Opposition Based Computing – A Survey

Fares S. Al-Qunaieer, Hamid R. Tizhoosh, Shahryar Rahnamayan

***Abstract*—In algorithms design, one of the important aspects is to consider efficiency. Many algorithm design paradigms are existed and used in order to enhance algorithms' efficiency. Opposition-based Learning (OBL) paradigm was recently introduced as a new way of thinking during the design of algorithms. The concepts of opposition have already been used and applied in several applications. These applications are from different fields, such as optimization algorithms, learning algorithms and fuzzy logic. The reported results confirm that OBL paradigm was promising to accelerate or to enhance accuracy of soft computing algorithms. In this paper, a survey of existing applications of opposition-based computing is presented.**

I. Introduction

DESIGNING efficient algorithms is considered to be one of the most important concerns in computer science. There are many existing design paradigms which are commonly used for designing algorithms. In order to mention some examples, *Divide and conquer* algorithms are iteratively or recursively reducing the given problem into smaller instances until they could be easily solved. In *reduction* algorithms, the aim is to transform the input problem into a known one which there are some efficient algorithms to solve it. *Brute-force* algorithms exploit every possible solution to decide which one is the best. Recently, a new design paradigm was proposed [1], which is based on considering candidate and corresponding opposite-candidate.

"Opposition is concerned with the relationship between entities, objects or their abstractions of the same nature which are completely different in some manner" [2]. For example, *hot* and *cold* are description of temperature (same kind) but completely different. The opposition-based thinking is basic element of human thinking, it has been used in many fields. In natural language, opposition is occurred frequently, such as, north-south in direction and long-short in adjectives. In psychology, rewards and punishment are two opposite reinforcements that are used in the learning process. Antiparticles in physics are subatomic particles with the same mass but opposite electric charges and magnetic moment. Similarly in mathematics, if the probability of an event is *p*, then the probability of its contrary is *1-p*. Also, the pair $(x, -x)$ are representing opposite numbers in $\mathbb{R}$.

It had been proven mathematically and experimentally [3] that utilizing opposition in learning yields more efficient algorithms than using only pure randomness. For this reason, algorithms from different fields were enhanced by using OBL and achieved promising results. In this paper, a review of previous applications and algorithms that were enhanced by OBL is conducted.

The remaining of this paper is organized as follows: An overview of opposition-based learning is provided in Section II. In Section III, a review of the applications of opposition-based learning in soft computing is conducted. Then, open areas for further research are suggested in Section IV. Finally, Section V concludes this paper.

II. Opposition-Based Learning

The concept of Opposition-Based Learning (OBL) was recently introduced by Tizhoosh [1]. The basic idea behind OBL is that whenever we seek the solution in a direction, that is beneficial to consider the opposite direction as well. Many machine intelligence algorithms consider finding the solution of a given problem as function approximation. Thus, if the objective is to search for the solution x, the algorithm makes an estimation $\hat{x}$ which should resemble the closest value to x. Such algorithms can be computationally expensive if the required solution must be very accurate. Starting point of search can dramatically affect the accuracy of the found solution (among others due to local maxima or minima) and the convergence time. In many cases, the starting points are chosen randomly, such as weights of a neural network, initial population of evolutionary algorithms, and action policy of reinforcement agents. If the starting point is close to the optimal solution, this results a faster convergence. On the other hand, if it is very far from the optimal solution, such as opposite location in worst case, the convergence will take much more time or even the solution can be intractable. Looking simultaneously for a better candidate solution in both current and opposite directions may help to solve the problem efficiently. Following are some important definitions.

Definition (Type-I Opposite Points) [2] – Let x be a real number defined on the interval $[a, b]$. The *opposite number* $\breve{x}$ is defined as follows

$$\breve{x} = a + b - x. \tag{1}$$

For $a = 0$ and $b = 1$, we have

$$\breve{x} = 1 - x. \tag{2}$$

In the same manner, the opposite number in a multidimensional search space can be defined [2].

Let $P(x_1, x_2, ..., x_n)$ be a point in an n-dimensional coordinate system with $x_1, ..., x_n \in \Re$ and $x_i \in [a_i, b_i]$. The opposite point $\breve{P}$ is defined by its coordinates $\breve{x}_1, ..., \breve{x}_n$ where

$$\breve{x}_i = a_i + b_i - x_i \qquad i = 1, ..., n \tag{3}$$

Fares S. Al-Qunaieer, Systems Design Engineering, University of Waterloo, 200 University Ave. West, Waterloo, ON N2L 3G1, Canada, e-mail: falqunai@engmail.uwaterloo.ca

Hamid R. Tizhoosh, Systems Design Engineering, University of Waterloo, 200 University Ave. West, Waterloo, ON N2L 3G1, Canada, e-mail: tizhoosh@uwaterloo.ca

Shahryar Rahnamayan, Electrical and Computer Engineering, University of Ontario Institute of Technology (UOIT), 2000 Simcoe Street North, Oshawa, ON L1H 7K4, Canada, email: shahryar.rahnamayan@uoit.ca

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

Definition (Type-I Super-Opposite Points) [2] – Let $P = (a_1, a_2, \ldots, a_n)$ be a point in an n-dimensional search space with $a_i \in [X^i_{min}, X^i_{max}]$ and $\breve{P} = (\breve{a}_1, \breve{a}_2, \ldots, \breve{a}_n)$ is its opposite point. Then all points $\breve{P}^s$ are type-I super-opposite of P when $d(\breve{P}^s, P) > d(\breve{P}, P)$, where $d(.,.)$ denotes a metric such as Euclidean distance.

Definition (Type-I Quasi-Opposite Points) [2] – Let $P = (a_1, a_2, \ldots, a_n)$ be a point in an n-dimensional search space with $a_i \in [X^i_{min}, X^i_{max}]$ and its opposite point $\breve{P} = (\breve{a}_1, \breve{a}_2, \ldots, \breve{a}_n)$. Then all points $\breve{P}^q$ are type-I quasi-opposite of P when $d(\breve{P}^q, P) < d(\breve{P}, P)$, where $d(.,.)$ denotes a metric such as Euclidean distance.

Definition (Type-II Opposite Points) [2] – Assuming the function $f(x_1, x_2, \ldots, x_n)$ is not known, but y_{min} and y_{max} are given or can be estimated. Let $y = f(x_1, x_2, \ldots, x_n) \in \mathbb{R}$ be an arbitrary function with $y \in [y_{min}, y_{max}]$. For every point $P = (a_1, a_2, \ldots, a_n)$ the type-II opposite point $\breve{P} = (\breve{a}_1, \breve{a}_2, \ldots, \breve{a}_n)$ is defined by

$$\breve{a}_i = \{x | \breve{y} = y_{min} + y_{max} - y\}. \quad (4)$$

Opposition-Based Learning (OBL) [1] – Let $f(x)$ be the function in focus and $g(.)$ a proper evaluation function. If $x \in [a, b]$ is a candidate guess and $\breve{x}$ is its opposite value, then the learning continues with x if $g(f(x)) > g(f(\breve{x}))$, otherwise with $\breve{x}$.

III. Opposition-Based Computing

Opposition-based learning concepts were applied to enhance various problems in different soft computing fields. In this section, a review of the OBL-based algorithms is presented. These algorithms are categorized into five main areas, namely, reinforcement learning algorithms, neural networks, optimization algorithms, fuzzy logic, and miscellaneous applications. Each area is reviewed in a separate subsection.

A. Reinforcement Learning

Reinforcement Learning (RL) is an approach in artificial intelligence that was inspired from the concepts of actions and rewards in psychology. Several researches were attempted to incorporate OBL concepts to enhance RL algorithms.

An enhancement to reinforcement learning based on opposition based computation was proposed by Tizhoosh [4] in 2005. The main idea of the algorithm is to consider actions and opposite actions and/or opposite states. This makes the traversal of the states of the environment shorter, which mean faster convergence time. The proposed algorithm has been extended to Q-learning algorithm and was compared with the parent algorithm using two sizes of the grid world problem. The results of OQL was better than the results for QL.

Tizhoosh [5] tested OQL by introducing three versions of that. The first one, the opposition based algorithm considers the opposite action $\breve{a}$ and opposite reward $\breve{r}$ for each taken action a with reward r. The second variant used a second learning step $\breve{\alpha}$ which is defined as a decreasing function of the number of episodes. The third algorithm considered the opposite actions only for a limited number of episodes. Experiments were conducted to compare these three variants of OQL along with QL algorithm. Three sizes of the grid world problem were used for benchmarking. Algorithm 2 achieved the best results among the three proposed variants and was much better than QL in term of convergence speed.

Shokri et al. [6] proposed an opposition-based $Q(\lambda)$ algorithm ($OQ(\lambda)$). In this algorithm, eligibility traces for the actions and opposite actions, which called opposition traces, are used. The updating mechanism for opposite action is the same as mechanism used in [5], but in this paper opposition traces were introduced. $OQ(\lambda)$ was compared with the conventional $Q(\lambda)$ using three sizes of the grid world problem, and performed better in terms of running time and success rate. However, this approach could not work with non-deterministic environments. This was solved in [7] by using non-markovian update of the opposition traces, the new algorithm is called opposition-based $Q(\lambda)$ with non-markovian update, $NOQ(\lambda)$. This is performed by introducing a weight $W \in [0, 1]$ to the opposition update. For the applications which have no clear definition of opposite action, the weight is set low and it is gradually increased. On the other hand, in problems which have clear definition of opposite action the weight is set to 1. $NOQ(\lambda)$ was tested against both $Q(\lambda)$ and $OQ(\lambda)$ again using three sizes of grid world problem, and $NOQ(\lambda)$ achieved the best results in overall. In order to have trade-off between exploration and exploitation in the processes of non-markovian update of the opposite actions; Shokri et al. [8] suggested an increasing function of the weight. Thus, the value of W is increasing as the iterations increase. The rationale behind this is that, the values of state-action in later stages is more meaningful. The modified $NOQ(\lambda)$ was tested using a simple elevator control problem, which is dynamic and non-deterministic, and was compared with $Q(\lambda)$. Lower average running time and overall average iteration were achieved using the modified $NOQ(\lambda)$.

Mahootchi et al. [9] investigated the use of opposition-based reinforcement learning for the management of single reservoir operations. Two algorithms were proposed, first one using type-I opposition and the second one using type-II. A multilayer perception was used for the purpose of function approximation to find type-II opposite actions. Only opposite actions were considered, but opposite states were not.

In his PhD [10], Mahootchi investigated the enhancement of several RL algorithms and their applications in storage management. His extended algorithms are $Q - learning$, $Q(\lambda)$, $sarsa$, and $sarsa(\lambda)$.

The use of opposition-based reinforcement learning for image segmentation was proposed by Sahba et al. [11]. In order to segment prostate ultrasound images, thresholding is performed, then morphological opening was applied to remove the remaining artifacts and noise. The segmentation was performed locally, so the input image was divided into sub images. The task of RL agents was to determine both threshold and structuring element for each sub-image. The use of opposite actions in updating the Q-map could make the learning faster. The proposed algorithm was tested on 20 medical images and compared with the standard Q-learning algorithm. In terms of convergence speed, the proposed algorithm presented much better results, but it got accuracy lower than the standard Q-learning algorithm.

B. Neural Networks

Artificial Neural Networks (ANN) is a sub-field of machine learning which was introduced in order to mimic the way humans learn. ANN received too much attention and have many models and configurations.

Ventresca and Tizhoosh [12] investigated the use of opposition based computation to improve the performance of back-propagation algorithm. In their approach, they considered the opposite of the transfer function for a subset of neurons. The opposite transfer function of $f(x)$ defined as $\breve{f}(x) = f(-x)$. The opposite network was defined as a network which has same weights as the original network and at least one neuron having the opposite transfer function. Four common benchmark problems were used to test the proposed algorithm (OBP). The results were compared with the results of back-propagation algorithm (BP). OBP outperformed BP in terms of speed on the four problems, and achieved better accuracy in three of them.

An improvement to Back-Propagation Through Time ($BPTT$) algorithm, which is a discrete-time recurrent neural network training algorithm, was proposed based on opposition based computation. Ventresca and Tizhoosh [13] used the opposite neural network defined in [12] for Elman recurrent topologies [14] to construct Opposition-based Back-propagation Through Time ($OBPTT$) algorithm. The algorithm was dynamically determining the opposite transfer function at runtime. $OBPTT$ algorithm was tested using three benchmark problems, and results were compared with $BPTT$. The results obtained by $OBPTT$ were at least as good as those obtained by $BPTT$. Furthermore, $OBPTT$ was better than $BPTT$ in terms of learning reliability and stability and also convergence rate.

Ventresca and Tizhoosh [15] utilized the opposition-based computation concept to improve the performance of large scale neural networks that have hundreds or thousands of parameters. During the learning process, in each iteration a set of neurons were selected based on a probabilistic rule. The selected neurons were assigned the opposite transfer function. The formed network represent the opposite network. In each iteration, both current and opposite networks were evaluated and the network that have best results was labeled as the current network. Using five benchmark problems, this technique was tested against two variants of Back-propagation (BP) algorithm, namely, BP with adaptive learning rate and BP with adaptive learning rate and momentum. Opposite version achieved the best results and was the fastest on all benchmark problems. In addition, it had the most reliable results.

C. Optimization

The problem of minimization or maximization of a function is called optimization problem, it is an important field in all science and engineering fields.. Because of its importance, many different algorithms have been proposed to solve optimization problems efficiently. In this subsection, the extensions of three categories of optimization algorithms are reviewed. The categories are differential evolution algorithms, particle swarm optimization, and other optimization techniques.

1) Differential Evolution: Differential Evolution (DE) is a population-based optimization technique which uses evolutionary concepts. Some researches were attempted to utilize opposition concepts to improve performance of DE.

As pioneers, Rahnamayan et al. [16], [17], [18] proposed the inclusion of opposition-based computation in evolutionary algorithms. They proposed a mean of population initialization based on opposition concept. This is performed by initializing a random population $P(n)$ and calculating the corresponding opposite population $OP(n)$. Then, the fittest individuals are selected from union set of $P(n)$ and $OP(n)$. In addition to that, based on a jumping rate, dynamic opposite population of the current population is calculated and the fittest individuals are selected in the same manner. An extensive experimentations were preformed in [18] using 58 benchmark functions in order to test the performance of ODE. Sets of experiments were conducted to study the effect of dimensionality, opposite points, population size, different strategies of mutation and jumping rates, and the speed and robustness. In these experiments, ODE was compared with DE and achieved superior results. In addition, ODE was compared with $FADE$, and achieved much better overall results. In [19], Rahnamayan and Wang investigated the application of ODE for large scale optimization problems. Comparison was conducted between ODE and DE using seven large scale benchmark functions (500 and 1000 dimensions). Results of ODE was much better that that of DE in both accuracy and convergence speed.

Additional modification of this algorithm was introduced by locally improving the fittest member in the current population [20], [21]. This is performed by calculating difference-offspring (*newbest*) of the fittest member (*best*) of the current population, and then calculate the opposite of newbest (*op_newbest*). Current fittest members are replaced by the fittest members of the set {*best*, *newbest*, *op_newbest*}. This modified algorithm was tested on nine well-known minimization functions with added variable amount of noise, and compared with DE algorithm. opposition-based version outperformed DE on eight functions in terms of convergence rate with the same success rate.

Rahnamayan et al. [22] enhanced the previous methods by replacing opposite points with quasi-opposite points. The authors proved that a quasi-opposite point has a higher probability of being closer to the solution than an opposite point. The same procedure of population initialization and generation jumping that proposed in [18] was used, except that quasi-opposite points were used and a smaller jumping rate is employed. This algorithm ($QODE$) was compared with both ODE and classical DE algorithms using 15 benchmark functions with two different dimensions (total of 30). $QODE$ outperformed both ODE and DE in terms of number of function calls and success performance on 22 functions. DE achieved marginally better average success rate than $QODE$ and ODE.

The jumping rate for the previous algorithms is a predefined constant value (i.e., $J_r = 0.3$ for ODE and $J_r = 0.05$ for $QODE$). Variable jumping rate was introduced by Rahnamayan et al. [23]. The authors introduced two types of time varying jumping rates, namely, linearly increasing and decreasing functions. The former has lower jumping rate during exploration, and higher jumping rate during exploitation, and vice-versa for the latter. Based on 15 benchmark functions, it had been found that, the linearly decreasing jumping rate achieves better performance than both constant and linearly increasing jumping rates..

Opposition-Based DE (ODE) was applied for the task of finding the best threshold value for images [24]. As an optimization problem, the following objective function was defined

$$f(T) = \sum_{i=1}^{M} \sum_{J=1}^{N} |I_{ij} - B(T)_{ij}|, \tag{5}$$

where M and N are the dimensions of the image I, and $B(T)$ is the corresponding thresholded image by the threshold value T. By this way, an image thresholding task is modeled to a minimization problem. In order to solve problem faster, ODE with very small population size ($N_p = 5$), which called micro-ODE, was used. The algorithm is tested against the well-known kittler algorithm and its non-opposition-based version (micro-DE) by utilizing 16 test images. micro-ODE achieved better average accuracy than both Kittler and micro-ODE and also it was faster than micro-ODE by 13%.

Similar to ODE works, Peng et al. [25] utilized opposition-based learning in the initialization of the population for a Multi-Objective Differential Evolution. In order to extend DE to solve multi-objective problems, the concepts of dominance, non-dominated sorting and crowding distance metric were used. The performance of the algorithm ($OMODE$) was tested using five two-objective benchmark functions and compared with six different algorithms, namely, NSGA-II [26] (real and binary coded), PAES [26], SPEA2 [27], and IBEA [27], [28]. Also, $OMODE$ was tested on Earth-March transfer problem and compared with NSGA-II. In both test sets $OMODE$ achieved the best overall results

To solve the problem of tuning chess program, Boškovi$\acute{c}$ et al. [29] used DE with the concepts of adaptation and opposition. Population initialization was performed by using opposition points as described in [18]. Opposite concept was also utilized in current population using a jumping rate. Adaptation was performed by using the $rand/2$ mutation strategy and adaptive mutation scale factor F. A simplified chess evaluation function of the chess program BBchess was used to test the algorithm. The algorithm was tested using opposition initialization only and also by using opposition through entire evolutionary process. By using opposition initialization only, convergence of the algorithm was better at the beginning, while the use of opposition through the whole algorithm obtained poor convergence at the beginning, but same results at the end.

Omran [30] investigated the using of OBL to improve Particle Swarm Optimization (PSO) and Barebones Deferential Evolution ($BBDE$) [31]. For PSO, the particle with the lowest fitness is replaced by its opposite particle in each iteration, this is performed by using opposite point. Similarly, in $BBDE$ the individual with lowest fitness is replaced by the opposite individual. The improved $iPSO$ and $iBBDE$ were compared with PSO and $BBDE$, respectively, using seven benchmark functions. In general, the improved algorithms outperformed PSO and $BBDE$.

Free Search Differential Evolution ($FSDE$) was introduced by Omran and Engelbrecht [32]. $FSDE$ was the result of hybridization of of Free Search (FS) [33], Differential Evolution (DE), and opposition-based learning. The concept of sense was taken from FS, and mutation operator from DE. In each iteration, the solution x that has lowest fitness is replaced by its opposite point $\breve{x}$, defined as

$$\breve{x} = LB + UB - r.x \tag{6}$$

where LB, UB, and r are lower bound, upper bound, and a uniformly generated random number in [0,1], respectively. This algorithm was tested on ten benchmark functions and compared with both DE and one of its variants, $BBDE$. $FSDE$ outperformed DE on seven functions and outperformed $BBDE$ on eight functions.

Omran and Salman [34] proposed a new population-based meta-heuristic optimization algorithm called $CODEQ$. $CODEQ$ is a combination of concepts from chaotic search, opposition-based learning, Deferential Evolution (DE), and quantum mechanics. Population in this algorithm is initialized randomly, and then in each iteration t a trial vector, $v_i(t)$ is created by mutating the parent vector $x_i(t)$ as,

$$v_i(t) = x_i(t) + (x_{i1}(t) - x_{i2}(t))ln(\frac{1}{u}), \tag{7}$$

where $u \sim U(0,1)$, and $x_{i1}(t)$, $x_{i2}(t)$ are randomly selected indices where $i_1 \neq i_2 \neq i$. $v_i(t)$ will replace $x_i(t)$ if it has a better fitness value. After that, a new vector is generated for each iteration as,

$$w(t) = \begin{cases} LB + UB - r.x_b(t) & \text{if } n \leq 0.5 \\ x_g(t) + |x_{i1}(t) - x_{i2}(t)|.(2c(t) - 1) & \text{otherwise} \end{cases} \tag{8}$$

where $r \sim U(0,1)$, LB and UB are lower and upper bounds, respectively, $x_b(t)$ is the least fit vector, $x_g(t)$ is the fittest vector. If the fitness of the generated vector $w_i(t)$ is better than the worst vector $x_b(t)$, then it will replace it. The algorithm is repeated until it satisfy a stopping condition. The performance of $CODEQ$ was tested using five constrained problems. The results were compared to other techniques.

An evolutionary algorithm, biogeography-based optimization (BBO) [35] was extended to oppositional BBO ($OBBO$) by utilizing opposition-based learning (OBL) [36]. The authors introduced a new opposition-based method called quasi-reflection. Mathematical proof was provided to show that quasi-reflection point has a higher probability of being closer to the solution than an opposite point. The opposite population was computed during generation jumping as described in [18]. Using 16 benchmark functions, $OBBO$ algorithm was compared with BBO. The former had a success rate of 94%, while the latter had 70%. Also, $OBBO$ had 98% less average function evaluations compared to BBO.

2) *Particle Swarm Optimization:* Particle Swarm Optimization (PSO) is a stochastic population-based optimization algorithm inspired from the social behavior of bird flocking or fish schooling. Several approaches were considered in order to utilize OBL for improving PSO algorithms.

Han and He [37] introduced $OPSO$ by utilizing OBL to enhance swarm initialization, generation jumping and improving the swarm's best member. First, swarms are initialized with random positions and velocities. The opposite swarm is calculated by computing the opposite of position and velocity. The fittest of swarm and opposite swarm is selected. The same procedure is applied to current generations using jumping rate and dynamic constriction factor (CF) in the calculation of opposite points as follows

$$ox_{ij} = L_j + U_j - CF_{ij}.x_{ij}, \tag{9}$$

where

$$CF_{ij} = 1 - \lambda r_{ij}, \tag{10}$$

where r_{ij} is a Cauchy random number, and λ is starting at 1.0 and decreased every 50 generations. Best members jumping was applied as described in [18]. The modified algorithm ($OPSO$) was tested against PSO algorithm using six benchmarking functions. $OPSO$ achieved a better performance in terms of convergence speed and global search ability (i.e., escaping from local optima).

OBL with a Cauchy mutation [38] were used to enhance PSO [39]. Opposition concept was introduced during the initialization and improving current populations. In every generation, Cauchy mutation was applied to the global best particle. It helps to decrease the probability of being trapped in a local optimum. Four unimodal and four multimodal functions were used to compare the performance of $OPSO$ and PSO. Faster convergence was achieved by $OPSO$ for unimodal functions and better global search for the multimodal functions. Just in one function, $OPSO$ was trapped in local optima.

Wu et al. [40] extended the previously proposed Comprehensive Learning Particle Swarm Optimization ($CLPSO$) [41] to Opposition-Based $CLPSO$ ($OCLPSO$). Again opposition concept was used in population initialization as in [17]. Opposition-based exemplar selection also was introduced. First, two particles are chosen from the population. The fitness of the selected particles and their opposites are compared. The best fitter particle is used as the exemplar to learn from that dimension. Based on 10 benchmarking functions, $OCLPSO$ performance was compared with $CLPSO$. The reported results achieved by $OCLPSO$ was much better.

Omran and al-Sharhan proposed three variants of opposition-based PSO [42]. The first variant ($OPSO$) used opposition concepts just for population initialization as described by [17]. Second variant, which called improved $OPSO$ ($iOPSO$), utilized opposition for every iteration by replacing the particle with lowest fitness value by its opposite. The third variant, which named improved PSO ($iPSO$), is the same as $iOPSO$, but without opposition-based population initialization. The performance of the three proposed variants were compared with PSO using eight benchmark functions. In general, $iOPSO$ and $iPSO$ outperformed PSO and $OPSO$ in both accuracy and convergence speed. The results of $iOPSO$ and $iPSO$ were very close.

Jabeen et al. [43] tested the effect of using opposition-based population initialization in more details. Four benchmark functions were used to test $OPSO$. The algorithm is compared to three variants of PSO, namely, $PSO1$, $PSO2$, and PPO [44]. Opposition-based algorithm outperformed $PSO1$ and $PSO2$ on all four functions and PPO on three of them. The $O - PSO$ algorithm is also compared with a PSO that initialize its population randomly ($2 - PSO$). The comparison was conducted using three functions. $O - PSO$ achieved its results in less number of iterations that $2 - PSO$.

A variant of PSO that uses velocity clamping ($VCPSO$) was extended by using OBL ($OVCPSO$) [45]. Opposition was employed during the initialization and iteration phases. Performance of $OVCPS$O was compared with PSO, $VCPSO$ and opposition-based PSO with Cauchy Mutation ($OPSOCM$) by using eight benchmark functions. In overall, $OVCPSO$ achieved a better performance than PSO, $VCPO$ and $OPSOCM$.

3) *Other Optimization Methods:* Simulated annealing (SA) is a well-known global search algorithm. An enhancement to the vanilla version of SA was proposed using an opposition based technique [46] (OSA). The proposed algorithm calculates a neighborhood and an opposite neighborhood of the current solution and evaluates the quality of both, then the best solution is chosen as the current one. OSA was compared with vanilla version of SA and achieved better accuracy and convergence rate. Furthermore, OSA results were more reliable than SA results.

Malisia and Tizhoosh [47], [48] investigated the use of OBL ideas for Ant Colony System (ACS). The authors proposed five variants for employing opposition concept to extend the construction phase of ACS, namely, Synchronous Opposition, Free Opposition, Free Quasi-Opposition, Opposite Pheromone per Node (OPN), and Opposite Pheromone per Edge (OPE). In addition, an extension to the update phase was investigated in [48].

D. *Fuzzy Set Theory*

Fuzzy set is a generalization of the classical crisp set. Rather than considering an element to either belong or does not belong to a set , a membership of the element is calculated to determine its membership degree. The notion of opposition have always been a part of fuzzy sets since it has been introduced.

Tizhoosh [49] defined the preliminary concepts of opposite fuzzy sets. Following definitions were provided for a basic framework: opposite fuzzy set, type I opposite fuzzy sets, type I super-opposite fuzzy sets, type I quasi-opposite fuzzy sets, type II opposite fuzzy sets, and opposition-based fuzzy inference systems ($OFIS$). Based on the defined concepts, a new image segmentation approach was proposed. The algorithm defines a set A as dark pixels, and then calculate the entropy of A. Then, iteratively defines $\breve{A}$, the opposite fuzzy set of A with different sizes starting from the brightest region, calculates entropy of $\breve{A}$ and the difference between the entropies. The minimum difference indicates that $\breve{A}$ is the most probable opposite of A. The algorithm was tested on four breast ultrasound images and the results were compared to the results of Otsu algorithm.

An extension to the previous method was proposed by Tizhoosh [50]. First, the center of the object of interest is determined interactively by user input. A window is constructed around the central point, and its size is increased incrementally in each iteration. For each iteration, dark pixels fuzzy set is determined for current window, and its opposite bright fuzzy set is found. The location of the window that has a maximum entropy difference between the two fuzzy sets is found. Then, the threshold value is calculated as the average of representative numbers of both fuzzy sets of the selected window. The performance of this algorithm was tested using eleven prostate ultrasound images.

E. Other Applications

Fuzzy c-mean clustering for data with tolerance ($FCM{-}T$) was proposed by Kanzawa et al. [51]. Two variants of the algorithm were defined, namely, standard type $FCM - T$ ($sFCM - T$) and entropy regularized type ($eFCM - T$). The essence of both algorithms is solving an optimization problem which minimize the objective function with respect to membership (μ) , centers (v), and tolerance (ε). In this way, tolerance is determined such that the data heads to the center of the cluster. In addition, the authors proposed another two algorithms that maximize the objective function with respect to tolerance. Thus, the data is moving away from cluster center. If the first two algorithms and the latters obtains same clustering result, it could be considered as reliable results, otherwise it is unreliable.

Khalvati et al. [52] considered the use of opposition-based concept to enhance window memoization. The case study was gray-scale morphological algorithm which uses 3×3 non-flat structuring elements. A lookup technique which uses multi-thresholding values was developed to increase reusing rate. Each time a lookup is performed on a window, the response of the opposite window is calculated. This method presents a reduction in the number of calculations.

IV. Further Research Directions

OBL can be applied to wide range of areas. As it can be observed from previous section, extended methods usually obtain very promising results. Still, it is early to say that the possibilities of employing OBL concepts is fully exploited. The door is widely open for many researches to utilize the opposition concept in other soft computing areas.

Optimization algorithms received the most attention from researchers. The modified algorithms achieved better results than the parent algorithms. More studies on enhancing of Particle Swarm Optimization and Ant Colony System is required. In addition, many studies of utilizing OBL to enhance other existing optimization algorithms, such as gradient descent and hill climbing, can be conducted.

Learning algorithms, either supervised or unsupervised, have good potential to enhance by OBL. Already Neural Networks and Reinforcement Learning was extended and tested. But still, new methods can be investigated; such as, the definition of opposite networks. Other types of networks could be studied, such as Cascade-Correlation and Neuro-Fuzzy networks. Embedding OBL in other learning algorithms, such as Support Vector Machines, Hidden Markov Model and clustering algorithms can be investigated.

Utilizing OBL in image processing applications is an area with many possibilities. The opposite of a pixel can be defined with respect of color, intensity, location, or direction. Some of the suggested applications are image thresholding and segmentation, edge detection and registration.

More extensive studies on opposite fuzzy sets and its applications are required.

V. Conclusions

There are several existing paradigms of thinking through the process of developing algorithms, such as brute-force search, dynamic programming, and greedy method. Thinking of opposite possibilities during the design of algorithms is a recent suggested paradigm. A review of the algorithms which used opposition approach was presented. The reviewed algorithms achieved very promising result, which indicates that OBL can be beneficial if applied in an efficient way. OBL concepts can be applied in many research and applications areas, such as optimization algorithms, learning algorithms, fuzzy sets and image processing. In each area, there are many OBL-based potential ideas to enhance algorithms. It is still early to say that all applications of OBL is reasonably exploited.

Acknowledgment

The authors would like to thank King Abdulaziz City for Science and Technology for their scholarship and for their valuable support.

References

[1] H. R. Tizhoosh, “Opposition-based learning: A new scheme for machine intelligence,” in *Proceedings - International Conference on Computational Intelligence for Modelling, Control and Automation, CIMCA 2005 and International Conference on Intelligent Agents, Web Technologies and Internet*, vol. 1, pp. 695–701, 2005.

[2] H. R. Tizhoosh and M. Ventresca, *Oppositional Concepts in Computational Intelligence*, vol. 155 of *Studies in Computational Intelligence*. 2008.

[3] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, “Opposition versus randomness in soft computing techniques,” *Applied Soft Computing Journal*, vol. 8, no. 2, pp. 906–918, 2008.

[4] H. R. Tizhoosh, “Reinforcement learning based on actions and opposite actions,” in *Proc. of the first ICGST International Conference on Artificial Intelligence and Machine Learning AIML 05*, vol. 05, (Cairo, Egypt), pp. 94–98, ICGST, Dec. 2005.

[5] H. R. Tizhoosh, “Opposition-based reinforcement learning,” *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 10, no. 4, pp. 578–585, 2006.

[6] M. Shokri, H. R. Tizhoosh, and M. Kamel, “Opposition-based Q(λ) algorithm,” in *IEEE International Conference on Neural Networks*, pp. 254–261, 2006.

[7] M. Shokri, H. R. Tizhoosh, and M. S. Kamel, "Opposition-based Q(λ) with non-markovian update," in *Proceedings of the 2007 IEEE Symposium on Approximate Dynamic Programming and Reinforcement Learning, ADPRL*, pp. 288–295, 2007.

[8] M. Shokri, H. R. Tizhoosh, and M. S. Kamel, "Tradeoff between exploration and exploitation of OQ(λ) with non-markovian update in dynamic environments," in *Proceedings of the International Joint Conference on Neural Networks*, pp. 2915–2921, 2008.

[9] M. Mahootchi, H. R. Tizhoosh, and K. Ponnambalam, "Opposition-based reinforcement learning in the management of water resources," in *Proceedings of the 2007 IEEE Symposium on Approximate Dynamic Programming and Reinforcement Learning, ADPRL*, pp. 217–224, 2007.

[10] M. Mahootchi, *Storage System Management Using Reinforcement Learning Techniques and Nonlinear Models*. Dissertation, University of Waterloo, Waterloo, Canada, 2009.

[11] F. Sahba, H. R. Tizhoosh, and M. M. M. A. Salama, "Application of opposition-based reinforcement learning in image segmentation," in *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Image and Signal Processing, CIISP*, 2007.

[12] M. Ventresca and H. R. Tizhoosh, "Improving the convergence of backpropagation by opposite transfer functions," in *IEEE International Conference on Neural Networks - Conference Proceedings*, pp. 4777–4784, 2006.

[13] M. Ventresca and H. R. Tizhoosh, "Opposite transfer functions and backpropagation through time," in *Proceedings of the 2007 IEEE Symposium on Foundations of Computational Intelligence, FOCI*, pp. 570–577, 2007.

[14] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[15] M. Ventresca and H. R. Tizhoosh, "Improving gradient-based learning algorithms for large scale feedforward networks," in *Proceedings of the International Joint Conference on Neural Networks*, pp. 3212–3219, 2009.

[16] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution algorithms," in *IEEE Congress on Evolutionary Computation*, pp. 2010–2017, 2006.

[17] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "A novel population initialization method for accelerating evolutionary algorithms," *Computers and Mathematics with Applications*, vol. 53, no. 10, pp. 1605–1614, 2007.

[18] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 64–79, 2008.

[19] S. Rahnamayan and G. G. Wang, "Solving large scale optimization problems by opposition-based differential evolution (ODE)," *WSEAS Transactions on Computers*, vol. 7, no. 10, pp. 1792–1804, 2008.

[20] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution for optimization of noisy problems," in *IEEE Congress on Evolutionary Computation, CEC*, pp. 1865–1872, 2006.

[21] S. Rahnamayan, *The Power of Oppositeness in Optimization: Toward Accelerating of Evolutionary Algorithms*. Verlag Dr. Müller, May 2009.

[22] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Quasi-oppositional differential evolution," in *IEEE Congress on Evolutionary Computation, CEC 2007*, pp. 2229–2236, 2008.

[23] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution (ODE) with variable jumping rate," in *Proceedings of the 2007 IEEE Symposium on Foundations of Computational Intelligence, FOCI*, pp. 81–88, 2007.

[24] S. Rahnamayan and H. R. Tizhoosh, "Image thresholding using micro opposition-based differential evolution (micro-ODE)," in *IEEE Congress on Evolutionary Computation, CEC*, pp. 1409–1416, 2008.

[25] L. Peng, Y. Wang, and G. Dai, *A novel opposition-based multi-objective differential evolution algorithm for multi-objective optimization*, vol. 5370 LNCS. 2008.

[26] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithms: NSGACII," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 6, pp. 182–197, 2002.

[27] J. Zhang, "Research on Indicator-Based Evolutionary Algorithm and Its Application in Constellation Design," Master's thesis, China University of Geosciences, Wuhan, China, 2008.

[28] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *8th Int'l Conf. on Paralle Problem Solving from Nature (PPSN VIII)*, pp. 832–842, Springer, Heidelberg, 2004.

[29] B. Bošković, S. Greiner, J. Brest, A. Zamuda, and V. Žumer, *An adaptive differential evolution algorithm with opposition-based mechanisms, applied to the tuning of a chess program*, vol. 143. 2008.

[30] M. G. Omran, *Using Opposition-based Learning with Particle Swarm Optimization and Barebones Differential Evolution*, ch. 23, pp. 343–384. Particle Swarm Optimization, InTech Education and Publishing, 2009.

[31] M. G. H. Omran, A. P. Engelbrecht, and A. Salman, "Bare bones differential evolution," *European Journal of Operational Research*, vol. 196, no. 1, pp. 128–139, 2009.

[32] M. G. H. Omran and A. P. Engelbrecht, "Free search differential evolution," in *IEEE Congress on Evolutionary Computation, CEC*, pp. 110–117, 2009.

[33] K. Penev and G. Littlefair, "Free search - a comparative analysis," *Information Sciences*, vol. 172, no. 1-2, pp. 173–193, 2005.

[34] M. G. H. Omran and A. Salman, "Constrained optimization using CODEQ," *Chaos, Solitons and Fractals*, vol. 42, no. 2, pp. 662–668, 2009.

[35] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[36] M. Ergezer, D. Simon, and D. Du, "Oppositional biogeography-based optimization," in *IEEE Conference on Systems, Man, and Cybernetics*, pp. 1035–1040, 2009.

[37] H. Lin and H. Xingshi, "A novel opposition-based particle swarm optimization for noisy problems," in *Proceedings - Third International Conference on Natural Computation, ICNC*, vol. 3, pp. 624–629, 2007.

[38] H. Wang, C. Li, Y. Liu, and S. Zeng, "A hybrid particle swarm algorithm with cauchy mutation," in *Proceedings of the 2007 IEEE Swarm Intelligence Symposium, SIS*, pp. 356–360, 2007.

[39] H. Wang, H. Li, Y. Liu, C. Li, and S. Zeng, "Opposition-based particle swarm algorithm with cauchy mutation," in *2007 IEEE Congress on Evolutionary Computation, CEC*, pp. 4750–4756, 2008.

[40] W. Zhangjun, N. Zhiwei, Z. Chang, and G. Lichuan, "Opposition based comprehensive learning particle swarm optimization," in *Proceedings of 2008 3rd International Conference on Intelligent System and Knowledge Engineering, ISKE*, pp. 1013–1019, 2008.

[41] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[42] M. G. H. Omran and S. Al-Sharhan, "Using opposition-based learning to improve the performance of particle swarm optimization," in *IEEE Swarm Intelligence Symposium, SIS*, 2008.

[43] H. Jabeen, Z. Jalil, and A. R. Baig, "Opposition based initialization in particle swarm optimization (O-PSO)," in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*, pp. 2047–2052, 2009.

[44] A. Silva, A. Neves, and E. Costa, "Chasing the swarm: A predator prey approach to function optimization," in *8th International Conference on Soft Computing*, 2002.

[45] F. Shahzad, A. R. Baig, S. Masood, M. Kamran, and N. Naveed, *Opposition-Based Particle Swarm Optimization with Velocity Clamping (OVCPSO)*, vol. 116 of *Advances in Computational Intelligence*, pp. 339–348. Springer Berlin, 2009.

[46] M. Ventresca and H. R. Tizhoosh, "Simulated annealing with opposite neighbors," in *Proceedings of the 2007 IEEE Symposium on Foundations of Computational Intelligence, FOCI*, pp. 186–192, 2007.

[47] A. R. Malisia and H. R. Tizhoosh, "Applying opposition-based ideas to the ant colony system," in *Proceedings of the 2007 IEEE Swarm Intelligence Symposium, SIS*, pp. 182–189, 2007.

[48] A. R. Malisia, "Investigating the application of opposition-based ideas to ant algorithms," Master's thesis, University of Waterloo, Waterloo, Canada, 2007.

[49] H.R.Tizhoosh, "Opposite fuzzy sets with applications in image processing," in *Proceedings of IFSA World Congress (International Fuzzy Systems Association)*, pp. 36–41, 2009.

[50] H. R. Tizhoosh and F. Sahba, "Quasi-global oppositional fuzzy thresholding," in *International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1346–1351, 2009.

[51] K. Yuchi, E. Yasunori, and M. Sadaaki, "Fuzzy c-means algorithms for data with tolerance based on opposite criterions," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E90-A, no. 10, pp. 2194–2202, 2007.

[52] F. Khalvati, H. R. Tizhoosh, and M. D. Aagaard, "Opposition-based window memoization for morphological algorithms," in *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Image and Signal Processing, CIISP*, pp. 425–430, 2007.