IPMP Journal of Management & Science (IPMP-JMS), ISSN: 2590-9681, Scientific Journal Impact Factor (SJIF): 4.309
Vol 2, Issue 06, June, 2018, Pages 131 - 148, Available @ www.ipmp-jms.org

Evolutionary Algorithms (EAs) for Vehicle Routing Problem (VRP)

Andrew Owusu-Hemeng[1], Peter Kwasi Sarpong[2], Joseph Ackora-Prah[3]
[1,2,3]Department of Mathematics, Kwame Nkrumah University of Science and Technology,Kumasi,Ghana
Email: owusuhemengandrew@gmail.com, kp.sarp@yahoo.co.uk, ackph@yahoo.co.uk

Abstract
Genetic Algorithms (GAs) are powerful and widely applicable stochastic search technique and optimization methods based on the principles of genetics, natural selection and natural evaluation. Thus GA is a stochastic global search method that mimics the metaphor of natural biological evolution. This work discusses the concept of design procedure of Genetic Algorithm and explores a well-established methodology of the literature to realize the workability and application of genetic algorithm. The Vehicle Routing Problem (VRP) is a complex combinatorial optimization problem where one or more vehicles can be used in the solution. The optimization can be described as follows: given a fleet of vehicles, a common depot and several requests by the customers, find the set of routes with overall minimum route cost which service all the demands. Because of the fact that VRP is already a complex, namely an NP-complex problem, heuristic optimization algorithms, like Genetic Algorithms (GAs) need to be taken into account. This requires special, interpretable encoding to ensure efficiency. In this paper, GA is used to solve these problems and propose a novel, easily interpretable. Genetic algorithms are used to model the Vehicle Routing Problem. MATLAB simulations was carried out to find the optimal route of Amponsah Efah Pharmaceuticals Limited as. The corresponding distances (fitness) of their distribution route was found to be 7560m.

I. INTRODUCTION

Evolutionary Algorithms (EAs) are computer programs that attempt to solve complex problems i.e., Combinatorial Optimization Problems (COPs) by mimicking the processes of Darwinian evolution. In an EA a number of artificial creatures search over the space of the problem. They compete continually with each other to discover optimal areas of the search space. It is hoped that over time the most successful of these creatures will evolve to discover the optimal solution. The artificial creatures in EAs, known as individuals, are typically represented by fixed length strings or vectors. Each individual encodes a single possible solution to the problem under consideration. Over recent years the application of EAs to Combinatorial Optimization Problems (COPs) in computational chemistry has become common place and there is now a growing collection of published applications. One of the numerous algorithms proposed for applying to these COPs is Genetic Algorithm. Genetic algorithms (GAs) are search methods based on principles of natural selection and genetics (Fraser, 1957; Bremermann, 1958; Holland, 1975). GAs encode the decision variables of a search problem into finite-length strings of alphabets of certain cardinality. The strings which are candidate solutions to the search problem are referred to as *chromosomes*, the alphabets are referred to as *genes* and the values of genes are called *alleles*. For example, in a problem such as the traveling salesman problem, a chromosome represents a route, and a gene may represent a city. GAs work with coding of parameters, rather than the parameters themselves. Genetic algorithms are often viewed as function optimizer, although the range of problems to which genetic algorithms have been applied are quite broad.

II. BACKGROUND

Many human inventions were inspired by nature. One example is Genetic Algorithm (GA) which is inspired by Charles Darwin's theory about evolution – "the survival of the fittest". In nature, competition among individuals for scanty resources according to the principle of selection (survival of the fittest) results in the fittest individuals dominating over the weaker ones to reach certain remarkable tasks. An implementation of genetic algorithm begins with a population of (typically random) chromosomes. John Holland's pioneering book *Adaptation in Natural and Artificial Systems* (1975, 1992) showed how the evolutionary process can be applied to solve a wide variety of problems using a highly parallel technique that is now called the *genetic algorithm*. This particular branch of Evolutionary Algorithms (EAs) was inspired by the way living things evolved into more successful organisms in nature. Before applying the genetic algorithm to the problem, the user designs an artificial chromosome of a certain fixed size and then defines a mapping (encoding) between the points in the search space of the problem and instances of the artificial chromosome. GAs search by simulating evolution, starting from an initial set of solutions or hypotheses, and generating successive "generations" of solutions. One then evaluates these structures and allocated reproductive opportunities in such a way that these chromosomes which represent a better solution to the target problem are given more chances to `reproduce' than those chromosomes which are poorer solutions. The 'goodness' of a solution is typically defined with respect to the current population.

Changes occur during reproduction. The *genetic algorithm* (GA) transforms a *population* (set) of individual objects, each with an associated *fitness* value, into a new *generation* of the population using the Darwinian principle of reproduction and survival of the fittest and analogs of naturally occurring genetic operations such as *crossover* (*sexual recombination*) and *mutation*.

Each *individual* in the population represents a possible solution to a given problem. The genetic algorithm attempts to find a very good (or best) solution to the problem by genetically breeding the population of individuals over a series of generations. The genetic algorithm differs from other search methods in that it searches among a population of points, and works with a coding of parameter set, rather than the parameter values themselves. It also uses objective function information without any gradient information. The transition scheme of the genetic algorithm is probabilistic, whereas traditional methods use gradient information. Because of these features of genetic algorithm, they are used as general purpose optimization algorithm. They also provide means to search irregular space and hence are applied to a variety of function optimization, parameter estimation and machine learning applications.

III. REVIEW OF RELATED WORK

Kannan, Sasikumar and Devika (2010) recast a Genetic Algorithm approach for solving a closed supply chain model. A case of battery recycling. In this paper the authors developed a closed loop mixed integer linear programming model to determine the raw material level, production level, distribution and inventory level, disposal level and recycling level at different facilities with the objective of minimizing the total supply chain costs. The model is solved by the proposed heuristics based genetic algorithm and for smaller size problem the computational results obtained through Genetic Algorithm are compared with the solutions obtained by GAMS optimization software. The problem of drawing graphs nicely contains several computationally intractable sub-problems. It is on this note that Eloranta and Makinen (2001) presented a paper Tim GA: A genetic algorithm for drawing undirected graphs. The authors indicated that it is natural to apply genetic algorithms to graph drawing. Their paper introduces a genetic algorithm (Tim GA) which nicely draws undirected graphs of moderate sizes. The aesthetic criteria used are the number of edge crossing, even distribution of nodes, and edge length deviation. Eloranta and Makinen (2001) indicated that although Tim GA usually works well, there are some unsolved problems related to the genetic crossover operation of graphs and concluded that Tim GA's search is mainly guided by the mutation operations.

Various applications of Genetic Algorithms to the problem of image segmentation are explored by Keri Woods (2007) on his paper Genetic Algorithm: Colour image segmentation to discuss the feasibility of using genetic algorithms to segment general colour images and also discuss the issues involved in designing such algorithms. Keri Woods (2007) indicated that Genetic Algorithms are commonly used approach to optimizing the parameters of existing image segmentation algorithms and stated that the major decisions are choosing a method of segmentation to which genetic algorithms will be applied, finding a fitness function that is a good measure of the quality of image segmentation and finding a meaningful way to represent the chromosomes. Keri Woods (2007) used modified GAs and Hybrid GAs to solve this problem. A recast of genetic algorithms and the Evolution of Neutral Networks for language processing by Jaine T. (2009) presents ways in which he have used GAs to find which Neutral Network (NN) parameter values produce natural language task. In addition to this, the system has been modified and studied in order to evaluate ways in which coding methods in the GA and the NN can affect performance. In the case of GA coding, an evolution method based on schema theory is presented. This methodology can help determine optimal balances between different evolutionary operators such as crossover and mutation, based on the effect of different ways of presenting words and sentences at the output layer is examined with binary and floating point schemes.

A dynamic routing control based on genetic algorithm can provide flexible real time management of the dynamic traffic changes in broadband networks. It was demonstrated through computer simulations using genetic algorithms by Shimanoto N. (2000). The proposed technique can generate the exact solution of path arrangement that keeps the traffic loss rate below the target value, even after changes in traffic. Takagi-Sugeno-Kang (TSK) type recurrent fuzzy network is proposed by Juang (2002), which develops from a series of fuzzy if-then rules with Takagi-Sugeno-Kang (TSK) type consequent parts. Takagi-Sugeno-Kang (TSK) type recurrent fuzzy network with supervised learning is suggested for the problems having on-line training data. To demonstrate the superiority of Takagi-Sugeno-Kang (TSK) type recurrent network, it is applied to dynamic system. By comparing the results the efficiency of Takagi-Sugeno-Kang (TSK) type recurrent fuzzy network is verified.

Design of direct form of a finite word length, finite impulse response (FIR) low pass filter was proposed by Xu and Daley (1995). The results of the proposed design techniques are compared with an integer programming technique

and it is inferred from the results that genetic algorithm based technique outperforms the traditional approach. Design of optimal disturbance rejection using genetic algorithm was suggested by Krohling and Rey (2001). The method was proposed to design an optimal disturbance rejection proportional integral derivative (PID) controller. A condition for disturbance rejection of control system is described which is further formulated as a constrained optimization problem. A constraint optimization problem to optimize integral of time and absolute error (ITAE) was tested by proportional integral derivative (PID) controller as applied to servo motor system. A double genetic algorithm was applied for solving constraint optimization problem. Simulation results demonstrate the performance and validity of the methods.

Scheduling of hydraulically coupled plants can be approximated by genetic algorithms. An effective approach was suggested by Chen and Chang (1996) to 24 hours ahead generation scheduling of hydraulically coupled plants. Experimental results show that the genetic algorithm approach obtains a more highly optimal solution than the conventional dynamic programming model. Hong Y. (2002) applied genetic algorithms on economic dispatch for congregation units considering multi-plant multi-buyer wheeling, which transmits microwaves to design load buses via wheeling. Varying the weights coeficient for penalty functions and determination of gene variables using genetic algorithms was discussed. The IEEE 30 and IEEE 188 bus system were used as test system to illustrate the applicability of the proposed method.

Genetic algorithms applied to scheduling and optimization of refinery operations was discussed by Oliveira, Almeida and Hamacher (2008). This paper presents a Genetic Algorithm-based method to optimize the production schedule of the fuel oil and asphalt section in a petroleum refinery. Two Genetic Algorithm models were developed to establish the sequence and size of all production shares. It shows the development of a methodology that applies GA to solve the scheduling problem of fuel oils and asphalt. In this study, the proposed GA aims at minimizing the demand that cannot be supplied, minimizing the production that cannot be allocated to the tanks, and minimizing the number of operational changes. Two GA models were proposed to solve the optimization of a lotsizing and sequencing problem in a multi-product plant with two-stage serial machines Direct and Indirect representation. The first uses a direct representation of the production schedule, dividing the scheduling horizon into discrete intervals of one hour which achieves some interesting results, which are further improved with the use of the NM operator. The second model uses an indirect representation which must be decoded into a production scheduling and achieved outstanding performance levels concerning demand fulfillment and production allocation; a satisfactory performance level (according to the refinery's real production scheduling) was observed when it comes to operational mode change minimization. They also showed that the implementation of the modified EM method allowed the GA to find better solutions due to weight updates during the evolutionary process avoiding meeting some specific objectives while neglecting others. The obtained results confirm that the proposed Genetic Algorithm models, associated with the multi-objective energy minimization method, are able to solve the scheduling problem, optimizing the refinery's operational objectives.

Within the classical resource-constrained project scheduling problem (RCPSP), the activities of a project have to be scheduled such that the makespan of the project is minimized. Considering Resource-constrained project scheduling problem (RCPSP) with makespan minimization as objective Hartmann (1998) propose a new genetic algorithm approach to solve this problem and compared it to two genetic algorithm concepts. While our approach makes use of a permutation based genetic encoding that contains problem-specific knowledge, the other two procedures employ a priority value based and a priority rule based representation, respectively. The representation is based on a precedence feasible permutation of the set of the activities. The genotypes are transformed into schedules using a serial scheduling scheme. Among several alternative genetic operators for the permutation encoding, a ranking selection strategy was chosen, a mutation probability of 0.05, and a two point crossover operator which preserves precedence feasibility. The initial population was determined with a randomized priority rule method. In order to evaluate the approach, two GA concepts which make use of a priority value and a priority rule representation, respectively was compared with. As further benchmarks, seven other heuristics known were considered. The outcome reveals that the procedure is the most promising genetic algorithm to solve the RCPSP since in-depth computational study revealed that their GA outperformed the other GAs as well as the other approaches. Finally, computational study show that genetic algorithm yields better results than several heuristic procedures presented in the literature.

The accuracy of the localized face center coordinates and orientation has a heavy influence on the recognition performance so, finding the exact location of face after localization algorithm is crucial. In view of this Kanan and Moradi (2005) introduced a new method using genetic algorithms (GA's) for face localization: A genetic Algorithm based Method for Face Localization and pose Estimation. Face images often have a background that can effect on the

face localization algorithm. In their algorithm, input image is first enhanced by means of histogram equalization. Then connected components are determined by applying a region growing algorithm (coarse segmentation), followed by computing the fit ellipse for face area and at least exact location of face is found by genetic algorithms method. Then the best-fit ellipse for face area is computed. We have used genetic algorithms to find the best location (includes the best orientation and the best position) of face in image. To check the utility of the proposed algorithm, experimental studies were carried out on the ORL database images.

Many automated analytical techniques such as Curie-point pyrolysis mass spectrometry (Py-MS) have been used to identify bacterial spores giving use to large amounts of analytical data. Hence the rapid identification of Bacillus spores and bacterial identification are paramount because of their implications in food poisoning, pathogenesis and their use as potential biowarfare agents. In view of this, Correa and Goodacre (2011) proposed A genetic algorithm-Bayesian network approach for the analysis of metabolomics and spectroscopic data: application to the rapid identification of Bacillus spores and classification of Bacillus species. They developed a novel genetic algorithm-Bayesian network algorithm that accurately identifies sand selects a small subset of key relevant mass spectra (biomarkers) to be further analysed which once identified, it becomes a subset of relevant biomarkers used to identify Bacillus spores successfully and to identify Bacillus species via a Bayesian network model specifically built for this reduced set of features. This final compact Bayesian network classification model is parsimonious, computationally fast to run and its graphical visualization allows easy interpretation of the probabilistic relationships among selected biomarkers. In addition, they compared the features selected by the Genetic Algorithm-Bayesian Network (GA-BN) approach with the features selected by partial least squares-discriminant analysis (PLS-DA). The classification accuracy results show that the set of features selected by the GA-BN is far superior to PLS-DA.

Several heuristic approaches for the flowshop scheduling problem have been developed. In recent years, meta-heuristic approaches, such as Simulated Annealing, Tabu Search, and Genetic Algorithms, have become very desirable in solving combinatorial optimization problems because of their computational performance. In a paper, which introduced the fundamental model and described a GA-based heuristic for solving the flowshop scheduling problems: A Model to Study Genetic Algorithm for the Flowshop Scheduling Problem proposed by Tyagi and Varshney (2012). Many scheduling problems are NP-hard problems. For such NP-hard combinatorial optimization problems, heuristics play a major role in searching for near-optimal solutions. In this GA-based heuristic, a different parameter set was generated for the Genetic operators which protect the best schedule which has the minimum make-span, at each generation and then transfer this schedule to the next population with no change. This operation enables us to choose the higher crossover and mutation probability pc = 1 (crossover probability) and pm = 0.05 (mutation probability). This increase the diversity of the population to get a better solution and also show the excellent performance of the LOX operator. Their heuristic was compared with the NEH (Nawaz, Enscore, Ham) Algorithm which is the most popular heuristic in the literature. The computational experience shows that the Genetic Algorithm approach provides competitive results for flowshop scheduling problems.

Alba and Troya (2009) proposed a paper on A Survey of Parallel Distributed Genetic Algorithms. In this work we have presented the fundamental syntax and semantics of sequential genetic algorithms (GAs). The paper deals with very popular extensions of these algorithms known as parallel distributed GAs, in which many sub-algorithms run in parallel with sparse migrations of strings. A structured and extensive overview on the more important and up-to-date PGA systems is discussed. In it, much of the existing software and criteria for their classification is used. In addition, we present in the paper useful technical information about PGAs relating operators, structured-population paradigms, and parameters guiding the parallel search. We have included a brief theoretical foundation of a distributed GA to make the paper relatively self-contained. In particular, we have offered a location of PGAs in the taxonomy of search techniques, a nomenclature revision, algorithmic descriptions of techniques, future trends, a classification of a large portion of the existing software, open questions relating generational versus steady-state evolution modes and heterogeneous versus homogeneous parallel algorithms, and many other minor details and major concepts relating parallel GAs in general. Our main interest has been in parallel distributed GAs, since the impact of the research in this kind of algorithms is a priori larger than for other kinds of parallel genetic algorithms. We are especially concerned with offering useful and rigorous material that could help new and expert practitioners. Although our overview is obviously not complete, it represents a good starting point to conduct future research in this domain or to make new applications by using parallel distributed GAs.

Recovery of used products has become increasingly important recently due to economic reasons and growing environmental or legislative concern. Product recovery, which comprises reuse, remanufacturing and materials

recycling, requires an efficient reverse logistic network. In view of this a paper: An Optimization Model for Reverse Logistics Network under Stochastic Environment Using Genetic Algorithm was proposed by Hosseinzadeh and Roghanian (2012). One of the main characteristics of reverse logistics network problem is uncertainty that further amplifies the complexity of the problem. The degree of uncertainty in terms of the capacities, demands and quantity of products exists in reverse logistics parameters. With consideration of the factors noted above, this paper proposes a probabilistic mixed integer linear programming model for the design of a reverse logistics network. The demand of manufacturing centers and recycling centers are regarded as random variables. This probabilistic model is first converted into an equivalent deterministic model. In this paper multi-product, multi-stage reverse logistics network problem which consider the minimizing of total shipping cost was proposed and fixed opening costs of the disassembly centers and the processing centers in reverse logistics. Then, we propose priority based genetic algorithm to find reverse logistics network to satisfy the demand imposed by manufacturing centers and recycling centers with minimum total cost under uncertainty condition. Finally the proposed model was applied to the hypothetical problem. And then, computing results show that it can obtain solutions for reverse logistics network design problem with some stochastic parameters. In fact, this type of network design problem belongs to the class of NP-hard problems.

After suitable modifications, genetic algorithms can be a useful tool in the problem of wavelength selection in the case of a multivariate calibration performed by PLS. Unlike what happens with the majority of feature selection methods applied to spectral data, the variables selected by the algorithm often correspond to well-defined and characteristic spectral regions instead of being single variables scattered throughout the spectrum. This leads to a model having a better predictive ability than the full-spectrum model; Application of genetic algorithm–PLS for feature selection in spectral data sets by Leardi (2000), furthermore, the analysis of the selected regions can be a valuable help in understanding which the relevant parts of the spectra are. Nowadays, spectral data are perhaps the most common type of data to which chemometric techniques are applied. Owing to the development of new instrumentation, data sets in which each object is described by several hundreds of variables can be easily obtained. Methods such as partial least squares (PLS) or principal component regression (PCR), being based on latent variables, allow one to take into account the whole spectrum without having to perform a previous feature selection. The present study shows that the GA can be a good method for feature selection in spectral data sets. The results obtained on five different data sets demonstrate that the predictive ability of the models obtained with the wavelengths selected by the algorithm is very often much better, and anyway never worse, than the predictive ability of the full spectrum. Another relevant point is that the selected variables almost always clearly identify spectroscopically relevant regions.

In this paper, the authors discussed the problem of selecting suppliers for an organisation, where a number of suppliers have made price offers for supply of items, but have limited capacity. Selecting the cheapest combination of suppliers is a straight forward matter, but purchasers often have a dual goal of lowering the number of suppliers they deal with. This second goal makes this issue a bicriteria problem – minimisation of cost and minimisation of the number of suppliers. Hence a Genetic Algorithm for a Bicriteria Supplier Selection Problem by Weintraub and Basnet (2005) presented a mixed integer programming (MIP) model for this scenario in which Quality and delivery performance are modeled as constraints. Smaller instances of this model may be solved using a MIP solver, but large instances will require a heuristic. To this a multi-population genetic algorithm for generating pareto-optimal solutions of the problem was presented. The performance of this algorithm is compared against MIP solutions and Monte Carlo solutions. The contributions of this paper are two-fold: the inclusion of number of selected suppliers as a criterion in supplier selection in a MIP model and the development and testing of a multi-population genetic algorithm for the generation of Pareto-optimal solutions. Three algorithms were presented to generate the Pareto-optimal solutions. The exact (MIP) algorithm solved problems with up to 50 suppliers within reasonable time, but not higher sized problems. The performance of multi-population genetic algorithm was close to the MIP algorithm results. The genetic algorithm performed better than Monte Carlo optimization, particularly in regard to the number of solutions generated. In regard to the purchase cost of the solutions, the genetic algorithm performed better for all but the largest problems for which the performances of the two algorithms were almost even.

Frustration or competition between interactions is a common feature of many systems in condensed matter. One of the most common frustrated systems is ordinary water ice, in which hydrogen ions follow the so-called ‘‘ice rules’’. Using a genetic algorithm to study properties of minimum energy states and geometrical frustration in artificial ‘‘spin ice’’ systems by Leon and Pozo (2007) reports the results of a study on the base state of artificially frustrated ‘‘spin ice’’ systems. We have studied the states of minimum energy reported by experimental studies on nanoscale ferromagnetic islands and the protocols employed to reach those states. The main technique employed in this study is a genetic algorithm that has been contrasted with two Montecarlo methods. Nanoscale islands are modeled through

dipolar moments placed on a plane, rectangular array. Studies include the correlation between nanoscale islands, statistics on vertex types formed in the array for the minimum energy state and intermediate states. The system studied shows a minimum energy state with degeneration two, and the configuration of moments. The minimum energy state is reached when considering only the effective, long-range dipolar interaction, using the three techniques described above, namely, the genetic algorithm, the Montecarlo method for ''spin ice'', and the general Montecarlo method. When we simulate the experimental protocol using a wide range of values for we cannot make the system achieve the minimum energy that can be achieved when only the effective dipolar interaction is considered. This methodology allowed us to assess the efficiency of the numerical techniques used to study this type of systems. It was concluded that the 3-color Montecarlo method is more efficient when searching is done only in the space of ''spin ice'' configurations (type I and II vertices), but the genetic algorithm is far more efficient when searching starts at general configurations, that is, the four types of vertices.

Computer software marketed by companies such as the Heat Transfer Research Institute (HTRI), HTFS, and B-JAC International are used extensively in the thermal design and rating of HEs. A primary objective in HE design is the estimation of the minimum heat transfer area required for a given duty, as it governs the overall cost of the HE. However, because the possible design configurations of heat transfer equipment are numerous, an exhaustive search procedure for the optimal design is computationally intensive. Tayal, Fu, and Diwekar (2009) presented a paper: Optimal Design of Heat Exchangers: A Genetic Algorithm Framework for solving the combinatorial problem involved in the optimal design of HEs. The problem is posed as a large-scale, combinatorial, discrete optimization problem involving a blackbox model. This paper demonstrates the first successful application of genetic algorithms to optimal HE design with a black-box model. It also incorporates methodologies to avoid process infeasibilities and design vibrations. In addition this paper compares the performance of SA and GAs in solving this problem and presents strategies to improve the performance of the optimization framework. From this study, it was concluded that (1) The optimal design obtained using combinatorial algorithms such as GAs and SA significantly improves base-case designs; (2) these algorithms also result in considerable computational savings compared to an exhaustive search; and (3) GAs have an advantage over other methods in obtaining multiple solutions of the same quality, thus providing more flexibility to the designer. A vehicle designer can do little to improve road surface roughness, so designing a good suspension system with good vibration performance under different road conditions becomes a prevailing philosophy in the automobile industry. The ride quality of a vehicle is significantly influenced by its suspension system, the road surface roughness, and the speed of vehicle. A paper, Research on Suspension System Based on Genetic Algorithm and Neural Network Control by Tang and Guo (2009) showed a five degree-of-freedom half body of vehicle suspension system model which can describe both the vertical movement and the pitching movement of the body, what's more, it can demonstrate the effect of the passenger, which makes it to be a relatively ideal model for suspension dynamic. In this work, the specified half body vehicle model with passenger involving five degree of freedom is presented to achieve the excellent ride comfort and drive stability of the system description. Genetic algorithm and neural network control are used to control the suspension system. The desired objective is proposed as the minimization of a multi-objective function formed by the combination of not only sprung mass acceleration, pitching acceleration, suspension travel and dynamic load, but also the passenger acceleration.

The model is assumed to have five masses attached with linear springs and nonlinear dampers. It is also assumed that the system does not vibrate in lateral direction, only oscillates in vertical and longitudinal directions. Furthermore, the tires are assumed not losing the contact with the road surface. Approaches are presented for suspension design which uses genetic algorithm and neural network control algorithm. It is obvious from the response plots that vehicle body vertical acceleration, passenger response and pitching angular response decreased compared with the passive suspension system, which naturally brings ride comfort. And the suspension travel and dynamic load reduced compared with the passive suspension system, which indicates that the proposed controller proves to be effective in the stability improvement of the suspension system. A mechanical dynamic model of the five degree of freedom half body of vehicle suspension system is also simulated and analyzed by using software Adams. Simulation results demonstrate that the proposed active suspension system proves to be effective in the ride comfort and drive stability enhancement of the suspension system. People often need to make decisions based on different kinds of information, but the explosion of information is hard to handle and reading everything may be very time consuming. Various kinds of summaries (e.g.: titles, abstracts, keywords, outlines, previews, reviews, biographies and bulletins) help reduce this problem. Summarizing Jewish Law Articles Using Genetic Algorithms is a paper by HaCohen-Kerner, Malin, Chasson (2005). This paper describes the first summarization model for texts in Hebrew. The summarization is done by extraction of the most relevant sentences. The introduction of summaries offers the readers the option whether or not to read the entire text. In addition, summaries can serve as brief substitutes of full documents. Automatic text

summaries can be produced with two main approaches: Natural Language Processing (NLP) and information extraction (IE). Three machine learning methods have been tried: perceptron learning, Naive Bayesian learning, and genetic algorithm. The best results have been achieved by the genetic algorithm. To the best of our knowledge, this model is also the first to use successfully genetic algorithm for sentence extraction. This model belongs to the sentence extraction approach. That is, it selects the most important sentences from the article and proposes them as a summary. In contrast to many summarization models that were designed and checked mostly for English articles taken from magazines and newspapers, the model deals with articles referring to Jewish law written in Hebrew.

Using Genetic Algorithm for Distributed Generation Allocation to Reduce Losses and Improve Voltage Profile by Sedighizadeh, and Rezazadeh (2008) is a paper that presents a method for the optimal allocation of Distributed generation in distribution systems. In this paper, the aim would be optimal distributed generation allocation for voltage profile improvement and loss reduction in distribution network. Genetic Algorithm (GA) was used as the solving tool, which referring two determined aim; the problem is defined and objective function is introduced. Considering to fitness values sensitivity in genetic algorithm process, there is needed to apply load flow for decision-making. Load flow algorithm is combined appropriately with GA, till access to acceptable results of this operation. It was implemented on part of Tehran electricity distributing grid. The resulting operation of this method on some testing system is illuminated improvement of voltage profile and loss reduction indexes. The impact of DG in system operating characteristics, such as electric losses, voltage profile, stability and reliability needs to be appropriately evaluated. The installation of DG units at non-optimal places can result in an increase in system losses, implying in an increase in costs and, therefore, having an effect opposite to the desired. As a contribution to the methodology for DG economical analysis, in this paper it is presented an algorithm for the allocation of generators in distribution networks, in order to voltage profile improvement and loss reduction in distribution network. The Genetic Algorithm is used as the optimization technique. In this paper the results of application of GA algorithm to the optimal allocation of DGs in distribution network is presented. The Khoda Bande Loo distribution test feeder in Tehran has been solved with the proposed algorithm and, the simple genetic algorithm.

Brain Computer Interfaces (BCIs) measure brain signals of brain activity intentionally and unintentionally induced by the user, and thus provide a promising communication channel that does not depend on the brain's normal output pathway consisting of peripheral nerves and muscles. Ghanbari et al, (2012) paper on Brain Computer Interface with Genetic Algorithm Genetic Algorithm to select the effective number of electrodes and Redundancy Reduction. BCI operation depends on the interaction of two adaptive controllers, the user, who must maintain close correlation between his or her intent and these phenomena, and the BCI, which must translate the phenomena into device commands that accomplish the user's intent. They might also control a neuroprosthesis that provides hand grasp to those with mid-level cervical spinal cord injuries. With adequate recognition and effective engagement of these issues, BCI systems could provide an important new communication and control option for those with disabilities that impair normal communication and control channels. They might also provide to those without disabilities a supplementary control channel or a control channel useful in special circumstances. Their paper is on one hand to reduce the redundancy and on the other hand to increase the BCI speed and making use of it in real time form. Hence, the linear filtering method is applied to Artifact removal which is a relatively simple method with fairly low complexity computations

Since the EEG is non-stationary in general, it is most appropriate to use time–frequency domain methods like wavelet transform (WT) as a mean for feature extraction. The simulation results confirm this fact that the Genetic algorithm is applied in order to choose the best features from the feature space as well as the best channels from the many channels that have been used. In this case, the increased number of electrodes causes a non-linear increase in computational complexity (decrease transfer rate). To overcome these problems in this article evolutionary intelligent method for selecting the effective number of electrodes and redundancy reduction was used. One of the main privileges of the mixed methods used in this paper is that, the redundant data are removed by the selection power of the genetic algorithm. This fact reduces the data dimension and reduced time response of system significantly.

IV. EVOLUTIONARY ALGORITHMS

Evolutionary algorithms (EAs) are based on the process of Darwin's theory of evolution. Evolutionary algorithms (EAs) use biologically derived techniques such as inheritance, mutation, natural selection and recombination. In 1882, Charles Darwin defined natural selection or survival of the fittest as the preservation of favorable individual differences and variations and the destruction of those that are injurious. In nature, features that make an individual

more suited to compete are preserved when reproducing and the weakening features are eliminated. This process is called *evolution*, where features are controlled by units called genes which form sets called chromosomes. Over generations, the fittest individuals survive and their fittest genes are transmitted to their descendants during a sexual recombination process called crossover. To their basic components one can subsume population (set of solutions), chromosomes (individuals), fitness of the chromosomes, process of reproduction (selection of parents and children generation), replacement (death of the individuals) and generation completion. Typically evolution starts from a population of completely random individuals (solutions), represented by chromosomes, and happens in generations. Traditionally, solutions occur as binary strings of zeros (0's) and ones (1's), but different encodings are also possible. Each individual is characterized by its fitness. Each generation is defined by population size, as well as the birth and death processes. In every generation, multiple individuals are stochastically selected from the current population, and next — modified through mutation or recombination to form a new population, which becomes current in the following iteration of the algorithm. Solutions which form the offspring are selected according to their fitness — the more suitable they are the more chances they have to reproduce.

This is motivated by a hope, that the new population will be better than the old one. In such a manner, an approximation algorithm evolves towards better solutions. The procedure stops when the desired stopping criterion, like number of populations or improvement of the best solution, is reached. As a result of this simulated evolution one obtains highly evolved solution to the original problem, that is, the best chromosome picked out of the final population. The main purpose of evolutionary algorithms is to imitate this evolutionary process in computers. The general procedure of EAs is as follows:

1. Initialize the population by randomly selecting or generating a set of potential solutions (also called chromosomes or individuals). Such individuals evolve during several generations (i.e., they constitute off-springs) through steps 2 to 4 below;
2. Evaluate each individual in the population by calculating its fitness;
3. Reproduce selected individuals to form a new population (best individuals are kept, while the others are discarded);
4. Perform evolutionary operations, such as crossover and mutation, on the population according to pre-specified probabilities. Crossover exchanges the genetic material of a pair of individuals to create the population of the next generation, while mutation randomly changes a gene of a chromosome;
5. Loop to step 2 until some condition is met (e.g., reaching a specified number of generations).

An idea of evolutionary algorithms for combinatorial optimization problems, inspired by Darwin's theory of evolution, was introduced by John Holland in 1975. Categorically, three of the nearly contemporaneous sources of the evolutionary algorithms (EA) have been kept alive over three decades and experienced an amazing increase of interest during the last fifteen years. Two of them are lying in the United States of America, the source of evolutionary programming (EP) in San Diego (Fogel, 1962; Fogel et al., 1966), the source of genetic algorithms (GA) in Ann Arbor (Holland, [1962, 1975]). Evolution strategies (ES), the third main variant of EA, were founded by students at the Technical University of Berlin (TUB) (Rechenberg, [1965, 1971]; Schwefel, [1965, 1975]).

Evolutionary algorithms have been generally applied in the recent decade to different disciplines, such as DSS research, scheduling, engineering, chemistry, health, management and finance (Chambers 2001; Carlsson and Turban 2002; Osyczka 2002; Marczyk, 2004; Mora et al. 2006). They are also increasingly applied to the economics field. Excellent surveys and discussion of relevant issues about the applications in economics can be found in Dawid (1999); Arifovic (2000); Tsang, Lsasi and Quintana (2009); Safarzynska and Bergh (2009). Below is a summary of the economics areas of applications and a number of examples. The string representation in GAs has been successfully used to code consumer preferences (Aversi et al. 1997); production functions (Birchenhall, Kastrinos and Metcalfe 1997); pricing strategies (Curzon Price 1997) and production rules in cobweb models (Dawid and Kopel 1998; Frenke 1998). In addition, genetic programming has been used to develop an optimal price-setting rule (Dosi et al. 1999) and an optimal trading rule (Allen and Karjalainen 1999).

Furthermore, Hidalgo et al. (2008) used an EA to find correct parameters for technical indicators applied to interpret stock market trending and investing decisions. Lately, Jina, Tsang and Li (2009) applied a constraint handling EA to search equilibriums for bargaining problems. Safarzynska and Bergh (2009) showed that applications of evolution strategies in economics are rare; therefore, *this paper contributes* to utilizing evolution strategies in the field of economics.

V. EVOLUTIONARY STRATEGIES

Evolutionary strategies (ES) were developed in Germany by Ingo Rechenberg and H.P. Schwefel in the 1960's. It imitates mutation, selection and recombination by using normally distributed mutations, a deterministic mechanism for selection (which chooses the best set of offspring individuals for the next generation) and a broad repertoire of recombination operators. The primary operator of the ES is Mutation. There are two variants of selection commonly used in ES. In the elitist variant, the 'ρ' parent individual for new generation are selected from the set of both 'ρ' parent and 'ϑ' offspring at old generation. This is called Plus Strategy (ρ+ϑ). Additionally, each individual contains a number of strategy parameters, these being the variances and covariances of the object variables (the covariances are optional, but when used are normally defined using the rotation angles of the covariance matrix). The strategy parameters are used to control the behavior of the mutation operator and are not required when decoding an individual. The recombination operator produces one child and requires two parents for each object variable and strategy parameter in the child. Historically, the same parents are used to generate all object variables in the child, then the parents are re-selected for each strategy parameter. The parents are selected randomly from the current population (i.e., there is no selection pressure at this point). Mutation, which is the main operator in the ES acts upon strategy parameters as well as object variables. The mutation operator first perturbs the strategy parameters. The object variables are then mutated using the resulting probability distribution defined by the modified strategy parameters. This special mutation operator allows the ES to evolve good strategy parameters for the problem and has been termed self-adaptation.

Selection in EAs is deterministic: the best ρ individuals are taken from the ϑ new offspring ..m; l/ ρ ϑ selection) or from the union of ρ parents and ϑ offspring ..m C l/ ρ ϑ selection). The preferred method is .m; l/ ρ ϑ selection, since .m C l/ ϑ selection can disrupt the self-adaptation mechanism. If parent individual do not take part in the selection, then 'ρ' individuals of the next generation are selected from 'ϑ' offspring, which is called Comma Strategy and denoted by (ρ, ϑ). The notation (ρ,+ ϑ) is used to submuse both selection schemes. Historically ESs were designed for parameter optimization problems. The encoding used in an individual is therefore a list of real numbers: these are called the object variables of the problem. Like the GA, EAs run until some termination criteria are satisfied. However, in ESs remains an attractive alternative to GAs, especially in the field of parameter optimization, where in model systems they appear to outperform GAs.
An ES-algorithm run can be described briefly as follows

ES as developed by Rechenberg and Schwefel.

1. A current population of m individuals is randomly initialized.
2. Fitness scores are assigned to each of the m individuals.
3. *l* new offspring are generated by recombination from the current population.
4. The *l* new offspring are mutated.
5. Fitness scores are assigned to the *l* new offspring.
6. A new population of m individuals is selected, using either .m; l/ ρ or .m C l/ ρ selection.
7. The new population becomes the current population.
8. If the termination conditions are satisfied exit, otherwise go to step 3.

VI. EVOLUTIONARY PROGRAMMING

Evolutionary programming is an optimization strategy that is based on the stochastic modification of a set of trial solutions. Evolutionary Programming (EP) was developed by L. J. Fogel et al. (1966) in the USA . This technique is especially well suited for combinatorial problems and situations where the fitness landscape has many local minima. EP (which is a Stochastic Optimization Strategy) is a useful method of optimization when other techniques such as gradient descent or direct analytical discovery are not possible.

Illustrates of the form of an EP scheme

1. A current population of *m* individuals is randomly initialized.
2. Fitness scores are assigned to each of the *m* individuals.
3. The mutation operator is applied to each of the *m* individuals in the current population to produce *m* offspring.
4. Fitness scores are assigned to the *m* offspring.
5. A new population of size *m* is created from the *m* parents and the *m* offspring using tournament selection.
6. If the termination conditions are satisfied exit, otherwise go to step 3.

An important point is that the strings do not have to be of a fixed length, they could mutate into longer or shorter forms. If you look at the algorithm, you can see why this is - there is no crossover operation. All in all, this means that system representation in ES or EP can be direct and simple.
However, not using crossover also has one major disadvantage – that of speed, mutation is a slow way to search for good solutions.

VII. REVIEW OF GENETIC ALGORITHM

The term *genetic algorithms,* almost universally abbreviated nowadays to GAs, are search methods based on principles of natural selection and genetics (Fraser, 1957; Bremermann, 1958; Holland, 1975). Genetic Algorithms (GAs) are powerful and widely applicable stochastic search technique and optimization methods based on the principles of genetics, natural selection and natural evaluation. A GA allows a population composed of many individuals to evolve under specified selection rules to a state that maximizes the "fitness" (i.e., minimizes the cost function). The method was developed by John Holland (1975) over the course of the 1960s and 1970s and finally popularized by one of his students, David Goldberg (at the University of Michigan), who was able to solve a difficult problem involving the control of gas-pipeline transmission for his dissertation (Goldberg, 1989). John Holland, whose book *Adaptation in Natural and Aritificial Systems* of 1975 and Goldberg (with his successful applications and excellent book (1989)) was instrumental in creating what is now a flourishing field of research and application that goes much wider than the original GA.

GA is a method for moving from one population of "chromosomes" (e.g., strings of ones and zeros, or "bits") to a new population by using a kind of "natural selection" together with the genetics−inspired operators of crossover, mutation, and inversion. Each chromosome consists of "genes" (e.g., bits), each gene being an instance of a particular "allele" (e.g., 0 or 1). For example, in a problem such as the traveling salesman problem, a chromosome represents a route, and a gene may represent a city. In contrast to traditional optimization techniques, GAs work with coding of parameters, rather than the parameters themselves. The selection operator chooses those chromosomes in the population that will be allowed to reproduce, and on average the fitter chromosomes produce more offspring than the less fit ones. Crossover exchanges subparts of two chromosomes, roughly mimicking biological recombination between two single−chromosome ("haploid") organisms; mutation randomly changes the allele values of some locations in the chromosome; and inversion reverses the order of a contiguous section of the chromosome, thus rearranging the order in which genes are arrayed. Although there are a variety of operators such as crossover, mutation and inversion as defined above, the two main operators used is:-

- Crossover, which creates new individuals by combining parts from two individuals like the bit-string crossover in which two strings are used as parents and new individuals are formed by swapping a sub-sequence between the two strings
- Mutation, which creates new individuals by making changes in a single individual like the bit-flipping mutation, in which a single bit in the string is flipped to form a new offspring.

Genetic algorithm maintains a population of individuals, say $P(t)$, for generation t. Each individual represents a potential solution to the problem at hand. Each individual is evaluated to give some measure of its fitness. Some individuals undergo stochastic transformations by means of genetic operations to form new individuals. The new individuals, called offspring $C(t)$, are then evaluated. A new population is formed by selecting the more fit individuals from the parent population and offspring population. After several generations, genetic algorithm converges to the best individual, which hopefully represents an optimal or suboptimal solution to the problem.

The general structure of the Genetic algorithms is as follow:

```
Begin
    {
    t = 0;
    Initialize P(t);
    Evaluate P(t);
    While (not termination condition) to
    Begin
        {
        Apply crossover and mutation to P(t) to yield C(t);
        Evaluate C(t);
        Select P(t+1) from P(t) and C(t);
        t = t + 1;
        }
     End
     }
End
```

Another important concept of GAs is the notion of population. Unlike traditional search methods, genetic algorithms rely on a population of candidate solutions. The population size, which is usually a user-specified parameter, is one of the important factors affecting the scalability and performance of genetic algorithms. For example, small population sizes might lead to premature convergence and yield substandard solutions. On the other hand, large population sizes lead to unnecessary expenditure of valuable computational time. Once the problem is encoded in a chromosomal manner and a fitness measure for discriminating good solutions from bad ones has been chosen, we can start to *evolve* solutions to the search problem. Note that the offspring population created by selection, crossover (recombination), and mutation replaces the original parental population.

***A. Genetic Programming* (GP)**

The Genetic Programming (GP) is an EC technique which was developed in 1992 by Koza John. Genetic programming is a collection of methods for the automatic generation of computer programs that solve carefully specified problems, via the core, but highly abstracted principles of natural selection. In a sentence, it is the compounded breeding of (initially random) computer programs, where only the relatively more successful individuals pass on genetic material (programs and program fragments) to the next generation. Genetic Programming represents a special type of genetic algorithm in which the structures that undergo adaptation are not data structures, but hierarchical computer programs of different shapes and sizes. In GP, the individuals do not represent the solution of a given problem. Now, they represent algorithms/procedures to solve such problem. Thus, the GP find out the best procedure to solve a problem. In GP, the individuals are defined by a function set (subprograms, mathematical functions, etc.) and a terminal set (constants, variables, etc.). The GP process starts by creating an initial population of randomly-generated programs and continues by producing new generations of programs based on the Darwinian principle of *"the survival of the fittest"*. The automatically-generated computer programs are expressed as function composition, and the main breeding operations are *reproduction* and *cross-over*. By reproduction we mean that a program from generation *i* is copied unchanged within generation *i+1*, while the cross-over takes two parent-programs from generation *i*, breaks each of them in two components, and adds to generation *i+1* two children programs that are created by combining components coming from different parents. In order to create a GP-based application, the user has to specify a *set of building blocks* based on which the population of programs is constructed, and an *evaluation function* that is used to measure the fitness of each individual program.

There are two types of primitive elements that are used to build a program: *terminals* and *functions*. Both terminals and functions can be seen as LISP-functions, the only difference between them consisting of the number of arguments that they are taking: terminals are not allowed to take arguments, while functions take at least one argument. The *individuals* generated by the GP system represent computer *programs* that are built by *function composition* over the set of terminals and functions. Consequently, GP imposes the *closure property (Koza 1994)*: any value returned by a function or a terminal must represent a valid input for any argument of any function in the function set. As we have already mentioned, the GP problem specification must include a domain-specific *fitness evaluation function* that is used by the GP system to estimate the "fitness" of each individual of a generation. More specifically, the fitness function takes as input a GP-generated program *P*, and its output represents a measure of how appropriate *P* is to solve the problem at hand Both cross-over and reproduction are performed on randomly chosen individuals, but they are biased for highly fit programs. Such an approach has two major advantages: on one hand, the "highly fit" bias leads to the potentially fast discovery of a solution, while on the other hand, GP is capable of avoiding local minima by also using in the breeding process individuals that are less fit than the "best" offsprings of their respective generations.

Normally, the GP uses two operators: *crossover* and *mutation*. The crossover operator exchanges two sub-trees from two tree randomly selected. In this way, two new trees are created. The mutation operator randomly selects a tree and creates a new tree by taking a sub-tree and replacing it by other one, which is randomly generated. These operators preserve the syntactic constraints of the models. This model is easy to implement in GP, through the utilization of the ADF (Automatic Definition Function) technique. This extension of GP permits to define functions to evolve in parallel with the main procedure. These functions can be called by other functions, or by the main procedure, during the evolution.

Rather than blindly searching the fitness space, or searching from randomly initialized states, genetic programming attempts to extract the useful parts of the more successful programs and use them to create even better solutions. How does the system know which parts of a program are useful, and how to combine them to form more fit solutions? By randomly selecting parts of the more successful programs, and randomly placing those parts inside other successful programs. Genetic programming relies upon the fitness function to tell if the new child received something useful in the process. Often the child is worse for its random modification, but often enough the right code is inserted in the right place, and fitness improves. Given the programming language such as Lisp and a fitness metric, the steps executed by a genetic programming algorithm are straightforward:

- Initial Population: With an algorithm that allows random generation of code, an initial population of potential solutions can be generated. All will be quite inept at solving the problem, as they are randomly generated programs. Some will be slightly better than others, however, giving evolution something to work with.
- **Fitness Ranking:** Via the fitness metric, the individual programs are ranked in terms of ability to solve the problem.
- **Selection:** The closer (better) solutions are selected to reproduce because they probably contain useful components for building even better programs.
- **Mating:** At random, chunks of those selected programs are excised, and placed inside other programs to form new candidate solutions. These "children" share code from both parents, and (depending on what code was selected) may exhibit hybrid behavior that shares characteristics of both.
- **Mutation:** To simulate genetic drift/stray mutation, many genetic programming systems also select some of the more fit programs and directly duplicate them, but with a few of their statements randomly mutated.
- **Repetition until Success:** From here, the process starts over again at Fitness Ranking, until a program is found that successfully solves the problem.

Not every child will be more fit than its parent(s), and indeed, a very large percentage will not be. The expectation, however, is that some children will have changes that turn out to be beneficial, and those children will become the basis of future generations. Note that the random makeup of the initial population has a large effect on the likelihood that GP will find a successful program. If a single run does not succeed after a reasonable period of time, often it will succeed if it is restarted with a new random population. These steps constitute the core of most genetic programming systems, though all systems tweak, or completely change, many aspects of these steps to suit the theoretical interests pursued by their designers. The field is still young and there is no standard of what a genetic programming system must include, or how it must proceed from step to step.

VIII. CONCLUSION

Genetic Algorithms can be applied to solve combinatorial optimization problems (COPs) such as VRP. Optimal solutions among the search space can be found by Genetic algorithm with the use of the operators like crossover and mutation. They are not instantaneous, but can perform an excellent search. In this work, Genetic algorithm is tested to find the optimal route for the VRP which shows the superiority of Genetic Algorithm over the company's normal route.

It is also proven that if Amponsah Effah Pharmaceutical Limited in retrospective uses this work, they would be able to reduce their operational distance by 3776m (3.7760km) thereby reducing their cost of fuelling their delivery vans which intend reduces the cost of operations of the company. We are of the view that this work if adopted would increase the profit margin of the company y and as well help the company to improve remuneration of all staff members of the company.

References

A. Leon and J. Pozo, Using a genetic algorithm to study properties of minimum energy states and geometrical frustration in artificial ''spin ice'' systems, Journal of Magnetism and Magnetic Materials 320, pp. 210–216, (2008).

A. A. Ghanbari, A. Broumandnia, H. Navidi, A. Ahmadi, Brain Computer Interface with Genetic Algorithm, International Journal of Information and Communication Technology Research Vol. 2 No. 1, (2012).

A. Ghosh and S. Dehuri, Evolutionary algorithms for Multi-Criterion Optimization: A Survey, International Journal of Computing and Information Sciences, vol.2, pp 35-62, (2004)

Arifovic, J., Evolutionary algorithms in macroeconomic models. Macroeconomic Dynamics 4 (5): pp 373-414, (2000).

Birchenhall, C., N. Kastrinos, and J. Metcalfe, Genetic algorithms in evolutionary modeling, Journal of Evolutionary Economics 7 (4): pp 375-393, (1997).

Bremermann, H. J., The evolution of intelligence. The nervous system as a model of its environment, Technical Report No. 1, Department of Mathematics, University of Washington, Seattle, WA, (1958).

Carlsson, C., and E. Turban, DSS: Directions for the next decade. Decision Support Systems 40 (1): pp1-8, (2002).

Cassaigne, N., Aversi, R., G. Dosi, G. Fagiolo, M. Meacci, and C. Olivetti, Demand dynamics with socially evolving preferences. IIIASA Working Paper, Luxemburg, Austria, (1997).

C. Darwin, The Origin of Species, Dent Gordon, London, (1973).

C. Basnet, A. Weintraub, A Genetic Algorithm for a Bicriteria Supplier Selection Problem,(2001)

Chia-Feng Juang. A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithm, IEEE Transactions on Fuzzy Systems, 10(2), pp. 155-170, (2002)

Chuan-Yin T. and Li-Xin G. Research on Suspension System Based on Genetic Algorithm and Neural Network Control, The Open Mechanical Engineering Journal, pp. 72-79, (2009).

Dosi, G., L. Marengo, A. Bassanini, and M. Valente., Norms as emergent properties of adaptive learning. Journal of Evolutionary Economics 9 (1): pp 5-26, (1999).

D. B. Fogel, 'Evolutionary Computation: Toward a New Philosophy of Machine Intelligence', IEEE Press, Piscataway, NJ, (1995).

D. B. Fogel, 'Evolving Artificial Intelligence', Ph.D. Thesis, University of California, San Diego, CA, (1992).

D. E. Goldberg, Sizing populations for serial and parallel genetic algorithms. In Schaffer, J. D. (editor) Proceedings of the Third International Conference on Genetic Algorithms. San Mateo, CA: Morgan Kaufmann Publishers Inc. (1989).

D. E. Goldberg, 'Genetic Algorithms in Search Optimization and Machine Learning', Addison-Wesley, Reading, MA, (1989).

D. Goldberg and R. Lingle, Alleles, loci and traveling salesman problem, In J.J. Grefenstette, editor, Proceedings of International Conference on GAs, Lawrence Erlbaum, (1985).

D. E. Goldberg, B. Korb, K. Deb, Messy Genetic Algorithms: Motivation, Analysis, and First Results, Complex Systems 3, pp. 493-530, (1989).

D. E. Goldberg, Genetic algorithms in search, optimization, and machine learning, Addison Wesley Longman, Inc., ISBN 0-201-15767-5, (1989).

Dawid, H. Adaptive learning by genetic algorithms: Analytical results and application to economic models. Berlin: Springer, (1999).

Davis, L., Applying algorithms to epistatic domains, in: *Proc. Int. Joint Conf. on Artifical Intelligence*, pp. 162–164, (1985).

Davis, Lawrence (editor), *Genetic Algorithms and Simulated Annealing*. London: Pittman, (1987).

Davis, Lawrence, *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold, (1991).

De Jong K., An Analysis of the Behaviour of a Class of Genetic Adaptive Systems, PhD Dissertation, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, (1975).

E. Alba and J. M. Troya, A Survey of Parallel Distributed Genetic Algorithms, Journal of Information and Operations Management
ISSN: 0976–7754 & E-ISSN: 0976–7762 , Vol. 3, pp 38-42, (2012).

E. Correa, R. Goodacre, A genetic algorithm-Bayesian network approachfor the analysis of metabolomics and spectroscopic data: application to the rapid identification of Bacillus spores and classification of Bacillus species, Correa and Goodacre BMC Bioinformatics, (2011)

Forrest, S., Genetic algorithms: Principles of natural selection applied to computation, *Science* **261**:872–878,(1993).

F. Oliveira, M.R. Almeida and S. Hamacher, Genetic algorithms applied to scheduling and optimization of refinery operations, **(2001).**

Frenke, R., Co-evolution and stable adjustment in the cobweb model, Journal of Evolutionary Economics 8 (4): pp 383-406, (1998).

Fraser, A. S., Simulation of genetic systems by automatic digital computers II: Effects of linkage on rates under selection, Austral. J. Biol. Sci. 10: pp 492–499, (1957).

Forel, D. B., and Atmar, W., eds., Proceedings of the second on Evolutionary Programming. Evolutionary Programming Society, (1993)

Fogel L. J., Autonomous automata. Industrial Research 4: pp 14–19, (1962).

Fogel L. J., Owens A. J. and Walsh M. J., Artificial Intelligence through Simulated Evolution, Wiley, New York, (1966).

G. Kannan, P. Sasikumar and K. Devika, A genetic algorithm approach for solving a closed loop supply chain model: A case battery recycling Journal of Applied Mathematics, vol.34, pp. 655-670, (2010)

Goldberg, D. E., Deb, K., and Korb, B. Messy genetic algorithms revisited: Studies in mixed size and scale. Complex Systems 4. pp. 410-470, (1990)

Goldberg, D. E. Genetic Algorithms and the minimal deceptive problem. In L. D. Davis, ed., Genetic Algorithms and Simulated Annealing. Morgan Kaufmann, (1987)

Goldberg, D. E. A note on Boltzman tournament selection for genetic algorithms and population-oriented simulated annealing. Complex Systems 4, pp. 445-460.

Goldberg, D. E. Genetic Algorithms and Walsh Functions: Part I, A gentle introduction. Complex Systems 3: pp. 130-155 (1998).

Goldberg, D. E., Genetic algorithms and Walsh functions: Part II, deception and its analysis, *Complex Syst.* **3**:153–171, (1989a).

Goldberg, D. E., 1989b, *Genetic Algorithms in Search Optimization and Machine Learning*, Addison-Wesley, Reading, MA, (1989b).

Goldberg, D. E., Sastry, K. and Latoza, T., On the supply of building blocks, in: *Proc. of the Genetic and Evolutionary Computation Conf.*, pp. 336–342, (2001).

Goldberg, D. E. and Lingle, R., Alleles, loci, and the TSP, in: *Proc. 1st Int. Conf. on Genetic Algorithms*, pp. 154–159, (1985).

Gen, M. and Cheng, R., Genetic algorithms and engineering optimization, John Wiley, New York, (2000).

Holland, John H., Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems. In Michalski, Ryszard S., Carbonell, Jaime G. and Mitchell, Tom M. Machine Learning: An Artificial Intelligence Approach, Los Altos, CA: Morgan Kaufman, vol II. P. 593-623, (1986).

Hong Y. Y. and Li C. . Genetic algorithms based Economic Dispatch for Cogeneration Units Considering Multiplant Multibuyer Wheeling, IEEE Power Engineering Review, 22(1), pp. 66-69, (2002).

H.-P. Schwefel, 'Numerical Optimization of Computer Models', Wiley, Chichester, (1981).

H. G. Beyer and H. P. Schwefel, Evolution strategies: A comprehensive introduction, Natural Computing 1, pp. 3-52, (2002).

H. P. Schwefel. Evolutions strategies and Numeric Optimization, Dr.-Ing. Thesis, Technical University of Berlin, Department of Process Engineering, (1975).

H. R. Kanan, M. H. Moradi, A Genetic Algorithm based Method for Face Localization and pose Estimation, 3rd International Conference: Sciences of
Electronic, Technologies of Information and Telecommunications, (2005)
Hidalgo, J., F. Soltero, D. Bodas-Sagi, and P. Fernandez-Blanco, Technical market indicators optimization using evolutionary algorithms. Paper presented at the thirteenth IEEE International conference on evolutionary computation, Atlanta, Georgia, USA, (2008).

Holland J. H., Outline for a logical theory of adaptive systems. JACM 9: pp 297–314, (1962).

J. Blazewicz, M. Szachniuk and A. Wojtowicz, Evolutionary algorithm for a reconstruction of NOE paths in NMR spectra of RNA chains, Bulletin Of The Polish Academy Of Sciences Technical Sciences, Vol. 52, No. 3, pp. 8-33, (2004).

J. H. Holland, 'Adaptation in Natural and Artificial Systems', MIT Press, Cambridge, MA, (1992).

J. H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor, Michigan, (1975).

Jina, N., E. Tsang, and J. Li., A Constraint-guided method with evolutionary algorithms for economic problems. Applied Soft Computing 9 (3): pp 924-935, (2009).

J. D. Schaffer and A. Morishima, "An adaptive crossover distribution mechanism for genetic algorithms," in *Proceeding of the Second International Conference on Genetic Algorithms*, pp. 36-40, (1987).

J.F. Gonçalves, J.M. Mendes, and M.C.G. Resende. A hybrid genetic algorithm for the job shop scheduling problem. *European Journal of Operational Research*, vol. 167, pp. 77-95, (2005).

Kinnear, Kenneth E. Jr. (editor), Advances in Genetic Programming. Cambridge, MA: MIT Press, (1994).

Koza, J. R., Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA: The MIT Press, (1992).

Koza, J., Genetic Programming II: Automatic Discovery of Reusable Programs, The MIT Press,USA, (1994).

Krohling R. A. and Rey J. P. Design of optimal disturbance rejection PID controllers using genetic algorithm, IEEE Transactions on Evolutionary Computation, 5(1), pp. 78-82, (2001).

L. J. Fogel, A. J. Owens, and M. J. Walsh, 'Artificial Intelligence through Simulated Evolution', Wiley, New York, (1966).

Muhlenbein, H. and Schlierkamp-Voosen, D., Predictive models for the breeder genetic algorithm: I. continous parameter optimization, *Evol. Comput.* **1**, pp. 25–49, (1993).

Mora, M., G. Forgionne, J. Gupta, L. Garrido, F. Cervantes, and O. Gelman, "A strategic review of the intelligent decision-making support systems research: The 1980-2004 period". In Intelligent decision-making support systems: Foundations, applications and challenges, eds., pp. 441-462. Germany: Springer-Verlag, (2006)

Marczyk, A., Genetic algorithms and evolutionary computation, (2004).

M. Younes, M. Rahli and L. A. Koridak, Economic Power Dispatch Using Evolutionary Algorithm, Journal of Electrical Engineering, Vol. 57, No. 4, pp 211–217, (2006).

M. Sedighizadeh, and A. Rezazadeh, Using Genetic Algorithm for Distributed Generation Allocation to Reduce Losses and Improve Voltage Profile, World Academy of Science, Engineering and Technology 37, (2008).

Manish C. T, Yan F., and Urmila M. D., Optimal Design of Heat Exchangers: A Genetic Algorithm Framework, Ind. Eng. Chem. Res., 1999, 38 (2), 456-467 (1998).

M. H. Khorshid and H. A. Hassan, An Economic Optimization-Oriented Decision Support Model Based On Applying Single-Objective Evolutionary Algorithms To Computable General Equilibrium Models, Working Paper No. 9, pp. 15-39, (2010).

M. H. Khorshid and H. A. Hassan, An Economic Optimization-Oriented Decision Support Model Based On Applying Single-Objective Evolutionary Algorithms To Computable General Equilibrium Models, Working Paper No. 9, pp 4-12, (2010).

M. Hosseinzadeh and E. Roghanian, An Optimization Model for Reverse Logistics Network under Stochastic Environment Using Genetic Algorithm, International Journal of Business and Social Science Vol. 3 No. 12, (2012)

M. Kumar, M. Husian, N. Upreti and D. Gupta, Genetic Algorithm: Reviewand Application, International Journal of Information Technology and Knowledge Management, Vol. 2, No. 2, pp. 451-454, (2010)

N. Tyagi and R. G. Varshney, A Model To Study Genetic Algorithm For The Flowshop Scheduling Problem, Journal of Information and Operations Management, Vol 3, pp-38-42 (2012)

Osyczka, A. Evolutionary algorithms for single and multi-criteria design optimization. Germany: Physica Verlag, (2002).

Oliver, J. M., Smith, D. J. and Holland, J. R. C., A study of permutation crossover operators on the travelling salesman problem, in: *Proc. 2nd Int. Conf. on Genetic Algorithms*, pp. 224–230, (1987).

O'Reilly, U.M. and Oppacher, F., 'Program search with a hierarchical variable length representation: Genetic programming, simulated annealing and hill climbing', Davidor, Y. ,Schwefel, H.P. and Manner, R. (Ed.), *Parallel Problem Solving in Nature - PPSN III*, Vol.866 of Lecture Notes in Computer Science, Springer-Verlag, Germany, pp.397-406, (1994).

P.J.B. Hancock, An empirical comparison of selection methods in evolutionary algorithms. In T.C. Fogarty (ed.), *Evolutionary Computing: AISB Workshop, Leeds, UK, April 1994; Selected Papers.* Springer-Verlag, Berlin, pp. 80–94, (1994).

Po-Hung Chen and Hong-Chan Chang. Genetic aided scheduling of hydraulically coupled plants in hydro-thermal coordination, IEEE Transactions on Power Systems, 11(2), pp. 975-981, (1996)

R. Kumar, Blending Roulette Wheel Selection & Rank Selection in Genetic Algorithms, International Journal of Machine Learning and Computing, Vol. 2, No. 4, (2012).

R. A. Fisher. The Design of Experiments, Oliver and Boyd, Edinburgh, (1935).

R. Leardi, Application of genetic algorithm–PLS for feature selection in spectral data sets, Journal Of Chemometrics; vol.14, pp. 643–655, (2000).

Rechenberg I., Cybernetic solution path of an experimental problem. Royal Aircraft Establishment, Farnborough p. Library Translation 1122, (1965).

Rechenberg I., Evolutions Strategy: Optimization technique System and Principle in Biological Evolution. Dr.-Ing. Thesis, Technical University of Berlin, Department of Process Engineering, (1971).

Safarzynska, K., and J. Bergh, Evolutionary models in economics: A survey of methods and building blocks. Journal of Evolutionary Economics, (2009)

S. Hartmann, A Competitive Genetic Algorithm for Resource-Constrained
Project Scheduling, (1998).

S. L. K. Pond, D. Posada, M. B. Gravenor, C. H. Woelk and S. D.W. Frost, GARD: a genetic algorithm for recombination detection, Bioinformatics Applications Note, Vol. 22 no. 24 , pp 3096–3098, (2006).

S. N. Sivanandam, S. N. Deepa, Introduction to Genetic Algorithms, ISBN 978-3-540-73189-4 Springer Berlin Heidelberg New York, Springer-Verlag Berlin Heidelberg , pp. 39-71,(2008).

Schwefel H-P., Kybernetische Evolution als Strategie der exprimentellen Forschung in der Strömungstechnik. Master's thesis, Technical University of Berlin, Germany, (1965).

Shimamoto, N., Hiramatsu, A. and Yamasaki, K. A dynamic routing control based on a genetic algorithm, IEEE International Conference on Neural Networks, 1993, vol.2, pp. 1123-1128, (2000).

Spears, W., Recombination parameters, in: *The Handbook of Evolutionary Computation*, T. B¨ack, D. B. Fogel and Z. Michalewicz, eds, Chapter E1.3, IOP Publishing and Oxford University Press, Philadelphia, PA, pp. E1.3:1–E1.3:13, (1997).

Spears, W. M. and De Jong, K. A., On the virtues of parameterized uniform crossover, in: *Proc. 4th Int. Conf. on Genetic Algorithms*, (1994).

Syswerda, G., Uniform crossover in genetic algorithms, in: *Proc. 3rd Int. Conf. on Genetic Algorithms*, pp. 2–9,(1989).

Tsang, E., P. Lsasi, and D. Quintana. A special session on evolutionary computation in finance and economics. The annual congress on evolutionary computation, Norway, (2009).

Timo Eloranta and Erkki Makinen, TimGa: A genetic algorithm for drawing Undirected Graphs. Divulgacioncs Mathematics, vol.9 pp. 155-171, (2001).

T. Miquelez, E. Bengoetxea, P. Larranaga, Evolutionary Computation Based On Bayesian Classifiers, Int. J. Appl. Math. Comput. Sci., Vol. 14, No. 3, pp 335–349, (2004).

Tank K. S., Man K. F., Kwong S and He Q. Genetic algorithms and their applications, IEEE Signal Processing Magazine, 13(6), pp. 22-37, (1996).

T. Back and Frank Hoffmeister. Adaptive search by evolutionary algorithms. In W. Ebeling, M. Peschel, and W. Weidlich, editors, *Models of Selforganization in Complex Systems (MOSES)*, Akademie–Verlag, Berlin, pp. 156–163, (1992).

Tanese, Reiko. Distributed Genetic Algorithm for Function Optimization. PhD. dissertation. Department of Electrical Engineering and Computer Science. University of Michigan. (1989).

T. Back and F.Hoffmeister, "Extended Selection Mechanisms in Genetic Algorithms",ICGA4, pp. 92-99, (1991).