Abstract

Operation research is a discipline that allows organizations to make efficient decisions, by solving abstract problems in different areas. Computational tools such as heuristics and metaheuristics are used for supporting operation research. These tools have been used with success in the last decades, however, the creation of new metaheuristic is required due to the rise of the complexity of the problems. The goal of this project is to develop a new metaheuristic based on human manners, precisely, the selection of the best individuals using tournaments. Even though metaheuristics based on league championships or tournaments have been developed, none of these has incorporated the concept of learning curves. It is expected that this metaheuristic incorporates components of exploitation, exploration, learning curves and that it provides feasible solutions at a reasonable time.

Key words: Optimization methods, metaheuristic, tournaments.

1. Introduction

Operation research is an applied science that uses analyzing methods to solve abstract and complex problems from different practices, helping the decision-making process [1].

There are different methods for solving problems in engineering and applied sciences, some of these are the exact methods, which through strategies of searching the solution space find the best alternative [2]. Even so, when the problems are complex and on a real scale, they tend to become NP-Hard problems, where the feasible solution space grows exponentially, therefore, the processing time of these techniques becomes too high [3], which decreases its efficiency and leads to the use of heuristic and metaheuristic methods [4].

Heuristic methods are algorithms developed for giving a solution to a specific problem [5], these methods search for the best solution, on a lower computational cost. However, heuristics tend to stagnate on a local optimum [6]. On the other hand, metaheuristics are methods, which iteratively try to give a solution to a complex problem, from a staring solution to a better one inside the feasible region [7]. These methods represent approximation algorithms, which can be applied to a variety of subjects such as, engineering, economy, science, and business, among others [8]. However, metaheuristics cannot assure the global optima, they provide feasible solutions closed to the global optima without recurring to a high computational cost [22].

Metaheuristics have two main components: exploration and exploitation. Exploration is a process where the algorithm explores randomly the feasible region looking for new solutions [9]. Exploitation consists on searching for better solutions into the regions found at the diversification [10]. An important part on the success of a metaheuristic lays on the balance between exploration and exploitation, since exploration gives the
algorithm the capability of finding out new searching spaces, whilst exploitation increases the odds of finding a local optimum in the searching process [10].

Metaheuristics are classified into different categories according to its searching method. The evolutionary metaheuristics are algorithms inspired by evolution, generating new solutions by selection and posterior combination or mutation [11], in this category are found algorithms such as: Genetic Algorithm [12], Genetic Programming (GP) [44], evolutionary strategies algorithms [13], Scatter Search [14], among others. Constructive metaheuristics consist in an iterative process that adds random items to a structure to find a better solution [15], some of the most recognize algorithms of this category are Path Relinking [16] and GRASP [17]. Search algorithms make motions on each iteration to allow the solutions to travel the space looking for a feasible solution [18], some of these algorithms are Tabu Search [19], Cuckoo Search [20], Particle Swarm Optimization (PSO) [21], among others.

Metaheuristics have been implemented with success for solving a variety of problems in applied sciences, engineering, and economy, among others. Some of the classic problems that have been studied are: Vehicle Routing Problem (VRP), where there is searched the best route for minimizing costs of risks and transportation [24, 25, 26, 27]. Scheduling problems where is searched a heterogeneous workflow containing every task required [28, 29, 30, 31, 32]. Localization problems, where the objective is to set the least locations to satisfy all the customer demands [33, 34, 35, 36, 37]. The knapsack problem (KP) looks for the best combination of items packed on a knapsack maximizing the value of the items without exceeding its weight limit [38, 39, 40, 41, 42]. Nurse Rostering Problem (NRP) tries to assign the least number of nurses possible, satisfying the restrictions of the problem [43, 44, 45, 46].

In order to measure the capacity of a metaheuristic to give great answers there has been used different methods. Such as, compare the results given by exact methods on small instances of an original problem, grade the results from the metaheuristic on some numerical functions, compare the results against known instances created from other authors, or compare the quality of the answers given by other metaheuristics.

Along with the advances on computational technology and the growth of the complexity of the optimization problems during the last years, the use of metaheuristics has become more popular. These is due to the rise of the time spent using exact methods for solving complex problems, as the time used for finding a solution grows along to the complexity of the problem, therefore the need to create new methods capable of giving good answers on an acceptable amount of time arises [48]. Although there have designed hybridizations and modifications to some of the most popular metaheuristics [51, 52, 53, 54], there has been proved that some metaheuristics are better solving some problems in particular [49].

The need of creating new metaheuristics arises from the complexity of some problems for finding a good solution. Metaheuristics have a more efficient capacity of giving a great answer to a complex problem than other classic problem-solving methods [50]. This creates the motivation of building searching strategies, which could be able to adapt to a great variety of new and complex problems that arise every day.

Some metaheuristics proposed for solving optimization problems have been inspired by tournaments and competitions [55]. Competitions are a common phenomenon in the humanity. In nature, beings or groups compete between each other for a specific purpose, where the losers often get eliminated [56], therefore competitions, tournaments or leagues are recurred when attempting to find the best being from a specific group or a specific category. Using these structures to create search strategies to solve optimization problems is due to its simple and nature architecture that allows, through competitions, to find the best beings.

Some tournaments structures are Knockout tournament, where the beings are eliminated as soon as they lose the first time. Another type of tournaments are competitions measured by points, where the number of points earned during a specific period defines the winner. There are also tournaments of double eliminations, where a
participant is eliminated from the competition after losing two matches; this structure inspires the idea of creating a k elimination tournament, where the competitors are eliminated after losing k matches. Although some metaheuristics are based on tournaments, none has been inspired by k elimination tournaments.

Learning curves could be described as the improvement on the performance of certain task while realizing it repeatedly, due to the experience earned [57]. At a tournament, the learning process takes place before, during and after individuals confront each other. Even though there are some metaheuristics that have used a learning process and make dentitions based on the experience of other individuals from its environment, like PSO [21], any search structure inspired in tournaments and including a learning component has been found on the literature.

Therefore, in this project it is proposed the design of a new metaheuristic based on a k elimination tournament, where n individuals compete during t periods. As the matches are held, each being has the capacity to modify its way of playing according to its learning curve and the result of the match. The intention is to keep the individuals with the best skills, which had been able to learn from its opponents, therefore had come closer to a better solution.

2. Background

Due to the growth on the complexity of the problems, there have been arisen new NP-hard problems [3], therefore the use of new methods of research or metaheuristic algorithms have been required for trying to give the best answer to these problems [58]. During the last 35 years, great interest and study have been devoted on creating new metaheuristic methods in order to give good solutions to these complex problems [59]. The success of these metaheuristics consists on their capacity for exploitation and exploration the searching space [58].

The first step taken to develop evolutionary strategies, starts with genetic programing, where the initial population evolves under the individual selection, based on physical capacities; through the applications of genetic operators on the future generations was possible to achieve a better solution [60]. Later at the 70’s there was developed the genetic algorithm by John Holland, this method starts with an initial solution, then using tools such as combination and mutation looks for a new solution [61]. At the beginning of the 80’s GRASP (Greedy Randomized Adaptive Search) was settled, which is an iterative algorithm, with a constructive and improvement phase [62].

In 1982, Kirk Patrick proposed Simulated Annealing; this is a metaheuristic with a single solution based on the metal tempering [63]. In 1986, Fred Glover brought forward Tabu Search, one of the representative search methods [64]. In 1992, Dorigo Marco came through with the ant colony optimization, an idea that came from the study of the ant’s behavior which helped the development of that algorithm [65]. Later, in 1995 Kennedy and Eberhart have had a significant progress developing the Particle Swarm Optimization (PSO), this strategy identifies the solutions as particles with a given pattern values, establishing the best position of a particle and the relation between the particle and its neighborhood [66].

In 1997, Storn and Price proposed the Differential Evolution, an evolutionary algorithm which mutation is based on the differences between the individuals instead of using a randomized mutation for the offspring [67]. Genetic Algorithm (GA) has developed some variations on its selection method, while studying the way of finding new parents that help the algorithm to achieve new solutions, some of those methods are based on tournament selection, where the winners of the tournament are those with the highest fitness and are capable to procreate the next generation [75].
In the last 5 years, the interest to development of new metaheuristics had risen as the complexity of the problems grows, which is the reason of creating new methods based on nature or human behavior. The mankind with the years had faced the need to identify the individual who may achieve crucial skills for the best performance, this is the reason why tournaments were developed as leagues or championships. Each one of the methods named before were replicated for the design of new metaheuristics, as an example we find some algorithms focused on tournaments, with some variation depending on the sport or the league.

One of the representative metaheuristics inspired by competitions is the League Championship Algorithm (LCA) by Kashan [51]. The algorithm is based on populations, simulating an artificial tournament with a limited time (quantity of weeks until the end of the tournament). At the beginning, teams are paired, then the pairs change, and the teams face each other week by week, for this process the team can make some changes on the formation they use, a tool used for achieving new solutions.

Moghdani came through with the Volleyball Premier League algorithm (VLP), a new algorithm inspired on the concept of volleyball, for this simulation, teams had a process called Team training where they can evaluate different alignments for a better performance at the match [68]. Considering other metaheuristics related to tournaments Moosavian designed the metaheuristic Soccer League Competition (SLC), an algorithm based on soccer, it has a different method where the players of a team face each in other to improve its performance and win with the official team at the league that holds the tournament between teams [69].

Yousef Mssoudi and Habib Motieghader brought forward the World Competitive Contest, an algorithm based on sport tournament, looking for the best individual. This algorithm is explained by parts, the first stage of the contest holds a little tournament on the search space that is considered as the local search (exploitation), at the second stage the teams with the best performance at the regional tournament face each other, where at this point a global search takes place (exploration) [56].

---

*Fig 1. Metaheuristics classification*
Algorithms based on leagues have its score system, which helps the selection of the best participant [51, 68, 69], some of these algorithms add the knockout tournament after k matches to its league [56] where seeding takes place [76], which allows a better selection of its participants. Within the learning process exist tools for inertia that allows players to resemble the best players of the league [51, 68], there are also individual learning tools such as backward Q-learning [77] that makes way for exploration and exploitation relying on the learning speed.

Within a metaheuristic development, guarantying its capability to find good answers its necessary. According to the literature, some authors have compared the results of their metaheuristics with others obtained by: mathematical functions such as Rosenbrock, Dixon-Price and Beale, exact methods, instances found on the literature [10, 23, 53, 54] or metaheuristics with a similar behavior [70, 71, 72, 73].

Keeping in mind that metaheuristics have been created to answer complex problems, delve into new searching strategies could allow creating new ways or perspectives for problem resolution. Therefore, the purpose of this project is to develop a new metaheuristic based on tournaments including a learning process for its participants.

3. Objectives

To design a metaheuristic algorithm inspired on k elimination tournaments where n competitors confront each other to find the best competitor.

Specific

- To design the logic of a metaheuristic with its characteristics of exploitation and exploration.
- To program the algorithm of the metaheuristic that has been proposed in order to apply it for solving problems.
- To evaluate the performance of the metaheuristic against mathematical functions, problems with instances and other metaheuristics.

4. K eliminations tournament metaheuristic (KET)

Combats take place in between living organisms, which co-exist in the same environment. Animals compete over supplies, food and other biological resources. Humans usually compete for food and territory when these
needs arise; deep rivalries often arise over the pursuit of wealth, power, prestige, or fame. The ancient Greeks held a religious and athletic festival every four years at the sanctuary of Zeus in Olympia. This ancient Olympic Games took place from the 8th century BC to the 4th century AC. At the 3rd century BC, the Roman Empire offered combats between gladiators and wild animals, inspiring admiration and popular acclaim for the strongest fighter. Since 1896, the modern Olympic Games are leading international sporting events in which thousands of athletes form more than 200 nations participate at a competition where the single round robin as the structure of the competition.

At 1972 the E-sports arises as video games competitions with real time strategy, such as multiplayer online battle arena and battle royal games where the teams face each other, in these games usually who wins 2 out of 3 round games is considered as the winner of the match. In E-sports fighting games are common, a video game genre that is played by combats, where the top players are those who can get out of a plateau, a plateau is considered as a stagnation, players develop strategies as conditioning or using baits, that help the learning process against rival’s game mode, trying to have an advantage. The characters in fighting games has some features and a game mode, it is fundamental for a player to recognize his own strengths, weaknesses and opponent features to take a better play mode according to the opponent’s level.

In order to produce a fair matchmaking at E-sports, it is necessary to classify the players according to their performance. As an example, League of Legends a famous multiplayer online battle arena game around the world, in which players control “the champion” with unique abilities. In League of Legends, the players are classified by public rankings based on the Elo rating system, which is a method that evaluates the player’s performance. Into the game, rankings allow to organize teams with players with a similar performance level.

At competitions, the most common types of tournaments are round robin competition and elimination tournaments. A round robin competition is a tournament where each competitor face all of the other rivals; this type of tournament is usually used in sports with a large number of matches without constraints of time. At soccer leagues, it is common to implement a double round robin tournament, where each team faces each rival twice, once at home and once as a visitor. The ranking score at the end of the season determine the winner in a round robin tournament, which is the fairest way to select a champion in a competition. A single elimination or knockout tournament is a competition where individuals are organized by pairs in brackets to compete in a match, the loser of each match is immediately eliminated, and the winner advance to the next round. This continues repeatedly until the final match, which defines the champion.

There are variations of the single elimination tournament, for instance, the double elimination tournament, which consists in a system where competitors are organized by pairs in brackets to compete in matches, unlike single elimination, the loser, is not immediately eliminated, and it has another opportunity to advance in the tournament, at the losers bracket, until after losing twice. The champion in a double elimination tournament is determined at the final match, where the winner of the winners’ brackets faces the winner of the losers’ brackets. As well, a k elimination tournament is a system competition that gives the individuals a greater opportunity to advance in the tournament since they compete until losing k matches. For eliminations tournaments it is recommended a number of competitors equal to a power of two, in order to organize matches by pairs. Fig. 3 shows a representation of this process.
The winner of a match advances to next round in the winners’ bracket while the loser descends to compete at losers’ bracket. For instance, in Fig. 3, player 1 faced player 8 in round one, where the player 1 as a winner of the match advanced to round two in the winners’ bracket and player 8 (the loser) descends to the losers’ bracket. At the losers’ bracket, the winner of a match advances to next round in the losers’ bracket, while the loser is immediately eliminated. In the final round, the winner of winners’ bracket (player 3) and winner of losers’ bracket (player 1), face each other. Player 3 was the winner of final match, so he was the champion, if player 1 had won the final match, a second final match is held since no one had lost two matches.

*KET* metaheuristic structure is inspired by *k* elimination tournaments. The architecture of *k* eliminations tournament is based on double elimination tournaments, nevertheless in *k* eliminations tournaments, players are not limited to lose just twice, at this kind of tournament participants are eliminated after losing *k* number of matches.

**Fig. 4** shows a general representation of the architecture of KET metaheuristic where, each phase of the tournament is shown. Throughout the tournament after each match, the players have a learning phase to improve their performance. Initially a round robin phase is held; where all players face each other in order to evaluate their performance. At the seeding phase, players are classified in a ranking according their performance, which the Elo rating system helps with the classification process. At the groups’ phase, the players with the higher Elo are selected to be group heads and the rest of the group members are randomly selected. In this phase all the players compete with each other in their group, and once all matches are played the best players of each group advance to bracket’s phase. Players start the bracket’s phase without any losses, in this phase the players are
organized by pairs to compete in the brackets; they are eliminated after $k$ number of losses. At the end of the tournament, a match or matches between winner of winners’ brackets and winner of losers’ brackets determine the champion.

**Elo**

Most of sports and competitions have a rating or classification system. Arpad Elo developed a system based on statistical estimation assuming the average performance of a chess player changes slowly over time.

The ELO system is a classification method for chess players, where the players accumulate points and are classified according to their performance along their carriers. The players earn or lose points according to the result obtain in each match and the expected results for each match, which depends on their own and opponent’s previous performances.

The Performance level is calculated from wins, losses and draws. When a player wins, he has performed on a higher level than the opponent has. When he loses, he has performed on a lower level and a draw indicates that both players have performed on the same level.

The rating of a player will move up faster when the player wins against a player who has a higher rating, compared to one who against somebody who plays at the same or a lower level. A player who has been playing chess for a short period and has improved his performance has a greater chance to move higher on a faster rate compared to someone who has been played chess for a longer period.

The expected result of a match is determined by:

$$ E_a = \frac{1}{1 + 10^{(R_b - R_a)/400}} \quad (1) $$

Where $R_a$ and $R_b$ correspond to the ELO of each participant.

The ELO punctuation of each participant after each match is determined by:

$$ ELO_a = P_a + K (P_a - E_a) \quad (8) $$

Where $E_a$ as the expected punctuation of the individual and $P_a$ the final punctuation of the match achieved by the individual (0-winner, 1-looser).

**Winner selection**

Each tournament defines how to determine who wins a match, for example, in soccer the player who wins is the player who has the greatest number of goals. In the case of this new metaheuristic, the winner and loser individual for each match are selected by the Swiss system of Els. In each match two individuals compete to be the winner, the expected value for the match is defined by the next equation($E_a$):

$$ E_a = 11 + \frac{10 \times (R_b - R_a)}{400} $$

$$ E_a = 11 + \frac{10 \times (R_b - R_a)}{400} $$

$$ E_1 = E_a $$
\[ E_2 = 1 - E_1 \]

This equation defines the winning probability of each individual (\( E_1 \) and \( E_2 \)). Where \( R_A \) and \( R_B \) correspond to the fitness function of each participant, so the individual who has a higher fitness function value will be the participant with higher winning probability. Once the expected value is calculated for each player, a random number between zero and one is generated to determine the winner, thus if the random number is smaller or equal than \( E_1 \) value, the first individual is the winner, otherwise the second individual wins.

**Example of a match point.**

If the winning probability of competitors take the following values:

\( E_1 = 0.6 \) \( E_2 = 0.4 \)

If random=0.5; the random number is smaller than \( E_1 \), thus the winner is the individual 1.

If random=0.8; the random number is higher than \( E_1 \), thus the winner is the individual 2.

**Learning**

In psychology, the learning curve denotes a graphical representation of the rate at which a person makes progress learning new information. Through repetition, any person becomes more efficient and effective at a task. The progress made during the learning and repetition phases can be represented graphically in the learning curve.

Each participant of the tournament has a learning phase after each match. The time or trainings required to become more efficient and improve is denoted by the learning curve and the next equation:

\[ X = Y \frac{\log(2)}{\log(b)} \]

The \( X \) trainings needed to improve are used as the iterations and changes a participant can take to change their skills and become a better player. One training generates two random numbers, the first indicates the skill’s position that will change and the second random number indicates the new skill’s value. If a participant wins a match tries to improve by a local search, it has \( X \) trainings (number of changes), each change is evaluated, if it improves the participant’s objective function it is preserved, otherwise the change is discarded. When a participant loses a match tries to resemble the opponent by reducing the distances between him and his opponent, in this case a change is preserved if it reduces the distance respect to the opponent or if the participant’s objective function improved.

In this phase the exploration and exploitation strategies are given. The exploitation is given when the participants make changes by local search or changes orientated to reduce the difference between their rivals. Otherwise, the exploration takes place when a participant with a better fitness function loses against one with lower fitness and has to make changes orientated to reduce the distance against his rival, which may cause a decrease in the FO, given him a chance to leave local optimal areas.

**Seeding**

In order to make tournaments fair and attractive a seeding phase were developed. In the seeding method, the participants are classified based on their previous results in other tournaments, and then they are “planted” into the brackets in a way the strongest participants face each other in an advance phase. This method was designed first for Tennis tournaments; however, it was successfully use in other sports as NFL, WNBA, NBA, NCAA and MLS. Likewise, the same method is applied in E-sports as League of legends, CSGO, Call of duty and others.
For KET, the seeding consists in a previous phase in which the participants compete against each other winning or losing ELO depending on their results, then they are classified and set in groups depending the quantity of ELO earned. The best participants are set as head of the groups, and the rest of the participants are set randomly in the groups, finally the participants compete against the members of their group earning points. Depending of their results, at the end the best participants advance the brackets phase.

**Brackets phase**

The best players of each group arrive to the brackets phase and they are organized in pairs for the matches. There are several sets of brackets, the number of brackets sets is determined by the number \( k \) of eliminations of the tournament. At the beginning of this phase the number of losses of all players is zero, when a player loses a match descends to the next bracket set and earns a loss until completing \( k \) number of loses before being eliminated. When a player wins a match continues to the next round and stays in the same bracket. At the end, there is a winner in each bracket, so the winner of the \( k \) bracket faces the winner of the \( k-1 \) bracket and so on with \( k-2, k-3, \) etc., until there is a winner from losers’ bracket set who faces the winner of winners’ bracket set, and finally, matches are held until any of finalists complete \( k \) losses and there is only on winner who had not lost more than \( k-1 \) times.

### 4.1 KET Steps

**Step 1:** Initialize the tournament population.

**Step 2:** Choose the initial parameters \( k \) eliminations, coding problem and distance.

**Step 3:** Set the Round Robin Tournament.

**Step 4:** Calculate the ELO at each round between individuals.

**Step 5:** Calculate the fitness value of the individuals at the tournament.

**Step 5:** State the seeding stage with the individuals.

**Step 6:** Select the individuals at the first and second position after the group stage.

**Step 7:** Define the learning curve of the individuals for each round at the \( k \) elimination tournament.

**Step 8:** Winner competitors advance to the tournament, and those who were defeated learn and change.

**Step 9:** The \( k \) elimination tournament finishes with an individual.

### 4.2 Pseudocode and flowchart

**Parameters**

\( k = \text{Tournament Eliminations} \)

\( b = \text{change percentage} \)

\( P = \text{Initial population} \)

\( X = \text{Individual training sessions generated by the learning curve} \)

\( Y = \text{counter} \)

\( Z = \text{Individuals that advance to the brackets} \)

**Algorithm \( k \) elimination tournament**

**Input:** The \( k \) elimination tournament population, eliminations.

**Output:** The tournament winner (The individual with the best Objective function)

1: procedure: KET
2: Initialize the parameters coding problem, \( k \), \( b \) and \( P \)
3: \hspace{1cm} While (\( y<P/2 \)) do *Round Robin*
4: \hspace{2cm} For each match do
5: \hspace{3cm} Calculate ELO
6: \hspace{3cm} Set match winner and punctuation
7: \hspace{3cm} Update \( x \) training sessions
8: \hspace{3cm} Set the individual learning phase
9: \hspace{2cm} End For
10: \hspace{1cm} ‘Seeding phase’
11: \hspace{2cm} Set groups of 4 individuals
12: \hspace{2cm} Select groups leaders
13: \hspace{2cm} Fill the groups randomly

For each group do
  For each match do
    Calculate ELO
    Set match winner and punctuation
    Update x training sessions
    Set the individual learning phase
  End For
End For

Fill the brackets k elimination tournament
'Selecting the best individuals of the group'
While (y=1) do’ k elimination tournament
  For each bracket do
    Calculate ELO
    Set match winner
    Update x training sessions
    Set the individual learning phase
  End For
  For each individual do
    Update b learning percentage
  End For
  For each bracket do
    If individual = loser then
      individual elimination =individual elimination+1
    End If
  Next For
Fig 5. KET algorithm Flowchart

1. Start
2. Initialize the tournament solution
3. Outline the initial parameters
4. Set the round robin stage
5. State the groups stage
6. Elimination Tournament
7. End
8. Set the round robin stage
9. Set new positions
10. Calculate the ELO
11. Set match winner and punctuation
12. Update x training sessions
13. Set the individual learning phase
14. End round robin stage
15. State the groups stage
16. Set the groups
17. Select the group leaders (seeding)
18. Fill the groups randomly
19. Set the round robin stage by group
20. End Seeding Stage
21. Elimination Tournament
22. For x:
23. Fill the brackets
24. Individual i vs individual j
25. Individual (winn)
26. Fill the brackets, round 1, elimination x
27. Fill the brackets, round 1, elimination x+1
28. Elimination x+1
29. Individual i eliminated
30. R=1
31. End k
32. Elimination Tournament
5 Experimental results

This metaheuristic based on $k$ eliminations tournaments is a new algorithm, therefore its performance should be evaluated; thus, this section presents a comparison of the results obtained by this algorithm against the results obtained by other metaheuristics in different classical and mathematical problems. To perform the validation of this results there was used some instances of the knapsack problem, which are usually coded as binary problems; some mathematical functions, which are coded as scalar numbers; and some instances of the Travelling Salesman Problem (TSP), coded as permutations. The results obtained by this new metaheuristic to the problems mentioned were compared against the results obtained by known algorithms like genetic (GA), Particle Swarm Optimization (PSO) and Tabu search.

5.1 Benchmark test functions

Some benchmark functions were used to evaluate the performance of the proposed metaheuristic, these functions are divided into two categories: unimodal functions ($f1$-$f6$) and multimodal functions ($f7$-$f9$). Unimodal functions are characterized by having a global optimum and not having locals optimum while in multimodal functions exist multiple local solutions in addition to the global optimum [49].

Table 1. Unimodal functions graphics

<table>
<thead>
<tr>
<th>Function</th>
<th>Dim</th>
<th>Range</th>
<th>$f_{\text{min}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beale</td>
<td>5</td>
<td>[-4.5, 4.5]</td>
<td>0</td>
</tr>
<tr>
<td>Rosenbrock</td>
<td>5</td>
<td>[-5, 10]</td>
<td>0</td>
</tr>
<tr>
<td>Booth</td>
<td>5</td>
<td>[-10, 10]</td>
<td>0</td>
</tr>
<tr>
<td>Sumsquare</td>
<td>5</td>
<td>[-10, 10]</td>
<td>0</td>
</tr>
<tr>
<td>Dixon price</td>
<td>5</td>
<td>[-10, 10]</td>
<td>0</td>
</tr>
<tr>
<td>Sphere</td>
<td>5</td>
<td>[-5.12, 5.12]</td>
<td>0</td>
</tr>
<tr>
<td>Griewank</td>
<td>5</td>
<td>[-600, 600]</td>
<td>0</td>
</tr>
<tr>
<td>Rastrigin</td>
<td>5</td>
<td>[-5.12, 5.12]</td>
<td>0</td>
</tr>
<tr>
<td>Schwefel</td>
<td>5</td>
<td>[-500, 500]</td>
<td>0</td>
</tr>
</tbody>
</table>

This new metaheuristic is compared by running a 4-elimination tournament where 1000 participants compete to be the best, the quantity of participants wasn’t bigger due to the computational memory used. The table 2 shows the parameters used for programming each algorithm for the mathematical functions. Table 1 shows the dimension and range limits of the search space for each function; this table also indicates the minimum value found for each function.
Table 2. Parameter table Metaheuristics

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tabu</td>
<td>Iterations</td>
<td>1000</td>
</tr>
<tr>
<td>GA</td>
<td>Population size</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Number of generations</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>Mutation coefficient</td>
<td>5%</td>
</tr>
<tr>
<td>PSO</td>
<td>Inertia Coefficient</td>
<td>0.75</td>
</tr>
<tr>
<td></td>
<td>Number of particles</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>Number of generations</td>
<td>1000</td>
</tr>
<tr>
<td>KET</td>
<td>K eliminations</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Number of participants</td>
<td>1000</td>
</tr>
</tbody>
</table>

5.1.1 Unimodal functions

Table 3 shows the plots of the unimodal functions (f1-f6). These functions have just one global optimum, the absence of local optimum at unimodal functions represents great utility to evaluate the exploitation capacity of metaheuristic algorithms. To measure this characteristic on the proposed metaheuristic were implemented six unimodal functions, which are the Rosenbrock, Dixon-Price, Beale, Sphere, Sum square and Booth function [68].

Table 3. Unimodal functions plots

<table>
<thead>
<tr>
<th>Unimodal functions</th>
<th>Unimodal functions</th>
<th>Unimodal functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="f1. Beale plot" /></td>
<td><img src="image" alt="f2. Rosembrock plot" /></td>
<td><img src="image" alt="f5. Dixon Price plot" /></td>
</tr>
<tr>
<td><img src="image" alt="f3. Booth plot" /></td>
<td><img src="image" alt="f4. Sum square plot" /></td>
<td><img src="image" alt="f6. Sphere plot" /></td>
</tr>
</tbody>
</table>

To validate the performance of the proposed metaheuristic its results on unimodal functions are compared against the results obtained by the Genetic Algorithm, Tabu search and PSO. The table 4 shows the results obtained on unimodal functions.
Table 4 shows that even if the proposed algorithm (KET) does not reach the minimum value for these functions, it is a competitive algorithm since the results obtained by KET are closed to the optimum values. This new metaheuristic has a good exploration capability and beat the Generic Algorithm in the Beale and Sumsquare functions (f1 and f4). PSO algorithm had the best performance for these functions.

Table 4. Unimodal functions results

<table>
<thead>
<tr>
<th>Function</th>
<th>Name</th>
<th>Best</th>
<th>GA</th>
<th>Tabu</th>
<th>PSO</th>
<th>KET</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Beale</td>
<td>0</td>
<td>0.131</td>
<td>75.680</td>
<td>0.040</td>
<td>0.080</td>
</tr>
<tr>
<td>2</td>
<td>Rosenbrock</td>
<td>0</td>
<td>0.006</td>
<td>58.210</td>
<td>0.020</td>
<td>0.001</td>
</tr>
<tr>
<td>3</td>
<td>Booth</td>
<td>0</td>
<td>0.014</td>
<td>30.966</td>
<td>0.060</td>
<td>0.003</td>
</tr>
<tr>
<td>4</td>
<td>Sumsquare</td>
<td>0</td>
<td>0.106</td>
<td>80.250</td>
<td>0.040</td>
<td>0.001</td>
</tr>
<tr>
<td>5</td>
<td>Dixon price</td>
<td>0</td>
<td>0.033</td>
<td>44.749</td>
<td>0.140</td>
<td>0.006</td>
</tr>
<tr>
<td>6</td>
<td>Sphere</td>
<td>0</td>
<td>0.007</td>
<td>4.126</td>
<td>0.080</td>
<td>0.005</td>
</tr>
</tbody>
</table>

5.1.2 Multimodal functions

Multimodal functions include many local optima, unimodal mathematical functions are implemented on the proposed metaheuristic to measure its exploration capacity (f7-f9), in which they are the Griewank, Rastrigin and Schwefel functions [81]. These types of functions were selected because they have several local optima.

Table 5. Multimodal functions plots

<table>
<thead>
<tr>
<th>Multimodal functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>f7. Griewank</td>
</tr>
<tr>
<td>f8. Rastrigin</td>
</tr>
<tr>
<td>f9. Schwefel</td>
</tr>
</tbody>
</table>

The table 6 shows the results obtained for the functions (f7-f9) for the metaheuristic proposed and the classical algorithms.

The results found for these functions show that the KET algorithm has a good performance on the Rastrigin (f8), in this function, KET had better results than the Genetic Algorithm and Tabu Search. Considering the
performance of the proposed metaheuristic for unimodal and multimodal functions, it shows that KET has a better capacity of exploitation rather than exploration.

Table 6. Multimodal functions results

<table>
<thead>
<tr>
<th>Function</th>
<th>Name</th>
<th>Genético</th>
<th>Tabu</th>
<th>PSO</th>
<th>KET</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best</td>
<td>Avg</td>
<td>T(s)</td>
<td>Best</td>
<td>T(s)</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0.030</td>
<td>0.440</td>
<td>0.900</td>
<td>0.007</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0.020</td>
<td>17.310</td>
<td>0.170</td>
<td>0.496</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0.027</td>
<td>68.390</td>
<td>0.300</td>
<td>0.950</td>
</tr>
</tbody>
</table>

5.1.3 Convergence

The aim of the convergence analysis is to have a better understanding about the KET’s exploration and exploitation capacity. Fig 6 shows the converge curves for the functions: Beale and SumSquare. KET has a fast convergence during the first iterations and as the iterations run the algorithm leads to the optimum point in a slower haste and it reaches a good answer. This convergence behavior represents a good balance between the exploration and exploitation of the KET algorithm for these problems.
5.2 Knapsack problem

The knapsack problem is based on decision-making, this problem describes a situation on which the person must decide what items to carry in a knapsack, to maximize the benefit carried without overpassing its limit capacity. This problem has been applied in a wide variety of fields. Being a binary optimization problem, it is considered a NP-Hard problem because of its complexity. Therefore, delve into its study is a challenge that in recent years has led to find new solution methods for real problems [78].

Below are the parameters of the instances of the knapsack problem that were programmed to evaluate the performance of this new metaheuristic.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tabu</td>
<td>Iterations</td>
<td>1000</td>
</tr>
<tr>
<td>GA</td>
<td>Population size</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Number of generations</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>Mutation coefficient</td>
<td>5%</td>
</tr>
<tr>
<td>PSO</td>
<td>Inertia Coefficient</td>
<td>0.75</td>
</tr>
<tr>
<td></td>
<td>Number of particles</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>Number of generations</td>
<td>1000</td>
</tr>
<tr>
<td>KET</td>
<td>K eliminations</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Number of participants</td>
<td>1000</td>
</tr>
</tbody>
</table>

The first 10 functions with which the performance of metaheuristics will be compared are of small instances, the following five functions are instances that are more robust. For the development of these instances, three parameters are considered: capacity (W), weight (w) and benefit (v) [79]. The performance was evaluated among Tabu Search, Genetic Algorithm, and PSO and k eliminations tournament metaheuristics.
Table 8. Knapsack problem results

<table>
<thead>
<tr>
<th>Function</th>
<th>Dim</th>
<th>GMPL</th>
<th>GA</th>
<th>Tabu</th>
<th>PSO</th>
<th>KET</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Best</td>
<td>T(s)</td>
<td>Best</td>
<td>Gap</td>
<td>Best</td>
</tr>
<tr>
<td>F1</td>
<td>10</td>
<td>295</td>
<td>5.00</td>
<td>295</td>
<td>206.80</td>
<td>0.00%</td>
</tr>
<tr>
<td>F2</td>
<td>20</td>
<td>5568</td>
<td>2.43</td>
<td>5548</td>
<td>3165.14</td>
<td>0.36%</td>
</tr>
<tr>
<td>F3</td>
<td>4</td>
<td>35</td>
<td>0.59</td>
<td>35</td>
<td>26.80</td>
<td>0.00%</td>
</tr>
<tr>
<td>F4</td>
<td>4</td>
<td>41</td>
<td>1.23</td>
<td>41</td>
<td>31.80</td>
<td>0.00%</td>
</tr>
<tr>
<td>F5</td>
<td>15</td>
<td>516.1</td>
<td>2.77</td>
<td>516</td>
<td>402.40</td>
<td>0.02%</td>
</tr>
<tr>
<td>F6</td>
<td>10</td>
<td>52</td>
<td>0.35</td>
<td>52</td>
<td>37.60</td>
<td>0.00%</td>
</tr>
<tr>
<td>F7</td>
<td>7</td>
<td>107</td>
<td>0.35</td>
<td>107</td>
<td>74.60</td>
<td>0.00%</td>
</tr>
<tr>
<td>F8</td>
<td>23</td>
<td>9767</td>
<td>0.36</td>
<td>9767</td>
<td>8376.00</td>
<td>0.00%</td>
</tr>
<tr>
<td>F9</td>
<td>5</td>
<td>130</td>
<td>0.35</td>
<td>130</td>
<td>99.70</td>
<td>0.00%</td>
</tr>
<tr>
<td>F10</td>
<td>20</td>
<td>4668</td>
<td>0.25</td>
<td>4634</td>
<td>4304.70</td>
<td>0.73%</td>
</tr>
<tr>
<td>F11</td>
<td>100</td>
<td>21987</td>
<td>0.58</td>
<td>21841</td>
<td>9267.80</td>
<td>0.66%</td>
</tr>
<tr>
<td>F12</td>
<td>200</td>
<td>45189</td>
<td>0.67</td>
<td>44960</td>
<td>31815.80</td>
<td>0.51%</td>
</tr>
<tr>
<td>F13</td>
<td>500</td>
<td>130611</td>
<td>4.90</td>
<td>129908</td>
<td>87839.37</td>
<td>0.54%</td>
</tr>
<tr>
<td>F14</td>
<td>1000</td>
<td>285810</td>
<td>3.78</td>
<td>284380</td>
<td>208490.40</td>
<td>0.50%</td>
</tr>
<tr>
<td>F15</td>
<td>2000</td>
<td>508680</td>
<td>3.78</td>
<td>505844</td>
<td>303009.00</td>
<td>0.56%</td>
</tr>
</tbody>
</table>

Table 8 shows the results obtain by the 15 instances of the Knapsack problem. KET has a great performance on this group of instances considering that it reaches the optimum value on 7 instances (F1, F3, F4, F5, F6, F7 and F9). In addition, this new metaheuristic obtains the best or equal results for the following instances against the Genetic Algorithm: F1, F4, F5, F6, F7, F10, F11, and F15; it beat or equals PSO in the instances F1, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11, F13 and F14, meanwhile it beats Tabu Search on the instance F1, F2, F3, F4, F5, F6, F7, F8, F9 and F15.

5.2.1 Convergence

Fig 7 shows the converge curves for the instances: F6 and F11. For instances F6 and F11 KET algorithm has a fast convergence during the first iterations and as the iterations run the algorithm leads to the optimum point in a slower haste. The instance F15 has a different convergence behavior, it has a fast convergence, it represents an efficient exploitation capacity.
5.3 The Travel Salesman Problem

The Travel Salesman Problem (TSP) is a NP-Hard problem which tries to find the best way to decrease the distance traveled visiting a set of nodes with established distances. The start point of the route may be any node, and from this point is searched the best permutation that generates the lowest objective function [80]. Twenty independent runs were made for 15 instances to evaluate the performance of the Genetic Algorithm, Tabu Search, PSO and the $k$ eliminations tournament.
### Table 9. TSP results

<table>
<thead>
<tr>
<th>Instance</th>
<th>Dim</th>
<th>FO</th>
<th>Best T(s)</th>
<th>Gap</th>
<th>Best T(s)</th>
<th>Gap</th>
<th>Best T(s)</th>
<th>Gap</th>
<th>Best T(s)</th>
<th>Gap</th>
<th>Best T(s)</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eil51</td>
<td>51</td>
<td>426</td>
<td>457.170</td>
<td>7.317%</td>
<td>450.260</td>
<td>5.695%</td>
<td>479.430</td>
<td>12.542%</td>
<td>463.100</td>
<td>0.990</td>
<td>463.100</td>
<td>0.990</td>
</tr>
<tr>
<td>Eil76</td>
<td>71</td>
<td>538</td>
<td>610.445</td>
<td>12.164%</td>
<td>569.550</td>
<td>5.864%</td>
<td>563.497</td>
<td>2.430</td>
<td>12.467%</td>
<td>10.854%</td>
<td>12.467%</td>
<td>10.854%</td>
</tr>
<tr>
<td>Rat195</td>
<td>52</td>
<td>2323</td>
<td>2731.624</td>
<td>17.590%</td>
<td>2708.144</td>
<td>16.580%</td>
<td>2639.640</td>
<td>13.647%</td>
<td>2612.625</td>
<td>12.467%</td>
<td>2612.625</td>
<td>12.467%</td>
</tr>
<tr>
<td>Kroa100</td>
<td>100</td>
<td>20749</td>
<td>22574.144</td>
<td>47.810</td>
<td>22471.985</td>
<td>24.350</td>
<td>22691.240</td>
<td>36.239</td>
<td>16509.339</td>
<td>3.019</td>
<td>16509.339</td>
<td>3.019</td>
</tr>
<tr>
<td>Kroa100</td>
<td>100</td>
<td>21282</td>
<td>24802.960</td>
<td>33.290</td>
<td>22744.373</td>
<td>28.090</td>
<td>23137.510</td>
<td>35.981</td>
<td>23592.184</td>
<td>13.520</td>
<td>23592.184</td>
<td>13.520</td>
</tr>
<tr>
<td>St70</td>
<td>70</td>
<td>675</td>
<td>785.300</td>
<td>6.540</td>
<td>742.583</td>
<td>8.290</td>
<td>717.530</td>
<td>10.012%</td>
<td>696.732</td>
<td>4.141%</td>
<td>696.732</td>
<td>4.141%</td>
</tr>
</tbody>
</table>

### Table 10. Parameter table Metaheuristics

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tabu</td>
<td>Iterations</td>
<td>1000</td>
</tr>
<tr>
<td>GA</td>
<td>Population size</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>Number of generations</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Mutation coefficient</td>
<td>5%</td>
</tr>
<tr>
<td>PSO</td>
<td>Inertia Coefficient</td>
<td>0.75</td>
</tr>
<tr>
<td></td>
<td>Number of particles</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>Number of generations</td>
<td>1000</td>
</tr>
<tr>
<td>KET</td>
<td>Number of tournaments</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Number of participants</td>
<td>1000</td>
</tr>
</tbody>
</table>

### Performance comparison

As an important objective, the evaluation of the metaheuristic for some TSP instances is crucial. The instances Eil51-St70 focused at the exploitation capability for metaheuristics which focused at permutations problems. At Table 9 KET efficiency surface at St70 and Eil76 which represent 23.6% as the better objective function evaluated whit the metaheuristics, the Eil51 instance has the best result with the Tabu search. The instances Kroa100, Kroc100, Lin105 focused at the exploration and exploitation performance of a metaheuristic, the instances are evaluated as medium capacity where the KET has 8% gap between the best answer founded by the algorithms. The Rat195 instance is the larger instance reviewed and the KET has the best objective function compared with the other metaheuristics.

#### 5.3.1 Convergence

The next graphs show that KET had a similar performance through the seven instances that had been evaluated, the Particle Swarm Optimization and the Tabu show a balance between exploration and exploitation, on the other hand the genetic algorithm had few changes what implies that for this problem the metaheuristic has not performed as well as in the other problems purposed. KET had the best result for 42% of the instances.
as the Tabu Search that had the same performance. KET has an exploitation performance where the objective function rate is progressive until it achieves the best answer recorded.

5.4 KET performance

Table 11 shows the percentages of instances according each problem where KET algorithm get the optimum answer, it also shows the percentages where KET beats the classic metaheuristics and the min/max gap for each problem.
Table 11. KET results comparison

<table>
<thead>
<tr>
<th>Problem</th>
<th>Optimum</th>
<th>Better or equal than GA</th>
<th>Better or equal than Tabu</th>
<th>Better or equal than PSO</th>
<th>Min gap</th>
<th>Max gap</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Unimodal functions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>33.333%</td>
<td>-</td>
<td>-</td>
<td>0.002</td>
<td>0.359</td>
</tr>
<tr>
<td><strong>Multimodal Functions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>33.333%</td>
<td>33.333%</td>
<td>-</td>
<td>0.005</td>
<td>1.750</td>
</tr>
<tr>
<td>Knapsack</td>
<td>46.667%</td>
<td>53.333%</td>
<td>66.666%</td>
<td>86.666%</td>
<td>0.000%</td>
<td>0.600%</td>
</tr>
<tr>
<td>TSP</td>
<td>-</td>
<td>66.666%</td>
<td>50.000%</td>
<td>66.666%</td>
<td>4.141%</td>
<td>14.816%</td>
</tr>
</tbody>
</table>

5.5 ANOVA

In order to understand how the parameters could affect the quality of the results of the metaheuristic, a statistical study was made with the help of the ANOVA test. The factors set for the ANOVA test are: participant quantity, elimination quantity and the instances used, this last one was blocked due to the significance of the instance on every problem. The sample taken for the study took up to 5 items per level. The ANOVA test showed that the participant and elimination quantity along with the iteration were significant.

Table 12. statistic parameters

<table>
<thead>
<tr>
<th>Factors</th>
<th>Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Instances</td>
</tr>
<tr>
<td></td>
<td>F14 F13 F12 F11</td>
</tr>
<tr>
<td></td>
<td>Eil51 Eil56 St70</td>
</tr>
<tr>
<td></td>
<td>Beale Rosenbrok</td>
</tr>
<tr>
<td>B</td>
<td>Individuals</td>
</tr>
<tr>
<td></td>
<td>4 16 64 256</td>
</tr>
<tr>
<td>C</td>
<td>Eliminations</td>
</tr>
<tr>
<td></td>
<td>2 3 4 5</td>
</tr>
<tr>
<td>N</td>
<td>Observations</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Alpha</td>
<td>0.05</td>
</tr>
</tbody>
</table>

The Tukey test shows that with a confidence of 95% for the KP problem and TSP problem, the best combination of parameters is among 64 and 1024 participants and among 4 and 5 eliminations, the same test shows that for numerical problems the best combination of parameters are among 16 and 1024 participants and among 3 and 5 eliminations.
5.6 Sensitivity analysis

**TSP Performance**

![TSP Performance Graph]

**Numeric Performance**

![Numeric Performance Graph]

**Knapsack Performance**

![Knapsack Performance Graph]
As could be seen from the graphs above, whilst the number of participants arises, the quality of the FO too.
However, the graphs show that in a specific quantity of participant the value of the FO does not improve, stagnates or improve just for units or less.

The ANOVA analysis also says that for more than 64 participants and 4 or 5 eliminations, the changes in the parameter participants does not affect the quality of the solutions, in other words it does not improve the solutions found.

6 Conclusions and recommendations

The k elimination tournament metaheuristic inspired on the confrontations structures was proposed to solve nine test functions, fifteen knapsack instances and seven TSP instances to analyze the exploitation, exploration and convergence behavior. Where was found that the best combination of parameters is among 64 and 1024 participants and among 4 and 5 eliminations. Also, it was found that the KET algorithm has a better performance solving problems which codification is binary, like knapsack, in this problem the KET results were compared with the PSO, GA and TS. KET algorithm reaches the optimum value for the 46.66% of the instances group and get competitive results for all the instances.

The convergence of the Metaheuristic shows that it has a good tuning for its exploitation and exploration since the algorithm reaches some of the optimum values of the instances and reaches good answers within 1000 participants and 4 elimination, and even that due to its better performance on unimodal functions compared to multimodal functions indicates that it has a greater capacity of intensification rather than diversification.

As future investigations on the development of the KET metaheuristic, solving other optimization problems could be done, as the multi-objective problems. Also, the program could be translated to other languages were the computational capacity will not affect the metaheuristics performance as insufficient memory. Thus, because due the quantity of participants and eliminations implicates sometimes a big computational cost.

References


