AI Communications 28 (2015) 591–603
DOI 10.3233/AIC-140652
IOS Press

Election algorithm: A new socio-politically inspired strategy

Hojjat Emami [a,*] and Farnaz Derakhshan [b]

[a] *Computer Engineering Department, Miandoab Branch, Islamic Azad University, Miandoab, Iran*
E-mail: hojjatemami@yahoo.com

[b] *Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran*
E-mail: derakhshan@tabrizu.ac.ir

Abstract. This paper describes Election Algorithm (EA), an optimization and search technique, inspired by presidential election. The EA is an iterative population based algorithm, which works with a set of solutions known as population. Each individual of the population is called a person and can be either a candidate or a voter. These persons form a number of electoral parties in the solution space. Advertising campaign is the core of this algorithm and contains three main steps: positive advertisement, negative advertisement and coalition. During positive advertisement, the candidates introduce themselves through reinforcing their positive images and qualities. In the negative advertisement, candidates compete with each other to increase their popularity and defame their opponents. Also in some cases, the candidates that have similar ideas can confederate together in order to increase the chance of success of the united party. Advertisements hopefully cause the persons to converge to a state of solution space that is the global optimum. All these efforts lead up to election day (stopping condition). On election day, the candidate who attained the most votes, is announced as the winner and equals to the best solution that is found for the problem.

In order to evaluate the performance of EA, we compared this algorithm with Continuous Genetic Algorithm (CGA), Comprehensive Learning Particle Swarm Optimizer (CLPSO), Adaptive Differential Evolution Algorithm (JDE) and Covariance Matrix Adaptation Evolution Strategy (CMA-ES) in finding the global optimum of four mathematical benchmark examples. Our experiments demonstrate the superiority of the EA for benchmark examples.

Keywords: Election algorithm, global optimization, meta-heuristic algorithms, presidential election

1. Introduction

Optimization means making something better. In other words, optimization is the process of finding an alternative approach with the highest efficiency by maximizing desired factors [15,23].

So far, many algorithms have been developed for solving various optimization problems. Two main groups of these algorithms include exact and heuristic algorithms. Exact algorithms guarantee to find optimal solutions in solving the small or moderately scale size optimization problems. Some of the well-known exact algorithms are branch and bound, dynamic programming, Lagrangian relaxation based algorithms, and linear and integer programming based methods [23]. The drawbacks of mathematical and exact algorithms are in using them for large-scale and non-linear complex optimization and engineering problems [23,27]. To overcome the limitation of exact algorithms, several heuristic algorithms have been developed. These algorithms are not guaranteed to find optimum solutions, but they can find near-optimum solutions in a limited time. Heuristic algorithms have shown good performance in solving large-scale, non-differentiable and complex nonlinear problems [6,23,27].

Various heuristic algorithms have been proposed and successfully applied to many scientific and practical fields. Beheshti and Shamsuddin [2] have been reviewed a number of population-based meta-heuristic algorithms. Some of the well-known algorithm that fall into the category of meta-heuristic algorithms include: Simulated Annealing (SA) [4], Tabu Search (TS) [10], Scatter Search (SS) [12], Iterated Local Search (ILS) [11], Variable Neighborhood Search (VNS) [14], and other population based evolutionary algorithms such as Genetic Algorithm (GA) [15], Particle Swarm Optimization (PSO) [20], Ant Colony Optimization (ACO) [5], Intelligent Water Drops (IWD) [26], River For-

*Corresponding author. E-mail: hojjatemami@yahoo.com.

mation Dynamics (RFD) [24], Imperialist Competitive Algorithm (ICA) [1], Artificial Bee Colony (ABC) [18], Charged System Search (CSS) [19], Firefly Algorithm (FA) [28], Gravitational Search Algorithm (GSA) [25] and Bat Algorithm (BA) [29]. These available search and optimization algorithms have been used in many fields such as economic, operations research, control engineering, chemistry and physic problems, business and industrial applications, computer science problems, image processing applications, speech recognition problems, data mining problems and mechanical design problems.

Most of optimization and search algorithms are inspired by some natural phenomena, such as ACO algorithm that is inspired by foraging behavior of real ants. ACO is useful for solving problems that require finding the shortest path. Another example is ABC algorithm, which models the intelligent behavior of honeybees. Some other meta-heuristic algorithms are non-natural inspired algorithms such as Tabu search and Iterated Local Search algorithms.

In this paper, we introduced another search and optimization algorithm, which is inspired by the socio-political phenomenon of presidential election. This algorithm is called Election Algorithm (EA) which is briefly described as follows.

EA begins its search and optimization process with a population of solutions. Each individual in the population is called a person and can be either a candidate or a voter. Forming a number of parties in the solution space, people can participate in their preferred party. Then these parties begin their advertising campaign. Advertising campaign forms the basis of this algorithm and causes the persons to converge to the global optimum of solution space. During advertisements, the popular candidates attract more voters using various techniques. Therefore, the unpopular ones lose their supporters and might resign from the election arena. Advertisement causes the persons to converge to the global optimum of solution space. On election day, voters cast their votes and the candidate that attains the most votes would be announced as the winner.

This paper describes how the EA algorithm is implemented. In addition, we evaluated and compared the performance of the proposed method with four well-known optimization algorithms including CGA [15], CLPSO [21], JDE [3] and CMA-ES [16]. These algorithms are chosen because they are popular population-based and swarm intelligence as the EA algorithm, and have been broadly applied in many applications as well. The results reveal the superiority of the proposed algorithm for the benchmark examples.

Having this short introduction, the rest of this paper would be organized as follows: After describing the election process in Section 2, we comprehensively introduce the working principle of the EA in Section 3. Then, in Section 4, the proposed algorithm is evaluated on some mathematical benchmark functions, and the simulation results are compared to the CGA, the CLPSO, the JDE and the CMA-ES algorithms. Finally, Section 5 makes conclusions and presents some future works.

2. General description of election

An election is a formal socio-political mechanism of selecting a person for public office or rejecting or accepting a political proposal by voting. Elections and voting are essential for a successful democracy. The people of society get to elect their representatives to rule them. These representatives supervise laws that directly affect the quality of people's life. Elections are held in limited geographical areas in given predefined times. In modern democracies, election is used as an important tool for selecting representatives [8,9].

In history, elections were used in ancient Athens, in Rome and in the medieval period to select Popes and Holy roman emperors. Also in the early medieval Rashidun Caliphate, ancient Arabs have used election to select their caliph, Uthman and Imam Ali. The origins of elections in the modern world emerged in the beginning of the 17th century when the representative government took hold in Europe and North America [7,22].

One of the most important elections that are held in most countries is presidential election [8]. The presidential election is held on national level and is a single winner election. In the most countries the president is the highest official elected by direct and popular vote. There are various types of election to select the president of a country. One of the free and fair election methods is direct election in which voters choose freely their favorite candidate. The method by which the winners of a direct election are selected depends on the electoral system used. The electoral system is responsible to implement election in a fair and sound way.

A most simple form of a presidential election involves the following steps. First, nominates called candidates declare their candidacy voluntarily. The candi-

dates want to be the president and hope their parties support them. Then, candidates get required supports, and ready for running a campaign all around the country. Political campaign is the process of gathering public supports for a candidate. The goal of a campaign is to provide as much information about the candidate and the party's platform to as many people as possible [9]. Beginning advertisement time, candidates through different techniques such as direct mail, personal appearance among voters and talking to them, printed materials and even through Internet, publicize themselves and compete with one another to increase their supporters.

Obviously, the voting outcomes are greatly influenced by the type of advertisements that voters are exposed to them and the voters' characteristics.

In general, advertisements are two types: positive and negative. During positive advertisement, candidates introduce their qualifications, agendas and ideas to the public. However, in negative advertisement, candidates focus exclusively on the negative aspects of the other opponents while emphasize on positive information about themselves [13]. Sometimes in campaign trial, two or more than two political parties that have same ideas and goals can union together. This process is called parties' coalition. All these efforts lead up to election day. On election day people go to the polls and choose their favorite candidate to be the next president. Then the polls are closed and after counting the votes, a candidate who collects the most votes is declared as the winner. In most elections, in order to win, a candidate must obtain at least an absolute majority of the votes.

In the following, an evolutionary algorithm called Election Algorithm (EA) developed based on the presidential election process is introduced.

3. Our proposed algorithm: Election algorithm

This section explains the Election Algorithm (EA) in detail. This algorithm is called election algorithm, since its procedure is similar to the election process in the real world. Like many other evolutionary algorithms, this algorithm is rely on an intelligent search of a large but limited solution space using its actuators including positive and negative advertisement, and coalition.

The EA is a multi-agent algorithm in which each agent is a person and can be either a candidate or a voter. Candidates together with their voters (supporters) form some political parties. At the beginning of the algorithm, all voters are divided among candidates based on their similar opinions and ideas. After forming initial parties, a candidate in each party begins his/her advertising campaign. Advertising campaign contains two types of advertisements including positive and negative advertisement. During the positive advertisement, candidates convey their agendas and ideas to the voters and try to attract the voters toward themselves. Then candidates use negative advertisement (will be described in Section 3.3.1) to increase their own popularity and decrease the popularity of other candidates. The negative advertisement has important effect on the result of election. Unpopular candidates lose their supporters and might resign from election arena. During advertising campaign, the candidates that have the same opinions and ideas might unite and create a new party which is a combination of these parties. The coalition increases the chance of the success of united candidates.

The EA works iteratively by successively applying three operators – positive advertisement, negative advertisement and coalition – in each iteration until a termination condition is satisfied. These operators will cause all the candidates to converge to global optimum of the eligibility function. In this paper, the termination condition is set to be the maximum iteration number. EA algorithm ends on election day. Then, at the end of the algorithm, the candidate who attained the majority of votes will be announced as the winner. The winner candidate is equal to the best solution found for the optimization and search problem.

In the following, first we select the variables and eligibility function in Section 3.1. Then, in Section 3.2, we explain how parties form in this algorithm. Finally, Section 3.3 describes three types of advertisement followed by termination condition of the algorithm. In addition to be more clear, a path through the EA's components is shown as a flowchart in Fig. 1. Each block in this flowchart is described in detail in the following sections. Also the pseudo code of the proposed algorithm is shown in Fig. 2.

3.1. Selecting the variable representation and eligibility function

The EA begins its work by defining the optimization variables, the eligibility function, and the eligibility. The EA in each iteration works with a set of solutions collectively known as the population. Usually, in the absence of any knowledge of the problem domain,

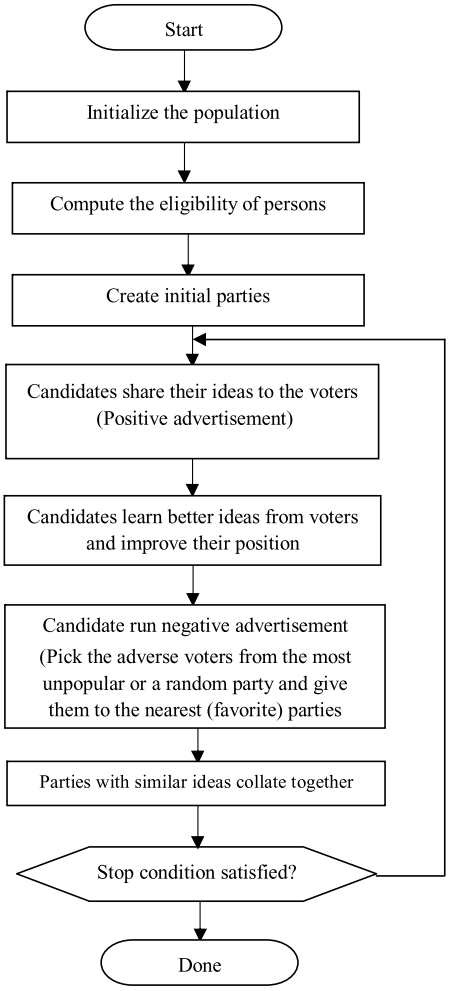


Fig. 1. Flowchart of the election algorithm.

this population is generated randomly. Each individual of the population is called a person and indicates a potential solution to the problem.

In an N_{var}-dimensional (N_{var} variables) problem, each person is formed an array of variable values to be optimized. If the person has N_{var}-variables given by $x_1, x_2, \ldots, x_{N_{\text{var}}}$ then the person is written as an $1 \times N_{\text{var}}$ element row vector such as follow

$$person = [x_1, x_2, \ldots, x_{N_{\text{var}}}].$$

Each variable in this array can be interpreted as a socio-political characteristic of a person such as language, culture, religion, nationality, economical policy and other opinions. The variable values in the person can be represented by all kinds of characters such as numbers, alphabets and any other form of data representation symbols.

Election Algorithm

BEGIN
Generate initial population;
Compute eligibility of each individual;
Create parties: initial candidates and their supporters
REPEAT //* advertising campaign (ad days) *//
For candidate size do
// positive advertisement
Candidates advertise their plans and improve their stance by learning new ideas;
// negative advertisement
Candidates try to attract the supporters from other parties toward themselves;
// coalition
Candidates collate if they have same ideas;
// revision the condition of parties
Reevaluate the eligibility of candidates;
END FOR
UNTIL population has converged //* (Election Day) *//
END

Fig. 2. Pseudo code of the election algorithm.

In the EA, each person must be assigned an eligibility value, which is related to the objective function of the problem. An eligibility function is used to assess the ability and qualifications of persons. In the optimization problems, the main goal is to find a solution having the minimum cost. Thus, in the EA, the persons who have smaller cost then larger eligibility values are assigned to them.

The eligibility of an individual is found by evaluating the eligibility function f at the variables $x_1, x_2, \ldots, x_{N_{\text{var}}}$ considering the related objective function. Therefore, we have:

$$\begin{aligned} eligibility &= f(person) \\ &= f(x_1, x_2, \ldots, x_{N_{\text{var}}}). \end{aligned} \tag{1}$$

In maximization problems, the eligibility of person is equal to the value of problem objective function. However, in minimization optimization problems, the persons with larger objective function value will get smaller eligibility value. Therefore, in the cost minimization problems, the following transformation is used:

$$\begin{aligned} eligibility &= -f(person) \\ &= -f(x_1, x_2, \ldots, x_{N_{\text{var}}}). \end{aligned} \tag{2}$$

3.2. Forming initial parties

To begin the EA, we define an initial population of N_{Pop} persons. The initial population is an $N_{\text{Pop}} \times N_{\text{var}}$ matrix filled with randomly generated persons. Each row in the matrix is representative of a person and is being an $1 \times N_{\text{var}}$ array of variable values. Then the worth of each one is evaluated by the eligibility function. So at this point, the persons are passed to the eligibility function to assess their capability. After forming the initial population, we divide the entire persons into some political parties in the solution space. To fulfill this aim, the N_c persons randomly are selected to be candidates and the remaining N_v persons will form the initial supporters (or voters) of these candidates. The candidate rate, C_r, is the fraction of N_{Pop} that selected as the initial candidates. Thus, the number of persons that are selected as initial candidates is given by

$$N_c = \lceil C_r \times N_{\text{Pop}} \rceil. \tag{3}$$

It is arbitrary to decide how many persons to select as the initial candidates. Letting only a few persons select as initial candidates may limits the convergence rate of the algorithm and causes the algorithm get stuck at local optimums. Also selecting too many persons as initial candidates increases the workload of the algorithm.

In order to choose a proper C_r value, we ran the algorithm several times. After experimental study, we concluded that 5–10% of the population is an appropriate value as the initial number of candidates in forming initial parties. The remaining $N_v = N_{\text{Pop}} - N_c$ persons are the total number of initial supporters (voters) of the mentioned candidates.

It is clear that the supporters are largely similar to their relevant candidate in politic, diplomacy, religion, beliefs and ideology. We used this characteristic to create initial parties. In the EA, the beliefs and ideas of the persons are determined using the eligibility function. All the supporters are divided among the candidates based on the similarity of beliefs and ideas. Thus, the eligibility distance between persons and candidates are computed to divide persons among candidates. To do so, Euclidean distance metric is used as similarity measure that is given by

$$\text{dist}(v_i, c_j) = \sqrt{\sum_{j=1}^{N_c} (E_{v_i} - E_{c_j})^2}, \tag{4}$$

where N_c is the number of all candidates, E_{v_i} indicates the eligibility of ith voter, E_{c_j} indicates the eligibility of jth candidate, and $\text{dist}(v_i, c_j)$ indicates the eligibility distance between the v_i voter and the c_j candidate. Each voter is assigned to the party whose the eligibility of candidate is closest to.

In other words, person v_p is considered as a supporter of candidate c_i, if the following predicate holds:

$$P_i = \{v_p : \ \|E_{v_p} - E_{c_i}\| < \|E_{v_p} - E_{c_i}\| \quad \forall 1 \leqslant j \leqslant N_c\}, \tag{5}$$

where P_i is the ith party, E_{v_p} and E_{c_i} indicates the eligibility of voter v_p and candidate c_i, respectively, $\|E_{v_p} - E_{c_i}\|$ is the selected dissimilarity measure between voter v_p and candidate c_i, where indicates the eligibility distance between them, and N_c is the number of all candidates. Here, each person v_p is assigned to exactly one party P_i.

3.3. Advertisement

Once the initial parties are formed, candidates begin their advertising campaign. In campaign trail, the popular candidates are reinforced; the unpopular candidates are weakened and may be resigned from the election arena. As mentioned before, advertisement is the main part of the EA and contains three main steps: positive advertisement, negative advertisements and coalition. The following subsections describe the different components of advertisements and how these mechanisms are implemented in the solution space.

3.3.1. Positive advertisement

As mentioned before, the aim of positive advertisement is to introduce a candidate through focusing on his/her abilities or stance on issues. During positive advertisement, candidates begin to expose their plans and ideas to voters and try to influence the decisions made by voters. Candidates attempt to create a lasting impression with the voters.

We have modeled this process by conveying some characteristics of the candidate to the voters. To fulfill this aim, in each party, the N_s variable values (characteristics) of the candidate are randomly selected. Next random numbers are chosen to select the position of variables in supporters (voters) to be replaced. Then, the selected variables from candidate are replaced with the selected variables of the supporters.

In this paper, selection rate (X_s) is selected as a random value in range [0, 1]. In other words, in each it-

eration, the number of variable values that transferred from a candidate toward its supporters is given by

$$N_s = \lceil X_s \times S_c \rceil, \tag{6}$$

where N_s indicates the number of selected variables of candidate to be replaced, X_s is the selection rate, and S_c is the total number of variables of candidate.

It is clear that, in an election party based on the eligibility distance between candidate and its relevant supporters, the effectiveness of advertisement varies. Therefore, it is reasonable to consider the effect of eligibility distance between candidate and its relevant supporters. To fulfill this aim, we defined eligibility distance coefficient (ω_e) as follows

$$\omega_e = \frac{1}{\text{dist}(E_c - E_v) + 1}, \tag{7}$$

where E_c and E_v are the eligibility of candidate c and voter v, respectively. In this paper, Euclidean distance metric is used to compute distance between E_c and E_v.

In advertising process, after choosing N_s and computing ω_e, the value of selected variables from candidate are multiplied in coefficient ω_e and then replaced with the selected variables of the relevant supporters. In other word, given $s_{i_{\text{old}}}$ be the value of ith selected variable of the supporter (that should be replaced) before advertising process, then after advertising, the new value of this variable is given as follows:

$$s_{i_{\text{new}}} = \omega_e \times s_{i_{\text{old}}}. \tag{8}$$

Based on Eq. (8), in advertisement process, the closer supporters are much more affected by their relevant candidate than farther supporters.

If a candidate has truly excellent plans and ideas, the number of his supporters will be more. During advertising, candidates try to learn good opinions and ideas from their supporters. From optimization point of view, the supporters move toward the more eligible (more popular) candidates. While positive advertisement, if a supporter has better ideas (i.e. has higher eligibility) than its relevant candidate, they can exchange these ideas with themselves. This fact increases the quality of party's plans.

3.3.2. Negative advertisement

Candidates through negative advertisement try to attract the supporters of other parties toward themselves. Negative campaigning leads to an increase in the popularity of popular parties and conversely a decrease in the popularity of unpopular parties. In the implementations of this algorithm, contrast advertisement is used among different negative campaigning techniques. To do this, candidates travel to the different areas of solution space (between other parties) and they try to attract those voters toward themselves using negotiation.

To model the contrast advertisement mechanism two new terms are defined, "defender party" and "attacker party". A party that other parties are going to take the attention of its supporters, is called *defender party*. The aggressor candidates are called *attacker party*. Obviously, in each party, if the supporters do not think as their relevant candidate and do not agree his/her ideology, then they change their willingness and easily influenced by the other candidates. There are various criteria to select a party as a defender. In this paper, randomly a party selected as the defender party.

For the implementation of the contrast advertisement, a number of supporters of the defender party are selected. Then, a race is taken place between attacker parties to possess these voters. There are various approaches to select the mentioned supporters. One of the simplest methods is to select some supporters at random. Another method is to select the farthest supporters. In this paper, the later approach is used. To do this, first, in the defender party, the distance of the eligibility between the supporters and the candidate is computed.

By using Euclidean distance metric, the distance is given by

$$\text{dist}(v_i, c_d) = \sqrt{\sum_{j=1}^{N_v} (E_{v_i} - E_{c_d})^2}, \tag{9}$$

where N_v is the number of all voters in defender party, c_d indicates the candidate of defender party and v_i indicates the ith supporter of the defender party. E_{c_d} and E_{v_i} are the eligibility of candidate c_d and ith supporter, respectively. Then, 5% of the farthest supporters are selected based on the distance measure.

Then the distance between selected supporters and all the attacker candidates are computed. These supporters are assigned to the closest candidates. Figure 3 depicts the big pictures of the modeled negative advertisement mechanism between defender and attacker parties.

Sometimes parties in the solution space may converge too quickly into the local optima of the solution space. To avoid the problem of fast convergence, the algorithm must explore other areas of solution space. Negative advertisement performs this work and keeps the EA away from converging too fast before exploring the entire solution space.

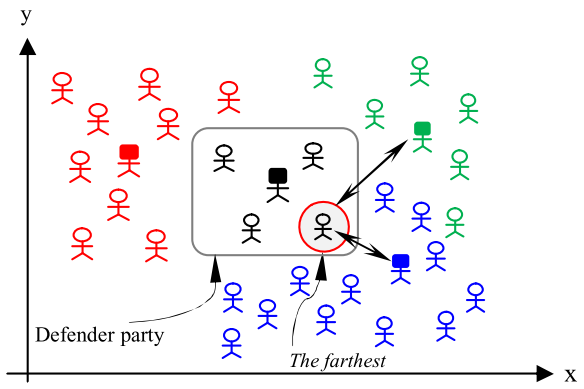


Fig. 3. Negative advertisement among candidates. The popular candidates will attract more supporters toward themselves. The defender party is shown in black color. The farthest voter is assigned to the nearest candidate (in this case the candidate by blue color). (The colors are visible in the online version of the article; http://dx.doi.org/10.3233/AIC-140652.)

3.3.3. Coalition

Similar to the process of candidates coalition in a real election, sometimes two or more than two candidates in the solution space become more closer together. In that case, they can join and create a new party. Therefore, some candidates leave the campaign and join to another candidate who is called "leader". The candidate who leaves the election arena is called "follower". The follower candidates collate to the leader one and encourage their supporters to follow the leader. All of the supporters of the follower candidates become the supporters of the leader one.

Different criteria can be defined to model the coalition mechanism and to specify a candidate to be either a leader or a follower. In our implementation, among the candidates that wish to collate together, a candidate is picked at random to be the leader candidate and the remaining is considered as the followers.

Figure 4(a) and (b) illustrate a great picture of the coalition process of parties before coalition and after being collated, respectively. In Fig. 4(a), the candidates that are shown by blue, green and red color, approximately are similar, and they want to collate together. As shown in Fig. 4(b), after coalition the leader candidate is shown by blue color and the follower candidates are considered as the new supporters of the leader shown by circular head. In addition, all the new supporters of the leader are shown by blue color.

3.3.4. Election day (stopping condition)

Until a termination condition is not satisfied three operators – positive advertisement, negative advertisement and coalition – are applied to update the population. Eventually a candidate that attains the most votes will declare as the winner and is equal to the best solution, which is found for the problem. In the implementation of this paper, the maximum iteration number is considered as the stopping condition of the algorithm.

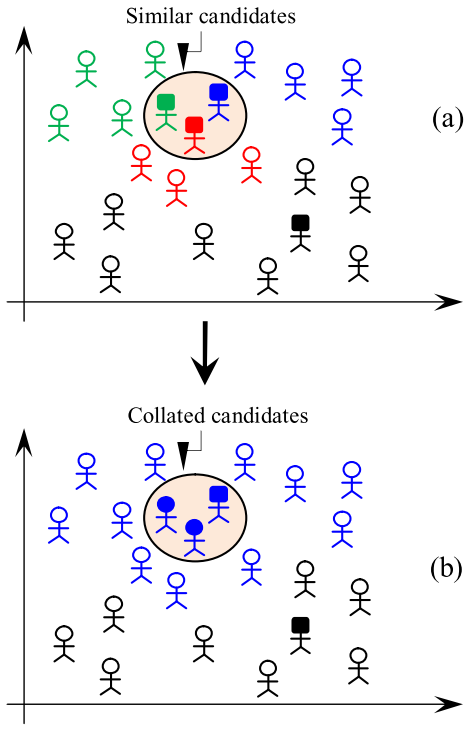


Fig. 4. Coalition between parties in the search space. (a) The parties before coalition; (b) The parties after coalition. (The colors are visible in the online version of the article; http://dx.doi.org/10.3233/AIC-140652.)

In the next section, we will give intuitive reasoning of why our proposed algorithm is a useful optimization and search technique. In order to assess the potential of the proposed algorithm, it is applied on some mathematical benchmark functions as described in the following.

4. Experimental results

To prove that the proposed EA technique works effectively, we used four benchmark functions [17,18]. We shall describe the details of these functions a little later. Since we are trying to find the global minimum of these problems, the cost function is as the negative of the eligibility function (presented in Eq. (2)), in order to put it into the form of a minimization algorithm.

In the following, the algorithms which include the proposed EA, CGA [15], CLPSO [21], JDE [3] and

CMA-ES [16] are compared in accordance with the benchmark functions. Each algorithm has its own parameters that affect its performance in terms of convergence rate and quality of solutions. Common control parameters of the algorithms are population size and the number of maximum generation. In our experiments, maximum number of generations is set to 1000 for EA and CGA, and 10,000 for CLPSO, JDE and CMA-ES. The population size is chosen to be 100 for all algorithms. Table 1 shows the parameter values adopted for each of the CGA, CLPSO, JDE, CMA-ES and EA algorithms, respectively.

4.1. Benchmark functions

There are many standard test functions for validating new algorithms. In this paper, four popular mathematical benchmark functions are employed to validate and compare our proposed EA algorithm with other algorithms. These are some well-known mathematical problems. Table 2 summarizes the characteristics of these problems including the general form, the domain, the dimension and the global minimum. Also the 3D plots of these benchmark functions are illustrated in Fig. 5. In these problems, the main goal is to minimize the value of cost function. Detailed information about these problems has been provided in [17,18].

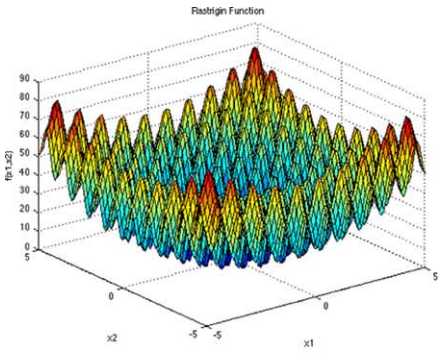
- f_1 = *Rastrigin*: The value of this function is 0 at its global minimum $(0, 0, \ldots, 0)$. The initial range for the function is set to $[-15, 15]$. Rastrigin is a multimodal function and has several local minima. So that an optimization algorithm easily can be trapped in a local optimum instead of reaching global optimal.
- f_2 = *Griewank*: This function is a continuous, differentiable, non-separable, scalable and multimodal problem. Initialization range for this function is $[-600, 600]$. The global minima of this function is located at $(0, 0, \ldots, 0)$ and its value is 0. The Griewank function has many widespread local minima, which are regularly distributed.
- f_3 = *Rosenbrock*: This function also referred to as the Valley or Banana function is a popular test problem for gradient-based optimization algorithms. Rosenbrock function is a popular benchmark function and repeatedly used to test the performance of various optimization algorithms. The function is continuous, differentiable, non-separable, scalable and unimodal. The global minimum lies in a narrow, parabolic valley. The value of this function is 0 at its global minimum $(1, 1, \ldots, 1)$. The initial range for the function is set to $[-5, 10]$. However, even though this valley is easy to find, convergence to the minimum is difficult.
- f_4 = *Ackley*: This function is a continuous, differentiable, non-separable, scalable and multimodal. Ackley function is widely used for testing optimization algorithms. This function has several local minima and it poses a risk for optimization algorithms, particularly hill climbing algorithms, to

Table 1
Control parameters of the related algorithms used in the tests

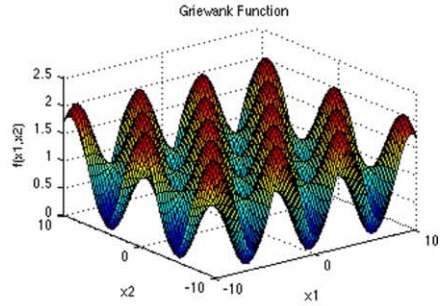
Algorithm	Control parameters
CGA	Mutation rate = 0.30, selection rate = 0.50
PSO	Cognitive parameter = 1, social parameter = 3, construction factor = 1
CLPSO	$c = 1.49445$, $m = 0$, $p_c = 0.5 \cdot \frac{e^t - e^{t(1)}}{e^{t(p_S)} - e^{t(1)}}$, where $t = 0.5 \cdot (0 : \frac{1}{p_S - 1} : 1)$
CMA-ES	$\sigma = 0.25$, $\mu = \lfloor \frac{4 + \lfloor 2 \cdot \log(N) \rfloor}{2} \rfloor$
JDE	$f_{\text{initial}} = 0.5$, $CR_{\text{initial}} = 0.90$, $\tau_1 = 0.1$, $\tau_2 = 0.1$
EA	$N_c = 0.7 \times P_{\text{size}}$, $N_v = P_{\text{size}} - N_c$, coalition rate = 0.2, selection rate = 0.30

Table 2
Characteristics of benchmark examples

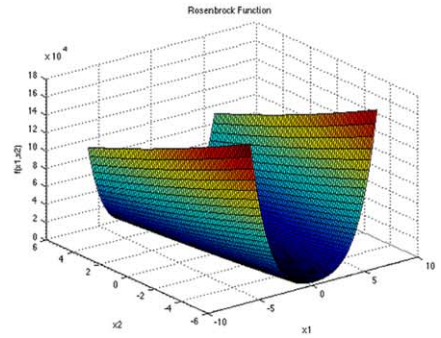
No.	Definition	Interval	Dimension	Global minimum
1	$f_1 = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$	$[-15, 15]$	$1, \ldots, D$	$f(0, 0, \ldots, 0) = 0$
2	$f_2 = \frac{1}{4000}(\sum_{i=1}^{D} (x_i^2)) - (\prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}})) + 1$	$[-600, 600]$	$1, \ldots, D$	$f(0, 0, \ldots, 0) = 0$
3	$f_3 = \sum_{i=1}^{D} 100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2$	$[-5, 10]$	$1, \ldots, D$	$f(1, 1, \ldots, 1) = 0$
4	$f_4 = 20 + e - 20e^{(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2})} - e^{\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i)}$	$[-32.768, 32.768]$	$1, \ldots, D$	$f(0, 0, \ldots, 0) = 0$



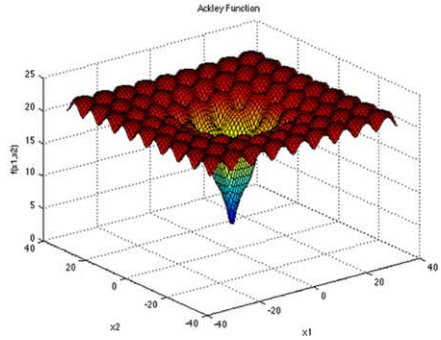
<*function* f_1>



<*function* f_2>



<*function* f_3>



<*function* f_4>

Fig. 5. 3D plot of the benchmark functions. (Colors are visible in the online version of the article; http://dx.doi.org/10.3233/AIC-140652.)

be trapped in one of its many local minima. The value of this function is 0 at its global minimum $(0, 0, \ldots, 0)$. The initial range for the function is set to.

4.2. *Performance measures*

The performance of the algorithms was compared using two criteria: (1) solution quality, and (2) the number of successful trial. The solution quality includes the minimum (best), the mean, the maximum (worst) cost value obtained in all simulation trials and the standard deviation of results. In addition, the number of successful trials is represented by the number of trial runs required for the eligibility function to reach its known global optimum value. In all experiments, the algorithms found a solution when they converge into ε tolerance and it is defined as:

$$\left| f_{\text{eligibility}}(T_i) - f_{\text{eligibility}}(T_{\text{go}}) \right| \leqslant \varepsilon, \tag{10}$$

where $f_{\text{eligibility}}(T_i)$ denotes the eligibility function value in ith iteration and $f_{\text{eligibility}}(T_{\text{go}})$ denotes the global optimum value of the f eligibility functions. ε is an acceptable tolerance of the known optimum value of the problem (i.e. the termination condition of the algorithm). To evaluate the performance of the algorithms, we define efficiency measure as below

$$Succ = \sum_{i=1}^{N_{\text{all}}} N_{\text{Succ}} | \varepsilon. \tag{11}$$

Efficiency is equal to the number of successful trials among all trials. N_{all} shows the number of all trials, N_{Succ} denotes the number of successful trials which the solution is found on ε. In the ideal state, all trials are successful so $Succ = N_{\text{all}}$. As the number of unsuccessful trials increases *Succ* deviates further from ideal state and decreases in value. In addition, the standard deviation (*Sigma*) of performance is proposed as a measure of fairness. *Sigma* is given by

$$Sigma = \sqrt{\frac{1}{N_{\text{all}}} \sum_{i=1}^{N_{\text{all}}} (O_i - O)^2}, \tag{12}$$

where O_i is the best solution found in ith trial, O is the average cost value over all trials and N_{all} is the total number of running trials. In an ideal state, the *Sigma* is 0 and how much the value of *Sigma* increases the reliability of the algorithm decreases.

4.3. Numerical examples and discussion

In this section, five algorithms were evaluated and compared: CGA, CLPSO, JDE, CMA-ES and the new proposed EA algorithm. All experiments took place on a 3 GHz, and 2 GB RAM, Personal Computer Intel® Pentium® 4. All the mentioned algorithms have been coded using the MATLAB language. The algorithms ran for 30 times. The results of solving the four test problems using the five algorithms are summarized in Tables 3–6. In these tables *min*, *mean* and *max* are respectively the minimum, mean and maximum cost values found over 30 simulation runs. *Sigma* is the standard deviation of the results. *Succ* is the number of times the global optimum is found. *Dim* indicates the dimension of the problem, and *Time* indicates the average simulation runtime of the algorithm.

As shown in Table 4, the performance of EA on f_1 function is better than others in terms of success rate and average consumed runtime. CMA-ES takes the second place and the third place belongs to JDE. The

Table 3
Statistical results obtained by CGA, CLPSO, CMA-ES, JDE and EA algorithms on benchmark function f_1 when $\varepsilon = 0.1$

Algorithm	*dim*	*min*	*mean*	*max*	*Sigma*	*Time*	*Succ*
GA	10	12.110	22.92	36.991	6.826	0.485	0
	20	155.762	214.217	281.514	29.5985	0.60	0
	30	413.194	544.858	633.555	51.228	0.69	0
CLPSO	10	20.900808	41.479	59.862	9.122	0.64	0
	20	31.486	44.461	60.263	6.465	0.75	0
	30	288.781	360.499	424.752	30.89	1.01	0
CMA-ES	10	5.3291e−14	2.025	4.058	1.089	1.28	1
	20	1.621	3.225	4.889	1.952	2.014	0
	30	3.038	7.738	19.881	2.927	3.159	0
JDE	10	3.265	5.643	6.922	0.789	22.68	0
	20	6.739	27.739	31.261	2.762	115.17	0
	30	21.125	28.165	47.739	4.381	124.14	0
EA	10	0.0005	1.077	6.209	1.531	0.37	10
	20	5.756	25.539	50.200	12.154	0.60	0
	30	20.009	27.581	45.581	7.102	0.95	0

Table 4
Statistical results obtained by CGA, CLPSO, CMA-ES, JDE and EA algorithms on benchmark function f_2 when $\varepsilon = 0.1$

Algorithm	*dim*	*min*	*mean*	*max*	*Sigma*	*Time*	*Succ*
GA	10	1.031	1.493	2.076	0.254	0.53	0
	20	10.116	27.698	44.426	6.979	0.65	0
	30	76.619	109.603	167.903	21.243	0.79	0
CLPSO	10	20.135	20.535	20.748	0.113	0.92	0
	20	20.875	20.980	21.014	0.08	1.10	0
	30	21.009	21.122	21.207	0.045	1.25	0
CMA-ES	10	20	21.292	21.6088	0.2016	0.32	0
	20	21.102	21.469	21.671	0.697	0.36	0
	30	21.304	21.470	21.627	0.0891	0.42	0
JDE	10	20.236	20.363	20.476	0.066	6.829	0
	20	20.336	30.341	20.402	0.012	17.15	0
	30	20.753	20.958	21.015	0.050	41.095	0
EA	10	2.053e−7	0.001	0.008	0.002	0.20	30
	20	0.034	0.102	0.208	0.040	0.37	18
	30	0.110	0.264	0.430	0.077	0.52	2

Table 5
Statistical results obtained by CGA, CLPSO, CMA-ES, JDE and EA algorithms on benchmark function f_3 when $\varepsilon = 0.1$

Algorithm	*dim*	*min*	*mean*	*max*	*Sigma*	*Time*	*Succ*
GA	10	48.27	348.983	1318.056	315.676	0.47	0
	20	19,572.055	70,793.853	153,114.842	32,914.984	0.62	0
	30	310,261.412	717,136.2815	1,228,734.341	240,546.508	0.63	0
CLPSO	10	6.023e+9	6.034e+9	6.026e+9	7.663e+6	0.66	0
	20	1.109e+10	1.154e+10	1.239e+10	2.988e+8	0.76	0
	30	2.014e+10	2.221e+10	2.340e+10	8.020e+8	0.82	0
CMA-ES	10	6.023e+9	6.026e+9	6.03e+9	1.726e+6	0.35	0
	20	1.152e+10	1.250e+10	1.630e+10	9.129e+8	0.36	0
	30	2.235e+10	2.520e+10	2.700e+10	1.089e+9	0.38	0
JDE	10	1.051e−12	0.030	0.264	0.058	2.66	18
	20	1.183	3.342	4.781	0.901	4.75	0
	30	4.223	19.338	69.481	21.909	6.39	0
EA	10	6.939	16.021	46.143	9.907	0.20	0
	20	85.236	255.051	430.206	85.362	0.46	0
	30	571.024	996.565	1120.755	105.413	0.62	0

Table 6
Statistical results obtained by CGA, CLPSO, CMA-ES, JDE and EA algorithms on benchmark function f_4 when $\varepsilon = 0.1$

Algorithm	*dim*	*min*	*mean*	*max*	*Sigma*	*Time*	*Succ*
GA	10	2.913	4.388	5.431	0.562	0.51	0
	20	11.268	12.758	14.276	0.722	0.62	0
	30	14.610	16.388	17.397	0.587	0.71	0
CLPSO	10	3.05	5.778	8.893	1.346	0.64	0
	20	35.461	62.488	80.698	11.015	0.64	0
	30	143.181	172.4125	206.414	15.135	0.62	0
CMA-ES	10	5.969	113.354	235.801	57.867	0.79	0
	20	38.803	298.449	446.727	89.293	0.90	0
	30	51.737	387.778	553.188	115.185	0.95	0
JDE	10	0	0	0	0	2.130	30
	20	0	0	0	0	5.67	30
	30	0	0	0	0	7.99	30
EA	10	0.007	0.158	0.954	0.190	0.20	27
	20	0.267	1.240	1.960	0.302	0.41	5
	30	1.205	2.318	2.846	0.310	0.65	0

performance of the CGA is better than that of CLPSO and worse than performance of others. On f_1 function for dimension 20 and 30 the performance order of algorithms is CMA-ES > EA > JDE > CLPSO > CGA.

For the f_2 function, the EA method outperformed all the other algorithms in terms of solution quality and the success rate. The CGA takes the second place on f_2 function for dimension 20. Approximately, the performance order of algorithms on for dimension 30 is EA > CLPSO > CMA-ES > JDE > CGA. As shown in Table 4, the success rate of EA for dimension 10, 20 and 30 is 30, 18 and 12, respectively, whereas other algorithms could not achieve to the global minimum of zero.

For the f_3 function, the success rate of the JDE algorithm is better than other algorithms in terms of the mean, the max and the standard deviation; but its average runtime is higher than those of others are. The worst results belong to CLPSO and CMA-ES in terms of the mean, the max and the standard deviation. For this function, the EA cannot achieve to the global minimum of zero.

For test function f_4, the JDE has the best performance in terms of the min, the mean, the max, and the standard deviation, but has the longest runtime among others. The EA takes the second place in terms of the quality of solution. Although the EA algorithm could not achieve to the global minimum value of this function for dimension 30, but it was able to achieve a solution close to the optimum value. Other algorithms could not achieve to the global minimum of zero.

From numerical simulations, it is clear that these algorithms have almost consistent behaviour for all benchmark functions. Generally, the EA algorithm almost performs well on all benchmark functions, exceeding or matching the best performance obtained by CGA, CLPSO and CMA-ES. Also, except f_3 and f_4 test functions, the EA outperformed the JDE algorithm in terms of solution quality and average runtime.

With the exception of the f_3 and f_4 functions, it can be seen that the solution quality obtained using the EA in 30 independent simulation runs are almost better than other algorithms and this testify that the EA has better stability behaviour rather than other algorithms. In summary, the experimental results prove the EA algorithm has the ability of getting out of local minimums in the search space and finding the global minimum.

5. Conclusions

In this paper, inspired by the presidential election process, an optimization and stochastic search algorithm has been presented which is called Election Algorithm (EA). EA simulates the candidates' behavior in presidential election and it is applicable for both continuous and discrete problems. This population-based technique contains persons, either candidates or voters that collectively form some electoral parties in the solution space. Advertising campaign – including positive advertisement, negative advertisement and coalition steps – is the main part of the algorithm and causes the persons to converge into the global optimum of solution space.

In order to evaluate the performance of the EA algorithm, CGA, CLPSO, CMA-ES and JDE algorithms were tested on four numerical benchmark functions and the comparative results were represented. From the simulation runs, it is concluded that the EA technique is almost found to perform better than others in terms of solution quality and success rate. One drawback of the EA is using of the Euclidean distance in the creation of initial parties and the negative advertisement steps that decrease the speed of the algorithm.

There remain some points to improve our research. First, we have to explore an appropriate and efficient similarity measure to substitute the Euclidean distance metric with it. Second work is to investigate the effect of control parameters on the performance and convergence speed of the EA algorithm. Another issue is that we hope to improve the execution speed of the algorithm and apply the proposed algorithm to solve some practical applications.

References

[1] E. Atashpaz-Gargari and C. Lucas, Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition, in: *2007 IEEE Congress on Evolutionary Computation*, 2007, pp. 4661–4667.

[2] Z. Beheshti, S. Mariyam and H. Shamsuddin, A review of population-based meta-heuristic algorithms, *Int. J. Adv. Soft Comput. Appl.* **5**(1) (2013), 1–35.

[3] J. Brest, S. Greiner, B. Boskovic, M. Mernik and V. Zumer, Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, *IEEE Transactions on Evolutionary Computation* **10**(6) (2006), 646–657.

[4] S.P. Brooks and B.J.T. Morgan, Optimization using simulated annealing, *Stat.* **44**(2) (1995), 241–257.

[5] M. Dorigo, V. Maniezzo and A. Colorni, Ant system: Optimization by a colony of cooperating agents, *IEEE Trans. Syst. Man. Cybern. B. Cybern.* **26**(1) (1996), 29–41.

[6] E. Elbeltagi, T. Hegazy and D. Grierson, Comparison among five evolutionary-based optimization algorithms, *Adv. Eng. Informatics* **19**(1) (2005), 43–53.

[7] Encyclopedia Britannica, Election (political science), available at: http://www.britannica.com [Accessed 22 May 2014].

[8] S.E. Finkel, Reexamining the 'Minimal effects' model in recent presidential campaigns, *J. Polit.* **55**(1) (1993), 1–21.

[9] P. Freedman and M. Franz, Campaign advertising and democratic citizenship, *Am. J. Pol. Sci.* **48**(4) (2004), 723–741.

[10] F. Glover, Tabu search: A tutorial, *Interfaces* **20**(4) (1990), 74–94.

[11] F. Glover and G.A. Kochenberger, *Handbook of Metaheuristics*, Springer, 2003.

[12] F. Glover, M. Laguna and R. Martí, Fundamentals of scatter search and path relinking, *Control Cybern.* **39**(3) (2000), 653–684.

[13] B.R. Gordon and W.R. Hartmann, Advertising competition in presidential elections, *Mark. Sci.* **32**(1) (2013), 19–35.

[14] P. Hansen and N. Mladenović, Variable neighborhood search, in: *Handbook of Metaheuristics*, Springer, 2003, pp. 145–184 (Chapter 6).

[15] R.L. Haupt and S.E. Haupt, *Practical Genetic Algorithms*, 2nd edn, Wiley, 2004.

[16] C. Igel, N. Hansen and S. Roth, Covariance matrix adaptation for multi-objective optimization, *Evol. Comput.* **15**(1) (2007), 1–28.

[17] D. Karaboga and B. Akay, A comparative study of artificial bee colony algorithm, *Appl. Math. Comput.* **214**(1) (2009), 108–132.

[18] D. Karaboga and B. Basturk, A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm, *J. Glob. Optim.* **39** (2007), 459–471.

[19] A. Kaveh and S. Talatahari, A novel heuristic optimization method: Charged system search, *Acta Mech.* **213**(3,4) (2010), 267–289.

[20] J. Kennedy and R. Eberhart, Particle swarm optimization, in: *Proc. ICNN'95 – Int. Conf. Neural Networks*, Vol. 4, 1995, pp. 1942–1948.

[21] J.J. Liang, A.K. Qin, S. Member, P.N. Suganthan, S. Member and S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *Evol. Comput. IEEE Trans.* **10**(3) (2006), 281–295.

[22] E. Mortimer, *Faith and Power: The Politics of Islam*, Random House, London, 1982.

[23] J. Puchinger and R. Raidl, Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification, in: *Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach*, Lecture Notes in Computer Science, Vol. 3562, Springer, Heidelberg, 2005, pp. 41–53.

[24] P. Rabanal, I. Rodríguez and F. Rubio, Using river formation dynamics to design heuristic algorithms, in: *Unconventional Computation*, 2007, pp. 163–177.

[25] E. Rashedi, H. Nezamabadi-pour and S. Saryazdi, GSA: A gravitational search algorithm, *Inf. Sci.* **179**(13) (2009), 2232–2248.

[26] H. Shah-Hosseini, Problem solving by intelligent water drops, in: *2007 IEEE Congress on Evolutionary Computation*, 2007, pp. 3226–3231.

[27] G. Venter, Review of optimization techniques, in: *Encyclopedia of Aerospace Engineering*, R. Blockley and W. Shyy, eds, Wiley, 2010, pp. 1–12.

[28] X.-S. Yang, Firefly algorithms for multimodal optimization, in: *Stochastic Algorithms: Foundations and Applications*, 2009, pp. 169–178.

[29] X.-S. Yang, A new metaheuristic bat-inspired algorithm, in: *Nat. Inspired Coop. Strateg. Optim. (NICSO 2010)*, Springer, Berlin, 2010, pp. 65–74.