 9761.0 | 9764.84 | 0.06 | 9761.0 | 9764.68 | 0.06 | 9762.0 | 9765.1 | 0.05 | 9761.0 | 9765.52 | 0.06 |
| KP9 | 130 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 |
| KP10 | 1025 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 |
| KP1 100 | 9147 | 5412.0 | 6174.74 | 40.83 | 5416.0 | 7031.06 | 40.79 | 5059.0 | 6574.48 | 44.69 | 5012.0 | 6814.52 | 45.21 | 5063.0 | 6679.1 | 44.65 |
| KP1 200 | 11238 | 5402.0 | 6910.97 | 51.93 | 5491.0 | 7580.1 | 51.14 | 5861.0 | 8612.13 | 47.85 | 4661.0 | 8260.16 | 58.52 | 5790.0 | 8199.1 | 48.48 |
| KP1 500 | 28857 | 8651.0 | 9749.9 | 70.02 | 8353.0 | 12546.35 | 71.05 | 7727.0 | 11999.9 | 73.22 | 7886.0 | 13013.03 | 72.67 | 8224.0 | 11591.06 | 71.5 |
| KP3 100 | 1514 | 1277.0 | 1335.58 | 15.65 | 1258.0 | 1345.58 | 16.91 | 1249.0 | 1337.06 | 17.5 | 1229.0 | 1347.1 | 18.82 | 1245.0 | 1326.97 | 17.77 |
| KP3 200 | 1634 | 1260.0 | 1341.81 | 22.89 | 1283.0 | 1411.58 | 21.48 | 1268.0 | 1417.9 | 22.4 | 1246.0 | 1420.9 | 23.75 | 1262.0 | 1414.68 | 22.77 |
| KP3 500 | 4566 | 2956.0 | 3128.55 | 35.26 | 2989.0 | 3564.68 | 34.54 | 2958.0 | 3581.7 | 35.22 | 3014.0 | 3595.7 | 33.99 | 2937.0 | 3436.0 | 35.68 |
| KP3 100 | 2397 | 1896.0 | 2070.9 | 20.9 | 1892.0 | 2215.68 | 21.07 | 1869.0 | 2234.23 | 22.03 | 1895.0 | 2287.19 | 20.94 | 1975.0 | 2265.52 | 17.61 |
| KP3 200 | 2697 | 1993.0 | 2101.61 | 26.1 | 1897.0 | 2328.68 | 29.66 | 1794.0 | 2303.61 | 33.48 | 1985.0 | 2446.94 | 26.4 | 1896.0 | 2401.74 | 29.7 |
| KP3 500 | 7117 | 4017.0 | 4245.16 | 43.56 | 3916.0 | 5201.58 | 44.98 | 3815.0 | 4607.45 | 46.4 | 3916.0 | 4830.52 | 44.98 | 3917.0 | 4910.83 | 44.96 |
| | | **2410.42** | 2631.39 | 17.22 | 2390.95 | 2955.91 | 17.46 | 2343.85 | 2926.59 | 18.04 | 2304.11 | 2997.54 | 18.18 | 2381.16 | 2903.29 | 17.54 |

Table 20: 0-1KP - WOA - SARSA

| | | SA1 | | | SA2 | | | SA3 | | | SA4 | | | SA5 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | Opt. | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| KP1 | 295 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 |
| KP2 | 1024 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 |
| KP3 | 35 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 |
| KP4 | 23 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 |
| KP5 | 481.0694 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 |
| KP6 | 52 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 |
| KP7 | 107 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 |
| KP8 | 9767 | 9762.0 | 9765.23 | 0.05 | 9761.0 | 9764.9 | 0.06 | 9761.0 | 9765.55 | 0.06 | 9762.0 | 9765.32 | 0.05 | 9763.0 | 9765.79 | 0.04 |
| KP9 | 130 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 |
| KP10 | 1025 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 |
| KP1 100 | 9147 | 5142.0 | 6121.55 | 43.78 | 5177.0 | 5977.94 | 43.4 | 5092.0 | 5864.55 | 44.33 | 5205.0 | 5990.81 | 43.1 | 5478.0 | 5981.74 | 40.11 |
| KP1 200 | 11238 | 5220.0 | 6413.52 | 53.55 | 5559.0 | 6241.13 | 50.53 | 5208.0 | 6189.61 | 53.66 | 5270.0 | 6080.16 | 53.11 | 5465.0 | 6293.58 | 51.37 |
| KP1 500 | 28857 | 7896.0 | 9086.35 | 72.64 | 7297.0 | 8718.61 | 74.71 | 6878.0 | 8604.9 | 76.17 | 7715.0 | 8899.55 | 73.26 | 7132.0 | 8442.39 | 75.29 |
| KP2 100 | 1514 | 1240.0 | 1326.35 | 18.1 | 1247.0 | 1304.35 | 17.64 | 1234.0 | 1302.35 | 18.49 | 1247.0 | 1298.26 | 17.64 | 1237.0 | 1295.26 | 18.3 |
| KP2 200 | 1634 | 1259.0 | 1346.87 | 22.95 | 1265.0 | 1314.19 | 22.58 | 1266.0 | 1325.52 | 22.52 | 1239.0 | 1309.68 | 24.17 | 1270.0 | 1313.55 | 22.28 |
| KP2 500 | 4566 | 3011.0 | 3134.39 | 34.06 | 2974.0 | 3069.19 | 34.87 | 2947.0 | 3036.27 | 35.46 | 2953.0 | 3053.46 | 35.33 | 2965.0 | 3072.32 | 35.06 |
| KP3 100 | 2397 | 1879.0 | 2034.81 | 21.61 | 1887.0 | 2037.26 | 21.28 | 1895.0 | 2002.42 | 20.94 | 1797.0 | 2018.6 | 25.03 | 1888.0 | 2005.71 | 21.23 |
| KP3 200 | 2697 | 1995.0 | 2106.74 | 26.03 | 1896.0 | 2014.1 | 29.7 | 1895.0 | 2058.84 | 29.74 | 1796.0 | 2044.52 | 33.41 | 1897.0 | 2050.06 | 29.66 |
| KP3 500 | 7117 | 4017.0 | 4244.81 | 43.56 | 4017.0 | 4176.97 | 43.56 | 4013.0 | 4112.52 | 43.61 | 4015.0 | 4174.14 | 43.59 | 4015.0 | 4174.35 | 43.59 |
| | | **2347.0** | 2565.93 | 17.7 | 2329.06 | 2515.3 | 17.81 | 2282.16 | 2496.56 | 18.16 | 2324.79 | 2516.14 | 18.35 | 2330.64 | 2503.52 | 17.73 |

Table 21: 0-1KP - GWO - Q-Learning

| | | QL1 | | | QL2 | | | QL3 | | | QL4 | | | QL5 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | Opt. | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| KP1 | 295 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 |
| KP2 | 1024 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 |
| KP3 | 35 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 |
| KP4 | 23 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 |
| KP5 | 481.0694 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 |
| KP6 | 52 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 |
| KP7 | 107 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 |
| KP8 | 9767 | 9760.0 | 9763.84 | 0.07 | 9762.0 | 9764.16 | 0.05 | 9760.0 | 9763.55 | 0.07 | 9759.0 | 9762.84 | 0.08 | 9759.0 | 9763.26 | 0.08 |
| KP9 | 130 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 |
| KP10 | 1025 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 |
| KP1 100 | 9147 | 6347.0 | 6947.19 | 30.61 | 5824.0 | 6547.26 | 36.33 | 5733.0 | 6508.65 | 37.32 | 5814.0 | 6622.1 | 36.44 | 5829.0 | 6551.06 | 36.27 |
| KP1 200 | 11238 | 6480.0 | 7254.06 | 42.34 | 6227.0 | 6961.65 | 44.59 | 6024.0 | 6949.94 | 46.4 | 6283.0 | 6929.77 | 44.09 | 6061.0 | 6867.58 | 46.07 |
| KP1 500 | 28857 | 8800.0 | 10045.16 | 69.5 | 8358.0 | 9773.97 | 71.04 | 8905.0 | 10083.84 | 69.14 | 8786.0 | 9920.03 | 69.55 | 8646.0 | 9807.03 | 70.04 |
| KP2 100 | 1514 | 1314.0 | 1369.84 | 13.21 | 1296.0 | 1353.71 | 14.4 | 1288.0 | 1346.58 | 14.93 | 1281.0 | 1346.48 | 15.39 | 1293.0 | 1350.65 | 14.6 |
| KP2 200 | 1634 | 1349.0 | 1402.94 | 17.44 | 1301.0 | 1377.61 | 20.38 | 1290.0 | 1367.19 | 21.05 | 1314.0 | 1360.23 | 19.58 | 1301.0 | 1374.16 | 20.38 |
| KP2 500 | 4566 | 3136.0 | 3217.06 | 31.32 | 3042.0 | 3124.84 | 33.38 | 3008.0 | 3160.48 | 34.12 | 3025.0 | 3144.13 | 33.75 | 3078.0 | 3136.9 | 32.59 |
| KP3 100 | 2397 | 1996.0 | 2124.94 | 16.73 | 1981.0 | 2080.45 | 17.36 | 1990.0 | 2050.71 | 16.98 | 1990.0 | 2057.87 | 16.98 | 1995.0 | 2079.13 | 16.77 |
| KP3 200 | 2697 | 2087.0 | 2158.1 | 22.62 | 1987.0 | 2129.45 | 26.33 | 1995.0 | 2112.87 | 26.03 | 1997.0 | 2116.65 | 25.95 | 1996.0 | 2135.35 | 25.99 |
| KP3 500 | 7117 | 4213.0 | 4335.16 | 40.8 | 4113.0 | 4306.1 | 42.21 | 4117.0 | 4315.35 | 42.15 | 4117.0 | 4344.68 | 42.15 | 4116.0 | 4293.26 | 42.17 |
| | | **2560.74** | 2725.81 | 14.98 | 2477.0 | 2662.7 | 16.11 | 2488.53 | 2675.33 | 16.22 | 2502.0 | 2672.47 | 16.0 | 2486.64 | 2659.5 | 16.05 |

Table 22: 0-1KP - GWO - SARSA

| Inst. | Opt. | SA1 | | | SA2 | | | SA3 | | | SA4 | | | SA5 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| KP1 | 295 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 |
| KP2 | 1024 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 |
| KP3 | 35 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 |
| KP4 | 23 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 |
| KP5 | 481.0694 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 |
| KP6 | 52 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 |
| KP7 | 107 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 |
| KP8 | 9767 | 9761.0 | 9764.39 | 0.06 | 9761.0 | 9764.32 | 0.06 | 9760.0 | 9764.97 | 0.07 | 9761.0 | 9764.52 | 0.06 | 9761.0 | 9763.87 | 0.06 |
| KP9 | 130 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 |
| KP10 | 1025 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 |
| KP1 100 | 9147 | 5890.0 | 6816.55 | 35.61 | 6185.0 | 6885.0 | 32.38 | 6011.0 | 6702.81 | 34.28 | 6086.0 | 6737.68 | 33.46 | 6082.0 | 6645.81 | 33.51 |
| KP1 200 | 11238 | 6597.0 | 7069.19 | 41.3 | 6348.0 | 7149.1 | 43.51 | 6593.0 | 7163.77 | 41.33 | 6464.0 | 7288.0 | 42.48 | 6349.0 | 7207.68 | 43.5 |
| KP1 500 | 28857 | 9362.0 | 10158.1 | 67.56 | 9337.0 | 10113.26 | 67.64 | 9166.0 | 10135.55 | 68.24 | 8784.0 | 10215.55 | 69.56 | 8994.0 | 10071.23 | 68.83 |
| KP2 100 | 1514 | 1299.0 | 1369.23 | 14.2 | 1309.0 | 1371.55 | 13.54 | 1311.0 | 1364.81 | 13.41 | 1313.0 | 1370.77 | 13.28 | 1317.0 | 1364.84 | 13.01 |
| KP2 200 | 1634 | 1329.0 | 1383.65 | 18.67 | 1341.0 | 1387.97 | 17.93 | 1329.0 | 1387.55 | 18.67 | 1337.0 | 1390.58 | 18.18 | 1339.0 | 1383.16 | 18.05 |
| KP2 500 | 4566 | 3126.0 | 3207.55 | 31.54 | 3121.0 | 3191.68 | 31.65 | 3124.0 | 3179.77 | 31.58 | 3111.0 | 3204.58 | 31.87 | 3140.0 | 3203.65 | 31.23 |
| KP3 100 | 2397 | 1997.0 | 2102.29 | 16.69 | 1996.0 | 2096.1 | 16.73 | 1997.0 | 2095.81 | 16.69 | 1996.0 | 2085.03 | 16.73 | 1996.0 | 2100.45 | 16.73 |
| KP3 200 | 2697 | 2089.0 | 2165.52 | 22.54 | 2092.0 | 2126.68 | 22.43 | 2080.0 | 2147.13 | 22.88 | 1996.0 | 2137.65 | 25.99 | 1997.0 | 2137.48 | 25.95 |
| KP3 500 | 7117 | 4215.0 | 4370.58 | 40.78 | 4117.0 | 4360.39 | 42.15 | 4217.0 | 4341.71 | 40.75 | 4216.0 | 4350.97 | 40.76 | 4216.0 | 4376.23 | 40.76 |
| | | **2570.37** | 2714.69 | 15.21 | 2567.32 | 2716.74 | 15.16 | 2566.32 | 2708.21 | 15.15 | 2538.74 | 2721.97 | 15.39 | 2545.42 | 2706.66 | 15.35 |

Table 23: 0-1KP - SCA - QL

| Inst. | Opt. | QL1 | | | QL2 | | | QL3 | | | QL4 | | | QL5 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| KP1 | 295 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 |
| KP2 | 1024 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 |
| KP3 | 35 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 |
| KP4 | 23 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 |
| KP5 | 481.0694 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 |
| KP6 | 52 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 |
| KP7 | 107 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 |
| KP8 | 9767 | 9758.0 | 9762.06 | 0.09 | 9759.0 | 9763.06 | 0.08 | 9759.0 | 9762.87 | 0.08 | 9758.0 | 9762.19 | 0.09 | 9759.0 | 9762.87 | 0.08 |
| KP9 | 130 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 |
| KP10 | 1025 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 |
| KP1 100 | 9147 | 5330.0 | 6174.55 | 41.73 | 5173.0 | 6138.61 | 43.45 | 5205.0 | 6652.0 | 43.1 | 4787.0 | 6307.84 | 47.67 | 5179.0 | 6529.35 | 43.38 |
| KP1 200 | 11238 | 5585.0 | 6644.39 | 50.3 | 5598.0 | 6950.35 | 50.19 | 5573.0 | 7194.94 | 50.41 | 5794.0 | 7016.19 | 48.44 | 5370.0 | 6661.84 | 52.22 |
| KP1 500 | 28857 | 8024.0 | 9218.26 | 72.19 | 7736.0 | 10809.55 | 73.19 | 7541.0 | 11591.32 | 73.87 | 7616.0 | 10568.84 | 73.61 | 7898.0 | 10635.29 | 72.63 |
| KP2 100 | 1514 | 1260.0 | 1322.13 | 16.78 | 1249.0 | 1311.48 | 17.5 | 1230.0 | 1317.87 | 18.76 | 1244.0 | 1320.19 | 17.83 | 1255.0 | 1328.29 | 17.11 |
| KP2 200 | 1634 | 1257.0 | 1335.35 | 23.07 | 1258.0 | 1334.58 | 23.01 | 1244.0 | 1357.94 | 23.87 | 1239.0 | 1344.29 | 24.17 | 1208.0 | 1333.32 | 26.07 |
| KP2 500 | 4566 | 3006.0 | 3111.71 | 34.17 | 2987.0 | 3113.03 | 34.58 | 2954.0 | 3179.23 | 35.3 | 2984.0 | 3210.74 | 34.65 | 2987.0 | 3210.71 | 34.58 |
| KP3 100 | 2397 | 1894.0 | 2027.9 | 20.98 | 1891.0 | 2149.97 | 21.11 | 1966.0 | 2120.55 | 17.98 | 1897.0 | 2135.29 | 20.86 | 1989.0 | 2116.58 | 17.02 |
| KP3 200 | 2697 | 1994.0 | 2090.77 | 26.07 | 1896.0 | 2199.1 | 29.7 | 1990.0 | 2178.81 | 26.21 | 1962.0 | 2131.39 | 27.25 | 1797.0 | 2207.13 | 33.37 |
| KP3 500 | 7117 | 4017.0 | 4200.16 | 43.56 | 3817.0 | 4646.52 | 46.37 | 3916.0 | 4486.52 | 44.98 | 3917.0 | 4484.48 | 44.96 | 3917.0 | 4482.23 | 44.96 |
| | | **2384.06** | 2582.07 | 17.31 | 2344.0 | 2715.17 | 17.85 | 2344.74 | 2790.22 | 17.61 | 2335.27 | 2708.08 | 17.87 | 2343.74 | 2707.35 | 17.97 |

Table 24: 0-1KP - SCA - SARSA

| Inst. | Opt. | SA1 | | | SA2 | | | SA3 | | | SA4 | | | SA5 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD | Best | Avg | RPD |
| KP1 | 295 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 | 295.0 | 295.0 | 0.0 |
| KP2 | 1024 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 | 1024.0 | 1024.0 | 0.0 |
| KP3 | 35 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 | 35.0 | 35.0 | 0.0 |
| KP4 | 23 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 | 23.0 | 23.0 | 0.0 |
| KP5 | 481.0694 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 | 481.07 | 481.07 | 0.0 |
| KP6 | 52 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 | 52.0 | 52.0 | 0.0 |
| KP7 | 107 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 | 107.0 | 107.0 | 0.0 |
| KP8 | 9767 | 9759.0 | 9762.86 | 0.08 | 9758.0 | 9761.67 | 0.09 | 9759.0 | 9761.97 | 0.08 | 9756.0 | 9761.74 | 0.11 | 9757.0 | 9762.06 | 0.1 |
| KP9 | 130 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 | 130.0 | 130.0 | 0.0 |
| KP10 | 1025 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 | 1025.0 | 1025.0 | 0.0 |
| KP1 100 | 9147 | 5565.0 | 6136.16 | 39.16 | 5225.0 | 6033.94 | 42.88 | 5282.0 | 6080.81 | 42.25 | 5178.0 | 5874.16 | 43.39 | 5313.0 | 6030.87 | 41.92 |
| KP1 200 | 11238 | 5602.0 | 6322.77 | 50.15 | 5232.0 | 6239.94 | 53.44 | 5492.0 | 6288.52 | 51.13 | 5701.0 | 6677.0 | 49.27 | 5342.0 | 6281.13 | 52.46 |
| KP1 500 | 28857 | 8428.0 | 9456.52 | 70.79 | 7478.0 | 8854.65 | 74.09 | 7158.0 | 8683.52 | 75.19 | 7557.0 | 9321.52 | 73.81 | 7455.0 | 9040.1 | 74.17 |
| KP2 100 | 1514 | 1242.0 | 1316.16 | 17.97 | 1248.0 | 1302.87 | 17.57 | 1238.0 | 1315.13 | 18.23 | 1236.0 | 1298.32 | 18.36 | 1247.0 | 1316.84 | 17.64 |
| KP2 200 | 1634 | 1258.0 | 1346.48 | 23.01 | 1259.0 | 1327.45 | 22.95 | 1251.0 | 1348.74 | 23.44 | 1252.0 | 1322.52 | 23.38 | 1239.0 | 1309.42 | 24.17 |
| KP2 500 | 4566 | 2997.0 | 3120.45 | 34.36 | 2979.0 | 3106.1 | 34.76 | 2934.0 | 3082.77 | 35.74 | 2960.0 | 3094.77 | 35.17 | 2968.0 | 3065.84 | 35.0 |
| KP3 100 | 2397 | 1896.0 | 2058.81 | 20.9 | 1893.0 | 2032.0 | 21.03 | 1896.0 | 1998.42 | 20.9 | 1896.0 | 2006.29 | 20.9 | 1897.0 | 2050.19 | 20.86 |
| KP3 200 | 2697 | 1989.0 | 2078.61 | 26.25 | 1991.0 | 2041.71 | 26.18 | 1897.0 | 2102.23 | 29.66 | 1896.0 | 2058.77 | 29.7 | 1894.0 | 2076.84 | 29.77 |
| KP3 500 | 7117 | 4015.0 | 4202.3 | 43.59 | 3917.0 | 4181.71 | 44.96 | 3917.0 | 4275.23 | 44.96 | 4016.0 | 4214.39 | 43.57 | 3917.0 | 4170.9 | 44.96 |
| | | **2417.0** | 2577.54 | 17.17 | 2323.79 | 2529.16 | 17.79 | 2315.58 | 2532.07 | 17.98 | 2348.42 | 2568.5 | 17.77 | 2326.37 | 2540.86 | 17.95 |

## 4.3 Statistical Test Results

If the p-value obtained, as explained in section 4.1, is a value less than and equal to 0.05, which means that the difference between the techniques is statistically significant, making the comparison of their means valid. The results obtained are shown in Tables (25-33), generating a matrix of the means obtained. The tables in their first row and first column present the versions of the MH to compare, where the reading per row and the obtaining of a p-value less than 0.05 means that the version in the row has obtained a better performance over the version located in the column, meaning that the difference between the results is statistically significant. Any value that is not less than or equal to 0.05 has been replaced by ">0.05" in order to facilitate the reading of the table.

Table 25: Statistical test of Metheuristics WOA and 85 Actions solving SCP

| Inst. | BCL | MIR | QL1 | SA1 |
|---|---|---|---|---|
| BCL | - | 0.0 | >0.05 | >0.05 |
| MIR | >0.05 | - | >0.05 | >0.05 |
| QL1 | 0.0 | 0.0 | - | >0.05 |
| SA1 | 0.0 | 0.0 | >0.05 | - |

Table 26: Statistical test of Metheuristics WOA and 40 Actions solving SCP

| Inst. | BCL | MIR | BQSA1 |
|---|---|---|---|
| BCL | - | 0.0 | >0.05 |
| MIR | >0.05 | - | >0.05 |
| BQSA1 | 0.0 | 0.0 | - |

Table 27: Statistical test of Metheuristics GWO and 85 Actions solving SCP

| Inst. | BCL | MIR | QL1 | SA1 |
|---|---|---|---|---|
| BCL | - | 0.03 | >0.05 | >0.05 |
| MIR | >0.05 | - | >0.05 | >0.05 |
| QL1 | >0.05 | >0.05 | - | >0.05 |
| SA1 | >0.05 | 0.02 | >0.05 | - |

Table 28: Statistical test of Metheuristics GWO and 40 Actions solving SCP

| Inst. | BCL | MIR | BQSA1 |
|---|---|---|---|
| BCL | - | 0.03 | >0.05 |
| MIR | >0.05 | - | >0.05 |
| BQSA1 | >0.05 | >0.05 | - |

Table 29: Statistical test of Metheuristics SCA and 85 Actions solving SCP

| Inst. | BCL | MIR | QL1 | SA1 |
|---|---|---|---|---|
| BCL | - | 0.0 | >0.05 | >0.05 |
| MIR | >0.05 | - | >0.05 | >0.05 |
| QL1 | 0.0 | 0.0 | - | >0.05 |
| SA1 | 0.0 | 0.0 | >0.05 | - |

Table 30: Statistical test of Metheuristics SCA and 40 Actions solving SCP

| Inst. | BCL | MIR | BQSA1 |
|---|---|---|---|
| BCL | - | 0.0 | >0.05 |
| MIR | >0.05 | - | >0.05 |
| BQSA1 | 0.0 | 0.0 | - |

Table 31: Statistical test of Metheuristics WOA solving 0-1KP

| Inst. | QL1 | QL2 | QL3 | QL4 | QL5 | SA1 | SA2 | SA3 | SA4 | SA5 |
|---|---|---|---|---|---|---|---|---|---|---|
| QL1 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL2 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL3 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL4 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL5 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| SA1 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 |
| SA2 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 |
| SA3 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 |
| SA4 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 |
| SA5 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - |

Table 32: Statistical test of Metheuristics GWO solving 0-1KP

| Inst. | QL1 | QL2 | QL3 | QL4 | QL5 | SA1 | SA2 | SA3 | SA4 | SA5 |
|---|---|---|---|---|---|---|---|---|---|---|
| QL1 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL2 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL3 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL4 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL5 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| SA1 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 |
| SA2 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 |
| SA3 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 |
| SA4 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 |
| SA5 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - |

Table 33: Statistical test of Metheuristics SCA solving 0-1KP

| Inst. | QL1 | QL2 | QL3 | QL4 | QL5 | SA1 | SA2 | SA3 | SA4 | SA5 |
|---|---|---|---|---|---|---|---|---|---|---|
| QL1 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL2 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL3 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL4 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| QL5 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 |
| SA1 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 | >0.05 |
| SA2 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 | >0.05 |
| SA3 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 | >0.05 |
| SA4 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - | >0.05 |
| SA5 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | >0.05 | - |

## 4.4 Action charts

We present average action charts Fig. (31-60), these charts give us information about the average choices made by our reinforcement learning techniques during the iterative process of the MH. This generates a weighting of the choices of the binarization schemes, allowing us to identify which are the preferences of the implemented techniques according to the state (exploratory or exploitative) in which they are during the execution of the algorithm. The charts presented in this section are composed as follows: on the abscissa axis is the average number of times the action has been selected for the respective state, while on the ordinate axis are the possible actions for the binarization scheme selector, in Fig.(29) and (30) you can distinguish the two versions with their respective actions.


Figure 29: y-axis with 85-action zoom.


Figure 30: y-axis with 40-action zoom.

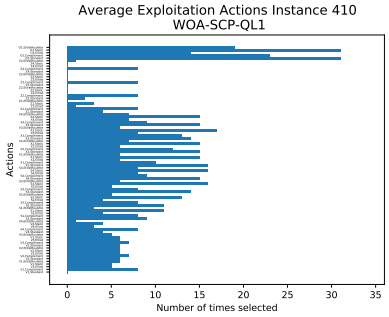


Figure 31: 85 Actions Chart - Average number of actions in exploitation state.

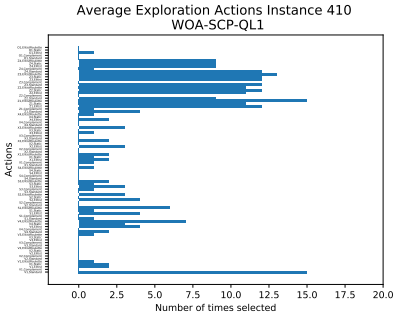


Figure 32: 85 Actions Chart - Average number of actions in exploration state.

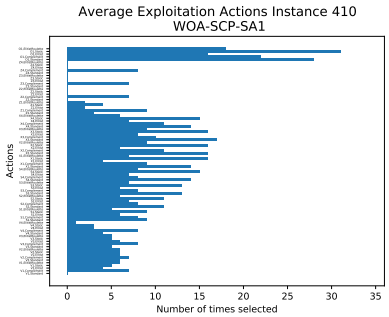


Figure 33: 85 Actions Chart - Average number of actions in exploitation state.

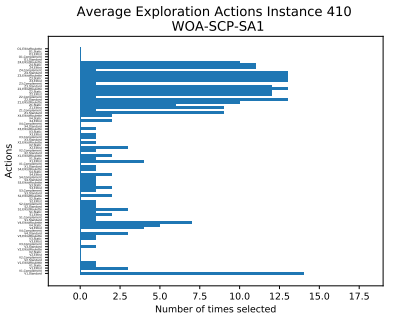


Figure 34: 85 Actions Chart - Average number of actions in exploration state.

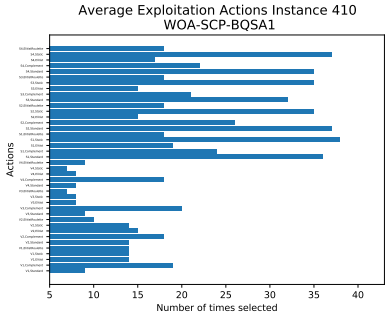


Figure 35: 40 Actions Chart - Average number of actions in exploitation state.

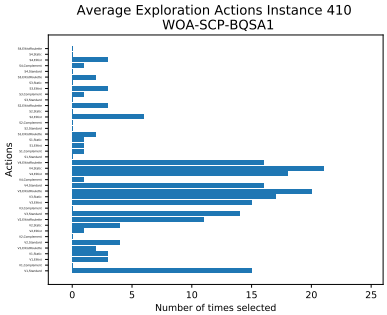


Figure 36: 40 Actions Chart - Average number of actions in exploration state.

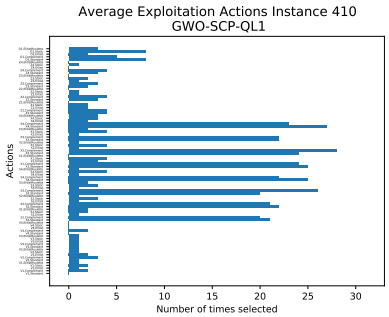


Figure 37: 85 Actions Chart - Average number of actions in exploitation state.

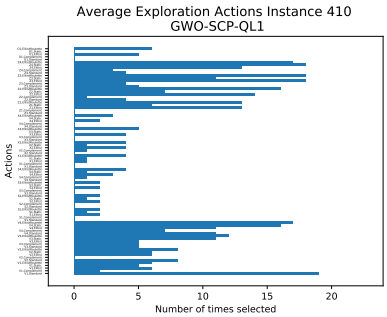


Figure 38: 85 Actions Chart - Average number of actions in exploration state.

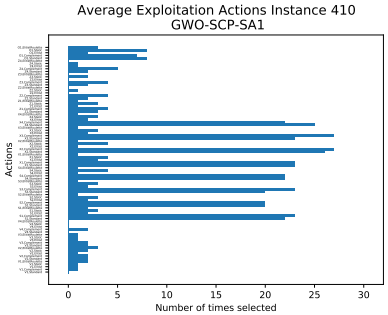


Figure 39: 85 Actions Chart - Average number of actions in exploitation state.

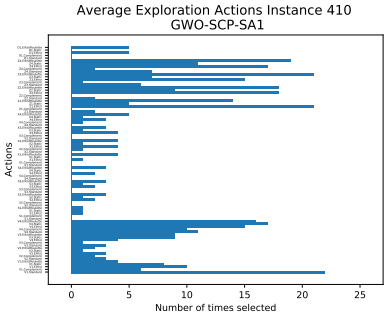


Figure 40: 85 Actions Chart - Average number of actions in exploration state.

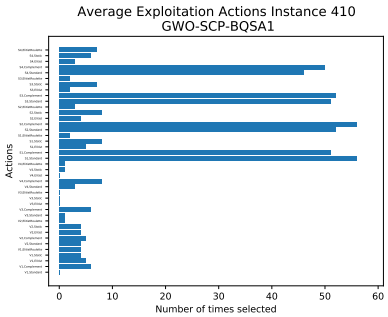


Figure 41: 40 Actions Chart - Average number of actions in exploitation state.

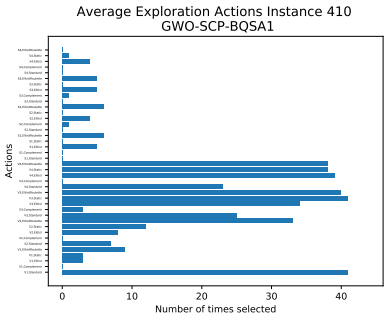


Figure 42: 40 Actions Chart - Average number of actions in exploration state.

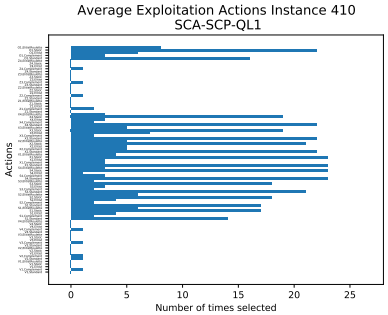


Figure 43: 85 Actions Chart - Average number of actions in exploitation state.

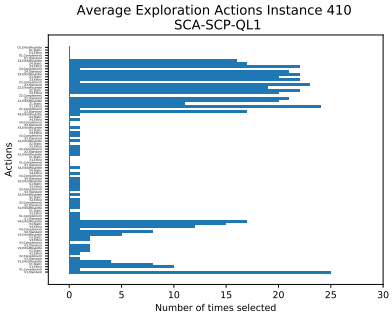


Figure 44: 85 Actions Chart - Average number of actions in exploration state.

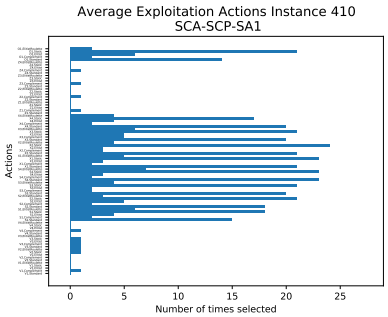


Figure 45: 85 Actions Chart - Average number of actions in exploitation state.

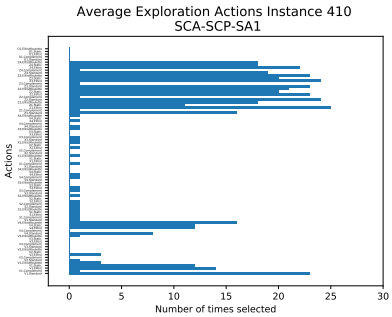


Figure 46: 85 Actions Chart - Average number of actions in exploration state.

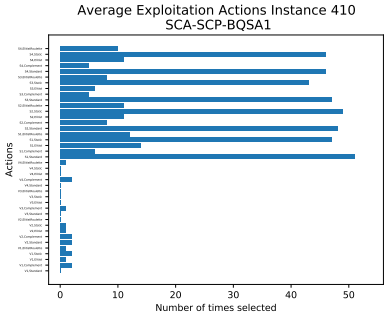


Figure 47: 40 Actions Chart - Average number of actions in exploitation state.

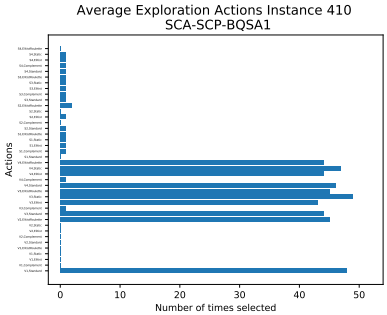


Figure 48: 40 Actions Chart - Average number of actions in exploration state.

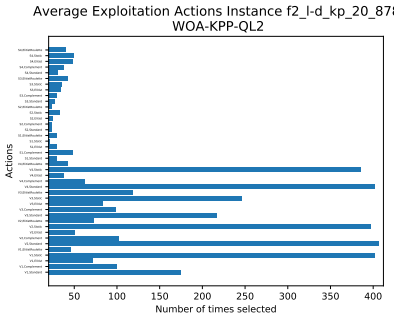


Figure 49: 40 Actions Chart - Average number of actions in exploitation state.

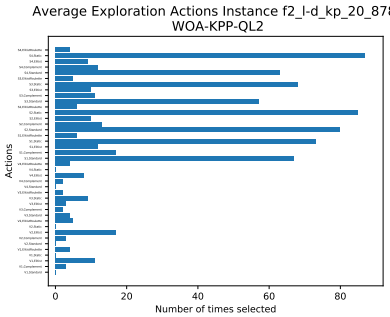


Figure 50: 40 Actions Chart - Average number of actions in exploration state.

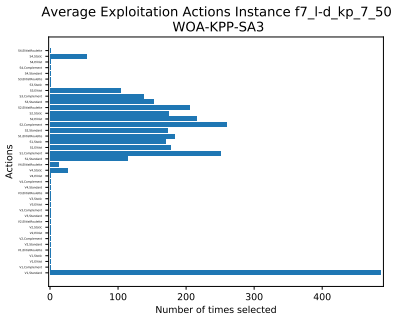


Figure 51: 40 Actions Chart - Average number of actions in exploitation state.

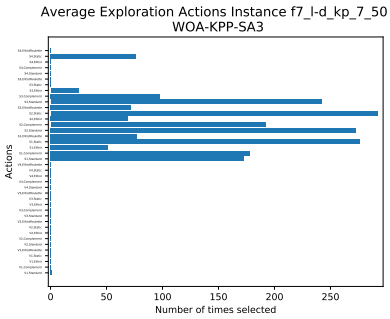


Figure 52: 40 Actions Chart - Average number of actions in exploration state.

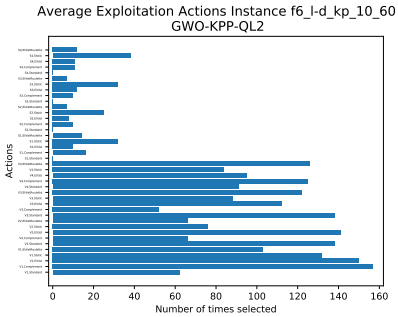


Figure 53: 40 Actions Chart - Average number of actions in exploitation state.

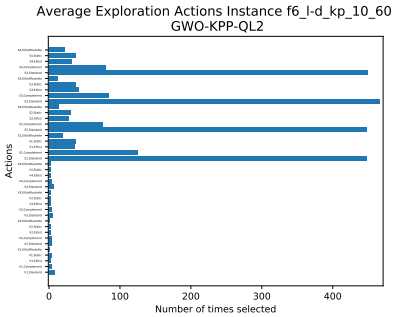


Figure 54: 40 Actions Chart - Average number of actions in exploration state.

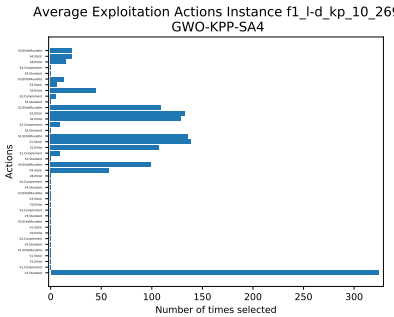


Figure 55: 40 Actions Chart - Average number of actions in exploitation state.

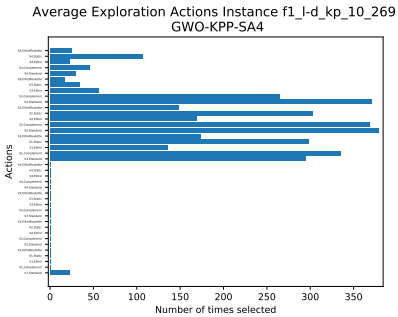


Figure 56: 40 Actions Chart - Average number of actions in exploration state.

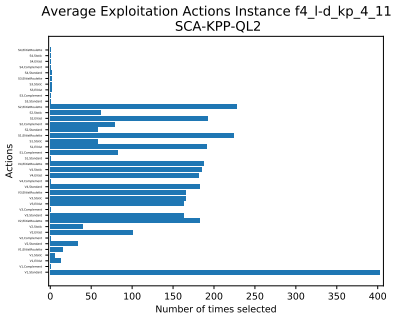


Figure 57: 40 Actions Chart - Average number of actions in exploitation state.

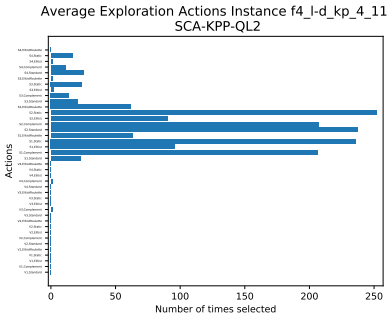


Figure 58: 40 Actions Chart - Average number of actions in exploration state.

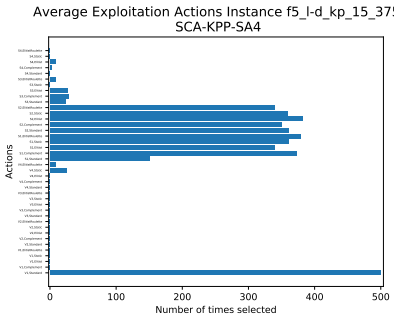


Figure 59: 40 Actions Chart - Average number of actions in exploitation state.

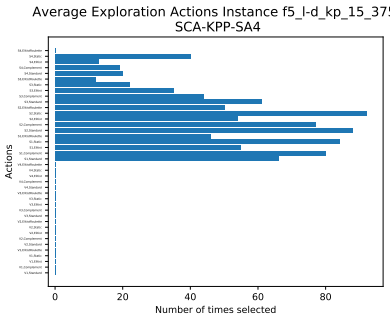


Figure 60: 40 Actions Chart - Average number of actions in exploration state.

## 4.5 Exploration-Exploitation Charts

In order to better understand the diversity and behavior of the MH in their exploratory and exploitation phases, it is necessary to use tools that allow us to visualize this process in a more graphic way. We present exploration and exploitation charts (Fig. 61-84) as presented by Morales-Castañeda et al. [84] with the way of calculating diversity presented in Eq. (78). The charts are composed of: the x-axis indicating the number of total iterations and the y-axis which is the percentage of exploration and exploitation that was obtained from Eq. (79) and (80).

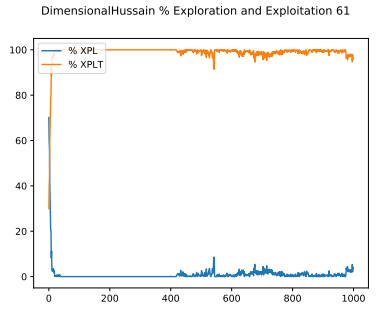


Figure 61: Exploration and Exploitation Chart of WOA - BCL - 61 solving SCP.

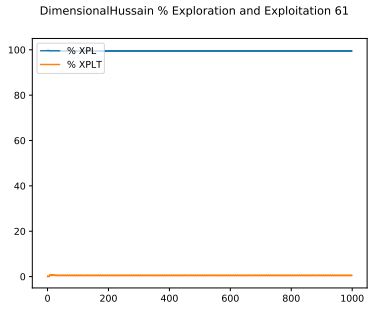


Figure 62: Exploration and Exploitation Chart of WOA - MIR - 61 solving SCP.

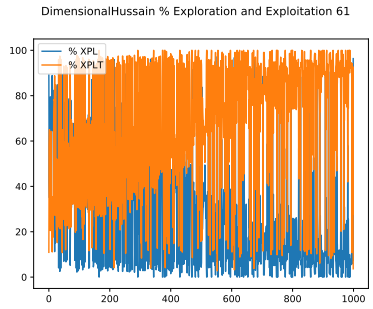


Figure 63: Exploration and Exploitation Chart of WOA - QL1 - 61 solving SCP with 85 actions.

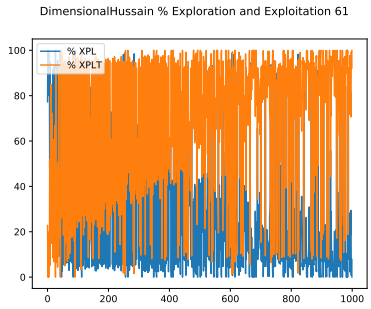


Figure 64: Exploration and Exploitation Chart of WOA - SA1 - 61 solving SCP with 85 actions.

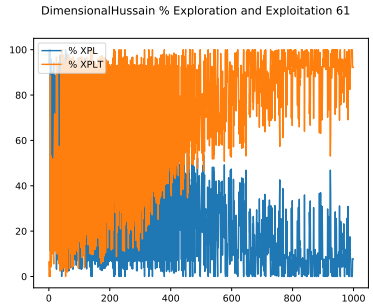


Figure 65: Exploration and Exploitation Chart of WOA - BQSA1 - 61 solving SCP with 40 actions.

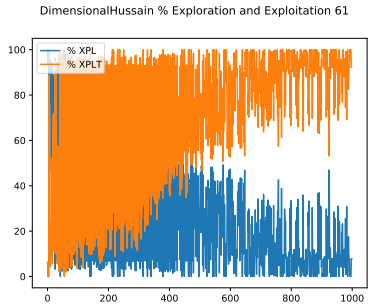


Figure 66: Exploration and Exploitation Chart of WOA - BQSA1 - 61 solving SCP with 40 actions.

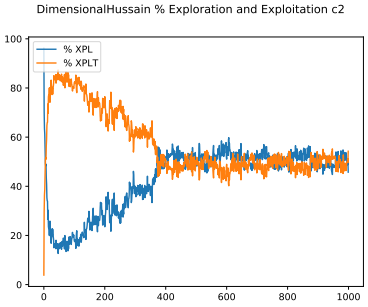


Figure 67: Exploration and Exploitation Chart of GWO - BCL - c2 solving SCP.

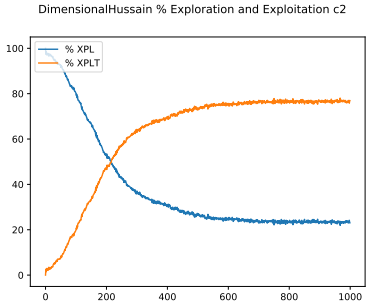


Figure 68: Exploration and Exploitation Chart of GWO - MIR - c2 solving SCP.

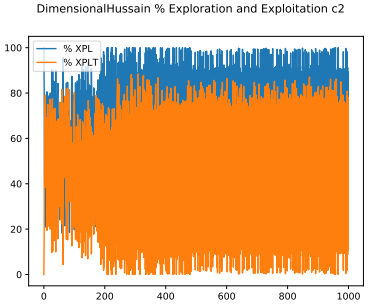


Figure 69: Exploration and Exploitation Chart of GWO - QL1 - c2 solving SCP with 85 actions.

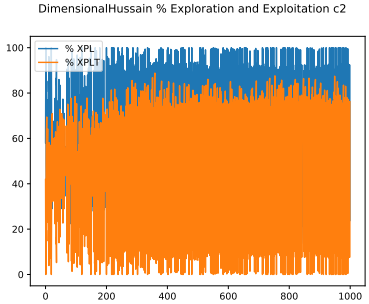


Figure 70: Exploration and Exploitation Chart of GWO - SA1 - c2 solving SCP with 85 actions.

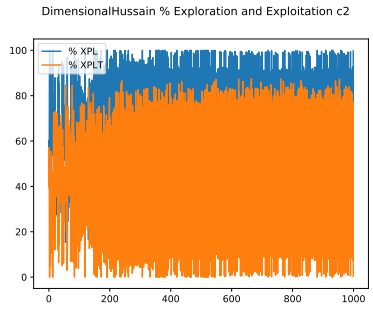


Figure 71: Exploration and Exploitation Chart of GWO - BQSA1 - c2 solving SCP with 40 actions.

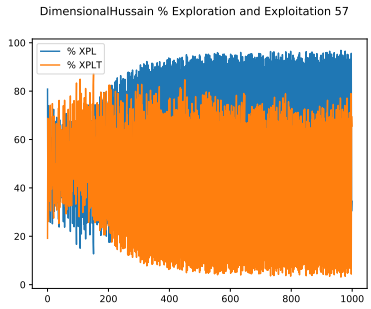


Figure 72: Exploration and Exploitation Chart of GWO - BQSA1 - 57 solving SCP with 40 actions.

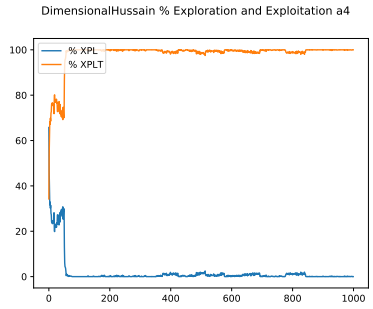


Figure 73: Exploration and Exploitation Chart of SCA - BCL - a4 solving SCP.

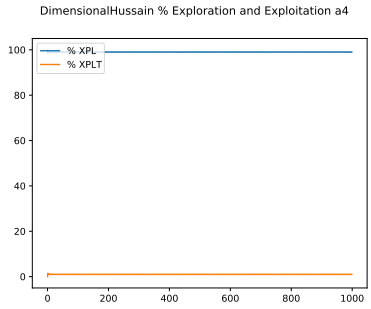


Figure 74: Exploration and Exploitation Chart of SCA - MIR - a4 solving SCP.

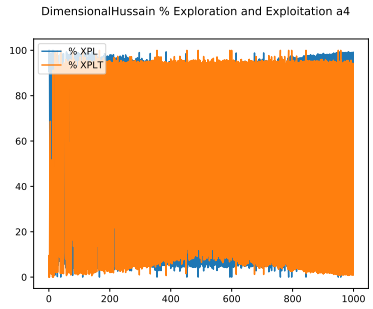


Figure 75: Exploration and Exploitation Chart of SCA - QL1 - a4 solving SCP with 85 actions.

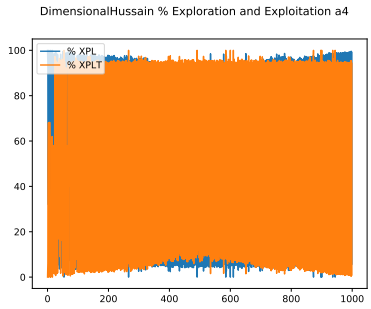


Figure 76: Exploration and Exploitation Chart of SCA - SA1 - a4 solving SCP with 85 actions.

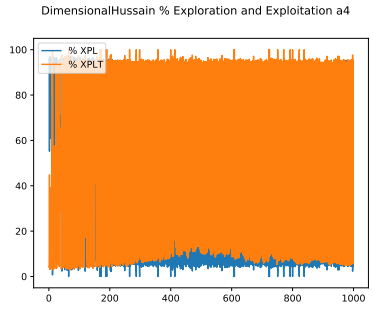


Figure 77: Exploration and Exploitation Chart of SCA - BQSA1 - a4 solving SCP with 40 actions.

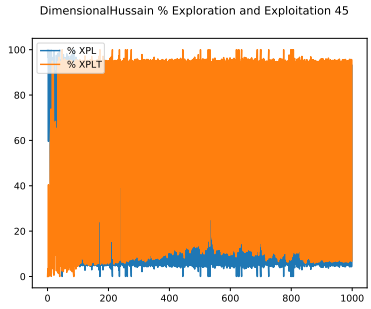


Figure 78: Exploration and Exploitation Chart of SCA - BQSA1 - 45 solving SCP with 40 actions.

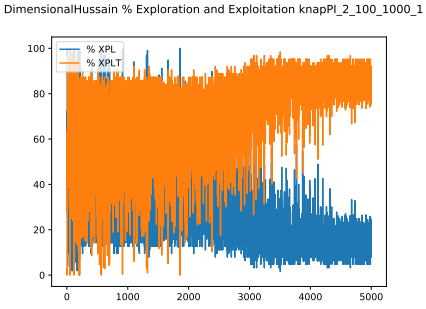


Figure 79: Exploration and Exploitation Chart of WOA - QL5 - KP2 100 solving 0-1KP with 40 actions.

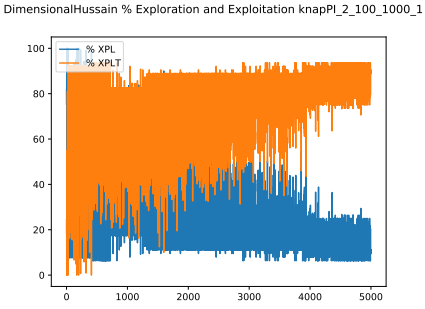


Figure 80: Exploration and Exploitation Chart of WOA - SA5 - KP2 100 solving 0-1KP with 40 actions.

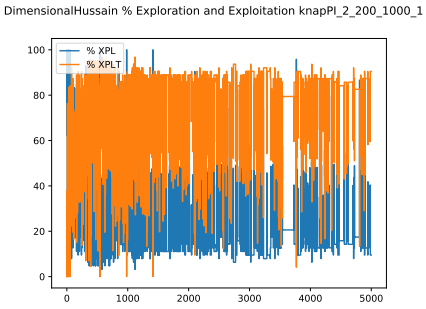


Figure 81: Exploration and Exploitation Chart of GWO - QL3 - KP2 100 solving 0-1KP with 40 actions.

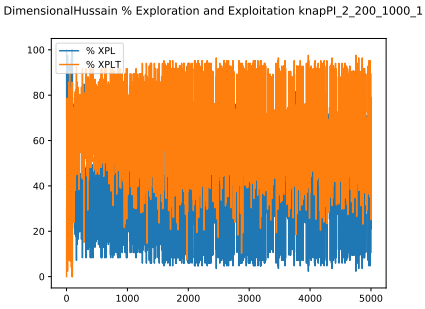


Figure 82: Exploration and Exploitation Chart of GWO - SA3 - KP2 100 solving 0-1KP with 40 actions.

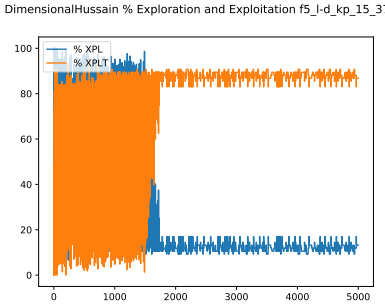


Figure 83: Exploration and Exploitation Chart of SCA - QL4 - KP5 solving 0-1KP with 40 actions.

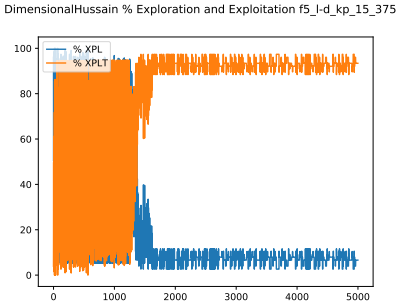


Figure 84: Exploration and Exploitation Chart of SCA - SA4 - KP5 solving 0-1KP with 40 actions.

# 5 Analysis and Discussions

We will analyze and discuss the results obtained in the section 4. We will start by analyzing the tables, passing through the Best fitness obtained, the avgs and RPDs and then the action and exploration-exploitation charts.

## 5.1 Tables Analysis and Discussions

By reviewing the summary tables (10-12) or the detailed tables (13-24), we can obtain quite interesting information about the behavior of the implemented techniques. In the tables obtained by solving the SCP problem with 85 actions (13, 15, 17), we can notice that in the MH: WOA and SCA the SARSA technique has stood out compared to QL and to the static versions BCL and MIR, however, for the GWO case, neither of the two implemented techniques has improved the BCL version. On the other hand, reviewing the statistical tables (25, 27, 29) for these 85 actions, we can notice that for WOA and SCA there IS statistically significant difference between the implemented techniques (QL1 and SA1) and the static versions (BCL and MIR), however, there is NO significant difference between the versions of QL1 and SA1. On the GWO side, we can notice that there is only a difference between the SA1 and MIR technique.

With the tables containing the versions with 40 actions (14, 16, 18), we can observe that the new implemented technique of BQSA has had a similar

behavior to QL and SARSA: in WOA and SCA the BQSA has stood out compared to the static versions and for GWO as in the 85 actions, BCL has not been improved by any of the implemented versions and techniques. Reviewing the statistical tests in tables (26, 28, 30): again in WOA and SCA we COUNT significant difference between the technique (BQSA) and the static versions and in GWO there is NO difference between the technique and the static ones.

Finally, with the tables that solve the 0-1KP (19-24), we can observe that for GWO and SCA the technique that has stood out has been SA1 in comparison with the QL versions, while in WOA it has been the opposite, QL1 has stood out in comparison with the SARSA versions. Thanks to the RPD and the statistical tests that record and analyze the 0-1KP results (31-33), we can observe that for all SARSA and QL versions there is NO statistically significant difference.

## 5.2 Charts Analysis and Discussions

If we read the work of Crawford et al. [10], we can conclude that binarization schemes have a quite strong impact on the performance of an MH, that is why Fig. (31-60) give us valuable information about this impact during the exploration and exploitation process. With the analysis of the average action charts of the versions implemented with 85 actions (31-34, 37-40, 43-46), for both binarization scheme selectors (SARSA and QL) in the exploited state have preference for transfer functions of type X, S and O (middle area of the y-axis) while in the explored state they are strongly attracted to transfer functions of type V and Z (lower and upper area of the y-axis).

In the charts recording the implemented technique with 40 actions (35-36, 41-42, 47-48), the behavior is similar to the charts having 85 actions with the QL and SARSA techniques. This time the BQSA technique in the exploitation for WOA, GWO and SCA feels attraction for S-type transfer functions, while in the exploration the attraction is for V-type ones.

In the action charts corresponding to 0-1KP (49-60), the behavior is different from BQSA. In this occasion for both QL and SARSA techniques the attraction is by V-type in exploitation and by S-type in exploration.

Thanks to the Fig. (61-84), we can see the behavior of the exploration and exploitation metrics during the iterative process, which allows us to

know the influence of the binarization schemes used between the different techniques, versions and MH. From these figures we can conclude that, in WOA for SCP and 01-KP we have a similar behavior according to the exploration and exploitation percentages, however, it can be observed that for SCP during the whole iterative process it has variations of greater amplitude, more pronounced and in accordance with what Morales-Castañeda presents us.

For the GWO versions we have the opposite in SCP, a trend is still seen thanks to the colors of the graph, but it is no longer as exploitative as the iterations progress, instead, it becomes exploratory. While in 0-1KP we again have a behavior similar to that presented by WOA, as the iterations progress the state becomes exploitative until the end of the optimization process.

Finally, for the versions with SCA, we still see the trend thanks to the colors of the graph, but it is no longer as pronounced as in WOA or GWO, it has constant variations in its phases with large amplitude variations for the case of SCP, while for 0-1KP we clearly see the trend of its behavior, more similar to those of WOA and GWO and to what was presented by Morales-Castañeda.

# 6 Conclusions

RL techniques have become famous in most research areas, as they allow the automation of several processes within the MH work. In the literature currently, there are several MH that are being supported by RL techniques from different approaches, aiming to improve the performance of these techniques. Due to this, the motivation to make this proposal arises, since the MH generate a large amount of data that are not always used for the operation of the same MH. It should be noted that in the proposed techniques the sizes of the Q-Tables are small, due to the small size of the sets of operators, unlike other techniques that work with large amounts of input variables.

In this work 2 different problems were proposed; Set Covering Problem and 0-1 Knapsack Problem being solved with 3 MH; Whale Optimization Algorithm, Grey Wolf Optimization and Sine-Cosine Algorithm, hybridized with three RL techniques; SARSA, QL and BQSA, giving some implementations 5 different reward versions (Table 6), giving a total of 45 different implementations. 12 implementations in SCP with 85 actions, 3 in SCP with

40 actions and finally 30 implementations in 0-1KP. The techniques of these RL to MH techniques have presented interesting fitness development and exploration and exploitation behavior. Different parameterizations have been given to the different implementations: for the SCP problem 45 instances with 40 population and 1000 iterations have been solved and in 0-1KP 19 instances with 20 population and 5000 iterations have been considered.

Experiments have shown that the MH: WOA and SCA tackling the SCP problem achieve significantly better results (although not statistically significantly) on the versions including SARSA and BQSA (in the case of the versions with 40 shares). In addition, they show near-optimal performance in most of the 45 situations considered. The behavior is similar for 0-1KP, where MH: GWO and SCA have been shown to achieve better results (again not statistically significant) in the versions with SARSA with the exception of GWO, where the best is QL.

Future work will aim at finalizing the experiments already performed, i.e. for the versions considering the 85 actions, to integrate BQSA to the tables and compare them, while for the versions with 40 actions to integrate QL and SARSA. In addition, the integration of other Temporal Difference approaches classified as RL techniques present in the literature is proposed.

## References


[1] Erik Cuevas, Fernando Fausto, and Adrián González. *New Advancements in Swarm Algorithms: Operators and Applications.* Springer, 2020.

[2] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997.

[3] David H Wolpert and William G Macready. Coevolutionary free lunches. *IEEE Transactions on evolutionary computation*, 9(6):721–735, 2005.

[4] Stefan Voss, V Maniezzo, and T Stützle. Matheuristics: Hybridizing metaheuristics and mathematical programming (annals of information systems). 2009.

[5] Angel A Juan, Javier Faulin, Scott E Grasman, Markus Rabe, and Gonçalo Figueira. A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations Research Perspectives*, 2:62–72, 2015.

[6] El-Ghazali Talbi. Combining metaheuristics with mathematical programming, constraint programming and machine learning. *Annals of Operations Research*, 240(1):171–215, 2016.

[7] El-Ghazali Talbi. Machine learning into metaheuristics: A survey and taxonomy of data-driven metaheuristics. 2020.

[8] Heda Song, Isaac Triguero, and Ender Özcan. A review on the self and dual interactions between machine learning and optimisation. *Progress in Artificial Intelligence*, 8(2):143–165, 2019.

[9] Laura Calvet, Jésica de Armas, David Masip, and Angel A Juan. Learnheuristics: hybridizing metaheuristics with machine learning for optimization with dynamic inputs. *Open Mathematics*, 15(1):261–280, 2017.

[10] Broderick Crawford, Ricardo Soto, Gino Astorga, José García, Carlos Castro, and Fernando Paredes. Putting continuous metaheuristics to work in binary search spaces. *Complexity*, 2017, 2017.

[11] Hanns de la Fuente-Mella, Diego Tapia, José Lemus-Romani, Mauricio Castillo, Marcelo Becerra-Rozas, Fernando Paredes, and Sanjay Misra.

A data-driven dynamic discretization framework to solve combinatorial problems using continuous metaheuristics. In *Innovations in Bio-Inspired Computing and Applications: Proceedings of the 11th International Conference on Innovations in Bio-Inspired Computing and Applications (IBICA 2020) held during December 16–18, 2020*, page 76. Springer Nature.

[12] Diego Tapia, Broderick Crawford, Ricardo Soto, Felipe Cisternas-Caneo, José Lemus-Romani, Mauricio Castillo, José García, Wenceslao Palma, Fernando Paredes, and Sanjay Misra. A q-learning hyperheuristic binarization framework to balance exploration and exploitation. In *International Conference on Applied Informatics*, pages 14–28. Springer, 2020.

[13] Diego Tapia, Broderick Crawford, Ricardo Soto, Wenceslao Palma, José Lemus-Romani, Felipe Cisternas-Caneo, Mauricio Castillo, Marcelo Becerra-Rozas, and Sanjay Misra. Embedding q-learning in the selection of metaheuristic operators: The enhanced binary grey wolf optimizer case. In *In proceedongs of 2021 IEEE International Conference on Automation/XXIV Congress of the Chilean Association of Automatic Control (ICA-ACCA)*. IEEE, 2021.

[14] Broderick Crawford, Ricardo Soto, Tapia Diego Cisternas-Caneo, Felipe, Hanns de la Fuente-Mella, Wenceslao Palma, José Lemus-Romani, Mauricio Castillo, and Marcelo Becerra-Rozas. A comparison of learnheuristics using different reward functions to solve the set covering problem. In *In proceedings of International Conference on Optimization and Learning (OLA 2021)*, 2021.

[15] El-Ghazali Talbi. *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons, 2009.

[16] S Binitha, S Siva Sathya, et al. A survey of bio inspired optimization algorithms. *International journal of soft computing and engineering*, 2(2):137–151, 2012.

[17] Seyedali Mirjalili and Andrew Lewis. The whale optimization algorithm. *Advances in engineering software*, 95:51–67, 2016.

[18] Ümit Can and Bilal Alataş. Physics based metaheuristic algorithms for global optimization. 2015.

[19] Meeta Kumar and Anand J Kulkarni. Socio-inspired optimization metaheuristics: a review. In *Socio-cultural inspired metaheuristics*, pages 241–265. Springer, 2019.

[20] Ali Abdullah Hassan, Salwani Abdullah, Kamal Z Zamli, and Rozilawati Razali. Combinatorial test suites generation strategy utilizing the whale optimization algorithm. *IEEE Access*, 8:192288–192303, 2020.

[21] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. Grey wolf optimizer. *Advances in engineering software*, 69:46–61, 2014.

[22] Seyedali Mirjalili. Sca: a sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, 96:120–133, 2016.

[23] SeyedAli Mirjalili and Siti Zaiton Mohd Hashim. Bmoa: binary magnetic optimization algorithm. *International Journal of Machine Learning and Computing*, 2(3):204, 2012.

[24] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Xin-She Yang. Binary bat algorithm. *Neural Computing and Applications*, 25(3):663–681, 2014.

[25] Majdi Mafarja, Ibrahim Aljarah, Ali Asghar Heidari, Hossam Faris, Philippe Fournier-Viger, Xiaodong Li, and Seyedali Mirjalili. Binary dragonfly optimization for feature selection using time-varying transfer functions. *Knowledge-Based Systems*, 161:185–204, 2018.

[26] Ricardo Soto, Broderick Crawford, Alexis Munoz, Franklin Johnson, and Fernando Paredes. Pre-processing, repairing and transfer functions can help binary electromagnetism-like algorithms. In *Artificial intelligence perspectives and applications*, pages 89–97. Springer, 2015.

[27] Ricardo Soto, Broderick Crawford, Rodrigo Olivares, Jorge Barraza, Ignacio Figueroa, Franklin Johnson, Fernando Paredes, and Eduardo Olguín. Solving the non-unicost set covering problem by using cuckoo search and black hole optimization. *Natural Computing*, 16(2):213–229, 2017.

[28] Ricardo Soto, Broderick Crawford, Rodrigo Olivares, Carla Taramasco, Ignacio Figueroa, Álvaro Gómez, Carlos Castro, and Fernando Paredes. Adaptive black hole algorithm for solving the set covering problem. *Mathematical Problems in Engineering*, 2018, 2018.

[29] Seyedali Mirjalili and Andrew Lewis. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation*, 9:1–14, 2013.

[30] Hossam Faris, Majdi M Mafarja, Ali Asghar Heidari, Ibrahim Aljarah, Al-Zoubi Ala'M, Seyedali Mirjalili, and Hamido Fujita. An efficient binary salp swarm algorithm with crossover scheme for feature selection problems. *Knowledge-Based Systems*, 154:43–67, 2018.

[31] Miguel OLIVARES-SUAREZ, Wenceslao PALMA, Fernando PAREDES, Eduardo OLGUÍN, and Enrique NORERO. A binary coded firefly algorithm that solves the set covering problem. *Science and Technology*, 17(3):252–264, 2014.

[32] Broderick Crawford, Ricardo Soto, Rodrigo Cuesta, and Fernando Paredes. Using the bee colony optimization method to solve the weighted set covering problem. In *International Conference on Human-Computer Interaction*, pages 493–497. Springer, 2014.

[33] Broderick Crawford, Ricardo Soto, Natalia Berrios, and Eduardo Olguin. Solving the set covering problem using the binary cat swarm optimization metaheuristic. *International Journal of Mathematical and Computational Sciences*, 10(3):113–117, 2016.

[34] Shahrzad Saremi, Seyedali Mirjalili, and Andrew Lewis. How important is a transfer function in discrete heuristic algorithms. *Neural Computing and Applications*, 26(3):625–640, 2015.

[35] James Kennedy and Russell C Eberhart. A discrete binary version of the particle swarm algorithm. In *1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation*, volume 5, pages 4104–4108. IEEE, 1997.

[36] Yanhong Feng, Haizhong An, and Xiangyun Gao. The importance of transfer function in solving set-union knapsack problem based on discrete moth search algorithm. *Mathematics*, 7(1):17, 2019.

[37] Gary Pampara, Nelis Franken, and Andries Petrus Engelbrecht. Combining particle swarm optimisation with angle modulation to solve binary problems. In *2005 IEEE congress on evolutionary computation*, volume 1, pages 89–96. IEEE, 2005.

[38] M Fernanda P Costa, Ana Maria AC Rocha, Rogério B Francisco, and Edite MGP Fernandes. Heuristic-based firefly algorithm for bound constrained nonlinear binary optimization. *Advances in Operations Research*, 2014, 2014.

[39] Gary Pampará and Andries P Engelbrecht. Binary artificial bee colony optimization. In *2011 IEEE Symposium on Swarm Intelligence*, pages 1–8. IEEE, 2011.

[40] Hong Zhu, Yichao He, Xizhao Wang, and Eric CC Tsang. Discrete differential evolutions for the discounted {0-1} knapsack problem. *International Journal of Bio-Inspired Computation*, 10(4):219–238, 2017.

[41] Kushal Kanti Ghosh, Pawan Kumar Singh, Junhee Hong, Zong Woo Geem, and Ram Sarkar. Binary social mimic optimization algorithm with x-shaped transfer function for feature selection. *IEEE Access*, 8:97890–97906, 2020.

[42] Zahra Beheshti. A novel x-shaped binary particle swarm optimization. *Soft Computing*, 25(4):3013–3042, 2021.

[43] Sha-sha Guo, Jie-sheng Wang, and Meng-wei Guo. Z-shaped transfer functions for binary particle swarm optimization algorithm. *Computational Intelligence and Neuroscience*, 2020, 2020.

[44] Wei-Zhong Sun, Min Zhang, Jie-Sheng Wang, Sha-Sha Guo, Min Wang, and Wen-Kuo Hao. Binary particle swarm optimization algorithm based on z-shaped probability transfer function to solve 0-1 knapsack problem. *IAENG International Journal of Computer Science*, 48(2), 2021.

[45] Seyedehzahra Mirjalili, Hongyu Zhang, Seyedali Mirjalili, Stephan Chalup, and Nasimul Noman. A novel u-shaped transfer function for binary particle swarm optimisation. In *9th International Conference on Soft Computing for Problem Solving, SocProS 2019*, pages 241–259. Springer Gabler, 2020.

[46] Md Jakirul Islam, Xiaodong Li, and Yi Mei. A time-varying transfer function for balancing the exploration and exploitation ability of a binary pso. *Applied Soft Computing*, 59:182–196, 2017.

[47] Mohammed Abdulrazaq Kahya, Suhaib Abduljabbar Altamir, and Zakariya Yahya Algamal. Improving whale optimization algorithm for feature selection with a time-varying transfer function. *Numerical Algebra, Control & Optimization*, 11(1):87, 2021.

[48] Hamouda Chantar, Thaer Thaher, Hamza Turabieh, Majdi Mafarja, and Alaa Sheta. Bhho-tvs: A binary harris hawks optimizer with time-varying scheme for solving data classification problems. *Applied Sciences*, 11(14):6516, 2021.

[49] Zong Woo Geem, Joong Hoon Kim, and Gobichettipalayam Vasudevan Loganathan. A new heuristic optimization algorithm: harmony search. *simulation*, 76(2):60–68, 2001.

[50] N Rajalakshmi, D Padma Subramanian, and K Thamizhavel. Performance enhancement of radial distributed system with distributed generators by reconfiguration using binary firefly algorithm. *Journal of The Institution of Engineers (India): Series B*, 96(1):91–99, 2015.

[51] Antonio Lavecchia. Machine-learning approaches in drug discovery: methods and applications. *Drug discovery today*, 20(3):318–331, 2015.

[52] Konstantina Kourou, Themis P Exarchos, Konstantinos P Exarchos, Michalis V Karamouzis, and Dimitrios I Fotiadis. Machine learning applications in cancer prognosis and prediction. *Computational and structural biotechnology journal*, 13:8–17, 2015.

[53] Rishikesh Magar, Prakarsh Yadav, and Amir Barati Farimani. Potential neutralizing antibodies discovered for novel corona virus using machine learning. *Scientific reports*, 11(1):1–11, 2021.

[54] Joon Lee, Joel A Dubin, and David M Maslove. Mortality prediction in the icu. *Secondary Analysis of Electronic Health Records*, pages 315–324, 2016.

[55] P Karthikeyan, Karunakaran Velswamy, Pon Harshavardhanan, R Rajagopal, V JeyaKrishnan, and S Velliangiri. Machine learning techniques application: Social media, agriculture, and scheduling in distributed systems. In *Research Anthology on Architectures, Frameworks, and Integration Strategies for Distributed and Cloud Computing*, pages 1396–1417. IGI Global, 2021.

[56] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24, 2007.

[57] M Emre Celebi and Kemal Aydin. *Unsupervised learning algorithms*, volume 9. Springer, 2016.

[58] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[59] A Inés, C Domínguez, J Heras, E Mata, and V Pascual. Biomedical image classification made easier thanks to transfer and semi-supervised learning. *Computer Methods and Programs in Biomedicine*, 198:105782, 2021.

[60] Aayush K Chaudhary, Prashnna K Gyawali, Linwei Wang, and Jeff B Pelz. Semi-supervised learning for eye image segmentation. *arXiv preprint arXiv:2103.09369*, 2021.

[61] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[62] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[63] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[64] José García, Paola Moraga, Matias Valenzuela, Broderick Crawford, Ricardo Soto, Hernan Pinto, Alvaro Peña, Francisco Altimiras, and Gino Astorga. A db-scan binarization algorithm applied to matrix covering problems. *Computational intelligence and neuroscience*, 2019, 2019.

[65] Alan Dávila de León, Eduardo Lalla-Ruiz, Belén Melián-Batista, and J Marcos Moreno-Vega. A machine learning-based system for berth scheduling at bulk terminals. *Expert Systems with Applications*, 87:170–182, 2017.

[66] Andrew W Lo. Reconciling efficient markets with behavioral finance: the adaptive markets hypothesis. *Journal of investment consulting*, 7(2):21–44, 2005.

[67] Tony Wauters, Katja Verbeeck, Patrick De Causmaecker, and Greet Vanden Berghe. Boosting metaheuristic search using reinforcement learning. In *Hybrid Metaheuristics*, pages 433–452. Springer, 2013.

[68] Edmund Burke, Graham Kendall, Jim Newall, Emma Hart, Peter Ross, and Sonia Schulenburg. Hyper-heuristics: An emerging direction in modern search technology. In *Handbook of metaheuristics*, pages 457–474. Springer, 2003.

[69] Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction mit press. *Cambridge, MA*, 22447, 1998.

[70] RS Sutton. Advances in neural information processing systems: Vol. 8. generalization in reinforcement learning: Successful examples using sparse coarse coding, 1996.

[71] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

[72] Matthew E Taylor, Peter Stone, and Yaxin Liu. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8(9), 2007.

[73] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[74] Florian Pawlik. The knapsack problem. 2014.

[75] Amira Gherboudj and Salim Chikhi. Bpso algorithms for knapsack problem. In *Recent Trends in Wireless and Mobile Networks*, pages 217–227. Springer, 2011.

[76] Jose M Lanza-Gutierrez, Broderick Crawford, Ricardo Soto, Natalia Berrios, Juan A Gomez-Pulido, and Fernando Paredes. Analyzing the effects of binarization techniques when solving the set covering problem through swarm optimization. *Expert Systems with Applications*, 70:67–82, 2017.

[77] Yue Xu and Dechang Pi. A reinforcement learning-based communication topology in particle swarm optimization. *Neural Computing and Applications*, pages 1–26, 2019.

[78] Shin Siang Choong, Li-Pei Wong, and Chee Peng Lim. Automatic design of hyper-heuristic based on reinforcement learning. *Information Sciences*, 436:89–107, 2018.

[79] Bilal H Abed-alguni. Bat q-learning algorithm. *Jordanian Journal of Computers and Information Technology (JJCIT)*, 3(1):56–77, 2017.

[80] Alexander Nareyek. Choosing search heuristics by non-stationary reinforcement learning. In *Metaheuristics: Computer decision-making*, pages 523–544. Springer, 2003.

[81] Ronghua Chen, Bo Yang, Shi Li, and Shilong Wang. A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem. *Computers & Industrial Engineering*, 149:106778, 2020.

[82] Seung-Seok Choi, Sung-Hyuk Cha, and Charles C Tappert. A survey of binary similarity and distance measures. *Journal of systemics, cybernetics and informatics*, 8(1):43–48, 2010.

[83] Kashif Hussain, William Zhu, and Mohd Najib Mohd Salleh. Long-term memory harris' hawk optimization for high dimensional and optimal power flow problems. *IEEE Access*, 7:147596–147616, 2019.

[84] Bernardo Morales-Castañeda, Daniel Zaldivar, Erik Cuevas, Fernando Fausto, and Alma Rodríguez. A better balance in metaheuristic algorithms: Does it exist? *Swarm and Evolutionary Computation*, page 100671, 2020.

[85] Shihua Zhan, Lijin Wang, Zejun Zhang, and Yiwen Zhong. Noising methods with hybrid greedy repair operator for 0–1 knapsack problem. *Memetic Computing*, 12(1):37–50, 2020.

[86] José García, Broderick Crawford, Ricardo Soto, and Gino Astorga. A clustering algorithm applied to the binarization of swarm intelligence continuous metaheuristics. *Swarm and evolutionary computation*, 44:646–664, 2019.

[87] John E Beasley and Kurt Jörnsten. Enhancing an algorithm for set covering problems. *European Journal of Operational Research*, 58(2):293–300, 1992.

[88] Mohamed Abdel-Basset, Reda Mohamed, and Seyedali Mirjalili. A binary equilibrium optimization algorithm for 0–1 knapsack problems. *Computers & Industrial Engineering*, 151:106946, 2021.

[89] Benyamin Abdollahzadeh, Saeid Barshandeh, Hatef Javadi, and Nicola Epicoco. An enhanced binary slime mould algorithm for solving the 0–1 knapsack problem. *Engineering with Computers*, pages 1–22, 2021.

[90] Ekaba Bisong. Google colaboratory. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 59–64. Springer, 2019.