Hindawi
Journal of Optimization
Volume 2017, Article ID 3082024, 13 pages
https://doi.org/10.1155/2017/3082024

Research Article

A Metaheuristic Algorithm Based on Chemotherapy Science: CSA

Mohammad Hassan Salmani and Kourosh Eshghi

Department of Industrial Engineering, Sharif University of Technology, Tehran 11365/8639, Iran

Correspondence should be addressed to Mohammad Hassan Salmani; mhsalmani@gmail.com

Received 3 June 2016; Revised 29 November 2016; Accepted 15 January 2017; Published 23 February 2017

Academic Editor: Bijaya Ketan Panigrahi

Among scientific fields of study, mathematical programming has high status and its importance has led researchers to develop accurate models and effective solving approaches to addressing optimization problems. In particular, metaheuristic algorithms are approximate methods for solving optimization problems whereby good (not necessarily optimum) solutions can be generated via their implementation. In this study, we propose a population-based metaheuristic algorithm according to chemotherapy method to cure cancers that mainly search the infeasible region. As in chemotherapy, Chemotherapy Science Algorithm (CSA) tries to kill inappropriate solutions (cancers and bad cells of the human body); however, this would inevitably risk incidentally destroying some acceptable solutions (healthy cells). In addition, as the cycle of cancer treatment repeats over and over, the algorithm is iterated. To align chemotherapy process with the proposed algorithm, different basic terms and definitions including Infeasibility Function (IF), objective function (OF), Cell Area (CA), and Random Cells (RCs) are presented in this study. In the terminology of algorithms and optimization, IF and OF are mainly applicable as criteria to compare every pair of generated solutions. Finally, we test CSA and its structure using the benchmark Traveling Salesman Problem (TSP).

1. Introduction

In the past few decades, various approaches have been proposed to solve optimization problems in two parts of exact and approximate methods. The exact ones such as dynamic programming and branch and bound algorithms are only applicable to small-scale hard problems while for solving large-scale models and highly nonlinear optimization heuristic approaches should be applied [1]. Therefore, the need to provide effective approximate solving procedures named metaheuristic algorithms is known to every researcher. It is claimed that a metaheuristic algorithm far surpasses the heuristic one as the latter is just applicable for solving a special class of problems while one can implement the former for a wide range of mathematical models and optimization problems. The majority of the proposed metaheuristic algorithms in the literature are nature-inspired with stochastic behavior which can be categorized into two groups of population-based and single point search ones. Nature is of course a great and immense source of inspiration for solving hard and complex problems in computer science since it exhibits extremely diverse, dynamic, robust, complex, and fascinating phenomena [2]. It always finds the optimal solution to solve its problem, maintaining a perfect balance among its components.

As a matter of fact, nature provides some efficient ways for solving problems via offering efficient methods to address mathematical models. Ant Colony Optimization (ACO), Simulated Annealing (SA), Genetic Algorithm (GA), and Particle Swarm Optimization (PSO) are the most well-known nature-inspired ones for solving optimization problems. Like these methods, this study also attempts to propose a natured-based metaheuristic algorithm whose origin is in chemotherapy cancer treatment.

Chemotherapy (sometimes called "chemo") uses more than 100 strong chemical drugs to treat cancer in a cycle and repetitive procedure, which is often used as the last resort to prevent the cancer from spreading, slow the cancer's growth, kill cancer cells that may have spread to other parts of the body, relieve symptoms such as pain or blockages caused by

the cancer, and cure the cancer [3]. It also has some side effects such as nausea and vomiting, hair loss, bone marrow changes, mouth and skin changes, memory loss, fertility problems, and mood swings. In chemo, the destruction of cancer cells that divide rapidly is targeted. In most cases, chemo drugs are put right into the bloodstream or taken as pills. They then travel throughout the body to kill cancer cells. Sometimes there is a need to get high doses of chemo to a specific area of the body. Regional chemotherapy directs the anticancer drugs into the part of the body where the cancer exists. The purpose is to get more of the drug to the cancer, while trying to limit effects on the whole body. Side effects will often still happen because the drugs can be partly absorbed into the bloodstream and travel throughout the body.

Our algorithm has the same structure as chemotherapy cancer treatment. In fact, we search the infeasible region where infeasible and feasible solutions are the same as cancer and healthy cells, respectively. Furthermore, while some healthy cells are killed during the treatment and their number decreases, chemo decreases the number of appropriate feasible solutions that may be loosened during the algorithm run. Moreover, each iteration of the algorithm is the same as each cycle of treatment and while the patient rests (between each two successive cycles) to recover the killed healthy cells, whereby increasing the number of cancer cells, the algorithm generates some other feasible and infeasible solutions to start the next iteration. Also, the assessment process in the algorithm is the same as the one used for studying the size of tumor and cancer cells. During these two producers, while we try to locate the exact location of the tumor, we must generate suitable initial infeasible solutions. On the other hand, determining the exact size and position of the cancer tumor is similar to calculating the value of objective function generated solution and its infeasibility value, respectively.

This study is structured as follows. To begin with, a concise and comprehensive review of literature is given on some important relative researches. Afterward, some important definitions are presented, followed by the body of CS algorithm. Next, a general discussion is given to clarify the structure of the presented algorithm. Then, the results of the study are discussed and a benchmark TSP instance is considered. Finally, the main points of the study are summed up.

2. Literature Review

As mentioned before, various nominal algorithms have been proposed in the literature, the most important of which are shown in Table 1.

Among the above-mentioned algorithms, GA, TS, SA, ACO, and PSO are the most common ones, widely used to solve optimization problems.

GA is a general metaheuristic algorithm based on genetics and human nature, which generally solves a range of optimization problems using different operators such as mutation and crossover [4]. Tabu Search (TS) algorithm is another typically used algorithm based on three short, medium, and long term memories, preventing the algorithm from generating repetitive solutions, getting stuck in local optimum solutions, and searching the regions which are not entirely investigated [5, 6].

SA tries to generate appropriate solutions for unconstrained and bound constrained optimization problems which act as the process of annealing metals [7]. ACO is a common algorithm proposed by Dorigo [8]. This nature-inspired algorithm based on the life of ant colonies is used to solve mathematical models, in particular the integer ones. Overall, ACO generates better qualified solutions in comparison to SA but the latter needs less time for finding the final solution. Particle Swarm Optimization (PSO) is another population-based algorithm for generating acceptable solutions (considered as particles), where position and velocity of the particles form the main structure of this algorithm [9].

3. The Main Body CSA

In this section, the general structure of CS algorithm is clarified and the main philosophy of algorithm and its adaptation to the process of chemotherapy treatment method for curing cancers is elaborated on. Generally speaking, we consider the standard canonical form of a mathematical model in which all types of variables can be embedded and as model (1)–(3), the objective function is in minimization form.

For more diversity and flexibility, a dynamic and stochastic structure, based on random approach, is proposed whose ignorance will raise a static algorithm that generates the same solutions in different runs.

$$\min \quad z = \sum_{j \in J} C_j X_j \tag{1}$$

$$\text{s.t:} \quad \sum_{j \in J} a_{ij} X_j \leq b_i \quad \forall i \in I \tag{2}$$

$$X_j \geq 0, \quad \text{Int} \quad \text{or } X_j \geq 0 \quad j = 1, 2, \ldots, J. \tag{3}$$

Table 2 explains different terms based on algorithm structure and chemotherapy science.

3.1. Limiting Cell Position Element (CPE) Bounds. In this phase of algorithm we limit the search space or determine the exact dimension of tumor by applying a developed method such as Constrained Programming (CP) to increase the effectiveness of our algorithm. This phase of algorithm is highly similar to the process of determining the exact position of cancer cells and tumor (infeasible solutions), the accuracy of which can go a long way towards curing the cancer (solving the problem).

Therefore, in an iterative approach the upper and lower bounds are calculated and tuned. In fact we can take the following steps to determine narrow bounds for CPEs:

(1) Calculating the initial CPE limits based on the ratio of b_i/a_{ij}, where the positive and negative ones help us to determine upper and lower bounds, respectively: obviously, we must select the minimum upper bound

Table 1: List of some metaheuristic algorithms (1975–2015) [10].

Number	Year	Algorithm
1	1975	Holland introduced the Genetic Algorithm (GA) [4].
2	1977	Glover proposed Scatter Search (SS) [11].
3	1980	Smith elucidated genetic programming [12].
4	1983	Kirkpatrick et al. proposed Simulated Annealing (SA) [7].
5	1986	Glover offered Tabu Search (TS) [5].
6	1986	Farmer et al. suggested the Artificial Immune System (AIS) [13].
7	1988	Koza registered his first patent on genetic programming [14].
8	1989	Evolver provided the first optimization software using the GA [15].
9	1989	Moscato presented Memetic Algorithm [16].
10	1992	Dorigo proposed the Ant Colony Algorithm (ACO) [8].
11	1993	Fonseca and Fleming provided Multiobjective GA (MOGA) [17].
12	1994	Battiti and Tecchiolli introduced Reactive Search Optimization (RSO) principles for the online self-tuning of heuristics [18].
13	1995	Kennedy and Eberhart proposed Particle Swarm Optimization (PSO) [9].
14	1997	Storn and Price suggested Differential Evolution (DE) [19].
15	1997	Rubinstein presented the Cross Entropy Method (CEM) [20].
16	1999	Taillard and Voss proposed POPMUSIC [21].
17	2001	Geem et al. provided Harmony Search (HS) [22].
18	2001	Hanseth and Aanestad offered Bootstrap Algorithm (BA) [23].
19	2004	Nakrani and Tovey presented Bees Optimization (BO) [24].
20	2005	Krishnanand and Ghose introduced Glowworm Swarm Optimization (GSO) [25].
21	2005	Karaboga proposed Artificial Bee Colony (ABC) Algorithm [26].
22	2006	Haddad et al. suggested Honeybee Mating Optimization (HMO) [27].
23	2007	Shah-Hosseini offered Intelligent Water Drops (IWD) [28].
24	2007	Atashpaz-Gargari and Lucas introduced Imperialist Competitive Algorithm (ICA) [29].
25	2007	Mucherino and Seref suggested Monkey Search (MS) [30].
26	2008	Yang presented Firefly Algorithm (FA) [31].
27	2009	Husseinzadeh Kashan provided League Championship Algorithm (LCA) [32].
28	2009	Rashedi et al. introduced Gravitational Search Algorithm (GSA) [33].
29	2009	Yang and Deb offered Cuckoo Search (CS) [34].
30	2010	Yang developed Bat Algorithm (BA) [35].
31	2011	Shah-Hosseini introduced the Galaxy-based Search Algorithm (GbSA) [36].
32	2011	Tamura and Yasuda designed Spiral Optimization (SO) [37].
33	2011	Rao et al. presented Teaching-Learning-Based Optimization (TLBO) algorithm [38].
34	2012	Gandomi and Alavi proposed the Krill Herd (KH) Algorithm [39].
35	2012	Civicioglu introduced Differential Search Algorithm (DSA) [40].
36	2013	Gandomi et al. introduced Cuckoo Search Algorithm (CSA): a metaheuristic approach to solving structural optimization problems [41].
37	2013	Gandomi et al. introduced Firefly Algorithm (FA) with chaos [42].
38	2014	Kaveh and Mahdavi developed Colliding Bodies Optimization (CBO) Algorithm [43].
39	2014	Beheshti and Shamsuddin presented CAPSO: centripetal accelerated Particle Swarm Optimization [44].
40	2014	Meng et al. designed Crisscross Optimization Algorithm (COA) [45].
41	2015	Javidy et al. proposed Lons Motion Algorithm (LMA) [46].
42	2015	Yu and Li developed a Social Spider Algorithm (SSA) [47].
43	2016	Rao proposed Jaya algorithm as a simple algorithm [48].
44	2017	Salmani and Eshghi introduced a Smart Structured Algorithm (SSA) to solve Mixed Integer Problem (MIP) [49].

Table 2: Terms and definitions.

Number	Chemotherapy term	Algorithm term	Definition
1	Out of tumor (OUT)	Feasible region	A set of all possible points of a predetermined space called a feasible region which satisfies the mathematical model's constraints.
2	Tumor (TU)	Infeasible region	A set of points which are located out of the feasible set and cannot satisfy at least one of the constraints of our optimization model.
3	Tumor Position (TP)	Infeasibility Function	TP indicates an approximate measure to calculate the infeasibility of a point from the border of a predetermined constraint. Also, Total TP (TTP) is an aggregation of TPs for calculating the total infeasibility.
4	Tumor size (TS)	Objective function	TS is a function that we want to optimize subject to different constraints by using mathematical programming techniques. In fact, it is going to minimize the tumor size.
5	Healthy cell	Feasible solution	Indicating a solution which is located in the feasible region.
6	Cancer and bad cell	Infeasible solution	Indicating a solution which is located in the infeasible region.
7	Initial cancer cell	Initial infeasible solution	Indicating the input solution of algorithm.
8	Cell Position Element (CPE)	Variable	Indicating element X_j in $X = (X_1, X_2, \ldots, X_J)$ as a solution.
9	Cell	Solution	Indicating set of variables $X = (X_1, X_2, \ldots, X_J)$ as a solution.

and maximum lower bound among all the bounds in different constraints.

(2) For each CPE (index k), we fix the previously calculated upper and lower bounds of the other CPEs (indexes $j = 1, \ldots, J$ & $j \neq k$) and then determine the new upper and lower bounds for the selected Cell Position Element (index k). For calculating upper (lower) bound of CPEk, if $a_{ik}/a_{ij} > 0$ the lower (upper) bound of CPEj or else its upper (lower) bound will be considered.

(3) Repeat phase (2) while for each successive repetition all the upper and lower bounds get into a confidence interval with a predetermined percentage of error (such as a value around 5%).

At the end of this step IL = (IL(1), IL(2), ..., IL(J)) and IU = (IU(1), IU(2), ..., IU(J)) result as initial lower (IL) and initial upper (IU) bounds, respectively.

3.2. Generating Initial Cancer Cells in CSA. Various approaches are available in this algorithm to determine the position of initial cancer cells and tumor or initial infeasible solution. There is a similarity between this phase and the process of chemotherapy. While we want to select the method of generating initial solutions, we may also want to determine which drugs and injection methods should be used. Also, calculating initial solutions in the proposed algorithm is the same as determining the initial position of cancer tumor and bad cells in the process of chemotherapy treatment.

Based on our problem, we may use completely random or exact methods or a trade-off method between these two. The following are four possible and proposed approaches which can be implemented in this phase to determine the initial position of tumor cells using:

(1) Relaxation methods including linear programming (LP) and Lagrangian.

(2) CPE-limited bounds which are calculated in the previous phase where we can use one of lower and upper bounds in order to improve the TS.

(3) Composition of random search, relaxation, variable bounds, and other possible approaches.

(4) Problem-based approaches such as greedy methods.

After this phase, the algorithm proposes a population ($p = 1, 2, \ldots, P$) of $X^{p0} = (X_1^{p0}, X_2^{p0}, \ldots, X_J^{p0})$ named initial cells.

3.3. Evaluating the Health Status and Position of the Tumor and Cells. CS algorithm searches the tumor to find a new healthy cell which means generating a new solution in the infeasible space. Therefore, for each pair of found cells (cancerous or healthy cells), two important factors must be considered for comparing them, which are the total value of Tumor Position (TP) and the value of tumor size (TS) in relation (1).

Simply put, it is possible to calculate TP_i^{pt} for constraint i based on inequality (2), in which $X^{pt} = (X_1^{pt}, X_2^{pt}, \ldots, X_J^{pt})$ that is the pth solution vector of the population generated at iteration $t = 1, 2, \ldots, T$.

In fact, by applying (4) and the aggregation in a theoretical and logical way using relation (5), the values of TP_i^{pt} and TTP^{pt} (Total TP based on all of the constraints), will result.

$$\mathrm{TP}_i^{pt} = \begin{cases} \sum_{j \in J} a_{ij} X_j^{pt} - b_i \sum_{j \in J} a_{ij} X_j^{pt} \geq b_i & \\ 0 \qquad \text{O.W,} & \end{cases} \qquad \forall i \in I, \quad (4)$$

$$\mathrm{TTP}^{pt} = \sum_{i \in I} \mathrm{TP}_i^{pt} V_i. \quad (5)$$

In this equation, V_i indicates the constraint weights while we can assume the same values for all model constraints.

Also, the value of TS^{pt} is calculated based on the relation

$$\mathrm{TS}^{pt} = \sum_{j \in J} C_j X_j^{pt}. \quad (6)$$

To analyze each pair of solutions or cells, the designer should create a tradeoff between the values of TTP^{pt} and TS^{pt}. We categorize all cells into four different groups to determine the best and worst ones. This type of classification is done according to Table 3. As it can be seen, it is an iterative and interactive approach between current cells and the ones which have been generated previously.

ATTP^{t-1} and ATS^{t-1} indicate the arithmetic mean of TTP^{pt} and TS^{pt} of the sets of the best solutions of the previous solution into average TTP and TS, respectively.

$$\begin{aligned} \mathrm{ATS}^{t} &= \frac{1}{|F(t)|} \sum_{p \in F(t)} \mathrm{TS}^{pt} \quad \forall t = 1, 2, \ldots, T, \\ \mathrm{ATTP}^{t} &= \frac{1}{|F(t)|} \sum_{p \in F(t)} \mathrm{TTP}^{pt} \quad \forall t = 1, 2, \ldots, T, \end{aligned} \tag{7}$$

where $|F(t)|$ is the cardinality of the set of the cells with the first rank at the end of iteration t.

In this ranking, the first group will be considered as the set of best generated cells and its combination with the second group constitutes the next input cells meaning $(X^{ft} \cup X^{pt}) \rightarrow X^{pt}$. The best cells of the third group based on TS are reserved for special use and finally the fourth group are thrown away. It should also be noted that when the first group is empty, the first group of previous iteration will be replaced.

3.4. Search Neighborhood Cells. In this phase, we want to extend our investigation where celerity and intelligently are its most important features. This phase is also flexible enough to be applied for developing a single point population-based algorithm.

3.4.1. Determining Cell Area (CA) in Tumor. CS algorithm proposes an innovative approach to solving mathematical models in which a distinct attitude towards investigating the tumor and search space is introduced. This method uses Cell Area (CA), which is a limited space around each cell (solution) and where easy generation of CAs increases its efficiency. To begin with, by using two different levels (r_{lj}^{pt} and r_{uj}^{pt} in relations (9)), CPE (Cell Position Element) lower (L_j^{pt}) and upper (U_j^{pt}) bounds are calculated via (8), and by their combination a subspace around each cell is created.

$$\begin{aligned} L_j^{pt} &= X_j^{pt} - r_{lj}^{pt}, \\ U_j^{pt} &= X_j^{pt} + r_{uj}^{pt}, \end{aligned} \tag{8}$$

where two values of r_{lj}^{pt} and r_{uj}^{pt} are determined using the following relations:

$$r_{lj}^{pt} = \begin{cases} \dfrac{X_j^{pt} C_j}{\sum_{j \in J(N)} C_j X_j^{pt} + \sum_{j \in J(Z)} C_j \left((\mathrm{IL}(j) + \mathrm{IU}(j))/2 \right)} & \forall j \in J(N) \\ \dfrac{C_j \left((\mathrm{IL}(j) + \mathrm{IU}(j))/2 \right)}{\sum_{j \in J(N)} C_j X_j^{pt} + \sum_{j \in J(Z)} C_j \left((\mathrm{IL}(j) + \mathrm{IU}(j))/2 \right)} & \forall j \in J(Z), \end{cases} \tag{9}$$

$$r_{uj}^{pt} = r_{lj}^{pt} \times \mathrm{Const}.$$

In these relations, $J(N)$ and $J(Z)$ are the set of CPEs with nonzero ($X_j^{pt} \neq 0$) and zero ones ($X_j^{pt} = 0$), respectively. Also, we mostly propose linear-based relations for more simplicity and decreasing complexity of CSA. Furthermore, we can calculate upper level (r_{uj}^{pt}) based on the lower one (r_{lj}^{pt}) by just an easy approach where the lower value is multiplied with a constant number to calculate the upper limit.

Also, if we want to determine these values for the CPEs without any coefficients in tumor size ($C_j = 0$), it should consider an equivalent weight (C_j^E) in relations (9). We can calculate this weight using an arithmetic mean of the CPEs (in which positive or negative sign is embedded) that are in the same constraints with CPE index j with $C_j = 0$, as the equation

$$\begin{aligned} C_j^E &= \frac{1}{|L|} \sum_{l \in L} C_l - \frac{1}{|M|} \sum_{m \in M} C_m; \\ & C_j = 0, \ \ j \notin L \ \& \ j \notin M, \end{aligned} \tag{10}$$

where $|L|$ and M are the cardinality of the sets of CPEs which are in the same constraints with CPEj, with different and same signs, respectively.

To clarify it, we can refer to Figure 1, which shows a CA for a sample instance with just two CPEs X_1 and X_2.

Once again, it is possible to round the value of integer CPEs to the nearest integer number. However, if L_j^{pt} (U_j^{pt}) gets less (greater) than the lower (upper) bound, the lower (upper) bound will be replaced.

3.4.2. Generating Random Cells in CA. To calculate the values of Random Cells (RCs), being cancerous or healthy, based on CA, we can multiply ($U_j^{pt} - L_j^{pt}$) with a random number and add it to its lower bound. Under this condition and because of using problem data and random number, RC will be a smart random number.

$$\begin{aligned} X_{rj}^{pt} &= L_j^{pt} + \mathrm{rand} \times \left(U_j^{pt} - L_j^{pt} \right) \\ & \forall j = 1, 2, \ldots, S, \ \ r = 1, 2, \ldots, R. \end{aligned} \tag{11}$$

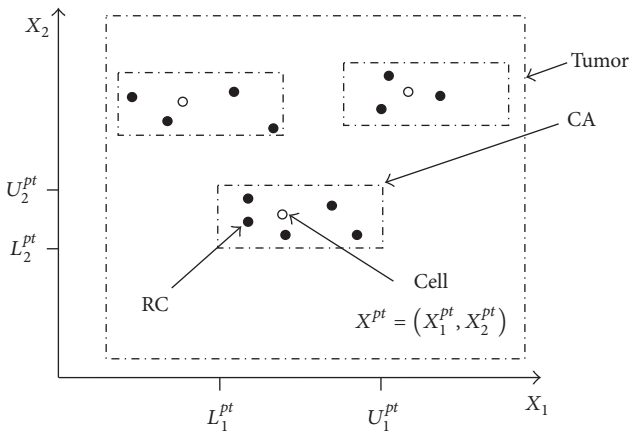

Figure 1: Sample Cell Area (CA).

Table 3: Cells ranking based on TS and TTP.

Rank	Optimality criterion	Infeasibility criterion	Healthy/cancer cells
1	$\text{ATS}^{t-1} > \text{TS}^{pt}$	$\text{ATTP}^{t-1} \geq \text{TTP}^{pt}$	X^{ft} and X^{pt}
2	$\text{ATS}^{t-1} > \text{TS}^{pt}$	$\text{ATTP}^{t-1} < \text{TTP}^{pt}$	X^{pt}
3	$\text{ATS}^{t-1} \leq \text{TS}^{pt}$	$\text{ATTP}^{t-1} \geq \text{TTP}^{pt}$	—
4	$\text{ATS}^{t-1} \leq \text{TS}^{pt}$	$\text{ATTP}^{t-1} < \text{TTP}^{pt}$	—

As in the case of upper and lower bounds of integer CPEs, again X_{rj}^{pt} should be rounded in special cases.

3.4.3. Rest Period Phase. In CS algorithm, we get near to the border of tumor from a determined cell out of the tumor using CA, RCs, and, of course, smart vectors. In fact, false direction and any inaccuracy or incorrectness in it can result in inappropriate cells. In the relevant literature, differentiation is the most common way among all the proposed effective ones. However, this approach is not applicable to different classifications of models and problems including LP, IP, and MIP. Therefore, a positive, negative, or zero direction for each CPE based on model coefficients and cells is determined, and then we take one step in this vector to find another cell nearer to the tumor and cancer cells and their border, with more qualified TS.

This phase is similar to the new cycle for drug injection to the body and tumor for killing the cancer cells. While we try to convert some of the cancer cells (infeasible solutions) to their corresponding healthy (feasible) ones in CS algorithm, doctors let the patient rest and refresh their healthy cells. As mentioned in Section 3.6, we are able to convert some of the bad cells to the healthy ones using a linear programming model or applying problem-related approaches such as a Greedy approach to TSP.

In two consecutive comparative steps we can determine the mentioned direction.

At first, we determine the direction type based on CPE values. If we consider that $C_{\max} = \max_{j\in J}(C_j)$ and $C_{\min} = \min_{j\in J}(C_j)$ and also $X_{\max}^{pt} = \max_{j\in J}(X_j^{pt})$ and $X_{\min}^{pt} = \min_{j\in J}(X_j^{pt})$, it is possible to determine $\overline{C} \cdot \overline{X}^{pt}$ using the following relations:

$$\overline{C} = \frac{C_{\max} + C_{\min}}{2},$$
$$\overline{X}^{pt} = \frac{C_{\max} \cdot X_{\max}^{pt} + C_{\min} \cdot X_{\min}^{pt}}{C_{\max} + C_{\min}}, \tag{12}$$
$$\overline{C} \cdot \overline{X}^{pt} = \frac{C_{\max} \cdot X_{\max}^{pt} + C_{\min} \cdot X_{\min}^{pt}}{2}.$$

Now, we should compare $C_j \cdot X_j^{pt}$ to $\overline{C} \cdot \overline{X}^{pt}$ to determine the direction type where using $R_j = C_j/(C_{\max} + C_{\min})$ as the adjustment coefficient helps us modify our comparison. In fact, we can use $R_j \cdot C_j \cdot X_j^{pt}$ instead of $C_j \cdot X_j^{pt}$ in our comparison whenever it is necessary. It means that this evaluation will be adjusted between $(C_{\max} \cdot X_{\max}^{pt} + C_{\min} \cdot X_{\min}^{pt})/2$ and $(C_j^2 \cdot X_j^{pt})/(C_{\max} + C_{\min})$ to determine the CPE direction type.

Relation (13) helps us to clarify this statement in which D_j^{pt} is the direction of CPE_j.

$$D_j^{pt} = \begin{cases} D_j^{pt} + d_1 R_j \cdot C_j \cdot X_j^{pt} < \overline{C} \cdot \overline{X}^{pt} \\ D_j^{pt} R_j \cdot C_j \cdot X_j^{pt} = \overline{C} \cdot \overline{X}^{pt} \\ D_j^{pt} - d_1' R_j \cdot C_j \cdot X_j^{pt} > \overline{C} \cdot \overline{X}^{pt}. \end{cases} \tag{13}$$

In relation (13), parameters d_1 and d_1' should be one or can be determined using a parameter tuning approach. In fact, we want to increase (decrease) the value of CPEj when its adjusted value $(R_j \cdot C_j \cdot X_j^{pt})$ in the tumor size (TS) is less (greater) than our criterion $(\overline{C} \cdot \overline{X}^{pt})$ and let it remain unchanged for the case of equality.

Secondly, we must determine the direction based on the position (infeasibility) of the cell (solution) using relation (14); afterwards, we update D_j^{pt} according to DTP_j^{pt} by using relation (15).

$$\text{DTP}_j^{pt} = \begin{cases} \text{DTP}_j^{pt} + d_2 \text{TP}_i^{pt} = 0 \\ \text{DTP}_j^{pt} - d_2' \text{TP}_i^{pt} > 0, \end{cases} \tag{14}$$

$$D_j^{pt} = \begin{cases} D_j^{pt} + d_3 \text{DTP}_j^{pt} < 0 \\ D_j^{pt} \text{DTP}_j^{pt} = 0 \\ D_j^{pt} - d_3' \text{DTP}_j^{pt} > 0. \end{cases} \tag{15}$$

To take small steps and, of course, adjust the directions, the values of D_j^{pt} can be normalized using the equation

$$D_j^{pt} = \frac{D_j^{pt}}{\max_{\forall j \in S} \{D_j^{pt}\}}. \tag{16}$$

In relations (14)-(15), parameters d_2, d_2', d_3, and d_3' should be valued in accordance with each other. Obviously, the

Table 4: Stopping conditions.

#	Stopping condition	Explanation
1	Reaching the border of tumor where it is cured approximately (feasible region).	Algorithm stops when TTP becomes zero or less for the best generated cell and an approximate acceptable solution is generated.
2	Reaching a fixed number of chemotherapy repetitions.	Algorithm stops after a predetermined number of runs and converting the cancer cells to their corresponding healthy ones.
3	Reaching a fixed value for TTP and getting near to the border of tumor.	Algorithm stops when it is in a predetermined special distance of feasible region and a predetermined TTP and TS result.
4	Reaching a percentage of improvement in TS or tumor size.	Algorithm stops when TS has a predetermined percentage of improvement in comparison to its initial value.

importance of criterion TP_i^{pt} increases as the number of constraints increases, resulting in bigger ratios. The number of CPEs and constraints increases, so do d_2'/d_2 and d_3'/d_3.

Finally, we can reach a new cell (solution) based on vectors D^{pt} and X_r^{pt} using the equation

$$X_{ro}^{pt} = X_r^{pt} + D^{pt}, \quad p = 1, 2, \ldots, P;\ t = 1, 2, \ldots, T,\ r = 1, 2, \ldots, R;\ o = 1, 2, \ldots, O, \tag{17}$$

where O indicates the number of directions, and based on X_{ro}^{pt}, we can determine the values of the other vectors such as X^{pt} and X^{ft}.

It should be mentioned that it is possible to calculate all these values based on matrix forms where using a powerful software program such as MATLAB can ease the implementation of this approach.

3.5. Stopping Conditions. CS algorithm has a completely different structure compared to the other proposed ones in the literature; therefore, the stopping conditions are distinct while different factors such as the structure of our problem (TSP, LP, MIP, etc.), improvement rate in TTP and TS, and also input data may bear on our selection. Table 4 indicates some suggested appropriate criteria.

3.6. Converting Cancerous Cells to Healthy Ones. In CSA we need a general approach to converting each cancerous cell to its equivalent healthy one. This method may be applied in different parts of the algorithm. Assume that we stop the algorithm and the final generated cells are not healthy where based on our mathematical model with hard constraints it is not an appropriate cell; therefore, a conversion method is required to generate the final applicable cell. Moreover, as in the process of cancer treatment using chemotherapy, here we need a fast conversion method to convert appropriate bad cells to healthy ones and maintain the best ones until stopping CSA repeating.

In the final stage of curing the cancer in which the surgeon removes the weakened tumor, we want to convert inapplicable infeasible solutions (cancer cells or tumor) to acceptable feasible ones (healthy cells). In fact, it is the final stage which can be implemented for stopping the algorithm. Generally, as doctors weaken the tumor during radiation therapy, we can convert the bad and cancerous cells (infeasible solutions) to healthy (feasible) ones during algorithm run. Apparently, this methodology increases flexibility and results in better-off cells.

We know that converting a cancerous cell to its corresponding healthy one decreases the quality of TS due to narrowing the space, and at the best situation with a small probability, a healthy cell with the same TS will result at best. Therefore, we should apply an approach to maintaining the TS in its maximum possible value. In this special case, the following model can help us to solve this issue. However, adapting it to our mathematical model is suggested. For instance, the combination of this model with a greedy approach seems more effective when we solve a TSP or Knapsack Problem (KP).

$$\max \quad W = \sum_{j \text{ where } X_j > 0} C_j X_j' \tag{18}$$

$$\sum_j a_{ij} X_j' \geq \mathrm{TP}_i \qquad \forall i = 1, 2, \ldots, |I| \text{ where } \mathrm{TP}_i > 0 \tag{19}$$

$$X_j' \leq X_j \quad \forall j = 1, 2, \ldots, n \text{ where } X_j > 0 \tag{20}$$

$$X_j' \geq 0, \quad \mathrm{Int}\ \forall j = 1, 2, \ldots, n \text{ where } X_j > 0, \tag{21}$$

where X_j' is the amount of CPE index j which should be decreased from its original value (X_j) to reach a new set of CPE vector (X^n) as a healthy cell (relation (22)).

$$X_j^n = X_j - X_j'. \tag{22}$$

In this model, we try to decrease the value of CPEs by considering the constraints with $\mathrm{TP}_j > 0$. We can say that relation sets (19) attempt to omit any occurring infeasibility while maximizing the total decrease of object function using relation (18).

Feasibility Theorem. *Model (18)–(21) has at least one feasible solution.*

Proof. Suppose a solution in which $X_j' = X_j$ (relation (20)) which means that $X_j^n = 0$, where the generated solution with $X = 0$ would be a feasible solution for model (1)–(3).

On the other hand, we know that the values $\mathrm{TP}_i > 0$ are created due to the positive values of variables ($X_j^n > 0$). Therefore, when $X_j' = X_j\ \forall j \in J$, summation $\sum_j a_{ij} X_j'$ will cover all the infeasibility of constraint i ($\mathrm{TP}_i > 0$), meaning that relation (19) is satisfied. □

4. Discussion about the Structure of CSA

The general structure of CSA, including the necessary steps and strategies in our algorithm, is graphically displayed in Figure 2. The rotating movements of CS algorithm between the inside and outside of the tumor in two general phases increase its flexibility by deeply exploring the space and subspaces of solutions.

As it can be seen, we map different parts of CS algorithm on the process of chemotherapy cancer treatment. Therefore, this algorithm is developed based on a firm background of treatment process whose efficiency has been proved in medicine in the past few decades. As a matter of fact, nature-inspired algorithms such as GA, ACO, and SA are significantly efficient for solving optimization problems. These types of algorithms such as CSA have a reliable background which theoretically build a logical structure.

The cyclic part of this algorithm is similar to the revolutionary part of chemotherapy process in which, after a drug injection phase, a resting phase follows while in the proposed algorithm we try to get close to the border tumor and the healthy space around it (as in drug injection phase). Finally, we may convert cancerous cells (killing cancer cells) to healthy ones (as resting phase).

Moreover, while developing CS algorithm, we try to consider mathematical exact solving methods, techniques, and formulas to improve its performance. In fact, by combining nature processes and mathematical relations, an effective algorithm is introduced where some parts of the main structure are based on nature and the other parts are based on mathematics, statistics, and science.

In CS algorithm, generating initial cancerous cells is critically important. In fact, based on the directing concept, the algorithm explores different parts of the space starting from the initial cells. However, the quality of initial cells depends on the first step in which algorithm determines the limits of CPEs to narrow the search space such as constraint programming.

On the other hand, generating cells using fundamental concepts such as CA and RCs as mathematical rules combined by the chemotherapy process provides a set of theoretical and practical tools for user of CSA to solve optimization problems. The importance of these techniques becomes clearer when considerable cells are generated particularly for solving hard problems such as TSP.

Likewise, it should be noted that converting strategy is another considerable part of this algorithm which lets us generate healthy cells during the algorithm and reserve them till the final stage. This approach helps users to modify directions (based on CA, RCs, and smart vectors) towards those parts of healthy space out of the tumor to reach healthy cells. This special feature creates a smart algorithm in searching the space discontinuously and widely. It means that CS algorithm tries to search different parts of the space which may include the optimum cell with more probability.

Finally, in special cases when the algorithm stops while it is still inside the tumor, a procedure for generating healthy cells is proposed. This phase lets algorithm stop before getting out of the tumor for saving time and prohibiting unnecessary runs of the algorithm. Moreover, for hard problems such as TSP, to use this methodology is strongly suggested for generating an acceptable cell in the healthy part of the body near the tumor with an efficient objective function.

As a matter of fact, as the degree of the hardness of prototype optimization problems increases, the deterioration of the cancer and its seriousness increases accordingly. As it is well known, TSP is one of the hardest problems in combinatorial optimization fields; therefore, its corresponding cancer prototype is more serious and dangerous.

5. Computational Results

To solve a sample problem using CSA, a benchmark example for TSP is solved (adopted from [50] named TSPTW). Our general approach is appraising the algorithm by comparing the generated cells of each problem with their optimum one. The results indicate an effective performance for this algorithm in solving NP-complete problems such as TSP.

We ran CSA on a Core i5 2.4 GHz computer with 4 GB RAM using MATLAB R2012a and GAMS 24.1.2.

To solve TSP, we need to modify some of the proposed relations based on the following new mathematical model [51]:

$$\min \quad z = \sum_i \sum_j C_{ij} X_{ij} \tag{23}$$

$$\text{s.t:} \quad \sum_j X_{ij} = 1 \quad \forall j = 1, 2, \ldots, n \tag{24}$$

$$\sum_i X_{ij} = 1 \quad \forall i = 1, 2, \ldots, n \tag{25}$$

$$u_i - u_j + nX_{ij} \leq n - 1 \qquad \forall i, j = 2, 3, \ldots, n \ \& \ i \neq j \tag{26}$$

$$X_{ij} \in \text{Binary} \ \& \ u_i \in \text{URS} \quad \forall i, j = 1, 2, \ldots, n. \tag{27}$$

Obviously, to determine the value of TP for constraints (24)-(25), we should apply the following two equations:

$$\text{TP}_i^{pt} = \sum_{j \in J} a_{ij} X_j^{pt} - b_i, \quad \forall i \in I, \tag{28}$$

$$\text{TTP}^{pt} = \sum_{i \in I} \left| \text{TP}_i^{pt} \right| V_i. \tag{29}$$

In this model, the CPE limits are zero or one for the binary variables and these bounds for the URS variables are infinite. Also, we generate the initial solution by using a combinatorial approach based on greedy and LP relaxation methods in which the generated solution using LP relaxation methodology is modified to create a new infeasible IP (binary) solution for TSP. On the other hand, for calculating the other parts and running the algorithm (such as the appraising phase, killing cancer cells, etc.), the previously mentioned relations and formulas are applied.

The benchmark data is available on its website and its optimum solution and the best generated cell by CS algorithm

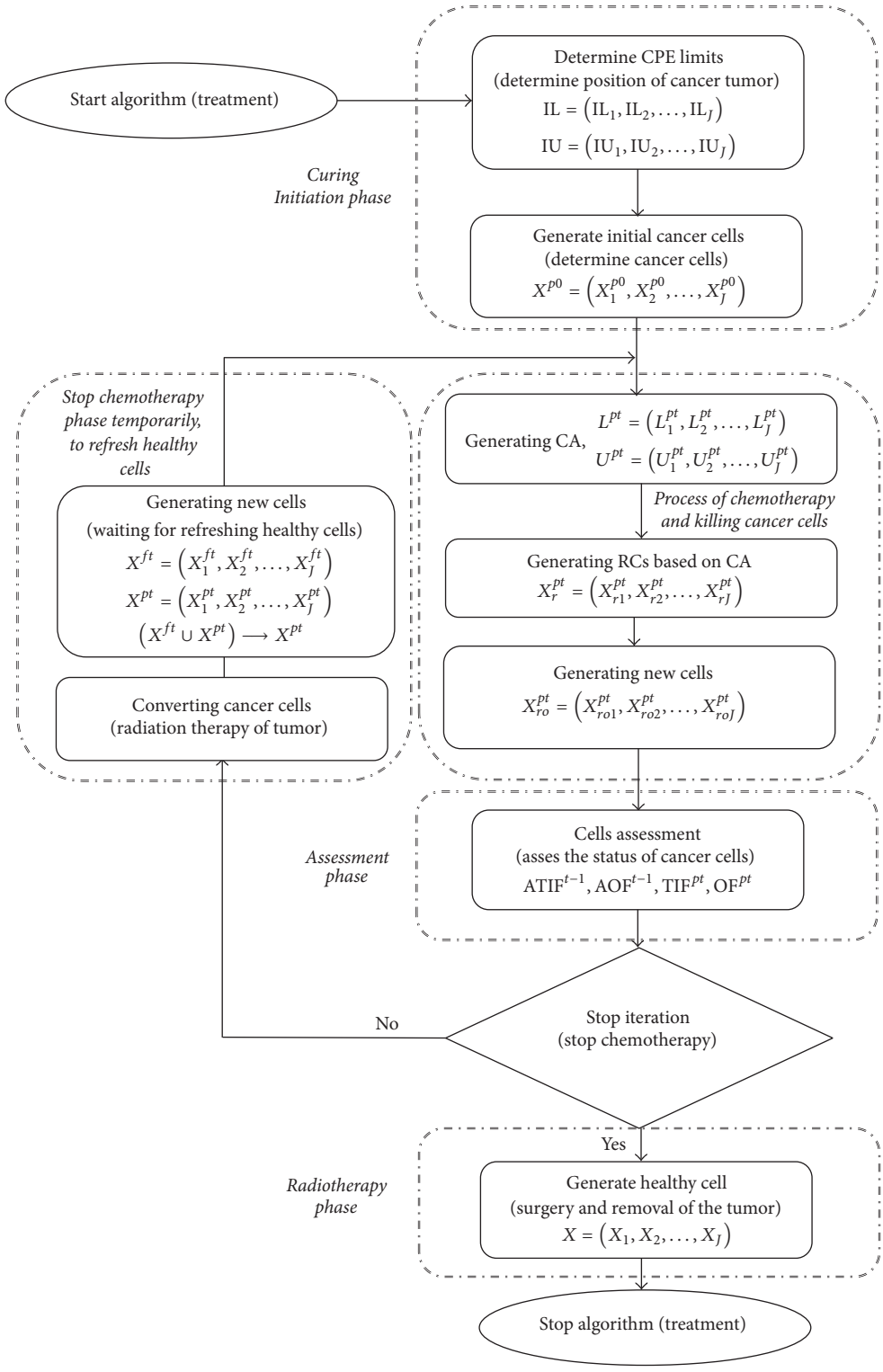

Figure 2: General structure of the algorithm.

is shown in Table 5. Just a short glance on the results indicates the optimum cell is approximately achieved using the proposed algorithm. Here, there are just 3 units different between TS values of the best generated cell (1457) and the optimum one (1454) which indicates a ratio of 0.21%.

We run the algorithms 20 times to solve the benchmark problem as is shown in Table 6. We know that in TSP all the constraints are hard and the final generated cell should be located outside the tumor, meaning that the value of TTP should be zero for the final generated cell. Generally,

Table 5: Optimum and algorithm best cell for sample TSP problem.

Optimum cell		Algorithm best cell	
Tumor size	1454	Tumor size	1457
Origin city	Destination city	Origin city	Destination city
1	23	1	7
2	3	2	23
3	25	3	17
4	5	4	16
5	6	5	6
6	17	6	3
7	18	7	18
8	14	8	10
9	2	9	20
10	1	10	1
11	16	11	19
12	11	12	25
13	10	13	2
14	22	14	22
15	19	15	9
16	15	16	15
17	8	17	8
18	12	18	12
19	21	19	24
20	4	20	21
21	20	21	14
22	9	22	11
23	24	23	4
24	8	24	5
25	13	25	13

it is possible to convert the cancer cell to its corresponding healthy one by using a greedy approach.

In fact, for a counterpart problem of TSP, we need to kill some of the cancer cells and after weakening tumor, we must remove it completely to cure it. As we know, TSP is an NP-complete problem; therefore its corresponding cancer prototype is also a serious and crucial one and in addition to chemotherapy radiation therapy is required and mandatory.

Also, we know that TSP is an NP-complete problem and generating a healthy cell with 0.21% minimum range of errors and an average around 1.10% is invaluable and these results indicate the effectiveness of CS algorithm and efficiency of its performance. It means that solving other types of optimization problems such as MIP ones by applying CS algorithm can result in great cells using appropriate software and professional programming methods.

In this instance, there is a small difference between the minimum and optimum solutions on the one hand and the maximum and optimum solutions on the other hand, which is 0.21% and 1.51%, respectively. Moreover, total difference between minimum and maximum ratio of TSs is 1.30% where the average is 1.51%. These values and of course Figure 3 again indicate a robust algorithm which effectively generates appropriate cells, and the range of TSs is narrow.

Table 6: Algorithm generated cells for sample TSP problem with 25 cities.

Iteration number	TS	TTP	Ratio*
Average	1469.95	0.00	1.10%
Minimum	1457.00	0.00	0.21%
Maximum	1476.00	0.00	1.51%
1	1467.00	0.00	0.89%
2	1474.00	0.00	1.38%
3	1471.00	0.00	1.17%
4	1457.00	0.00	0.21%
5	1468.00	0.00	0.96%
6	1471.00	0.00	1.17%
7	1476.00	0.00	1.51%
8	1467.00	0.00	0.89%
9	1472.00	0.00	1.24%
10	1468.00	0.00	0.96%
11	1471.00	0.00	1.17%
12	1463.00	0.00	0.62%
13	1475.00	0.00	1.44%
14	1472.00	0.00	1.24%
15	1474.00	0.00	1.38%
16	1475.00	0.00	1.44%
17	1472.00	0.00	1.24%
18	1465.00	0.00	0.76%
19	1474.00	0.00	1.38%
20	1467.00	0.00	0.89%

*[(TS − Optimum TS)/(Optimum TS) ∗ 100].

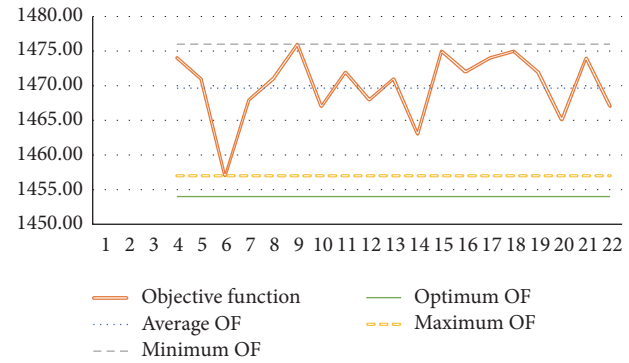

Figure 3: General analysis of the generated solution of the algorithm for the TSP problem with 25 cities.

In order to test the efficiency of proposed algorithm on large problem, a TSP problem with 100 cities was selected from benchmark problems [52] where its length of optimal tour is 2772.31.

As it can be seen in Table 7 and Figure 4, CSA generates good solutions where the average objective function of its tours is near, 2.53%, to the optimal tour. It is worth mentioning that this TSP sample is almost 4 times bigger than our first problem and its results indicate that CSA has the ability to produce good solutions for bigger and harder problems.

Table 7: Results of CSA for a large TSP problem with 100 cities.

Iteration number	TS	TTP	Ratio*
Average	2842.39	0.00	2.53%
Minimum	2786.77	0.00	0.52%
Maximum	2890.42	0.00	4.26%
1	2847.69	0.00	2.72%
2	2821.82	0.00	1.79%
3	2890.42	0.00	4.26%
4	2875.97	0.00	3.74%
5	2813.67	0.00	1.49%
6	2801.05	0.00	1.04%
7	2791.67	0.00	0.70%
8	2869.66	0.00	3.51%
9	2826.80	0.00	1.97%
10	2861.69	0.00	3.22%
11	2869.54	0.00	3.51%
12	2828.09	0.00	2.01%
13	2820.16	0.00	1.73%
14	2889.96	0.00	4.24%
15	2828.99	0.00	2.04%
16	2853.97	0.00	2.95%
17	2853.97	0.00	2.95%
18	2878.60	0.00	3.83%
19	2837.41	0.00	2.35%
20	2786.77	0.00	0.52%

*[(TS − Optimum TS)/(Optimum TS) ∗ 100].

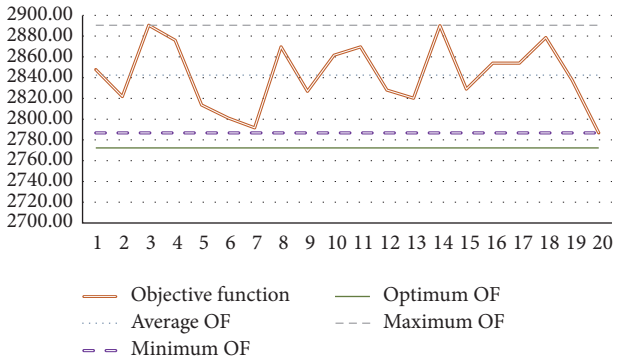

Figure 4: Analysis of the generated solutions of CSA for the TSP problem with 100 cities.

On the other hand, as it is shown in Figure 5, the average rate of 2.53% with a minimum percentage of 0.70% and maximum value of 4.26% indicates that CSA can generate solutions which are near to optimal solutions for small and big problems. Moreover, a narrow range in this regard indicates the robust manner of this algorithm in generating final solutions.

To check the process of running CSA, the total percentage of improvement in the quality of Cell Size (objective function) is reported. This value helps us to get a general overview about the quality of initial and final cells and the rate of improvement (Figure 6). However, we should note that the

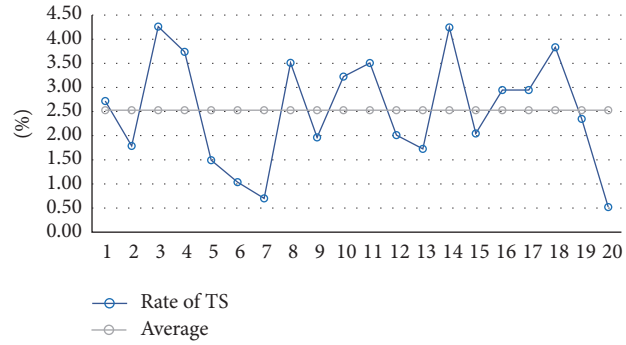

Figure 5: Analysis of the rate of TS for the generated solutions in comparison to the optimal one.

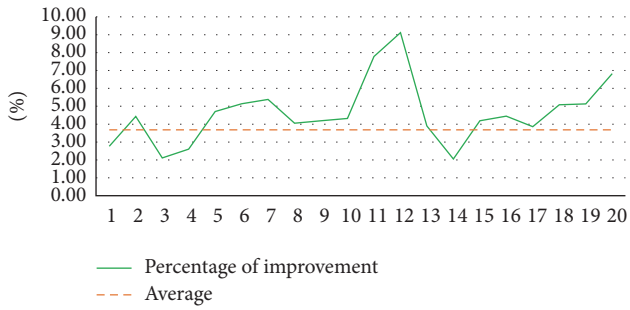

Figure 6: General analysis of the percentage of improvement during algorithm running time.

initial solution is an infeasible solution and the final solution is a feasible one. An average improvement rate of 4.60% indicates the robustness and effectiveness of CSA. As a matter of fact, the average percentage of 4.60% in solving a large TSP problem is valuable where a small amount of improvement in this type of problems usually needs an effective and time consuming approach.

It is worth noting that, as an effective approach, the concept of sparse matrix is used to solve the TSP problem where in the original form the number of CPEs and parameters is high and using the normal matrix may decrease the effectiveness of the algorithm particularly for large-scale problems.

We can see the graphical view of the best tour obtained by the proposed algorithm for the TSP problem with 100 cities in Figure 7.

6. Conclusion

In this study, an innovative algorithm is proposed which focuses on tumor space (infeasible region) to investigate and explore the cells. Intelligently, a nature-inspired algorithm based on chemotherapy cancer treatment is developed with a mathematical background according to the effective and exact solving techniques for optimization problems and the proposed approaches in the literature.

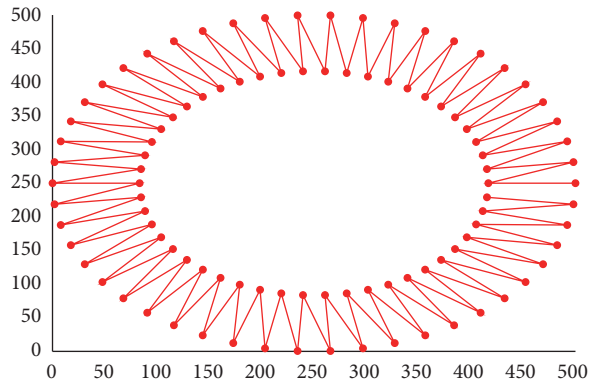

Figure 7: The graphical view of the best tour obtained by CSA.

A firm background for the proposed algorithm directs it towards generating appropriate healthy cells. This algorithm consists in searching the space far from the tumor border (convex hall) and investigating the tumor towards the outside of the tumor using CA, RC, and rational random directions. Theoretically, two TTP and TS criteria are applied for assessing each pair of cells where an effective approach is proposed to convert each cancer cell to its corresponding healthy one which can be applied in each iteration and part of the algorithm. However, based on the structure of our mathematical model, it is possible to develop other strategies in different parts of the algorithm, such as using greedy approaches for generating initial cancer cells or converting a bad cell to a healthy one when solving TSP.

The following can be some intriguing areas for future research regarding the proposed algorithm:

(i) Solving other categories of optimization problems such as nonlinear ones

(ii) Combining the algorithm with other well-known ones to develop a hybrid algorithm

(iii) Developing a toolbox and software-based solver to solve a wide range of parameters without having to effect drastic changes to programming codes.

Competing Interests

The authors declare that they have no competing interests.

References

[1] R. Neapolitan and K. Naimipour, *Foundations of Algorithms Using C++ Pseudo Code*, Jones & Bartlett Learning, Burlington, Mass, USA, 3rd edition, 2004.

[2] S. Binitha and S. S. Sathya, "A survey of bio inspired optimization algorithms," *International Journal of Soft Computing and Engineering*, vol. 2, no. 2, pp. 137–151, 2012.

[3] American Cancer Society, *Chemotherapy What It Is, How It Helps*, A.C. Society, Atlanta, Ga, USA, 2013.

[4] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[5] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Computers & Operations Research*, vol. 13, no. 5, pp. 533–549, 1986.

[6] F. Glover and C. McMillan, "The general employee scheduling problem: an integration of MS and AI," *Computers and Operations Research*, vol. 13, pp. 563–573, 1986.

[7] S. Kirkpatrick, J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[8] M. Dorigo, *Optimization, Learning and Natural Algorithms*, Politecnico di Milano, Milan, Italy, 1992.

[9] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia, 1995.

[10] Z. Beheshti and S. M. H. Shamsuddin, "A review of population-based meta-heuristic algorithm," *International Journal of Advances in Soft Computing and Its Applications*, vol. 5, no. 1, pp. 1–35, 2013.

[11] F. Glover, "Heuristics for integer programming using surrogate constraints," *Decision Sciences*, vol. 8, no. 1, pp. 156–166, 1977.

[12] S. F. Smith, *A Learning System Based on Genetic Adaptive Algorithms*, University of Pittsburgh, 1980.

[13] J. D. Farmer, N. H. Packard, and A. S. Perelson, "The immune system, adaptation, and machine learning," *Physica D: Nonlinear Phenomena*, vol. 22, no. 1–3, pp. 187–204, 1986.

[14] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, Mass, USA, 1988.

[15] I. Axcelis, "Evolver, the world's first commercial GA product for desktop computers," *The New York Times*, 1989.

[16] P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms," Caltech Concurrent Computation Program, Technical Report C3P 826, 1989.

[17] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: formulation, discussion and generalization," in *Proceedings of the 5th International Conference on Genetic Algorithms*, pp. 416–423, Urbana-Champaign, Ill, USA, 1993.

[18] R. Battiti and G. Tecchiolli, "The reactive tabu search," *ORSA Journal on Computing*, vol. 6, no. 2, pp. 126–140, 1994.

[19] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[20] R. Y. Rubinstein, "Optimization of computer simulation models with rare events," *European Journal of Operational Research*, vol. 99, no. 1, pp. 89–112, 1997.

[21] E. Taillard and S. Voss, "POPMUSIC: partial optimization metaheuristic under special intensification conditions," Tech. Rep., Institute for Computer Sciences, heig-vd, Yverdon-les-Bains, Switzerland, 1999.

[22] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[23] O. Hanseth and M. Aanestad, "Bootstrapping networks, communities and infrastructures. On the evolution of ICT solutions in heath care," in *Proceedings of the 1st International Conference on Information Technology in Health Care (ITHC '01)*, Erasmus University, Rotterdam, The Netherlands, 2001.

[24] S. Nakrani and C. Tovey, "On honey bees and dynamic server allocation in internet hosting centers," *Adaptive Behavior*, vol. 12, no. 3-4, pp. 223–240, 2004.

[25] K. N. Krishnanand and D. Ghose, "Detection of multiple source locations using a glowworm metaphor with applications to collective robotics," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '05)*, pp. 84–91, IEEE, June 2005.

[26] D. Karaboga, "An idea based on honey bee swarm for numerical numerical optimization," Tech. Rep. TR06, Computer Engineering Department, Engineering Faculty, Erciyes University, 2005.

[27] O. B. Haddad, A. Afshar, and M. A. Mariño, "Honey-bees mating optimization (HBMO) algorithm: a new heuristic approach for water resources optimization," *Water Resources Management*, vol. 20, no. 5, pp. 661–680, 2006.

[28] H. Shah-Hosseini, "Problem solving by intelligent water drops," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 3226–3231, Singapore, September 2007.

[29] E. Atashpaz-Gargari and C. Lucas, "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 4661–4667, Singapore, September 2007.

[30] A. Mucherino and O. Seref, "Monkey search: a novel metaheuristic search for global optimization," in *Proceedings of the AIP Conference Proceedings, Data Mining, Systems Analysis and Optimization in Biomedicine*, vol. 953, pp. 162–173, Gainesville, Fla, USA, March 2007.

[31] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, 2008.

[32] A. Husseinzadeh Kashan, "League Championship Algorithm: a new algorithm for numerical function optimization," in *Proceedings of the International Conference on Soft Computing and Pattern Recognition (SoCPaR '09)*, pp. 43–48, Malacca, Malaysia, December 2009.

[33] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[34] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, Coimbatore, India, December 2009.

[35] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, J. R. Gonzalez, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor, Eds., vol. 284 of *Studies in Computational Intelligence*, pp. 65–74, Springer, Berlin, Germany, 2010.

[36] H. Shah-Hosseini, "Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation," *International Journal of Computational Science and Engineering*, vol. 6, no. 1-2, pp. 132–140, 2011.

[37] K. Tamura and K. Yasuda, "Spiral dynamics inspired optimization," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 15, no. 8, pp. 1116–1122, 2011.

[38] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems," *Computer-Aided Design*, vol. 43, no. 3, pp. 303–315, 2011.

[39] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831–4845, 2012.

[40] P. Civicioglu, "Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm," *Computers & Geosciences*, vol. 46, pp. 229–247, 2012.

[41] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering with Computers*, vol. 29, no. 1, pp. 17–35, 2013.

[42] A. H. Gandomi, X.-S. Yang, S. Talatahari, and A. H. Alavi, "Firefly algorithm with chaos," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 1, pp. 89–98, 2013.

[43] A. Kaveh and V. R. Mahdavi, "Colliding bodies optimization: a novel meta-heuristic method," *Computers & Structures*, vol. 139, pp. 18–27, 2014.

[44] Z. Beheshti and S. M. Shamsuddin, "CAPSO: centripetal accelerated particle swarm optimization," *Information Sciences*, vol. 258, pp. 54–79, 2014.

[45] A.-B. Meng, Y.-C. Chen, H. Yin, and S.-Z. Chen, "Crisscross optimization algorithm and its application," *Knowledge-Based Systems*, vol. 67, pp. 218–229, 2014.

[46] B. Javidy, A. Hatamlou, and S. Mirjalili, "Ions motion algorithm for solving optimization problems," *Applied Soft Computing Journal*, vol. 32, pp. 72–79, 2015.

[47] J. J. Q. Yu and V. O. K. Li, "A social spider algorithm for global optimization," *Applied Soft Computing*, vol. 30, pp. 614–627, 2015.

[48] R. V. Rao, "Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 7, no. 1, pp. 19–34, 2016.

[49] M. H. Salmani and K. Eshghi, "A Smart Structural Algorithm (SSA) based on infeasible region to solve mixed integer problems," *International Journal of Applied Metaheuristic Computing*, vol. 8, pp. 24–44, 2017.

[50] AIULd Bruxelles, *TSPTW-Benchmark Problems*, Université libre de Bruxelles, Brussels, Belgium, 2006.

[51] G. Pataki, *The Bad and the Good-and-Ugly: Formulations for the Traveling Salesman Problem*, Department of Industrial Engineering and Operation Research, Columbia University, 2001.

[52] I. GitHub, "ViktorCollin lagt till test cases," in *TSP Problem, San Francisco*, Millions of developers use GitHub to build personal projects, support their businesses, and work together on open source technologies, 2012.