

RELEGATION BASED LEAGUE CHAMPIONSHIP ALGORITHM

*A thesis submitted in partial fulfillment of
the requirement for the award of the degree of*

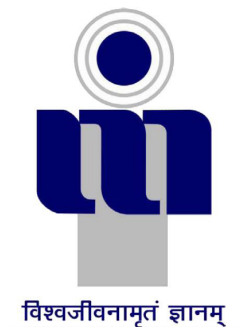
Bachelor of Technology
in
Computer Science and Engineering

by

Sandarbh Yadav
(2017BCS-027)

under the supervision of

Prof. Pramod Kumar Singh



**ABV INDIAN INSTITUTE OF INFORMATION
TECHNOLOGY AND MANAGEMENT
GWALIOR-474 015**

2021

THESIS CERTIFICATE

I hereby certify that the work, which is being presented in the report, entitled **REL-EGATION BASED LEAGUE CHAMPIONSHIP ALGORITHM**, in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** and submitted to the institute is an authentic record of my own work carried out during the period *Jan 2021 to May 2021* under the supervision of **Prof. Pramod Kumar Singh**. I have also cited the references about the table(s)/figure(s) from where they have been taken.

Name: Sandarbh Yadav

Roll No.: 2017BCS-027

Date: 28th May 2021

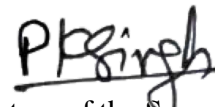


Signature of the Candidate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Name: Prof. Pramod Kumar Singh

Date: 29th May 2021



Signature of the Supervisor

The final copy of this thesis has been examined by the signatories and we find that both the content and the form meet acceptable presentable standards of scholarly work in the above mentioned discipline.

CANDIDATE'S DECLARATION

I hereby declare that I have properly checked and verified all the items as prescribed in the check-list and ensure that my thesis is in the proper format as specified in the guideline for thesis preparation.

I declare that the work contained in this report is my own work. I understand that plagiarism is defined as any one or combination of the following:

1. To steal and pass off (the ideas or words of another) as one's own
2. To use another's production without crediting the source
3. To commit literary theft
4. To present as new and original idea or product derived from an existing source.

I understand that plagiarism involves an intentional act by the plagiarist of using someone else's work/ideas completely/partially and claiming authorship/originality of the work/ideas. Verbatim copy as well as close resemblance to some else's work constitute plagiarism.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results and websites that are not my original contribution.

I affirm that no portion of my work is plagiarized and the reported results are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable. My faculty supervisor will not be responsible for the same.

Name: Sandarbh Yadav

Roll No.: 2017BCS-027

Date: 28th May 2021



Signature of the Candidate

ACKNOWLEDGEMENTS

I am very thankful to **Prof. Pramod Kumar Singh** who provided me the liberty to experiment various ideas. I am highly indebted to him for guiding me and showing high amount of interest in my work. He motivated me and boosted my confidence. This proved beneficial for me and infused trust within me. This project has been completed only because of his precious suggestions, intelligent judgement and fair criticism. My mentor resolved my each and every doubt and I never felt as beginner. He listened to my opinions and appreciated and improved them. The only reason my work has been completed successfully is due to his constant support and helpful attitude.

Lastly, I thank my companions and the institute for their continuously uplifting which helped in renewing my spirits, refocus my attention and helping me in taking this project.

Sandarbh Yadav

ABSTRACT

Classical optimisation techniques often prove insufficient for large scale combinatorial problems and non-linear problems. Consequently, heuristic based optimisation techniques have been introduced. Recently, a novel metaheuristic named league championship algorithm (LCA) has been proposed for global optimisation in continuous search space. LCA is a population based algorithm which mimics a league championship with a fixed number of teams. These teams compete against each other according to a pre-determined schedule and the winner is determined on the basis of playing strength of the teams. The formations of the teams keep on improving throughout the season and at the end an optimal solution is obtained. Since, its inception, LCA has been employed in solving many constrained optimisation problems. However, there are few drawbacks of LCA in form of premature convergence and slow convergence rate. This project attempts to overcome the shortcomings of LCA by incorporating the concept of relegation. In this project, relegation based LCA has been proposed. Comparative experiments have been conducted on 10 different test functions and promising results are obtained.

Keywords: Optimization, Metaheuristic, League Championship Algorithm, Relegation

TABLE OF CONTENTS

ABSTRACT	iv
LIST OF TABLES	vii
LIST OF FIGURES	viii
1 Introduction	1
1.1 Motivation	2
1.2 Objectives	2
1.3 Report Layout	2
2 Literature Review	3
2.1 Background	3
2.2 Key Research Work	3
2.3 Research Gaps	4
2.4 Problem Statement Formulation	4
3 Methodology	5
3.1 Analogy between Sports Leagues & Metaheuristics	5
3.1.1 League & Population	5
3.1.2 Week & Iteration	5
3.1.3 Team & Solution	5
3.1.4 Playing Strength & Fitness Value	6
3.1.5 Match Analysis & Updation	6
3.2 Assumptions	6
3.3 Flow Chart	7
3.3.1 League Schedule Generation	8
3.3.2 Determining Winner & Loser	8
3.3.3 Updating the Formations	9
3.4 Proposed Improvement	10
3.5 Parameters of the Algorithm	11
3.5.1 League Size (L)	11

3.5.2	Number of Seasons (S)	11
3.5.3	Number of Teams to be Relegated (R)	11
3.5.4	Number of Players in a Team (n)	11
3.5.5	Scale Coefficients (c_1 & c_2)	11
3.5.6	Control Parameter (p_c)	12
3.6	Proposed Algorithm	12
3.7	Tools and Technologies Used	13
3.7.1	Python	13
3.7.2	Jupyter Notebook	13
3.7.3	Popular Python Libraries & Modules	13
4	Results and Discussion	14
4.1	Test Functions	14
4.1.1	Ackley Function	15
4.1.2	Alpine Function	15
4.1.3	Griewank Function	16
4.1.4	Rastrigin Function	16
4.1.5	Rosenbrock Function	17
4.1.6	Schwefel Function	17
4.1.7	Sphere Function	18
4.1.8	Sum of Different Powers Function	18
4.1.9	Sum Squares Function	19
4.1.10	Zakharov Function	19
4.2	Experimental Results	20
4.3	Verification of Results	22
5	Conclusion	24
5.1	Future Scope	24
	REFERENCES	25

LIST OF TABLES

3.1	Analogy between Sports League and Metaheuristic	6
3.2	SWOT Analysis	9
4.1	Properties of Test Functions	20
4.2	Values of Parameters	20
4.3	Experimental Results	22
4.4	Values of t statistic	23

LIST OF FIGURES

3.1	Flow Chart	7
3.2	League Schedule Generation	8
3.3	Usage of Python, Matlab and R over the years	13
3.4	Jupyter Notebook	13
4.1	Ackley Test Function	15
4.2	Alpine Test Function	15
4.3	Griewank Test Function	16
4.4	Rastrigin Test Function	16
4.5	Rosenbrock Test Function	17
4.6	Schwefel Test Function	17
4.7	Sphere Test Function	18
4.8	Sum of Different Powers Test Function	18
4.9	Sum Squares Test Function	19
4.10	Zakharov Test Function	19
4.11	Convergence Graphs	21

ABBREVIATIONS

ACO	Ant Colony Optimization
ALCA	Adaptive League Championship Algorithm
CLCA	Chaotic League Championship Algorithm
DE	Differential Evolution
EA	Evolutionary Algorithm
GA	Genetic Algorithm
LCA	League Championship Algorithm
PSO	Particle Swarm Optimization
RLCA	Realistic League Championship Algorithm
SA	Simulated Annealing
SI	Swarm Intelligence
SWOT	Strength Weakness Opportunity Threat

CHAPTER 1

Introduction

An algorithm is a precise step by step plan for a computational procedure that transforms the input into a output in a finite number of steps. Since the emergence of algorithmic framework, many algorithms have been proposed. A special class of algorithms termed metaheuristics [1] are inspired from naturally occurring phenomena or behaviour of human beings and are used to solve many optimisation problems. Most of the metaheuristics involve randomness by making use of random numbers in the computational procedure. Metaheuristics are broadly divided into 2 categories: nature inspired algorithms and metaphor based metaheuristics. Nature inspired algorithms take inspiration from naturally occurring phenomena. Nature inspired algorithms are of 2 types: evolutionary algorithms and swarm intelligence algorithms. Evolutionary algorithms (EA) take inspiration from biological evolution of species. Genetic algorithm (GA) [2] and differential evolution (DE) [3] are 2 popular evolutionary algorithms. Swarm intelligence (SI) algorithms mimic the behaviour of group of living organisms. Some popular swarm intelligence metaheuristics include particle swarm optimisation (PSO) [4] and ant colony optimisation (ACO) [5]. Metaphor based metaheuristics are inspired from artificial processes and have resulted in highly effective solutions for many problems. Simulated annealing (SA) [6] is one popular example.

Apart from the extensive research going on to solve various problems using already existing metaheuristics, occasionally a new metaheuristic is introduced which solves optimisation problems by using a novel metaphor as guide. League championship algorithm (LCA) is one such metaheuristic which is proposed recently by Ali Kashan [7]. It is inspired by artificial sports leagues where a fixed number of teams compete against each other and at the end of every season, the best team is crowned as winner.

1.1 Motivation

The motivation for the proposed algorithm came by observing the concept of relegation in sports leagues. In established leagues, there are multiple divisions in a hierarchical manner. A fixed number of teams, generally 20, compete at each division for a season. After the end of every season, based on rankings of the teams, the worst 3 teams from each division (except the topmost division) are relegated to below division and best 3 teams from each division (except the lowest division) are promoted to the above division. This concept of relegation can be integrated in the basic LCA in which worst 3 solutions after every season can be discarded and replaced by newly generated random solutions.

1.2 Objectives

The main objectives of this project are:

1. To incorporate concept of relegation in LCA
2. Develop a fully functional python code of the proposed algorithm
3. Compare the performance of proposed algorithm with LCA
4. Statistically prove the findings

1.3 Report Layout

The report comprises of 5 chapters. Chapter 1 introduces the project. Chapter 2 gives the literature review. Methodology is covered in chapter 3. Results and discussions form the content of chapter 4. Chapter 5 concludes the project. References are given in the end.

CHAPTER 2

Literature Review

This chapter reviews the literature, discusses the research gaps and formulates the problem statement.

2.1 Background

The league championship algorithm (LCA) was introduced by Kashan [7]. He took inspiration from sports leagues and proposed LCA for global optimisation in continuous search space. He also compared the performances of LCA with particle swarm optimisation (PSO) algorithm on 5 benchmark functions: Sphere, Ackley, Rastrigin, Rosenbrock and Schwefel function. LCA gave better performances than PSO on all the benchmark functions. However, the basic version of LCA suffers from drawbacks like premature convergence and slow convergence speed. Exploration and exploitation are imbalanced which results in LCA getting stuck in local optima.

2.2 Key Research Work

Researchers have incorporated different strategies in an attempt to improve LCA. In 2012, Kashan et al [8] revamped LCA by incorporating halftime analysis. The modified version, termed realistic league championship algorithm (RLCA), was compared with LCA and PSO on 5 benchmark functions. Although RLCA performed better than PSO on all 5 benchmark functions, the difference between performances of RLCA and LCA was not very significant. Bingol et al [9] introduced chaotic league championship algorithm (CLCA) by incorporating concepts of chaos theory. 6 different versions of CLCAs were proposed based on usage of chaotic maps instead of random number arrays. Although, out of 6 versions, 2 provided better performance than LCA, there was no analytical result which guaranteed improvement in performance of LCA. Gade et al

[10] proposed adaptive league championship algorithm (ALCA) with improved learning rate and applied it for independent task scheduling in cloud computing.

Since its introduction, LCA has been used successfully to solve many complex constrained optimisation problems. Apart from optimisation problems, LCA has been extended to other applications, especially scheduling problems. Kashan [11] used LCA in mechanical engineering design. Pourali et al [12] used LCA for industrial optimisation. Sun et al [13] introduced a resource allocation scheme using LCA. Lenin et al [14] employed LCA to solve reactive power dispatch problem. Hamid et al [15] used LCA to propose a job scheduling scheme for IaaS cloud. Sajadi et al [16] used LCA to solve permutation flow shop scheduling problem. Jalili et al [17] used LCA for optimum design of pin jointed structures. Subbaraj et al [18] used LCA to solve real time task scheduling problem. Xu et al [19] used LCA as parameter learning method in neural networks and used them for scheduling in shop floor production. Rezaei et al [20] provide a detailed explanation of LCA with pseudo code. Abdulhamid et al [21] provide a detailed survey of LCA algorithm in literature.

2.3 Research Gaps

Since the introduction of LCA, researchers have used it to solve different optimisation problems in various fields. However, little effort has been made to improve the LCA itself. LCA suffers from issues of premature convergence and slow convergence rates. Exploration and exploitation are imbalanced which results in LCA getting stuck in local optima. Kashan [8] incorporated the concept of two halves analysis but the improvement in performance was not very significant. Incorporation of concepts of chaos theory [9] resulted in improvement in some cases but there was no analytical result guaranteeing improvement in performance of LCA. Thus, there is scope for improvement in LCA.

2.4 Problem Statement Formulation

Although LCA has been used to solve many optimisation problems successfully, there is still scope for improvement in it as it suffers from issues related to premature convergence and slow convergence rate. Ali Kashan, proposer of LCA, also mentioned in his research paper that it is possible to develop improved variants of LCA by incorporating different concepts. Hence, there is scope for improvement in LCA and this project attempts to improve LCA.

CHAPTER 3

Methodology

This chapter gives an overview of LCA algorithm and discusses the proposed improvement in it.

3.1 Analogy between Sports Leagues & Metaheuristics

LCA mimics sports leagues and hence there is a direct analogy between the concepts of sports leagues and metaheuristics. The following terms are equivalent to each other:

3.1.1 League & Population

In sports, league refers to a group of teams. These teams compete against each other for a season and in the end, a winner is crowned. A season generally starts in August and ends in April. League competitions are organised every season. In population based metaheuristics, the algorithm begins with a group of possible solutions. This is in stark contrast to single point based metaheuristics which begin with exactly one solution. The teams comprising the league are equivalent to members of the population in metaheuristic. A team typically consists of 11 players. This number is analogous to the dimensionality of search space.

3.1.2 Week & Iteration

A season comprises of many weeks. In each week, every team plays one game. The concept of week in sports is analogous to concept of iteration in metaheuristics.

3.1.3 Team & Solution

A team in a league is equivalent to a solution in the population. In sports like football, hockey etc, teams play using a formation which varies over the season depending upon

players, coach, opponents etc. The continuously changing formation of a team over the weeks is analogous to dynamically changing values of a solution across the iterations.

3.1.4 Playing Strength & Fitness Value

In a match between 2 teams, the winner is decided by playing strength. A strong team is likely to win against a weak team. In a similar manner, fitness value decides which solution is better.

3.1.5 Match Analysis & Updation

After the match ends, the coaching staff perform an analysis to improve the team based on the outcome of the match. This concept is equivalent to updation of solutions after every iteration.

Sports Term	Metaheuristic Term
League	Population
Week	Iteration
Team	Solution
Playing Strength	Fitness Value
Match Analysis	Updation

Table 3.1: Analogy between Sports League and Metaheuristic

3.2 Assumptions

LCA makes some assumptions which are as follows:

1. A team with higher playing strength is likely to beat a team with lower playing strength. However, sometimes there are upsets in which a team with lower playing strength might emerge victorious against a team with higher playing strength.
2. A game results in a win or a loss. The concept of ties is ignored.
3. Match analysis is performed keeping only next match in consideration. The upcoming matches, after the next match, are not taken into account. Formations are set by considering events of previous week.
4. Weakness is equivalent to lack of particular strength.

3.3 Flow Chart

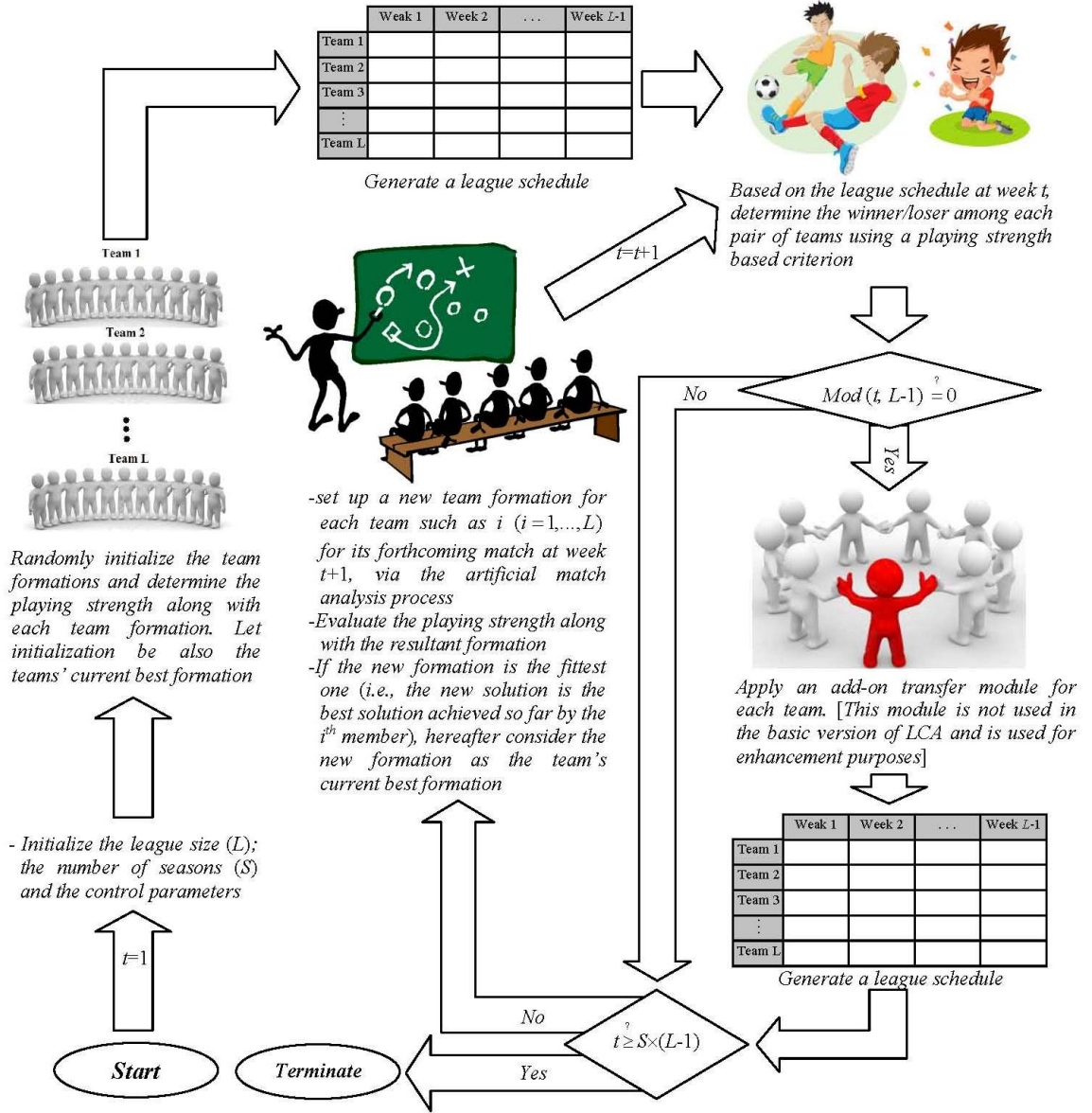


Figure 3.1: Flow Chart [22]

Being a population based metaheuristic, LCA employs a population of solutions. All the solutions are represented by n dimensional vectors. Here, n denotes the dimensionality of the search space. In terms of sports, n denotes the number of players in a team. Team i 's formation at week t is denoted by $X_i^t = (x_{i1}^t, x_{i2}^t, \dots, x_{in}^t)$. The fitness value of X_i^t is denoted by $f(X_i^t)$ where f denotes the fitness function. $B_i^t = (b_{i1}^t, b_{i2}^t, \dots, b_{in}^t)$ denotes team i 's best formation encountered until week t . B_i^t is determined by performing a greedy selection between $f(X_i^t)$ & $f(B_i^{t-1})$ at each iteration: If $f(X_i^t)$ is better than $f(B_i^{t-1})$, B_i^t is replaced by X_i^t , else B_i^t is replaced by B_i^{t-1} .

3.3.1 League Schedule Generation

In LCA, a league tournament is organised in which each team faces every other team exactly once. In a league comprising of L teams, each team participates in $L-1$ matches scattered over $L-1$ weeks. Thus, the total number of matches in a season are $L(L-1)/2$ as $L/2$ matches are held in each of the $L-1$ weeks. The schedule can be generated by keeping one team fixed (here team denoted by 1) and rotating the other teams in a clockwise manner as shown below.

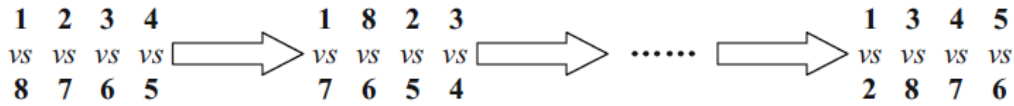


Figure 3.2: League schedule generation for 8 teams

In LCA, the competition is organised for S seasons resulting in $S(L-1)$ iterations.

3.3.2 Determining Winner & Loser

In sports, a team with higher playing strength is likely to beat a team with lower playing strength. However, sometimes there are upsets in which a team with lower playing strength emerges victorious against a team with higher playing strength. In order to capture this concept, the winner is decided by binary tournament selection based on the fitness values of the teams. Let p_i^t and p_j^t denote the winning probabilities of team i and team j respectively in a match against each other at week t . Clearly, the sum of p_i^t and p_j^t is equal to 1 as concepts of ties is ignored. Let f^* denote the optimal value of fitness function f . Since f^* is not known beforehand, it is assumed as $\min\{f(B_1^t), f(B_2^t), \dots, f(B_L^t)\}$ for week t in case of a minimisation problem. p_i^t can be determined using following equations:

$$\frac{f(X_i^t) - f^*}{f(X_j^t) - f^*} = \frac{p_j^t}{p_i^t} \quad (3.3.1)$$

$$p_i^t = \frac{f(X_j^t) - f^*}{f(X_i^t) + f(X_j^t) - 2f^*} \quad (3.3.2)$$

For determining the result of the match, we generate a random number from a uniform distribution between 0 and 1. If the generated random number is lesser than or equal to p_i^t , team i wins else team j wins.

3.3.3 Updating the Formations

SWOT analysis is performed to update the formations of the teams. SWOT is the abbreviation for strength, weakness, opportunity and threat. Strength and weakness are viewed as internal factors whereas opportunity and threat are viewed as external factors. Internal factors are conceived from analysis of previous game of the team itself while external factors are sensed from previous game of the opponent. Suppose team i won/lost against team j at week t . The win/loss is a direct result of its own strength/weakness or an indirect result of weakness/strength of team j . Suppose, team i faces team l in week $t+1$ and team l won/lost against team k at week t . The formation behind this win/loss can serve as threat/opportunity for team i . By concentrating on strength/weakness of team l , team i can avoid threats/take advantage of opportunities. Based on assumption 4, team i can indirectly concentrate on weakness/strength of team k . Table 3.2 summarizes the actions of team i based on outcomes of previous week.

Outcomes of matches	Team l won against team k	Team l lost against team k
Team i won against team j	S/T strategy Focus on own strength (or weakness of team j) and strength of team l (or weakness of team k)	S/O strategy Focus on own strength (or weakness of team j) and weakness of team l (or strength of team k)
Team i lost against team j	W/T strategy Focus on own weakness (or strength of team j) and strength of team l (or weakness of team k)	W/O strategy Focus on own weakness (or strength of team j) and weakness of team l (or strength of team k)

Table 3.2: Actions of team i based on outcomes of previous week

Gap analysis is performed to derive the update equations. $X_k^t - X_i^t$ denotes the gap between formations of team k and team i determined by concentrating on strengths of team k . Suppose, team k defeated team l at week t . Then a formation similar to team k can be beneficial for team i . Similarly, $X_i^t - X_k^t$ denotes the gap between formations of team i and team k determined by concentrating on weaknesses of team k . Suppose team l defeated team k at week t . Then a formation similar to team k might prove harmful for team i . Formations of team i at week $t+1$ can be derived using following update equations:

If team i and team l were winners,

$$x_{id}^{t+1} = b_{id}^t + y_{id}^t \{c_1 r_1 (x_{id}^t - x_{kd}^t) + c_1 r_2 (x_{id}^t - x_{jd}^t)\} \quad (3.3.3)$$

If team i and team k were winners,

$$x_{id}^{t+1} = b_{id}^t + y_{id}^t \{c_2 r_1 (x_{kd}^t - x_{id}^t) + c_1 r_2 (x_{id}^t - x_{jd}^t)\} \quad (3.3.4)$$

If team j and team l were winners,

$$x_{id}^{t+1} = b_{id}^t + y_{id}^t \{c_1 r_2 (x_{id}^t - x_{kd}^t) + c_2 r_1 (x_{jd}^t - x_{id}^t)\} \quad (3.3.5)$$

If team j and team k were winners,

$$x_{id}^{t+1} = b_{id}^t + y_{id}^t \{c_2 r_2 (x_{kd}^t - x_{id}^t) + c_2 r_1 (x_{jd}^t - x_{id}^t)\} \quad (3.3.6)$$

In the above equations, d denote the dimension of solution vector and can take integer values from 1 to n. r_1 and r_2 denote random numbers obtained from a uniform distribution between 0 and 1. c_1 and c_2 are coefficients which scale the weights of strength and weakness component respectively. y_{id}^t is a binary variable indicating whether there will be change in d^{th} element or not. A value of 1 means that value of d^{th} element will be changed while a value of 0 means that there will be no change. Let $Y_i^t = (y_{i1}^t, y_{i2}^t, \dots, y_{in}^t)$ be the binary change vector with number of 1's equal to q_i^t . Generally, the number of changes done by the coach is less. Hence, q_i^t should take a small value. The value of q_i^t is simulated using a truncated geometric distribution as given below:

$$q_i^t = \left\lceil \frac{\ln(1 - (1 - (1 - p_c)^n)r)}{\ln(1 - p_c)} \right\rceil \quad (3.3.7)$$

Here, r is a random number generated from a uniform distribution between 0 and 1 while p_c is a control parameter in range (0,1) which controls the value of q_i^t . The larger the value of p_c , smaller the value of q_i^t . After calculating the number of changes by above equation, q_i^t elements are randomly selected from B_{id}^t and their values are changed according to the update equations.

3.4 Proposed Improvement

Since LCA suffers from issues related to premature convergence and slow convergence rate, this project attempts to overcome the shortcomings of LCA by incorporating concept of relegation. In established leagues, there are multiple divisions in a hierarchical

manner. At each division, a fixed number of teams, generally 20, compete for a season. After the end of every season, based on rankings of the teams, the worst 3 teams from each division (except the topmost division) are relegated to below division and best 3 teams from each division (except the lowest division) are promoted to the above division. This concept of relegation is integrated in the basic LCA in which worst 3 solutions after every season are discarded and replaced by newly generated random solutions. The proposed algorithm is named relegation based league championship algorithm.

3.5 Parameters of the Algorithm

Relegation based LCA incorporates all the parameters of basic LCA. Additionally, it introduces a new parameter representing number of teams to be relegated at the end of each season. The parameters are discussed below:

3.5.1 League Size (L)

League size refers to the number of teams in the league. In metaheuristic terminology, this is equivalent to size of the population.

3.5.2 Number of Seasons (S)

S denotes the number of seasons for which league championship is held. It provides a termination condition in form of maximum number of iterations equalling $(L-1)*S$.

3.5.3 Number of Teams to be Relegated (R)

R denotes the number of teams which are relegated after each season. Generally, it is taken as 0.15 times of total number of teams. Thus, in a 20 team league, 3 teams are relegated each season.

3.5.4 Number of Players in a Team (n)

n denotes the number of players in a team. It basically denotes the dimensionality of the space in which optimization has to be done. The solution vectors have n dimensions.

3.5.5 Scale Coefficients (c_1 & c_2)

c_1 and c_2 are scale coefficients which determine how much strength and weakness component contribute to the updation.

3.5.6 Control Parameter (p_c)

p_c controls how many changes are done to a team's formation during updation process. To ensure a small number of changes, p_c should take a small value.

3.6 Proposed Algorithm

The pseudo code of the proposed algorithm is given below. The proposed improvement is highlighted in blue.

Algorithm 1: Relegation based League Championship Algorithm

Input: Algorithm Parameters

L: league size
S: number of seasons
R: number of teams to be relegated
n: dimension of search space
 c_1 & c_2 : scale coefficients
 p_c : control parameter

Output: Optimal value of objective function

Initialize team formations by randomly generating population of L solutions;

for $i = 1$ to L **do**

 Evaluate playing strength;
 Set the current formation as best formation;

end

Generate schedule of league matches;

iteration = 1;

while iteration $\leq (L - 1) * S$ **do**

 Determine the results of each match according to league schedule;

for $i = 1$ to L **do**

 Update the team formation using Eqs. 3.3.3 to 3.3.6;
 Determine the best formation by greedy selection;

end

if iteration % $(L-1) = 0$ **then**

 Relegate R worst performing teams by replacing them with newly
 generated random solutions;

end

 iteration = iteration + 1;

end

3.7 Tools and Technologies Used

3.7.1 Python

Most of the work related to LCA has been done in Matlab but Python has emerged as the language for the future. Python is a programming language which is known for its simplicity and widespread adoption [23]. Hence, Python is used as the programming language for this project.

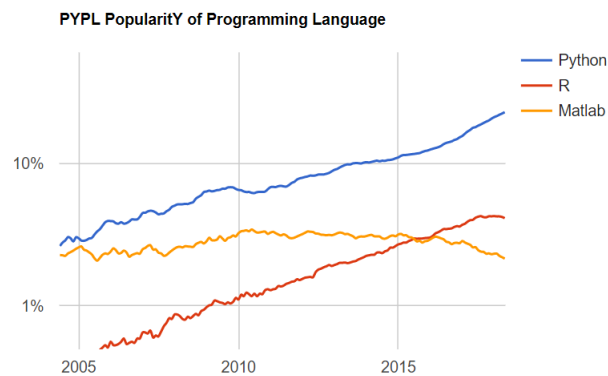


Figure 3.3: Usage of Python, Matlab and R over the years

3.7.2 Jupyter Notebook

Jupyter Notebook is the framework used for developing this project.



Figure 3.4: Jupyter Notebook

3.7.3 Popular Python Libraries & Modules

Numpy is used for keeping track of solutions in form of arrays. Random module is used for generating random numbers. Matplotlib is used for plotting graphs.

CHAPTER 4

Results and Discussion

This chapter discusses the test functions used and the experimental results obtained.

4.1 Test Functions

Test functions [24] are functions which are used to evaluate the effectiveness of optimisation algorithms. Test functions are generally described on the basis of 4 properties:

1. **Convexity:** Convexity refers to whether the function is convex or not. Some functions are convex in nature while others are non-convex.
2. **Differentiability:** Differentiability refers to whether a function is mathematically differentiable or not.
3. **Modality:** Some functions have only one optima and are known as unimodal functions. On the other hand, multimodal functions have many local optima apart from the global optima.
4. **Separability:** Separability signifies the difficulty of optimising a test function. Separable functions are easier to optimise compared to non-separable functions.

There are a huge number of test functions available in literature. In order to maintain a rich diversity, 10 different test functions are used: Ackley, Alpine, Griewank, Rastrigin, Rosenbrock, Schwefel, Sphere, Sum of different powers, Sum squares and Zakharov function. All these functions are n-dimensional and have global optima value as 0. The detailed description of these functions is given below. Table 4.1 summarizes the properties of test functions [25].

4.1.1 Ackley Function

Ackley function is a widely used test function. It is non-convex and multimodal. It is differentiable but non-separable in nature. Its formula is given below:

$$f(\mathbf{x}) = -20 \exp \left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2} \right) - \exp \left(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i) \right) + 20 + e \quad (4.1.1)$$

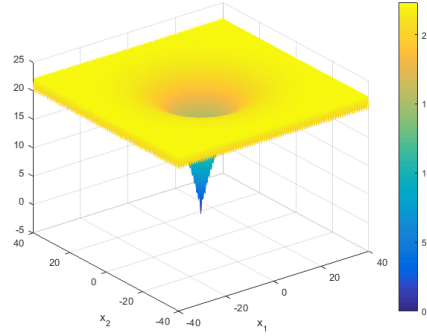


Figure 4.1: Ackley function for n = 2

4.1.2 Alpine Function

Like Ackley function, Alpine function is also a non-convex test function. It is multimodal with many local optima. However, it is non-differentiable and separable in nature. Its formula is given below:

$$f(\mathbf{x}) = \sum_{i=1}^n |x_i \sin(x_i) + 0.1x_i| \quad (4.1.2)$$

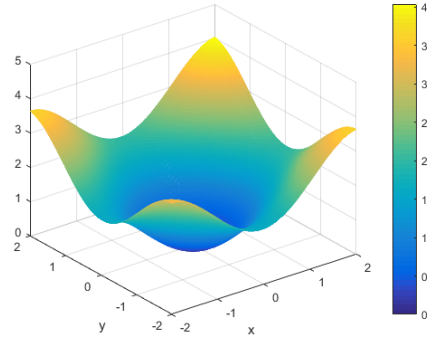


Figure 4.2: Alpine function for n = 2

4.1.3 Griewank Function

Griewank function is a non-convex test function. Like Ackley and Alpine function, it is multimodal with many local optima. It is differentiable but non-separable in nature. Its formula is given below:

$$f(\mathbf{x}) = 1 + \sum_{i=1}^n \frac{x_i^2}{4000} - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) \quad (4.1.3)$$

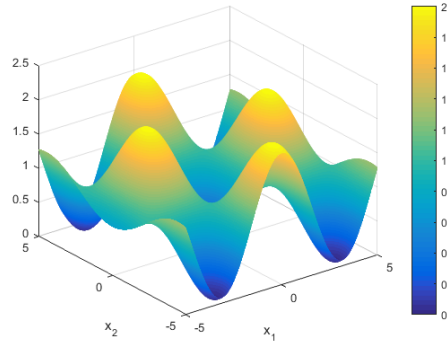


Figure 4.3: Griewank function for n = 2

4.1.4 Rastrigin Function

Rastrigin function comprises of many local optima. It is non-convex, differentiable and separable in nature. It is given by the following formula:

$$f(\mathbf{x}) = \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i) + 10) \quad (4.1.4)$$

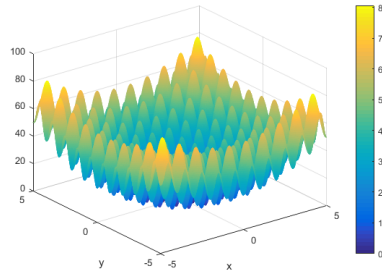


Figure 4.4: Rastrigin function for n = 2

4.1.5 Rosenbrock Function

The global optima in Rosenbrock function lies within a long and narrow valley making convergence difficult. It is non-convex, differentiable and non-separable in nature. It is given by the following formula:

$$f(\mathbf{x}) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2] \quad (4.1.5)$$

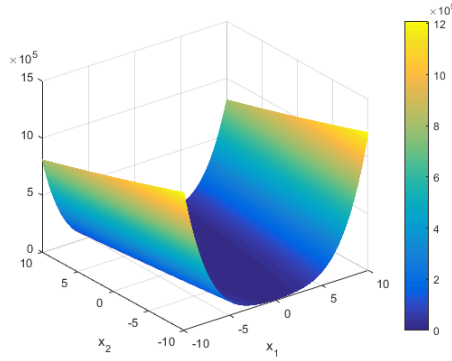


Figure 4.5: Rosenbrock function for n = 2

4.1.6 Schwefel Function

Schwefel function is a widely used test function. It is non-convex and multimodal. It is non-differentiable and partially separable. It is given by the following formula:

$$f(\mathbf{x}) = 418.9829n - \sum_{i=1}^n x_i \sin(\sqrt{|x_i|}) \quad (4.1.6)$$

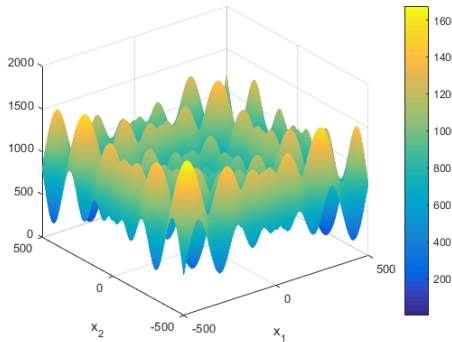


Figure 4.6: Schwefel function for n = 2

4.1.7 Sphere Function

Sphere function is a convex test function. It is unimodal with no local optima. It is differentiable and separable in nature. It is given by the following formula:

$$f(\mathbf{x}) = \sum_{i=1}^n x_i^2 \quad (4.1.7)$$

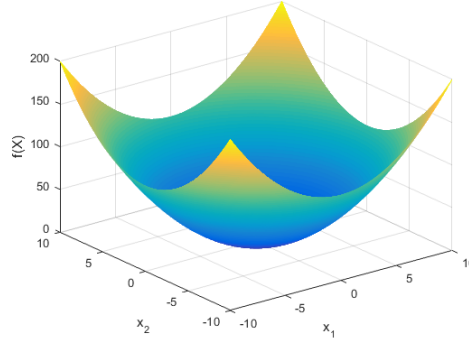


Figure 4.7: Sphere function for n = 2

4.1.8 Sum of Different Powers Function

Like sphere function, sum of different powers function is a convex test function. It is unimodal with no local optima. It is non-differentiable but separable in nature. It is given by the following formula:

$$f(\mathbf{x}) = \sum_{i=1}^n |x_i|^{i+1} \quad (4.1.8)$$

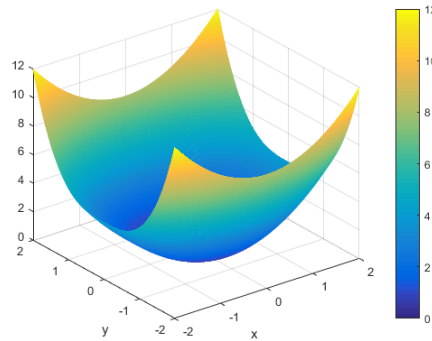


Figure 4.8: Sum of Different Powers function for n = 2

4.1.9 Sum Squares Function

Sum squares function is convex, differentiable, unimodal and separable in nature. It is given by the following formula:

$$f(\mathbf{x}) = \sum_{i=1}^n ix_i^2 \quad (4.1.9)$$

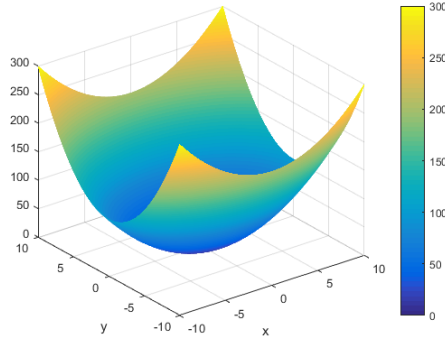


Figure 4.9: Sum Squares function for n = 2

4.1.10 Zakharov Function

Zakharov function is a convex test function. It is unimodal with no local optima. It is differentiable but non-separable in nature. It is given by the following formula:

$$f(\mathbf{x}) = \sum_{i=1}^n x_i^2 + \left(\sum_{i=1}^n 0.5ix_i \right)^2 + \left(\sum_{i=1}^n 0.5ix_i \right)^4 \quad (4.1.10)$$

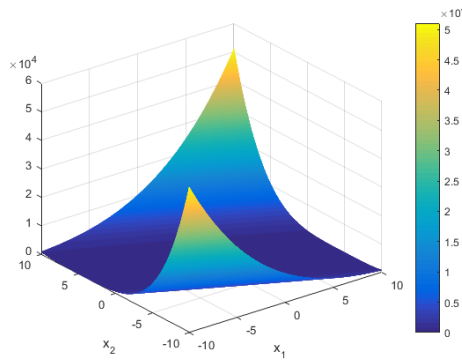


Figure 4.10: Zakharov function for n = 2

Test Function	Convexity	Differentiability	Modality	Separability
Ackley	Non-convex	Differentiable	Multimodal	Non-separable
Alpine	Non-convex	Non-differentiable	Multimodal	Separable
Griewank	Non-convex	Differentiable	Multimodal	Non-separable
Rastrigin	Non-convex	Differentiable	Multimodal	Separable
Rosenbrock	Non-convex	Differentiable	Unimodal	Non-separable
Schwefel	Non-convex	Non-differentiable	Multimodal	Partially separable
Sphere	Convex	Differentiable	Unimodal	Separable
Sum of Different Powers	Convex	Non-differentiable	Unimodal	Separable
Sum Squares	Convex	Differentiable	Unimodal	Separable
Zakharov	Convex	Differentiable	Unimodal	Non-separable

Table 4.1: Properties of Test Functions

4.2 Experimental Results

Comparison is done between basic LCA and relegation based LCA on 10 different test functions. Table 4.2 gives the values of the algorithm parameters. The league size (L) is taken as 20 which means the population comprises of 20 solutions. Number of seasons (S) is taken as 50. The number of teams to be relegated is taken as 3. Experiments are done in a search space of dimension $n = 10$. Scale coefficients c_1 and c_2 are set as 0.2 and 1.0 respectively while 0.3 is the value allocated to control parameter p_c . Both algorithms are run on same parameter values so that a fair comparison can be made between them in similar environment. Values of algorithm parameters do not matter significantly as long as it is ensured that both algorithms run on same parameter values.

Parameter	Value
L	20
S	50
R	3
n	10
c_1	0.2
c_2	1.0
p_c	0.3

Table 4.2: Values of Parameters

30 different runs of both algorithms are performed for each of the 10 test functions and comparative study is done on their performances. Figure 4.11 gives a visual representation of convergence of both algorithms for each test function during the 1st of the 30 runs. Table 4.3 summarizes the comparative results between basic LCA and relegation based LCA, obtained during the 30 different runs.

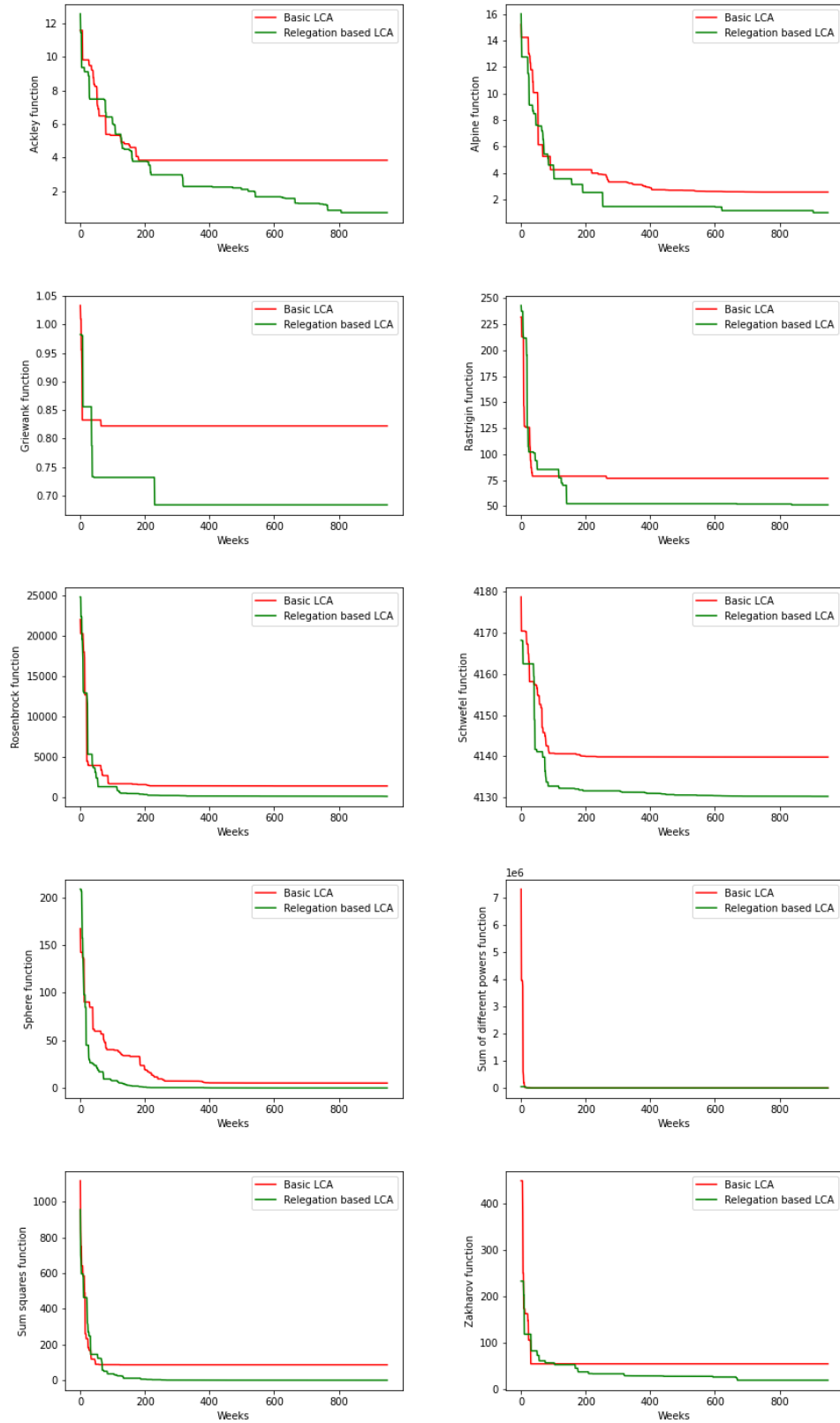


Figure 4.11: Convergence of Basic LCA and Relegation based LCA for each test function during first run

Test Function	Basic LCA			Relegation based LCA		
	Mean	Standard Deviation	Best	Mean	Standard Deviation	Best
Ackley	4.110765	1.514951	1.014089	1.858368	0.900234	0.182607
Alpine	2.553196	1.616669	0.121907	1.636277	0.898516	0.012698
Griewank	0.621392	0.216047	0.071970	0.458808	0.169806	0.064532
Rastrigin	47.065508	13.150035	17.6604587	38.748249	8.333133	15.967540
Rosenbrock	921.046607	780.905291	63.136866	128.369312	96.091490	7.160220
Schwefel	4130.866378	20.002376	4063.200173	4099.816674	28.852650	3994.254356
Sphere	9.112696	9.131957	0.243558	0.096699	0.374367	0.000454
Sum of Different Powers	99.829557	140.792658	0.000198	0.003878	0.0147568	0.000000
Sum Squares	33.099359	30.191493	2.386982	0.357874	1.031760	0.005929
Zakharov	75.045158	44.809268	27.451777	30.775599	23.465757	1.272265

Table 4.3: Experimental results obtained during the 30 different runs

4.3 Verification of Results

As observed from Table 4.3, relegation based LCA results in better function values compared to basic LCA. Moreover, the fluctuations in values are also less as signified by relatively lower values of standard deviations. Statistical test based on t statistic is performed to verify the results.

t statistic is defined as:

$$t = \frac{(\bar{x}_1 - \bar{x}_2) - (\mu_1 - \mu_2)}{\sqrt{\frac{s_1^2(n_1-1) + s_2^2(n_2-1)}{n_1+n_2-2}} \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \quad (4.3.1)$$

with degree of freedom $df = n_1+n_2-2$.

Here, \bar{x}_1 & \bar{x}_2 denote sample means; μ_1 & μ_2 denote population means; s_1 & s_2 denote sample standard deviations and n_1 & n_2 denote sample sizes. Null and alternate hypotheses are defined as:

$$H_0 : \mu_1 \geq \mu_2 \quad (4.3.2)$$

$$H_A : \mu_1 < \mu_2 \quad (4.3.3)$$

μ_1 corresponds to basic LCA while μ_2 corresponds to relegation based LCA. As test functions are getting minimised, null hypothesis should be accepted. The condition for null hypothesis to be accepted is:

$$t > t_\alpha \quad (4.3.4)$$

For level of significance $\alpha = 0.01$, the value of t_α is 2.392. The calculated values of t statistic for each test function are given in table 4.4.

Test Function	t statistic
Ackley	7.001
Alpine	2.715
Griewank	3.241
Rastrigin	2.926
Rosenbrock	5.518
Schwefel	4.844
Sphere	5.403
Sum of Different Powers	3.883
Sum Squares	5.936
Zakharov	4.794

Table 4.4: Values of t statistic for each test function

Clearly, the condition for accepting null hypothesis is satisfied for all 10 test functions. Hence, it has been verified statistically, with 99% confidence, that relegation based LCA provides better results than basic LCA.

CHAPTER 5

Conclusion

LCA is a metaheuristic which performs global optimisation in continuous search space. It is inspired by artificial sports leagues where a fixed number of teams compete against each other and at the end of the season, best team is crowned as winner. Although LCA has been used to solve many optimisation problems successfully, there is still scope for improvement in it as it suffers from issues related to premature convergence and slow convergence rate. In this project, LCA has been improved by integrating concept of relegation. Experiments have been conducted on 10 different test functions and the results indicate that relegation based LCA performs better than basic LCA. The results have been verified statistically using hypothesis test based on t statistic.

5.1 Future Scope

Although it has been statistically verified that relegation based LCA performs better than the basic LCA, there is scope of some improvement in case of complex test functions like Schwefel and Rosenbrock function. Also, efforts can be made to study the impact of number of relegated teams on performance of relegation based LCA.

REFERENCES

- [1] A. H. Gandomi, X.-S. Yang, S. Talatahari, and A. H. Alavi, “Metaheuristic algorithms in modeling and optimization,” *Metaheuristic applications in structures and infrastructures*, pp. 1–24, 2013.
- [2] D. Whitley, “A genetic algorithm tutorial,” *Statistics and computing*, vol. 4, no. 2, pp. 65–85, 1994.
- [3] K. V. Price, “Differential evolution,” in *Handbook of optimization*, pp. 187–214, Springer, 2013.
- [4] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of ICNN’95-international conference on neural networks*, vol. 4, pp. 1942–1948, IEEE, 1995.
- [5] M. Dorigo, M. Birattari, and T. Stutzle, “Ant colony optimization,” *IEEE computational intelligence magazine*, vol. 1, no. 4, pp. 28–39, 2006.
- [6] P. J. Van Laarhoven and E. H. Aarts, “Simulated annealing,” in *Simulated annealing: Theory and applications*, pp. 7–15, Springer, 1987.
- [7] A. H. Kashan, “League championship algorithm: a new algorithm for numerical function optimization,” in *Soft Computing and Pattern Recognition, 2009. SOCPAR’09. International Conference of*, pp. 43–48, IEEE, 2009.
- [8] A. H. Kashan, S. Karimiyan, M. Karimiyan, and M. H. Kashan, “A modified league championship algorithm for numerical function optimization via artificial modeling of the “between two halves analysis”,” in *The 6th international conference on soft computing and intelligent systems, and the 13th international symposium on advanced intelligence systems*, pp. 1944–1949, IEEE, 2012.
- [9] H. Bingol and B. Alatas, “Chaotic league championship algorithms,” *Arabian Journal for Science and Engineering*, vol. 41, no. 12, pp. 5123–5147, 2016.
- [10] A. Gade, N. B. Mundukur, and N. Thakare, “Adaptive league championship algorithm (alca) for independent task scheduling in cloud computing.,” *Ingénierie des Systèmes d Inf.*, vol. 24, no. 3, pp. 353–359, 2019.

- [11] A. H. Kashan, “An efficient algorithm for constrained global optimization and application to mechanical engineering design: League championship algorithm (lca),” *Computer-Aided Design*, vol. 43, no. 12, pp. 1769–1792, 2011.
- [12] Z. Pourali and M. Aminnayeri, “A novel discrete league championship algorithm for minimizing earliness/tardiness penalties with distinct due dates and batch delivery consideration,” in *International Conference on Intelligent Computing*, pp. 139–146, Springer, 2011.
- [13] J. Sun, X. Wang, K. Li, C. Wu, M. Huang, and X. Wang, “An auction and league championship algorithm based resource allocation mechanism for distributed cloud,” in *International Workshop on Advanced Parallel Processing Technologies*, pp. 334–346, Springer, 2013.
- [14] K. Lenin, B. Reddy, and M. Kalavathi, “League championship algorithm (lca) for solving optimal reactive power dispatch problem,” *International Journal of Computer and Information Technologies*, vol. 1, no. 3, pp. 254–272, 2013.
- [15] S. M. Abdulhamid and M. S. A. Latiff, “League championship algorithm based job scheduling scheme for infrastructure as a service cloud,” *arXiv preprint arXiv:1410.2208*, 2014.
- [16] S. M. Sajadi, A. H. Kashan, and S. Khaledan, “A new approach for permutation flow-shop scheduling problem using league championship algorithm,” *Proceedings of CIE44 and IMSS*, vol. 14, p. 2014, 2014.
- [17] S. Jalili, A. H. Kashan, and Y. Hosseinzadeh, “League championship algorithms for optimum design of pin-jointed structures,” *Journal of Computing in Civil Engineering*, vol. 31, no. 2, p. 04016048, 2017.
- [18] S. Subbaraj, R. Thiagarajan, and M. Rengaraj, “Multi-objective league championship algorithm for real-time task scheduling,” *Neural Computing and Applications*, pp. 1–12, 2019.
- [19] W. Xu, R. Wang, and J. Yang, “An improved league championship algorithm with free search and its application on production scheduling,” *Journal of Intelligent Manufacturing*, vol. 29, no. 1, pp. 165–174, 2018.
- [20] H. Rezaei, O. Bozorg-Haddad, and X. Chu, “League championship algorithm (lca),” in *Advanced Optimization by Nature-Inspired Algorithms*, pp. 19–30, Springer, 2018.

- [21] S. M. Abdulhamid, M. S. A. Latiff, S. H. H. Madni, and O. Oluwafemi, "A survey of league championship algorithm: prospects and challenges," *arXiv preprint arXiv:1603.09728*, 2015.
- [22] <https://drkashan.ir/>.
- [23] <https://blog.revolutionanalytics.com/2018/06/pypl-programming-language-trends.html>.
- [24] https://en.wikipedia.org/wiki/Test_functions_for_optimization.
- [25] <https://benchmarkfcns.xyz/fcns>.