A New Algorithm for Constrained Optimization Inspired by the Sport League Championships

Ali Husseinzadeh Kashan, Behrooz Karimi

***Abstract*— inspired from the competition of sport teams in a sport league, an algorithm is presented for solving constrained optimization problems. A number of individuals (solutions) as sport teams compete in an artificial league for several weeks (iterations). Based on the league schedule in each week, teams play in pairs and their game outcome is determined in terms of win or loss, given known the playing strength (fitness value) along with the teams' intended formations. Modeling an artificial match analysis, each team devises a new formation/ playing strategy (a new solution) for the next week contest and this process is repeated for a number of seasons (stopping condition). Performance of the proposed algorithm is measured using test functions from a well-known benchmark commonly adopted to validate new constraint-handling techniques/ algorithms. Results obtained by the proposed approach are very competitive with respect to other comparator algorithms already developed for constrained optimization and testify that the new algorithm can be regarded as an efficient tool for optimization in the presence of constraints.**

I. Introduction

Due to frequent appearance in the real world, solving constrained optimization problems, especially nonlinear optimization problems, are of scientists' great interests in recent decades. Structural optimization, engineering design, VLSI design etc, are just a few fields in which constrained optimization problems are met. A general constrained optimization problem (CO) can be defined as follows:

$$\begin{aligned} \text{(CO): Minimize} \quad & f(X) \\ & g_j(X) \le 0, \qquad j = 1,...,q \\ & h_j(X) = 0, \qquad j = q+1,...,m \\ & L \le X \le U \end{aligned} \tag{1}$$

where $X = (x_1, x_2, ..., x_n) \in \Re^n$ is an n-dimensional decision vector, $L = (l_1, l_2, ..., l_n)$ and $U = (u_1, u_2, ..., u_n)$ are the lower bound and the upper bound vectors of X, respectively. $f(X)$ is a numerical function to be minimized, $g_j(X) \le 0, j = 1,...,q$ are q inequality constraints and $h_j(X) = 0, j = q+1,...,m$ are m-q equality constraints. The feasible region in which every point satisfies all constraints is denoted with F and E is the whole search space in which every point satisfies the upper and lower boundaries; therefore $F \subseteq E$. At any point $X \in F$, the constraint g_k that satisfies $g_k(X) = 0$ is called the active constraint at X. All equality constraints h_j are active at all points of F [1].

There are several deterministic algorithms which are efficient to solve constraint optimization problems, such as recursive quadratic programming, projection method and the generalized reduced gradient method [2]. However, efficiency of these methods is under assumptions of differentiability and continuity of the objective functions which may rarely met in real world applications. Beside deterministic algorithms, metaheuristics, e.g., evolution strategies, particle swarm optimization, differential evolution, ant colony optimization etc, which are stochastic optimization techniques do not require any assumption on the objective function. However, such methods lack a mechanism able to bias efficiently the search towards the feasible region in constrained search spaces. To cover this deficiency, a considerable amount of research has been devoted and a wide variety of approaches have been suggested in the last few years to handle the constraints efficiently during the search [3], [4].

Inspired from the natural and social phenomena, metaheuristic algorithms have attracted many researchers from various fields of science in recent years. This interest is by far in applying the existing meta-heuristics for solving real word optimization problems in many fields such as business, industry, engineering, etc. However, beside all of these applications, occasionally a new metaheuristic algorithm is introduced that uses a novel metaphor as guide for solving optimization problems. The League Championship Algorithm (LCA) is a novel algorithm designed based on the metaphor of sporting competitions in sport leagues [5]. LCA can be metaphorically overviewed as follows: a number of individuals making role as sport teams compete in an artificial league for several weeks (iterations). Based on the league schedule in each week, teams play in pairs and their game outcome is determined in terms of win or loss given the known playing strength (fitness value) along with the particular team formation/arrangement (solution) followed by each team. Keeping track of the previous week events, each team devises the required changes in its formation/playing style (a new solution is generated) for the next week contest and the championship goes on for a number of seasons (stopping condition). The

The first author is with the Department of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran. e-mail: a.kashani@aut.ac.ir

The second author is with the Department of Industrial Engineering, Amirkabir University of Technology, Tehran, Iran. e-mail: b.karimi@aut.ac.ir

way in which a new solution associated to an LCA's team is generated is governed via imitating the match analysis followed typically by coaches to design a suitable arrangement for their forthcoming match. In a typical match analysis, coaches modify their arrangement on the basis of their own game experiences and their opponent's style of play.

Following our successful adaptation of LCA to solve unconstrained numerical optimization problems [5], in this paper we investigate the application of LCA on solving constrained optimization problems. The reminder of the paper is organized as follows. In section II we introduce LCA and give a detailed report of its algorithmic features for optimizing an unconstrained numerical function. Section III describes how LCA is adapted to solve constrained optimization problems. The experiments performed and the results obtained are shown in section IV. In this section a benchmark of 13 constrained optimization problems are adopted and performance of LCA is compared with other rivals. Finally section V concludes the paper.

II. The League Championship Algorithm (LCA)

Let us first have a look on the terminology related to team games, especially those terms which will be used metaphorically in LCA.

A *sports league* is an organization that exists to provide a regulated competition for a number of people to compete in a specific sport. League is generally used to refer to competitions involving team sports, not individual sports. A league championship may be contested in a number of ways. Each team may play every other team a certain number of times. In such a set-up, the team with the best record becomes champion, based on either a strict win-loss-tie system or on a points system where a certain number of points are awarded for a win, loss, or tie, while bonus points might also be added for teams meeting various criteria [6].

Generally each team has a playing style which is realized during the game via team formation. *Formations* are a method of positioning players on the pitch to allow a team to play according to their pre-set tactics. For example, the most common formations in soccer are variations of 4-4-2, 4-3-3, 3-2-3-2, 5-3-2 and 4-5-1 [7]. Usually each team has a *best formation* which is often related to the type of players available to the coach.

It is vital for a sport team to devise suitable game plans and formations for every match. After each match, coaches analyze their own game and their next opponent game to plan on how they can develop a style of play to improve on their weaknesses or afford more on their strengths. The Analysis also includes the evaluation of opportunities and threats that comes along with the unique dynamics of the team. This kind of *match analysis* is typically known as *strengths/weaknesses/opportunities/threats* (*SWOT*) analysis, which explicitly links internal (strengths and weaknesses) and external factors (opportunities and threats).

The SWOT analysis provides a structured approach to conduct the *gap analysis*. A gap is sometimes spoken of as "the space between where we are and where we want to be". When the process of identifying gaps includes a deep analysis of the factors that have created the current state of the team, the groundwork has been laid for improvement planning. The gap analysis process can be used to ensure that the improvement process does not jump from identification of problem areas to proposed solutions without understanding the conditions that created the current state of the team.

We can match the above terms to the standard evolutionary terms as follows: "league" stands for the population of solutions; "team *i*" stands for the i^{th} solution in the population; "week" stands for "iteration"; "playing strength" stands for the "objective/fitness function value" and "a new formation" stands for "a new solution". In the reminder of the paper we use both terminologies, alternately.

As a pseudo-evolutionary algorithm, the selection in LCA is the greedy selection which consists of always replacing the legacy formation, recognized as the best formation, with a more productive team formation having better playing strength. The algorithm terminates after a certain number of "seasons" (S) in which each season comprises *L*-1 weeks, yielding $S \times (L-1)$ weeks of contests.

Before giving details of LCA, we first put forward several idealized rules which define the characteristics of artificial championship modeled by LCA.

1) It is more likely that a team with better playing strength wins the game. The term "playing strength" refers to a team's ability to beat another team.
2) The outcome of the game is not predictable given known playing strength of the teams perfectly. In other words, it is not unlikely that FC Barcelona loss the game to Sandoghe_Zakhire_Robat_Karim from Iranian 3rd division.
3) The probability that team *i* beats team *j* is assumed equal from both teams point of view.
4) The outcome of the game is only win or loss; there is no tie.
5) When team *i* beats team *j*, any strength helped team *i* to win has a dual weakness caused team *j* to loss. In other words, any weakness is a lack of a particular strength. An implicit implication of this rule is that although the match outcome may be imputed to chance, technical staff may not believe it.
6) Teams focus only on their upcoming contest without regards of the other future matches. Formation settings are done just based on the previous week events.

The following algorithm gives the basic steps of LCA for optimizing an unconstrained numerical function.

The League championship algorithm (LCA)

1. *Initialize the league size* (*L*) *and the number of seasons* (*S*)*; t*=1*;*
2. *Randomly initialize the team formations* (*generate a population of L solutions*) *and determine the playing strength* (*objective function value*) *along with each team formation. Let initialization be also the teams' current best formation;*
3. *Generate a league schedule;*
4. *Based on the league schedule at week t, determine the winner/loser among each pair of teams using a playing strength based criterion;*
5. *If mod*(*t*,*L*-1)=0
6. *Generate a league schedule;*
7. *End if*
8. *If* $t \geq S \times (L\text{-}1)$
9. *Terminate the algorithm and report the best solution;*
10. *End if*
11. *For each team i* $(i = 1,...,L)$ *devise a new formation for its forthcoming match at week t*+1*, through an artificial match analysis. Evaluate the playing strength along with the resultant formation. If the new formation is the fittest one* (*i.e., the new solution is the best solution achieved so far by the* i^{th} *member*)*, hereafter consider the new formation as the team's current best formation;*
12. *t*=*t*+1*;*
13. *Go to* 4*;*

Like many other evolutionary algorithms, LCA works with a population of solutions. Each member of the population is a potential solution that is related to one of teams and is interpreted as the team current formation. Given a function *f* of *n* variables, each solution such as *i* can be represented with a vector of *n* real numbers. We can see each variable as one of players where changing in the value of the variables may reflect changing in the job of the relevant player. We use $X_i^t = (x_{i1}^t, x_{i2}^t, ..., x_{in}^t)$ to address the formation of team *i* at week *t*. By $f(X_i^t)$ we address the function value relevant to X_i^t. This value is called the playing strength along with formation X_i^t. By $B_i^t = (b_{i1}^t, b_{i2}^t, ..., b_{in}^t)$ we address the best formation for team *i*, experienced till week *t*. This is the best solution that has been obtained so far for the *i*th member. To determine B_i^t, a greedy selection is conducted at each iteration between X_i^t and B_i^{t-1} based on the objective values criterion.

Now we put forward greater details on the main steps of LCA; especially the manner of generating the league schedule, determining the winner/loser and setting up a new team formation.

A. Generating the league schedule

Since LCA mimics the championship in a sport league, it becomes required to schedule matches in the artificial league. A *single round-robin schedule* is utilized where each team plays every other participant once in each season. For a league of size *L*, single round robin tournament requires $L(L-1)/2$ matches, because in each of $(L-1)$ rounds (weeks), $L/2$ matches will be run in parallel (if *L* is odd, there will be *L* rounds with $(L-1)/2$ matches, and one team have no game in that round).

The scheduling algorithm is simple and we illustrate it using a league of 8 teams $(L = 8)$. Let assign each team a number and pair them off in the first week (Fig. 1a). Fig. 1a implies that team 1 plays 8, 2 plays 7 and so on. For the second week, fix one team, say team one, and rotate the others clockwise (Fig. 1b). In this week 1 plays 7, 8 plays 6 and so on. For the third week, once again rotate the order clockwise. So, 1 plays 6, 7 plays 5 and so on (Fig. 1c). We continue this process until getting the initial state. The last week (week 7) schedule can be obtained from Fig. 1d. If *L* is odd, a dummy team is added. In each week, the opponent of the dummy team does not play and gets rest.

a) 1 2 3 4
8 7 6 5

b) 1 8 2 3
7 6 5 4

c) 1 7 8 2
6 5 4 3

...d) 1 3 4 5
2 8 7 6

Fig. 1. An illustrative example of the league scheduling.

It is worth mentioning that the round-robin tournament can be modelled as an edge-coloring problem in a diagraph [8].

In LCA, championship continues for *S* successive seasons where the league schedule for each season is a single round robin schedule, yielding $S \times (L-1)$ weeks of contests. It is worth mentioning that in our implementation of LCA we use the same schedule for all of the *S* seasons. However, one can change the schedule at the start of each season.

B. Winner/loser recognition

In a regular league system, teams compete on weekends and the outcome in terms of win, loss or tie is determined for each team. Based on this, for example in soccer, each team is scored by 3 points for win, 0 for loss and 1 for tie. Disregarding the occasional crisis that may entrap even great teams in a continuum of abortive results, it is more likely that the more powerful team having better playing strength beats the weaker one (idealized rule 1). Therefore, proportional to its playing strength, each team may have a chance to win the game. Using the playing strength criterion, we determine the winner/loser in a stochastic manner by allowing teams to have their chance of win based on their degree of fit (recall that in the basic version of LCA there is no tie). The degree of fit is proportional to the team's playing strength and is measured based on the distance with an ideal reference point.

Let us consider teams *i* and *j* at week *t*, with formation strategies X_i^t and X_j^t and playing strengths $f(X_i^t)$ and $f(X_j^t)$, respectively. Let p_i^t be the chance of team *i* to beat

team j at week t (p_j^t can be defined accordingly). Let $\hat{f}$ be an ideal value (e.g., the optimal value or a lower bound on the optimal value). Based on the idealized rule 1 we can write

$$\frac{f(X_i^t)-\hat{f}}{f(X_j^t)-\hat{f}}=\frac{p_j^t}{p_i^t}. \quad (2)$$

In (2) we evaluate the teams based on their distance with a common reference point (the playing strength along with an ideal team formation), and thus the ratio of distances can determine the winning portion for each team.

Based on the idealized rule 3 we can also write

$$p_i^t + p_j^t = 1. \quad (3)$$

From (2) and (3) we get

$$p_i^t=\frac{f(X_j^t)-\hat{f}}{f(X_j^t)+f(X_i^t)-2\hat{f}}. \quad (4)$$

To simulate the win or loss, a random number in [0, 1] is generated and if it is less than or equal to p_i^t, team i wins and team j losses; otherwise team j wins and team i losses. This procedure is consistent with idealized rule 2 and 4.

If $f(X_i^t)$ be very close to $f(X_j^t)$, then $p_i \to \frac{1}{2}$ and if $f(X_j^t) >> f(X_i^t)$, then $p_i^t \to 1$. These extreme cases may validate our approach for determining the winner/loser. Since $\hat{f}$ is not known, we use the best function value found so far (i.e., $\hat{f}^t = \min_{i=1,\ldots,L}\{f(B_i^t)\}$), in place of $\hat{f}$.

C. Setting up a new team formation

Before any strategy or intervention is applied, it is important for a coach to evaluate the *strengths* and *weaknesses* of the individual members and the team as a whole. This will serve as a guide as to how to approach them and the kind of professional relationship that should be developed, which area to focus on, and how to teach the required game skills to enhance their performance. The analysis also includes the evaluation of *opportunities* and *threats* that comes along with the unique dynamics of the team. Strengths and weaknesses are internal factors while opportunities and threats often relate to external factors.

Likewise in LCA, the artificial analysis of the team's previous performance (at week t) is treated as internal evaluation (strengths/weaknesses) and analysis of the opponent's previous performance is accounted as external evaluation (opportunities/threats).

In an artificial post match analysis of team i, if it has won (lost) the game from (to) team j at week t, then it is assumed that the prosper (loss) is the direct consequence of the team strengths (weaknesses) or based on the idealized rule 5 of section 3, it is the direct consequence of the weaknesses (strengths) of team j.

Now, based on the league schedule at week $t+1$, assume that the next mach of team i is with team l. If team l has won (lost) the game from (to) team k at week t, then this success (loss) and the team formation behind it might be a direct threat (opportunity) for team i. Apparently, this success (loss) is the result of some strengths (weaknesses). Focusing on the strengths (weaknesses) of team l, gives us an intuitive way of avoiding the possible threats (affording the opportunities). Instead, based on idealized rule 5, we can focus on weaknesses (strengths) of team k.

Based on the previous week events (idealized rule 6), now the suitable actions derived from the artificial match analysis can be summarized in the SWOT matrix of Fig 2. For example, if team i was winner and team l was loser, then it is reasonable that team i focuses on the strengths which enabled it to win. At the same time it should focus on the weaknesses that caused team l to lose. These weaknesses may open opportunities for team i.

After adopting the suitable focus strategy with the aid of the artificial SWOT matrix of Fig 2, now teams should try to fill their gaps. For example, assume that during match analysis it has been detected that the reason of our loss was the weakness in man to man defence (which has allowed counter attacks by opponent). Therefore, there is a gap between the current penetrable defensive state and the state which ensures man to man pressure defence.

Let us first introduce the following indices:

l = Index of the team that will play with team i based on the league schedule at week $t+1$.

j = Index of the team that has played with team i based on the league schedule at week t.

k = Index of the team that has played with team l based on the league schedule at week t.

	i: winner ***l: winner*** ***Focusing on…***	***i: winner*** ***l: loser*** ***Focusing on…***	***i: loser*** ***l: winner*** ***Focusing on…***	***i: loser*** ***l: loser*** ***Focusing on…***
S	own strengths (or weaknesses of *j*)	own strengths (or weaknesses of *j*)	-	-
W	-	-	own weaknesses (or strengths of *j*)	own weaknesses (or strengths of *j*)
O	-	weaknesses of *l* (or strengths of *k*)	-	weaknesses of *l* (or strengths of *k*)
T	strengths of *l* (or weaknesses of *k*)	-	strengths of *l* (or weaknesses of *k*)	-

Fig. 2. Actions suitable for team i when devising its formation for the next match (based on the win/loss states).

Let us assume that X_i^t, X_j^t and X_k^t are the team formations associated to teams i, j and k at week t, respectively. By $X_k^t - X_i^t$ we address the gap between playing style of team i and team k, sensed via "focusing on

the strengths of team *k*". In this case, team *k* has won team *l* and to beat *l*, it is reasonable for team *i* to devise a playing style rather similar to that was adopted by team *k* at week *t* (for example playing counter attacking or high pressure defence). In a similar way we can interpret $X_i^t - X_k^t$ when "focusing on the weaknesses of team *k*". In other words, it may be reasonable to avoid a playing style rather similar to that was adopted by *k* (for example avoid playing counter attacking or high pressure defence). We can interpret $X_j^t - X_i^t$ or $X_i^t - X_j^t$ accordingly.

Given that normally teams play based on their current best formation (found it suitable over the times) while devising the required changes recommended by match analysis, therefore we can setup the new formation $X_i^{t+1} = (x_{i1}^{t+1}, x_{i2}^{t+1}, ..., x_{in}^{t+1})$ for team i $(i = 1,..., L)$ at week t+1 by one of the following equations.

If i was winner and l was winner, then

$$x_{id}^{t+1} = b_{id}^t + y_{id}^t (c_1 r_1 (x_{id}^t - x_{kd}^t) + c_1 r_2 (x_{id}^t - x_{jd}^t)) \ \forall d = 1,..., n \quad (5)$$

Else if i was winner and l was loser, then

$$x_{id}^{t+1} = b_{id}^t + y_{id}^t (c_2 r_1 (x_{kd}^t - x_{id}^t) + c_1 r_2 (x_{id}^t - x_{jd}^t)) \ \forall d = 1,..., n \quad (6)$$

Else if i was loser and l was winner, then

$$x_{id}^{t+1} = b_{id}^t + y_{id}^t (c_1 r_2 (x_{id}^t - x_{kd}^t) + c_2 r_1 (x_{jd}^t - x_{id}^t)) \ \forall d = 1,..., n \quad (7)$$

Else if i was loser and l was loser, then

$$x_{id}^{t+1} = b_{id}^t + y_{id}^t (c_2 r_2 (x_{kd}^t - x_{id}^t) + c_2 r_1 (x_{jd}^t - x_{id}^t)) \ \forall d = 1,..., n \quad (8)$$

End if

In the above formulas $d = 1,..., n$ is the dimension index. r_1 and r_2 are uniform random numbers in [0,1]. c_1 and c_2 are constant coefficients used to scale the contribution of the "retreat" and "approach" components, respectively. Note that the sign in parenthesis results in acceleration toward winner or retreat from loser. y_{id}^t is a binary change variable which indicates whether the *d*th element in the current best formation will be changed or not. The value of 1 for y_{id}^t allows making change in the value of b_{id}^t. Let us define $Y_i^t = (y_{i1}^t, y_{i2}^t, ..., y_{in}^t)$ as the binary change array with number of ones equal to q_i^t.

It is rather unusual that coaches do changes in all or many dimensions of the team. Generally the number of changes is relatively low. By analogy, it may seem suitable that the number of ones in Y_i^t (i.e., q_i^t) be small. To simulate the rate of changes (q_i^t), we use a truncated geometric distribution [9]. Using a truncated geometric distribution, we can set the rate of changes dynamically, while putting more weights on the smaller rate of changes. The following formula gives the random number of changes made in B_i^t to get the new formation X_i^{t+1}.

$$q_i^t = \left\lceil \frac{\ln(1-(1-(1-p_C)^n)r)}{\ln(1-p_C)} \right\rceil : \ q_i^t \in \{1,2,...,n\} . \quad (9)$$

Where r is a random number in [0, 1] and $p_C \in (0,1)$ is an input parameter. Typically p_C is known as the probability of success in the truncated geometric distribution. The larger the value of p_C, the smaller the number of changes are recommended. After simulating the number of changes by (9), q_i^t dimensions are selected randomly from B_i^t for change (i.e. their corresponding y_{id}^t gets 1).

Equations (5) to (8) use the teams' most recent formation as a basis to determine the new formation X_i^{t+1}. Let us use the notation "LCA/recent" to address such variant of LCA. It is possible to introduce another variant of LCA through introducing alternative equations for developing the new formation. Instead of considering the most recent formations (i.e. x^t) in the differential operations in equations (5) to (8), we may assume that teams do their analysis based on the teams' best formation. This means, using (b^t) in place of (x^t) in the right side of equations (5)-(8). Using the twofold notation, this variant is introduced by "LCA/best".

III. THE LEAGUE CHAMPIONSHIP ALGORITHM ADAPTED TO CONSTRAINED OPTIMIZATION

The sport driven algorithm introduced in section III is applicable only for optimization in the absence of any hard constraint except the boundary constraints. To adapt LCA to solve constrained optimization problems, the main idea is to preserve the main LCA structure while adding a mechanism to handle constraints. We use the notion of Deb's constraint-handling method [10] in the body of our algorithm. Deb's method uses a tournament selection operator, where two solutions are compared at a time, and the following selection criteria are always enforced:

- Between 2 feasible solutions, the one with better fitness value wins.
- If one solution is feasible and the other one is infeasible, the feasible solution wins.
- If both solutions are infeasible, the one with the lowest sum of constraint violations is preferred.

In Deb's approach, feasible solutions are always considered better than infeasible ones. Therefore, this approach may have difficulties with problems in which the global optimum lies on the boundary between the feasible and the infeasible regions [11]. To remedy this deficiency, similar to the approach used in [12], we try to preserve diversity by allowing solutions having good value of objective functions remain in the population.

To update the current best formation for team *i*, instead of conducting the greedy selection between X_i^t and B_i^{t-1} (as it

is typically used when there are no hard constraints), we employ the Deb's constraint-handling method in a probabilistic manner. Based on the value of a parameter called T^t (selection ratio), the selection will be performed either based only in the value of the objective function, regardless of feasibility, or based on the Deb's criteria [12].

Typically at the beginning of the search we expect searching in infeasible areas. Therefore, using larger values of T^t may make more sense since it allows keeping infeasible solutions located in promising areas of the search space. However, as the population evolves and feasible areas are detected the selection weight given to a better infeasible solution with a great sum of constraint violations value should be decreased in comparison with the one with a smaller sum of constraint violations value. This may calls for using a decreasing updating strategy for T^t (the superscript *t* denotes to different values of selection ratio at different time steps (weeks)). The updating formula for T^t is as follows:

$$T^{t+1} = \max\{0, T^t - \frac{aT^0 L}{NE}\} \quad (10)$$

Where $T^1 = T^0 = 0.55$ and $NE = 350{,}000$ is the total number of function evaluations. a is a scaling factor with conditions that: $a = 10$ if $n < 10$ and $a = 20$ otherwise.

Another issue that we should undertake is the adaptation of winner/loser recognition part of LCA to solve constrained optimization problems. Following subsection 3-B, suppose that p_i^t be the chance of team *i* to beat team *j* at week *t*. The calculation of p_i^t is subjected to the following conditions (we assume that all equality constraints $h_j(X) = 0$ are converted into inequality constraints $|h_j(X)| \le e$ and define $cv(X) = \sum_{j=1}^{q} \max(0, g_j(X)) + \sum_{j=q+1}^{m} \max(0, |h_j(X)| - e)$ as the total constraint violations value resulted by solution *X*):

If X_i^t *is feasible and* X_j^t *is infeasible then team i wins the game from j, that is* $p_i^t = 1$.

Else if X_i^t *is infeasible and* X_j^t *is feasible then team j wins the game from i, that is* $p_i^t = 0$.

Else if X_i^t *is feasible and* X_j^t *is feasible too, then the probability that team i wins the game from j is calculated based on the objective function criterion as follows:*

$$p_i^t = \frac{f(X_j^t) - \hat{f}}{f(X_j^t) + f(X_i^t) - 2\hat{f}}. \quad (11)$$

Else if X_i^t *is infeasible and* X_j^t *is infeasible too, then the probability that team i wins the game from j is calculated based on the total constraint violations criterion as follows:*

$$p_i^t = \frac{cv(X_j^t) - \hat{cv}}{cv(X_j^t) + cv(X_i^t) - 2\hat{cv}}. \quad (12)$$

End

In (12), $\hat{cv}$ is the lowest value among the total constraint violations values observed so far (similar to the definition of $\hat{f}$). As soon as the search approaches feasible regions we will have $\hat{cv} = 0$.

In order to increase the probability to generate better solutions, each team is allowed to generate a number of alternative formations in each week (multiple offspring are generated) [12]. The number of generated formations (n_f) is a user defined parameter. Among the n_f alternative formations generated for each team, we select one of them based on the sequential using of Deb's criteria with a modification on the third criterion as follows:

If both solutions are infeasible then

  If $r \le T^t$

    The selection is based on conducting the greedy selection between them based on the objective value criterion.

  Else

    The selection is based on conducting the greedy selection between them based on the total constraint violations criterion.

  End if

End if

The rational behind of the above idea is that at the beginning of the search when the algorithm searches in infeasible areas neither the sum of constraint violations nor the objective function value precedes directly the other one. The above modification has a significant role to prevent the search approaches early to unfavorable feasible regions. As an evident, the modification has a very significant effect on the rate of successfully hitting the global optimum of problem g13 (see section IV) by LCA/best.

We do not generate a predefined number of alternative formations (offspring) for each team as used in [12]. Instead we decrease the number of alternative formations systematically at certain milestones. Starting with $n_f = 5$, we decrease n_f by one every time that $\text{mod}(CE, NE/5) = 0$, where *CE* is the solution counter (*CE* is increased by one whenever a new solution is generated). Therefore, the algorithm performs its final searches with $n_f = 1$.

As our preliminary experiments indicate, the random perturbations used in (5)-(8) have a very significant effect on deteriorating the quality of the final solutions. This is because that the uniform distribution is a high-variance distribution and this causes frequently infeasibilities on problems with a narrow feasible area. On the other hand, discarding the random perturbations may be at the expense

of the lack of exploration ability. Hence, we kept the random numbers r_1 and r_2 constant for all vector dimensions $d = 1,...,n$.

TABLE I
COMPARISON OF OUR APPROACH (LCA) WITH OTHER CONSTRAINT-HANDLING ALGORITHMS

Problem	optimal		CHDE	ISR	DSS-MDE-2	LCA/ recent	LCA/ best
g01	-15.000	Best	**-15.000**	**-15.000**	**-15.000**	**-15**	**-15**
		Mean	-14.792134	**-15.000**	**-15.000**	**-15**	**-15**
		Worst	-12.743044	**-15.000**	**-15.000**	**-15**	**-15**
		Stdev	0.401	5.8E-14	0	5.35E-12	0
g02	-0.803619	Best	**-0.803619**	**-0.803619**	**-0.803619**	-0.803592	-0.803616
		Mean	-0.746236	-0.782715	-0.788011	**-0.803239**	**-0.801793**
		Worst	-0.302179	-0.723591	-0.744690	**-0.798205**	**-0.792602**
		Stdev	0.081	2.2E-02	1.5E-02	1.00E-03	3.8E-03
g03	-1.000	Best	**-1.00**	**-1.001**	**-1.0005**	-0.95402	**-1.00050**
		Mean	-0.640326	**-1.001**	**-1.0005**	-0.89587	**-1.00042**
		Worst	-0.029601	**-1.001**	**-1.0005**	-0.81096	-0.99968
		Stdev	0.239	8.2E-09	2.7E-09	3.40E-02	1.80E-04
g04	-30665.539	Best	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.539**
		Mean	-30592.154	**-30665.539**	**-30665.539**	-30665.537	**-30665.539**
		Worst	-29986.214	**-30665.539**	**-30665.539**	-30665.537	**-30665.539**
		Stdev	108.779	1.1E-11	2.7E-11	1.31E-03	1.09E-11
g05	5126.497	Best	**5126.49671**	**5126.497**	**5126.497**	-	**5126.497**
		Mean	5218.72911	**5126.497**	**5126.497**	-	**5126.497**
		Worst	5502.41039	**5126.497**	**5126.497**	-	**5126.497**
		Stdev	76.422	7.2E-13	0	-	5.067E-13
g06	-6961.814	Best	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**
		Mean	-6367.5754	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**
		Worst	-2236.9503	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**
		Stdev	770.803	1.9E-12	0	1.85E-12	1.85E-12
g07	24.306	Best	**24.306**	**24.306**	**24.306**	24.313	**24.306**
		Mean	104.599221	**24.306**	**24.306**	24.349	**24.306**
		Worst	1120.54149	**24.306**	**24.306**	24.453	**24.306**
		Stdev	176.761	6.3E-05	7.0E-08	3.19E-02	1.5E-04
g08	-0.095825	Best	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
		Mean	-0.091292	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
		Worst	-0.027188	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
		Stdev	0.012	2.7E-17	3.9E-17	2.82E-17	2.82E-17
g09	680.63	Best	**680.630**	**680.630**	**680.630**	680.631	**680.630**
		Mean	692.472322	**680.630**	**680.630**	680.632	**680.630**
		Worst	839.782911	**680.630**	**680.630**	680.635	**680.630**
		Stdev	23.575	3.2E-13	2.5E-13	9.02E-04	1.25E-10
g10	7049.248	Best	**7049.24802**	**7049.248**	**7049.248**	7070.021	**7049.248**
		Mean	8442.65694	7049.250	**7049.248**	7132.736	7049.271
		Worst	15580.3703	7049.270	**7049.249**	7251.561	7049.518
		Stdev	2186.49	3.2E-03	3.1E-04	42.953	4.91E-02
g11	0.75	Best	**0.749**	**0.75**	**0.7499**	**0.7499**	**0.7499**
		Mean	0.761823	**0.75**	**0.7499**	0.7536	**0.7499**
		Worst	0.870984	**0.75**	**0.7499**	0.7887	**0.7499**
		Stdev	0.020	1.1E-16	0	7.72E-03	1.12E-16
g12	-1.000	Best	**-1.000**	**-1.000**	**-1.000**	**-1**	**-1**
		Mean	**-1.000**	**-1.000**	**-1.000**	**-1**	**-1**
		Worst	**-1.000**	**-1.000**	**-1.000**	**-1**	**-1**
		Stdev	0.000	1.2E-09	0	0	0
g13	0.053942	Best	**0.053866**	**0.053942**	**0.053942**	-	**0.053942**
		Mean	0.747227	0.06677	**0.053942**	-	**0.053942**
		Worst	2.259875	0.438803	**0.053942**	-	**0.053942**
		Stdev	0.313	7.0E-02	8.3E-17	-	3.44E-08

"-" denotes no feasible solution was found.

IV. EXPERIMENTS

In this part we test the performance of LCA on some benchmark problems in the realm of constrained optimization. We adopt a set of 13 benchmark problems [13] used in most of constrained optimization research works. This set includes various forms of objective function such as linear, nonlinear and quadratic. The performance of the LCA is compared with that of the constraint-handling differential evolution algorithm (CHDE) [11], the improved version of stochastic ranking approach (ISR) [14] and the dynamic stochastic selection multimember differential evolution (DSS-MDE-2) [15]. All equality constraints are converted into inequality constraints with e =0.0001 (It should be noted that CHDE uses a tolerance of e =0.001, for problems g03, g11, and g13, which makes the task of hitting the optimum for this algorithm easier). All algorithms use the total number of 350,000 objective function evaluations (for CHDE this value is equal to 348,000).

We implemented both LCA/recent and LCA/best versions under the following settings of parameters: $L = \min(8n,64)$; $c_1 = c_2 = 1.1$; $p_C = 0.1$ if $n > 10$ and 0.001 otherwise.

Results obtained by CHDE, ISR, DSS-MDE-2, LCA/recent and LCA/best algorithms are reported in Table I. In this table, results are based on the best, mean, worst and the standard deviation of the lowest function values obtained in 30 independent runs of algorithms (except for DSS-MDE-2 in which the number of runs was 100) on each problem.

As can be seen from Table I, LCA/best could reach the global optimum of 12 out of 13 problems. For the other one problem (g02) LCA/best reaches -0.803616 which is extremely close to optimal value -0.803619. The average and worst case performance of LCA is also very well. For 10 problems, LCA/best finds always the true optimum solution and for the remaining three problems (g02, g03 and g10) it ensures finding near optimal solutions. Especially for g02, LCA/best reports on average an objective value of -0.801793 which is significantly better than -0.746236, -0.782715 and -0.788011 provided by CHDE, ISR, DSS-MDE-2 algorithms. The worst objective value reported by LCA/best, that is -0.792602, is also far better than that of the three algorithms, e.g., -0.302179, -0.723591, -0.744690. On g03 problem, there was only one run for which LCA/best could not reach the optimum solution. However the best objective value, i.e., -0.99968, obtained in that run is very close to -1, which is optimal objective. On g10 problem, the average and worst performance of LCA/best is a little bit worst than ISR and DSS-MDE-2 algorithms, but far better than CHDE algorithm.

Unlike LCA/best, the performance of LCA/recent is poor on several problems. Particularly, for g05 and g13 LCA/could not find any feasible solution. The reason may be due to the nature of these problems for which the density of the feasible search space is $|F|/|E| = 0.0000\%$. Also, both problems have nonlinear equality constraints. Besides the

complicated nature of these problems, the lack of intensification ability in LCA/recent seems also to be the main source of the deficiencies. Unlike LCA/best in which differential operations are calculated based on the teams' best formations, LCA/recent uses the most recent formations (solutions) as a basis to direct the search. In the absence of a bias component to direct the search towards elite solutions sitting in the promising areas of the search space, the search may stagnate in infeasible areas. However, our previous work [5] indicated that LCA/recent could deal successfully with unconstrained optimization problems. The only problem for which LCA/recent exhibits a very distinctive performance is g02. For this problem the average and worst performance of LCA/recent is -0.803239 and -0.798205 respectively, which is considerably better than the performance of other state of the art rivals. For most of optimization algorithms g02 problem presents many problems to consistently reach the vicinity of the best known solution.

From the results it can be observed that there is no problem for which the average and worst performance of CHDE algorithm surpasses those achieved by LCA/best. Although the CHDE algorithm finds the optimal solutions of all instances, however, the statistical measures suggest that this approach presents premature convergence in some case. While LCA/best is better than ISR in mean and worst values in g02 and g13, and is better than DSS-MDE-2 only in g02, it exhibits a weaker performance on g03 and g10. On the left 9 problems both performances are similar. From the comparisons, we can see that LCA/best produced very competitive results based on quality and robustness and can compete in the best way with current state of the art algorithms of constrained optimization problems.

V. CONCLUSIONS AND FUTURE DIRECTIONS

The adaptation of a new optimization algorithm called League Championship Algorithm (LCA) to solve the constrained optimization problems was investigated in this paper. LCA inspires the competition of sport teams in a sport league. A number of individual as artificial teams play in an artificial league for a number of seasons based on the win/loss system. At each week, teams devise a formation strategy (new solution) based on their current best formation (the best solution obtained so far by the corresponding team) and the post match analysis of the previous week events (to determine a search direction) using a SWOT like analysis. Based on the league schedule in each week, teams play in pairs based on their new formation strategy and their game outcome is determined in terms of win or loss, given known the playing strength (objective value) along with the team intended formation.

Using the notion of Deb's constraint-handling rule we adapt the winner/loser recognition part and the way of updating the teams' current best formation in LCA to solve a set of 13 well-known constrained optimization problems. Our results indicate that LCA is a competitive algorithm that can be efficiently used for solving constrained optimization problems. This suggests that further developments of the proposed algorithm and its applications to engineering optimization would be worth investigating in the future.

There would be a great interest on studying the effect of using other constraint-handling techniques, e.g., stochastic ranking, penalty function method etc, on the performance of LCA. Also, using a self or on-line adaptive adjusting for various parameters of the algorithm would be worth to investigate in the future works.

REFERENCE

[1] Z. Michalewicz, and M. Schoenauer, "Evolutionary algorithms for constrained parameter optimization problems," Evol Comput, vol. 4, pp. 1–32, 1995.

[2] D. G. Luenberger, "Linear and nonlinear programming," Addison-Wesley, 1984.

[3] E. Mezura-Montes, Ed., *Constraint-handling in Evolutionary Optimization*, ser. Studies in Computational Intelligence. Berlin Heideberg: Springer-Verlag, vol. 198, 2009

[4] C. A. Coello Coello, "Theoretical and numerical constraint handling techniques used with evolutionary algorithms: a survey of the state of the art," Comput Method Appl M, vol. 191, pp. 1245–1287, 2002.

[5] A. Husseinzadeh Kashan, "League Championship Algorithm: A new algorithm for numerical function optimization," in Proceedings of the International Conference of Soft Computing and Pattern Recognition (SoCPaR 2009), IEEE Computer Society, pp. 43-48, 2009.

[6] http://en.wikipedia.org/wiki/ Sports_league.

[7] http://www.talkfootball.co.uk/guides/football_ formations.html.

[8] E. Bruke, D. de Werra, and J Kingstone, "Colorings and related topics; Applications to timetabling," in Handbook of Graph Theory, J. L. Gross, and J. Yellen, Ed. CRC PRESS, 2004.

[9] A. Husseinzadeh Kashan, B. Karimi, and F. Jolai, "Effective hybrid genetic algorithm for minimizing makespan on a single-batch-processing machine with non-identical job sizes," Int J Prod Res, vol. 44, pp. 2337-2360, 2006.

[10] K. Deb, "An efficient constraint handling method for genetic algorithms," Comput Method Appl M, vol. 186, pp. 311–338, 2000.

[11] E. Mezura-Montes, C. A. Coello Coello, and E. I. Tun-Morales, "Simple feasibility rules and differential evolution for constrained optimization," in Proceedings of the 3rd Mexican International Conference on Artificial Intelligence (MICAI'2004), R. Monroy, G. Arroyo-Figueroa, L. E. Sucar, and H. Sossa, Eds. Heidelberg, Germany: Springer Verlag, lecture Notes in Artificial Intelligence No. 2972, pp. 707–716, 2004.

[12] E. Mezura-Montes, J. Vel´azquez-Reyes, and C. A. C. Coello, "Promising Infeasibility and Multiple Offspring Incorporated to Differential Evolution for Constrained Optimization," in Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2005), pp. 225–232, 2005.

[13] T. P. Runarsson, and X. Yao, "Stochastic Ranking for Constrained Evolutionary Optimization," IEEE Trans Evol Comput, vol. 4, pp. 284–294, 2000.

[14] T. P. Runarsson, and X. Yao, "Search biases in constrained evolutionary optimization", IEEE Trans. Systems, Man, Cybern. C, vol. 35, pp. 233-243, 2005.

[15] M. Zhang, W. Luo, and X. Wang, "Differential evolution with dynamic stochastic selection for constrained optimization," Inform Sciences, vol. 178, pp. 3043–3074, 2008.