GLOBAL OPTIMIZATION METHODS: A REVIEW

[1]Archana Gupta, [2]Sanjaya Kumar Sahu
[1]Associate Professor Department of Electrical Engineering, Bhilai Institute of Technology, Durg, India
[2]Associate Professor Department of Electrical Engineering, Bhilai Institute of Technology, Durg, India

Abstract:Global optimization finds wide applications in almost all branches of engineering, physical sciences, applied sciences, social science, finance, and management. It is the task of finding the absolutely best set of admissible conditions to achieve objective, the global minimum. Extensive efforts have been made to address the issue and many global optimization methods have been developed. The global optimization methods are classified into groups deterministic and stochastic depending on whether or not they incorporate any stochastic elements. The two broad classes of global optimization deterministic and stochastic have their own benefiting features and limitations. In recent years many modified algorithms of both the classes have been developed. Hybridization of algorithms of different classes is the newly arisen concept in the field of global optimization. This paper surveys the research work done in the field of global optimization.

Index Terms - global optimization; deterministic methods; stochastic methods; hybrid algorithms

I. INTRODUCTION

Optimization is ubiquitous in many applications, e.g., in advanced engineering design, biotechnology, data analysis, environmental management, financial planning, process control, risk management, scientific modeling, and others. Optimization is an act, process, or methodology of making something (as a design, system, or decision) as fully perfect, functional, or effective as possible. In design, construction and maintenance of any system, decisions have to be taken at many stages, either to minimize the effort required, or to maximize the desired benefit. Since the effort required or the benefit desired in any practical situation can be expressed as a function of certain decision variables, optimization can be defined as the process of finding the conditions that give the maximum or the minimum value of a function. Optimization can be taken to mean minimization, since the maximum value of the function can be found by seeking the minimum of the negative of the same function [1]. The local optimal methods converge to whatever local minimum is closest to the starting approximation. As a result, the global minimum of an objective function is not known to a local optimization method. Global optimization is the task of finding the absolutely best set of admissible conditions to achieve the objective, i.e. the global minimum. Global optimization thus aims at determining not just “a local minimum” but “the smallest local minimum” with respect to the search domain. There is no single method available for solving all optimization problems efficiently. Hence, a number of methods have been developed for solving different types of optimization problems [2, 3, 4, 5, 6 & 7].

In general, the global optimization methods are classified into groups deterministic and stochastic depending on whether or not they incorporate any stochastic elements. Deterministic methods are applicable to optimization problem involving certain mathematical structure. These methods involve no element of randomness and run in finite time and return a region that is guaranteed to contain the global optimum. However, for larger dimensional models, and for more complicated model functions, the associated computational burden can easily become excessive. Stochastic optimization (SO) methods using random variables are considered as infinite processes for which the probability of having visited the global optima approaches one as the number of steps tends to infinity. Some stochastic methods known as meta-heuristic methods introduce randomness to accelerate the search process. Meta-heuristic methods optimize a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. However, they do not guarantee an optimal solution. Simulated annealing and tabu search are the examples of point-to-point meta-heuristic methods and Evolutionary algorithms (EA) like Genetic algorithms, Differential Evolution algorithm, ant colony algorithm and bee algorithm are population-based meta-heuristic methods.

The two broad classes of global optimization deterministic and stochastic have their own benefiting features and limitations. This paper surveys some of the optimization techniques currently in use in an attempt to analyze the strengths and weaknesses inherent in them.

II. GLOBAL OPTIMIZATION PROBLEM STATEMENT

In many engineering design and optimization applications, we are concerned with finding parameter values that achieve the optimum of an objective function. Generally, the function to be optimized is of the form:

$$f(X): R^D \rightarrow R \tag{1}$$

The optimization goal is to minimize the objective function $f(X)$ by optimizing the values of its parameters:

$$X = (x_1, \ldots\ldots, x_D),\ X \in R^D \tag{2}$$

Where X denotes a vector composed of D objective function parameters. Usually, the parameters of the objective function are also subject to lower and upper boundary constraints, $X(L)$ and $X(U)$,respectively:

$$x_j^{(L)} \le x_j \le x_j^{(U)} \quad j = 1, \ldots.., D \tag{3}$$

Note that no further structural assumptions, such as convexity or differentiability, are imposed on the objective function, and there may exist many locally optimal solutions. In other words, our focus is on general global optimization problems with little known structure.

III. DETERMINISTIC METHODS

Deterministic methods are applicable to optimization problem involving certain mathematical structure. These methods involve no element of randomness and run in finite time and return a region that is guaranteed to contain the global optimum. However, if the relation between a solution candidate and its fitness are not so obvious or too complicated, or the dimensionality of the search space is very high, it becomes harder to solve a problem deterministically. Also for larger dimensional models, and for more complicated model functions, the associated computational burden can easily become excessive [2, 3, 8 & 9]. Examples of deterministic methods include naive approaches [10], trajectory methods and homotopy methods [11], relaxation methods [9], Lipschitzian methods [12], cutting plane methods [13], branch and bound procedures [9], interval methods[4, 14, & 15] etc. To provide a rigorous guarantee for finding at least one or all global solutions, these methods typically impose additional assumptions on the objective function. Therefore, these are tailor made methods designed for special problems [1].

Branch and Bound Procedures: Branch and bound methods improve the performance of exhaustive search by analyzing and reducing the search phase during the optimization process. To achieve the reduction, they divide the search space to sub regions and decide the lower bound for any optima inside. Now any region having worse lower bound than the current optimum can be discarded. The performance depends heavily on the accuracy of the lower bound approximating function. To generate the approximating function, some characteristics of the fitness function must be known. This limits the usability of the branch and bound methods to problems for which the generation of lower bound approximating function is possible [16].

Interval methods: Interval methods use interval arithmetic which is a generalized or an extension of real arithmetic using sets of intervals and works on the branch and bound technique [17]. Interval arithmetic technique provided a natural tool for range computation [18]. It was useful for solving ordinary differential equations, linear systems and global optimization problems [15]. The interval arithmetic and its interaction with established mathematical theory was introduced and included in to traditional literature collections, as well as electronic resources [14]. Interval techniques were used to compute global information about functions over large regions (box-shaped), for e.g. strict bounds on function values, Lipschitz constants, higher derivatives etc. [19]. The Moore-Skelboe algorithm for global optimization is credited to Moore and Skelboe because Moore was probably the first to discover that interval arithmetic is an excellent tool for computing the range of the function over a box or an interval. Then it was combined with branch and bound principle [20].

The algorithm was again improved as Skelboe's strategy was combined with monotonicity tests, the centered form, and Krawzyk's version of Newton's method for global optimization [17]. The phrase "Global Optimization" was not used in the early papers, the first time it was used in Eldon Hansen's paper [4]. Starting with the initial box, the interval algorithm discards the part that does not contain the global minimum and obtains small intervals of which the union contains the global optimum of the problems through continuously subdividing search interval and updating the approximation of the optimal solution. Interval algorithms for optimization always obtain guaranteed bounds on all solutions and would never terminate prematurely.

Even though convergence is monotonic, they tend to be slow [14, 15 & 18]. Interval algorithms for minimizing a function in a box frequently have the difficulty that the sub-boxes containing no solution cannot be easily eliminated if there is a nearby good minimum. This has the effect that near zero, many small boxes are created by splitting, whose processing may dominate the total work spent on the global search. This so-called cluster effect happens near all the local minimizers with function value close to or below the best function values [21]. Further the Moore-Skelboe algorithm was somewhat slow to converge in 'difficult' problems, when inclusion function of first and second orders are used. Also the shortage of memory space and the high computational cost raised in interval algorithm for the high dimensional problems, the main reasons of which were large over estimations of the inclusion function and its derivatives, the high dimension of the problem; the objective function is not differentiable [4 & 19].

One of the solutions for these could be the construction of the accelerated methods. Most of the accelerated methods were concerning the inclusion function of the objective function and its derivatives [19]. The improved version of interval algorithm included accelerating devices like, monotonicity test, cut-off test, local search procedures, concavity tests and interval Newton like steps, which required that the objective function to be twice continuously differentiable and the constraint functions be once continuously differentiable [4 & 19]. Due to possible over estimations, these accelerating devices did not influence the worst case behavior and hence the theoretical convergence speed of a particular algorithm. For an objective function that is 'flat' around the global minimizer points, cut-off test does not help much. Monotonicity test device was not effcient for an objective function that had several saddle points and local minimizers. The same function could cause problems for the concavity test that discarded intervals over which the objective function was strictly concave in a variable. These tests can lead to better results on wide classes of optimization problems, but in general, neither can these classes be determined explicitly, nor can the worst case speed be improved (Csallner et al., 2000).

Further ways to increase effciency are the use of different inclusion functions, e.g., the slope functions [12, 14, 22] or the centered forms with Baumann centers [23]. An improved interval global optimization algorithm for unconstrained global optimization in the framework of the Moore-Skelboe algorithm used a higher order inclusion function form called Combined Taylor-Bernstein form. Combined Taylor-Bernstein form was an improved version of Taylor-Bernstein form which had the property of higher order convergence. It was more successful in computing the range enclosures as the domain shrinks from large to small widths [24]. A further improved version of the algorithm used a variety of inclusion function forms for the objective function the simple natural inclusion, the Taylor model, and the combined Taylor Bernstein form. However the accelerating tools and the higher order inclusion function forms are invalid for non-differentiable objective function [25].

Hence the subdivision direction, subdivision point and number for splitting the interval are the areas of interest for the improvement of interval algorithm. A new multi-splitting technology which subdivides the interval into several number in one iteration step was proposed to speed up the convergent rate [26] The algorithm involves additional calculations for determining the splitting direction. This can also mean some inclusion function calls for the objective function and its derivative and that consume time. Also the speed of the convergence to the global minimum value is the best for bisection and worst with multi-splitting. Further the algorithm becomes computationally expensive for high dimensional problems. A heuristic rejection index which helped in selecting the interval containing global optimal solution for subdivision was developed in [27 &28].

IV. STOCHOSTIC METHODS

Stochastic optimization (SO) methods using random variables are considered as infinite processes for which the probability of having visited the global optima approaches one as the number of steps tends to infinity. Stochastic methods evaluate the objective function on a random sample of points from a feasible solution set, and subsequently manipulate the samples. Containing stochastic elements, these methods are robust and adaptable to different problems. They put emphasis on efficiency and, convergence is guaranteed only in probability. Stochastic methods can reach the global optimum with probability one, under mild and general analytical assumptions [29]. Stochastic optimization includes randomness as either noise in the measurements or Monte Carlo randomness in the search procedure, or both. For stochastic problems, the random variables appear in the formulation of the optimization problem itself, which involve random objective functions or random constraints. Methods of this class include stochastic approximation (Robbins Monro algorithm) [30], finite difference stochastic approximation (Kiefer Wolfowitz algorithm) [31] and stochastic gradient descent [32].

V. META-HEURISTIC METHODS

Meta-heuristic designates a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. The word ‘meta-heuristic’ was coined by Fred Glover in his seminal paper [33]. Meta-heuristic algorithm does not make any assumptions on the optimized problem. The only required information is the fitness value with a given input. The advantage of these methods is that their structure is simple enough to be realized on a computer and they normally are not sensitive to local irregularities of the objective function. They may fail to detect promising search directions especially in the vicinity of local minima due to their random constructions. Thus they do not guarantee an optimal solution is ever found. Many of the meta-heuristic approaches rely on probabilistic decisions made during the search. But, the main difference to pure random search is that in meta-heuristic algorithms randomness is not used blindly but in an intelligent, biased form. Typically meta-heuristic algorithms require a set of user-specified parameters, whose values determine the success of the algorithm in the given task. Thus the user may provide problem-specific information through the selection of suitable parameters. On the other hand, if such information is not readily available, finding a good parameter setup may become a difficult task by itself. Meta-heuristic optimization algorithms can be classified into trajectory based or non-population based methods and population based [34].

Trajectory based algorithm: A trajectory-based algorithm typically uses a single agent or one solution at a time, which will trace out a path as the iterations continue. Examples of trajectory-based algorithms include Hill-climbing [35], Simulated annealing [36] and Tabu search [37]. In simple Hill climbing, the first closer node is chosen, whereas in steepest ascent hill climbing all successors are compared and the closest to the solution is chosen. Both forms fail if there is no closer node, which may happen if there are local maxima in the search space which are not solutions. Hill-climbing explores only the region that can be reached by the neighborhood function with a probability depending on the construction of the neighborhood function [35]. Simulated annealing is inspired by the physical behavior of material during annealing process. Simulated annealing (SA) was first introduced for combinatorial optimization, but has been later generalized for real coded functions. The algorithm simulates the cooling of molecules to a state of minimum energy (optimum). The general framework of the algorithm requires the definition of three main parts: distribution for sampling new points, acceptance function, and cooling schedule. The idea is to start from a random initial point (state) and generate a new random point according to the sampling distribution. The acceptance function is then used to decide whether the algorithm moves to the new state or not. Simulated annealing requires proper number of iterations at each temperature to get a fine balance between the number of evaluations and solution quality. Further the global information is limited to the cooling schedule, as the framework does not implement a generational memory [36].

Tabu search uses memory and the search history and share the property of describing a trajectory in the search space during the search process. The essential idea behind Tabu search is the use of tabu lists, which are used as the generational memory to direct the search globally. Tabu lists define some areas of the search space as taboo and direct the search away from these areas. Different rules for the lists can be used. Typically areas around the already found solutions are excluded to prevent the algorithm from searching the same areas multiple times. Essentially, the algorithm implements an advanced direct elimination of optima-principle, as the tabu lists are often temporary and exceptions can also be specified [37].

Population based algorithm : Population based algorithms use multiple agents which will interact and trace out multiple paths. Evolutionary algorithms like Evolutionary programming [38], Evolutionary strategies [39], Genetic algorithms [40, 41], Genetic programming [42] are population based meta-heuristic methods. These all share a common conceptual base of simulating the evolution of individual structures via processes of selection, recombination, and mutation reproduction, thereby producing better solutions. Evolutionary algorithms use iterative progress, such as growth or development in a population. This population is then selected in a guided random search using parallel processing to achieve the desired end. Such processes are often inspired by biological mechanisms of evolution. Simulations of evolution using evolutionary algorithms and artificial life, started with the work of Nils Aall Barricelli in the 1960s, and was extended by Alex Fraser, who published a series of papers on simulation of artificial selection [43]. Evolutionary algorithms are robust with respect to noisy evaluation functions, and the handling of evaluation functions which do not yield a sensible result in given period of time is straightforward. The algorithms can easily be adjusted to the problem at hand. On the other hand, for a given problem it is difficult to decide which evolutionary algorithm is best suited. While the standard values usually provide reasonably good performance, different configurations may give better results. Furthermore, premature convergence to a local extremum may result from adverse configuration and may not yield the global extremum [44]. Evolutionary programming is an evolutionary approach that treats the instances of the genome as different species rather than as individuals. Hence, mutation and selection are the only operators used in Evolutionary programming and recombination is usually not applied. Evolutionary programming typically uses an adaptive mutation operator in which the severity of mutations is often reduced as the global optimum is approached. Evolutionary programming was introduced by Fogel in his PhD thesis back in 1964 [38]. One drawback of Evolutionary Programming is that it is very difficult almost impossible to know if the ideal termination criterion is going to be satisfied, or not. Evolutionary Strategies are numerical optimizations based on evolutionary principles pioneered by Rechenberg [39]. Evolutionary strategies are generally used for empirical experiments that are difficult to be modeled mathematically. Contrast to Evolutionary programming, Evolutionary strategies use deterministic selection. Also Evolutionary strategies optimize the standard deviations according to the actual topology of the objective function using crossover operation. Genetic algorithms (GA) were invented by John Holland in the early 1970s [41]. The main difference between evolutionary programming and evolutionary strategies is that the variations are not performed directly on the problem solution. There is a (often binary) representation of the solution which is varied and mapped onto the real solution by some function. The real solution is then tested on the problem to get the fitness of the individual. Genetic algorithms perform a search by evolving a population of candidate solutions through the use of a nondeterministic operator [40]. GA was originally developed for binary search-spaces, but optimization over real-valued search-spaces is also possible [45]. GAs become very inefficient if the fitness function is not decomposable, i.e. its parameters are not independent. Unfortunately, small mutation rates cause severe problems, when the parameters of the fitness function are mutually dependent. A rotation of the coordinate system makes the parameters dependent on each other. When optimizing a multi-modal function with dependent parameters, the complexity can increase. Further, the newly obtained solutions may not be always better than the previous ones. They may have a tendency to converge towards local optima or even arbitrary points rather than the global optimum. A survey of some theoretical and practical aspects of GAs is given in [46].

Genetic Programming (GP) is the most recent important new model for evolutionary computation. Genetic programming (GP) technique provides a framework for automatically creating a working computer program from a high-level statement of the problem. The main difference between GP and GA is the representation of the solution. GP creates computer programs in the LISP or scheme computer languages as the solution. Genetic Programming does not impose any fixed length of the solution. Computer programs evolved by GP have variable length, so that the algorithms can find the size necessary for a solution itself. GP requires data structures that are easy to handle and evaluate and are robust to structural manipulations. The set of functions and terminals that will be used in a specific problem has to be chosen carefully. If that set of functions is not powerful enough, a solution may be very complex or may not be found at all. Further, depending on the programming language that is evolved by the GP algorithm, the process of finding a good solution can take a very long time [42].

An interesting evolutionary strategy referred as Differential Evolution (DE) has been proposed by Price and Storn, which is a simple and yet powerful EA using real coding of floating point numbers [47]. Differential evolution algorithm is an improved version of genetic algorithm using similar operators; mutation, crossover and selection. While constructing better solutions, it relies on mutation operations. Also DE's self-referential population reproduction scheme is different from the other evolutionary algorithms. Further, the algorithm uses selection operations to direct the search towards the prospective regions in the search space. Thus, DE is a simple, efficient and extremely robust direct search algorithm for global optimization over continuous spaces [48 & 49]. Four different mutation schemes have been suggested for DE algorithm to improve its convergence [50]. Ten different strategies of DE were presented using these mutation schemes. Each mutation strategy is combined with either the 'exponential' type crossover or the 'binomial' type crossover [51]. However, a strategy that works out to be the best for a given problem may not work well when applied to a different problem. The DE algorithm has only a few control parameters like population size, scaling factor and crossover constant, which are kept fixed throughout the entire evolutionary process. However, depending upon the specific problem the controlling parameters are chosen, and are often difficult as the parameter tuning done mostly by trial-and-error [52]. One of the important aspects of any algorithm is the termination criterion. For practical applications the choice of stopping criteria can significantly influence the duration of an optimization run as well as the global minimum. DE terminates either with user defined number of generations or some error estimate. With pre-defined number of generations, the optimization run may terminate before the population has converged, or may terminate late thus, wasting computational resources. Also the algorithm uses trial-and-error methods to determine the maximum number of function evaluations. The second criterion requires some optimum value which is not possible for an arbitrary function. Thus the above termination criteria are not suitable for solving real world problems. The other drawback of DE is the slow convergence near the region of global minimum [49].

The rate of convergence of DE as well as its accuracy can be improved largely by applying different mutation and selection strategies. Modified versions of DE can be subdivided into two classes: algorithms using DE as an evolutionary framework assisted by additional algorithmic components and algorithms making a substantial modification within the DE structure [53]. Schemes that incorporate additional components in the standard DE framework include Trigonometric Differential Evolution (TDE) [54], Differential Evolution with Adaptive Hill Climbing Simplex Crossover (DEAHCSC) [55], DE with Population Size

Reduction (DEPSR) [56] etc. TDE used a trigonometric mutation operator which is a greedy operator that generates an offspring for three given points by exploiting the most promising search directions. The employment of this operator within TDE is supposed to offer an exploitative alternative to the standard exploration rule of DE [54]. In order to enhance performance of DE, DEAHCSC hybridizes DE as an evolutionary framework and a Local Searcher Simplex Crossover. A proper balance of the exploration abilities of DE and the exploitation abilities of a Local Searcher can lead to an algorithm with high performance. Local Searcher is deterministically applied to the individual of the DE population with the best performance (in terms of fitness value) [53].

The DEPSR employs, within a DE framework, a variable population size which is progressively reduced during the optimization process. During the early stages of the optimization process, the search requires a highly explorative search rule, i.e. a large population size, in order to explore a large portion of the decision space. During optimization, the search space is progressively narrowed by decreasing the population size and thus exploiting the promising search directions previously detected [56].

Algorithms which make a substantial modification within the DE structure, in the search logic, the selection etc., to enhance the performance of the original DE include self-adapting control parameters in Differential Evolution (Self-Adaptive DE) [57, 58 &59], Opposition Based DE (OBDE) [60]and Self Adaptive Coordination of Multiple Mutation Rules (SADE)[61, 62 &63]. These modifications tend to include extra moves within a standard DE and attempt to find logic for proposing the correct extra moves.

Self-Adaptive DE is based on the idea that fixed values of scaling factor and crossover constant can cause stagnation in a DE scheme. On the other hand, a full randomization of the control parameters would lead to an algorithm which is very similar to random search. For this reason, the restricted randomization proposed in Self-Adaptive DE seems to be an efficient compromise between the necessity of generating extra moves and the willingness to keep some exploitative features of the optimization algorithm [56, 57 &58]. The OBDE algorithm proposes an attempt to check, under certain probability conditions, unexplored areas of the decision space regardless of the stage of evolution of the population. The generation of extra individuals by means of projection with respect to a focus (opposition points) relies on the fact that the extra points test unexplored areas and refresh DE logic [60]. The SADE employs a multiple mutation strategy which allows an increase in the DE moves. On the other hand, the trial and error self-adaptation increases exploitative features of the algorithm since successful moves tend to be repeated and constitute a preferential choice during subsequent generations [53].

VI. OTHER ALGORITHM

In addition to EAs, another group of population based meta-heuristics, which takes its inspiration from biology, are the methods based on swarm intelligence [64]. Among these, Particle Swarm Optimization [65](PSO) is best suited for continuous optimization which is inspired by social behaviour of bird flocking or fish schooling. Population members are referred to as particles, which fly through the search space and evaluate the fitness values of visited points. The direction and velocity (step size) of each particle is affected by two attractors: the best location the particle has personally visited this far, and the best known point located by any particle in a certain neighborhood. PSO easily suffers from the partial optimism, which causes the less exact at the regulation of its speed and the direction. Then the method cannot work out the problems of scattering and optimization and also the problems of non-coordinate system, such as the solution to the energy field and the moving rules of the particles in the energy field [66]. The Ant Colony Algorithm (ACO) introduced by Dorigo in his Ph.D. thesis, is a technique for solving optimization problems that relies on probability [67]. The inspiring source of ACO is the foraging behavior of ants in their search for food, i.e., over a period of time ants are able to determine the shortest path from their home to a food source. In ACO, solutions of the problem are constructed within a stochastic iterative process, by adding solution components to partial solutions. Each individual ant constructs a part of the solution using an artificial pheromone, which reflects its experience accumulated while solving the problem, and the heuristic information dependent on the problem [68].

VII. HYBRID ALGORITHM S

The Hybrid Optimization methods are a combination of the deterministic and the evolutionary or stochastic methods, in the sense that they try to use the advantages of each of these methods. The Hybrid Optimization method usually employs an evolutionary or stochastic method to locate an approximate region where the global extreme point is located and then automatically switches to a deterministic method to reach the exact point faster [1]. Few algorithms are reported in the literature combining the interval algorithm and stochastic approaches together [69, 70, 71, 72, 73, 74 & 75]. A Branch and Bound Method for Stochastic Global Optimization is a stochastic version of the branch and bound method, which instead of deterministic bounds uses stochastic upper and lower estimates of the optimal value of sub-problems [71]. Algorithm exploits the stochastic nature of the problem together with different deterministic bounding techniques for the construction of stochastic lower bounds. However, the selection of deterministic bounding technique for a particular stochastic optimization problem is difficult. Also the algorithm is only applicable to stochastic optimization. A Parallel Interval Evolutionary Algorithm integrates interval arithmetic and Mind Evolutionary Computation to explorer the new parallel interval decomposition scheme that can solve computationally intensive problems and can determine all the optimal solution reliably [69 &70]. The algorithm uses Evolution algorithm to guide the split direction and interval algorithm to reduce the search space. However, the parallel characteristics and the scalability of algorithm is combined to the high speed parallel computers to improve the efficiency. Thus parallel computing is required. An indirect method was introduced which was based on mapping from the output into the input space using Cellular Evolutionary Strategies (CES) and Interval Arithmetic (IA) to obtain robust system design. CES and IA together obtain, by an iterative process, a robust design, that is, the maximum size of each variable deviation that allow to comply with a set of specifications. CES are an approach that combines the Evolution Strategy techniques with concepts from Cellular Automata in order to optimize a given function, while IA is used as a checking technique that guarantees the feasibility of the design by providing strict bounds (minimum and maximum values) with only one evaluation [72]. Evolutionary algorithm was used in place of Cellular Evolutionary Strategies together with Interval Arithmetic (IA) to obtain robust system design. The algorithm uses to Evolutionary algorithm to optimize a given function [73].

In a Hybrid Interval Genetic algorithm (HIG) an interval branch and bound algorithm was used to obtain small regions where candidate solution lie. Then the genetic algorithm was applied in such a way that all the above pieces of information are exploited. HIG constructed a mechanism using interval arithmetic that updated the lower and upper bound in each generation, and defined a new termination criterion based on the concept of shrinking box [74]. However, the algorithm gives only one global minimum thus is not useful where clusters of global minima exist. Further, use of GA algorithm makes the HIG algorithm computationally expensive.

A new interval genetic (NIG) algorithm used the genetic algorithm as local search device in the interval subdivision algorithm frame work. It used the improved upper bound of the global optimal value obtained by the genetic algorithm to delete the intervals not containing the global optimal solution from the work set at each iteration. Using the interval arithmetic the algorithm produced the reliable domains where the genetic algorithm was applied to search. Also the genetic algorithm proposed a direction of dividing the reliable interval and an upper bound of global optimum used to reduce the intervals at each iteration [75]. However, the algorithm inherits the drawbacks of genetic algorithm as explained earlier (2.1.3). Also the NIG algorithm is computationally expensive as the genetic algorithm may be trapped in local minimums.

VIII. CONCLUSION

An exhaustive survey on global optimization methods was presented here. Deterministic methods are applicable to optimization problem involving certain mathematical structure. These methods involve no element of randomness and run in finite time and return a region that is guaranteed to contain the global optimum. However, for larger dimensional models, and for more complicated model functions, the associated computational burden can easily become excessive. On the other hand stochastic methods evaluate the objective function on a random sample of points from a feasible solution set, and subsequently manipulate the samples. these methods are robust and adaptable to different problems. However the convergence is guaranteed only in probability. The selection of optimization method for a particular problem is quite difficult as the nature and characteristic of the optimization problem governs the applicability of method. Also a trade-off between the accuracy of solution and computational burden is required. In recent past years researchers focused on evolutionary algorithms.

REFERENCES

[1] Ray, S. A New Approach to Range Computation of Polinomials Using Bernstein Form. Ph.D dissertation, Indian Institute of Technology Bombay, 2007.
[2] Floudas, C. A. and Gounaris, C. E. A review of recent advances in global optimization. Journal of Global Optimization, Vol. 45, (1), pp.3-38, 2009.
[3] Floudas, C. A. and Pardalos, P. M. Recent Advances in Global Optimization. Prince ton University Press, Princeton, NJ, 1992.
[4] Hansen, E. Global optimization using interval analysis the multi-dimensional case, NumerischeMathematik, Springer Berlin, vol. 34(3), pp.247-270, 1980.
[5] Horst, R., Pardalos, P. M., and V.Thoai, N, Introduction to global optimization, Kluwer Academic Publishers, Dordrecht, 1995.
[6] Horst, R., Pardalos, P. M., and V.Thoai, N. , Introduction to global optimization, Kluwer Academic Publishers, Dordrecht, 1995.
[7] Pinter, J. D. Global optimization: software, test problems, and applications, vol.2. Kluwer Academic Publishers, London, 2002
[8] Floudas, C. A. Deterministic Global Optimization Theory, Methods and Applications. Kluwer Academic Publishers, London.1999.
[9] Horst, R. and Tuy, H. Global Optimization - Deterministic Approaches. Springer, Heidelberg, third edition, 1996
[10] Pinter, J. D. Global Optimization in Action (Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications). Kluwer Academic Publishers, Dordrecht. 1996
[11] Forster, W. Handbook of Global Optimization:Nonconvex optimization and its applications, chapter Homotopy methods, pp. 669–750. Kluwer Academic Publishers, Dordrecht, 1995.
[12] Hansen, P. Jaumard, B., and Lu, S. Global optimization of univariatelipschitzfunctions: I survey and properties. Mathematical Programming, vol, 55, pp.:251-272, 1992.
[13] Tuy, H., Thieu, T. V., and Thai, N. Q. A conical algorithm for globally minimizing a concave function over a closed convex set. Mathematics of Operations Research, vol. 10(3), pp.498-514, 1985.
[14] Kearfott, R. B. Interval computations: Introduction, uses, and resources. Euromath Bulletin, vol. 1(2), pp.95-112, 1994.
[15] Moore, R. E. Methods and applications of Interval analysis. SIAM, Philadelphia, 1979.
[16] Tawarmalani, M. and Sachinidis, N. Handbook of Global Optimization, vol, 2, chapter Exact Algorithms for Global optimization of mixed-integer nonlinear programs, pp. 65-85. Springer, 2002.

[17] Moore, R. E. On computing the range of values of a rational function of n variables over a bounded region. Computing, vol.16, pp.1–15. 1976.
[18] Ratschek, H. and Rockne, J. New Computer Methods for Global Optimization. Ellis Horwood Ltd, England, 1988.
[19] Hansen, E. Global optimization using interval analysis. Dekker, New York, 1992.
[20] Skelboe, S. Computation of rational interval functions. BIT, vol.14, pp.87-95, 1974.
[21] Kearfott, R. B. and Du, K. The cluster problem in multivariate global optimization. Journal of Global Optimization, vol. 5, pp.253-265, 1994.
[22] Ratz, D. A nonsmooth global optimization technique using slopes: The one dimensional case. Journal of Global Optimization, vol.14(4), pp.365-393, 1999.
[23] Sotiropoulos, D. G. and Grapsa, T. N. Optimal centers in branch and prune algorithms for univariate global optimization. Applied Mathematics and Computation, vol.169, pp.247-277, 2005.
[24] Nataraj, P. S. V. and Kotecha, K. An algorithm for global optimization using the taylorbernstein form as inclusion function. Journal of Global Optimization,vol. 24(4), pp.417-436, 2002.
[25] Nataraj, P. S. V. and Kotecha, K. An improved interval global optimization algorithm using higher order inclusion function forms. Journal of Global Optimization, vol.32(1), pp.35-63, 2005.
[26] Casado, L. G., Garca, I., and Csendes, T. A new multisection technique in interval methods for global optimization. Computing, vol,65, pp.263-269, 2000.
[27] Casado, L. G., Garca, I., and Csendes, T. A heuristic rejection criterion in interval global optimization algorithms. BIT Numerical Mathematics, vol.41 (4), pp.683-692, 2001.
[28] Casado, L. G., Garca, I., Csendes, T., and Ruiz, V. G. Heuristic rejection in interval global optimization. Journal of Optimization Theory and Applications, vol.118(1), pp.27-43, 2003.
[29] Pinter, J. D. Global optimization: software, test problems, and applications, volume 2. Kluwer Academic Publishers, London, 2002.
[30] Robbins, H. and Monro, S. A stochastic approximation method. Annals of Mathematical Statistics, vol.22 (3), pp.400-407, 1951.
[31] Kirkpatrick, S., Gelatt, C. D., and Vecchi, J. M. P. Continuous optimization by a variant of simulated annealing. Science, vol.220 (4598), pp.671-680, 1983.
[32] Mitchell, T. Machine Learning. McGraw-Hill, Boston, MA, 1997.
[33] Glover, F. . Future paths for integer programming and links to artifficial intelligence. Computers and Operations Research, vol.13 (5), pp.533-549, 1986.
[34] Glover, F. and Kochenberger, G. Handbook of Metaheuristics. Kluwer Academic Publishers, 2003.
[35] Russell, S. J. and Norvig, P. Artificial Intelligence: A Modern Approach, pp. 111-114. Prentice Hall, Upper Saddle River, NJ, 2 editions, 2003.
[36] Kirkpatrick, S., Gelatt, C. D., and Vecchi, J. M. P. Continuous optimization by a variant of simulated annealing. Science, vol.220 (4598), pp.671-680, 1983.
[37] Glover, F. and Laguna, M. Tabu Search. Kluwer Academic Publishers, London, 1997.
[38] Fogel, L. J. On the organization of intellect. PhD thesis, UCLA University of California, Los Angeles, California, USA, 1964.
[39] Rechenberg, I. Cybernetic solution path of an experimental problem,library translation. Technical Report 1122, Royal Aircraft Establishment, Farnborough, 1965.
[40] Goldberg, D. E. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Corporation, Inc, Reading, MA, 1989.
[41] Holland, J. Adaptation in Natural and Artificial Systems. University of Michican Press, Ann Harbor, MI. 1975.
[42] Koza, J. R. Genetic Programming. MIT Press, Cambridge, MA, 1992.
[43] Fraser, A. S. Monte carlo analyses of genetic models. Nature, vol.181, pp.208-209, 1958.
[44] Back, T. Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic algorithms. Oxford University Press, New York, 1996.
[45] Herrera, F., Lozano, M., and Verdegay, J. Tackling real-coded genetic algorithms: operators and tools for behavioural analysis. Artificial Intelligence Review, vol.12 (4), pp.265-319, 1998.
[46] Salomon, R. Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. a survey of some theoretical and practical aspects of genetic algorithms. Bio Systems, vol.39, pp.263-278, 1996.
[47] Storn, R. and Price, K. Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, International Computer Science Institute, 1995.
[48] Price, K. New Ideas in Optimization, chapter An Introduction to Differential Evolution. McGrawHill, London, 1999.
[49] Storn, R. and Price, K. Differential evolution, a simple and efficient heuristic strategy for global optimization over continuous spaces. Journal of Global Optimization, vol.11, pp.341-359, 1997.
[50] Storn, R. On the usage of differential evolution for function optimization. In Lee, M. A., Keller, J., Yen, J., and Smith, M. H., editors, Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS 1996) , Berkeley initia, North American Fuzzy Information Process, IEEE, IEEE Neural Networks Council, pp. 519- 523, Berkeley, California, USA, 1996.
[51] Storn, R., Price, K., and Lampinen, J. Differential Evolution - A Practical Approach to Global Optimization. Springer, Berlin, 2005.

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

[53] Neri, F. and Tirronen, V. Recent advances in differential evolution: A review and experimental analysis. Artificial Intelligence Review, vol.33 (1), pp.61-106, 2010.

[54] Fan, H. Y. and Lampinen, J. A trigonometric mutation operation to differential evolution. International Journal of Global Optimization, vol.27 (1), pp.105-129, 2003.

[55] Noman, N. and Iba, H. Accelerating differential evolution using an adaptive local search. IEEE Transaction on Evolutionary Computation, vol.12 (1), pp.:107-125, 2008.

[56] Brest, J. and Maucec, M. S. Population size reduction for the differential evolution algorithm. ApplIntell, vol.29(3), pp.228-247, 2008.

[57] Brest, J., Bokovi, C. B., Greiner, S., and umer, V. Performance comparison of self adaptive and adaptive differential evolution algorithms. Soft Computing, vol.11(7), pp.:617-629,2007.

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

[59] Tvrdik, J. Adaptive differential evolution: Application to nonlinear regression. In Oz, W. V. and Yannakakis, M., editors, Proceedings of the International Multi conference on Computer Science and Information Technology, pp. 193-202, 2007.

[60] Rahnamayan, S., Tizhoosh, H. R., and Salama, M. M. A. Opposition-based differential evolution algorithmsh. In Proceedings of IEEE Congress on Evolutionary Computation, 2006.

[61] Qin, A. K., Huang, V. N., and Suganthan, P. N. Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Transaction on Evolutionary Computation, vol.13 (2), pp.398-417, 2009.

[62] Qin, A. K. and Suganthan, P. N. Self-adaptive differential evolution algorithm for numerical optimization. In IEEE congress on evolutionary computation, volume 2, pp. 1785-1791, 2005.

[63] Yang, Z., Tang, K., and X.Yao Self-adaptive differential evolution with neighborhood search. In the world congress on computational intelligence, pp. 1110–1116, 2008.

[64] Kennedy, J., Ebenhart, R., and Shi, Y. Swarm Intelligence. Morgan Kaufmann, 2001.

[65] Kennedy, J. and Ebenhart, R. Particle swarm optimization. In IEEE International Conference on Neural Networks (ICNN 1995), pp, 1942–-948, Perth, Australia, 1995.

[66] Rini, D. P., Shamsuddin, S. M., and Yuhaniz, S. S. Particle swarm optimization: Technique, system and challenges. International Journal of Computer Applications, vol.14 (1), pp.19-27, 2011.

[67] Dorigo, M. Optimization, Learning and Natural Algorithms. PhD thesis, Politecnicodi Milano, Italy, 1992.

[68] Dorigo, M. and Sttzle, T. Ant Colony Optimization. The MIT Press, Cambridge, MA, 2004.

[69] Lei, Y. and Chen, S. A reliable parallel interval global optimization algorithm based on mind evolutionary computation. In Fourth ChinaGrid Annual Conference, pp, 205-209. IEEE Computer Society Washington, DC, USA, 209.

[70] Lei, Y., Chen, S., and Yan, Y. A novel parallel interval exclusion algorithm. In High Performance Computing and Applications, vol. 5938/2010, pp. 218-223. Lecture Notes in Computer Science, Springer, 2010.

[71] Norkin, V. I., Pflug, G. C., and Ruszczynski, A. A branch and bound method for stochastic global optimization. Mathematical Programming: Series A and B Springer Verlag, vol.83 (3), 1998.

[72] Sanseverino, C. M. R. A hybrid approach based on evolutionary strategies and interval arithmetic to perform robust designs. In Lecture Notes in Computer Science, vol.3449/2005, pp. 623-628. Springer, 2005.

[73] Sanseverino, C. M. R. and Aponte, D. E. S. A hybrid approach based on evolutionary strategies and interval arithmetic to perform robust designs. In Yang, S., Ong, Y., and Jin, Y., editors, Evolutionary Computation in Dynamic and Uncertain Environments, volume 51 of Studies in Computational Intelligence, pp. 543-564. Springer, 2007.

[74] Sotiropoulos, D. G., Stavropolous, E. C., and Vrahatis, M. N. A new hybrid genetic algorithm for global optimization. Nonlinear Analysis, vol.30 (7), pp.4529-4538, 1997.

[75] Zhang, X. and Liu, S. A new interval-genetic algorithm. In Natural Computation, ICNC, pp. 193-197, Haikou. IEEE Xplore Digital Library, 2007.