An Introduction to Genetic Algorithms and Evolution Strategies

[1]Mehrdad Dianati, [2]Insop Song, and [3]Mark Treiber

[1]Electrical & Computer Engineering, [2]Systems Design Engineering, and [3]Mechanical Engineering

[1,2,3]200 Univ. Ave. West, University of Waterloo, Ontario, N2L 3G1, Canada

***Abstract* – Genetic Algorithms and Evolution Strategies represent two of the three major Evolutionary Algorithms. This paper examines the history, theory and mathematical background, applications, and the current direction of both Genetic Algorithms and Evolution Strategies.**

I. INTRODUCTION

Evolutionary Algorithms can be divided into three main areas of research: Genetic Algorithms (GA) (from which both Genetic Programming (which some researchers argue is a fourth main area) and Learning Classifier Systems are based), Evolution Strategies (ES) and Evolutionary Programming. Genetic Programming began as a general model for adaptive process but has since become effective at optimization while Evolution Strategies was designed from the beginning for variable optimization.

In section II, the History of both Genetic Algorithms and Evolution Strategies will be examined including areas of research that apply both GA and ES. In section III the theory and mathematical background of GA and ES will be laid out. Additionally, both algorithms will be demonstrated in two separate examples. Finally in section IV a survey of current applications in which GA and ES have been applied is presented.

II. HISTORY

The origins of Evolution Computing can be traced to early work by Computer Scientists in the 1950s and 1960s with the idea that evolutionary processes could be applied to engineering problems of optimization. This led to three major independent implementations of Evolutionary Computing of which two are Evolution Strategies and Genetic Algorithms.

Genetic Algorithms were initially developed by Bremermann [10] in 1958 but popularized by Holland who applied GA to formally study adaptation in nature for the purpose of applying the mechanisms into computer science [21]. This work lead to the development of the Schema starting in 1968[22] which was explained in detail in his 1975 book *Adaptation in Natural and Artificial Systems* [23].

The Schema Theorem represented Holland's attempt to place Genetic Algorithms on firm theoretical framework. The first advancement on the Schema theory was by Goldberg who made the popular supposition known as the Building Block Hypothesis that crossover is the major source of Genetic Algorithm performance.[17] This is in contrast to the Schema theory which is focused mainly on the destructive behavior of the crossover and mutation operators,

In the 1990s, criticisms of the Schema theorem have appeared. Grefenstette argued [16] that the Schema theorem formulates that the GA will converge schemas that are winners of actual competition rather than on schemas with the best-observed fitness. Fogel and Ghozeil [11] criticized the Schema theorem for not being able to estimate the proportion of Schema in a population when fitness proportionate selection is used in the presence of noise or other stochastic effects. In Holland's defense, Poli argued [33] that Fogel and Ghozeil's criticisms were not based upon Holland's original theorem and that the original theorem is very good at modeling Schema in the presence of noise. Radcliffe also defended [34] the Schema theorem by explaining that many of the criticisms were not with Holland's theorem itself but with its over-interpretation.

To augment the Schema theory, more "exact" mathematical models have also been developed to make predictions about the population composition, the speed of population convergence and the distribution of finesses in the population over time which the Schema theory does not directly address. In 1991, Vose and Liepins [47] solved a simple genetic algorithm with an exact model that provided a geometric picture of the GA's behaviour and since then various other authors have provided additional "exact" models. Other less exact methods such as applying Markov Chain analysis to modeling GA by Nix and Vose [30] have been attempted; however, the formulations were difficult to solve due to high dimensions and non-linearities.

University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada

Recently, Statistical-Mechanics techniques from Physics have been successfully applied to GA by modeling a set of macroscopic variables that are assumed to characterize the system. The remaining degrees of freedom are then assumed to follow a maximum entropy distribution. Prugel-Bennett and Shaprio initially [40] applied this method to modeling simple GA, but it has recently been applied to understand the differences between different types of GA algorithms (for example Rogers and Prugel-Bennett) have compared Generational and Steady-State GA [37]).

Other recent advances in GA include the introduction of a variable-length chromosome by Kotani, Ochi, Ozawa, and Akazawa[27] in 2001. Kotani et al applied GA to determine a linear discriminator between two data sets. Using a fixed-length chromosome, they found that by increasing the number terms in the discriminator (using longer and longer chromosomes), the final fitness level of the GA increased. After they developed the variable-length chromosome, they found that there was an upper bound for extending the length of the chromosome after which there was not an increase in the average fitness of the GA.

However, while Holland popularized the GA, Bremermann made significant advances in the development of GA with the idea that in the future computers would be capable of implemented his more advanced methods. Bremermann was the first [12] to implement real-coded Genetic Algorithms as well as providing a mathematic model of GA known as the one-max function.

In contrast to Genetic Algorithms, Evolution Strategies were initially developed for the purpose of Parameter Optimization. According to Rechenberg[35], the first Evolution Strategies were developed in 1964 at the Technical University of Berlin (TUB). The idea was to imitate the principles of organic evolution in experimental parameter optimization for applications such as pipe bending or PID control for a nonlinear system. In his words "the method of organic evolution represents an optimal strategy for the adaptation of living things to their environment… [and] … it should therefore be worthwhile to take over the principles of biological evolution for the optimization of technical systems".[45]

The algorithm that was used a mutation-selection scheme known as two membered ES, or in short form (1+1)-ES. In this scheme, a child was generated from its parent and then its performance was compared with its parent's and the most fit of the two survived for the next generation.

To calculate the optimal mutation rate of this scheme, Rechenberg calculated the convergence rates of two model functions and calculated the optimum standard deviations for successful mutations. From this he postulated his 1/5 success rule [36]:

"The ratio of successful mutations to all mutations should be 1/5. If it is greater than 1/5, increase the variance; if it is less, decrease the mutation variance."

Since this method was not a purely Monte Carlo method, it was later enhanced, by adding the notion of population. Rechenburg proposed the multimembered ES where $\mu>1$ parents participate in the generation of 1 offspring. This has been denoted as $(\mu+1)$-ES. In this method, all the parents have the same mating probabilities and as with the two-membered ES, the least fit member of the population including all the parents and the one offspring is eliminated in each generation.

The $(\mu+1)$-ES is not- a widely used strategy but it led to further enhancements by Schwefel in 1975[43,44,45] to enable self adaptation of parameters such as the standard deviation of the mutations. The $(\mu+?)$-ES states that μ parents produce ? offspring $(?>\mu)$ that compete with the parents to select the μ most fit parents for the next generation. This scheme has problems with local optimum which lead to the $(\mu,?)$-ES where the life time of each individual is only one generation. While Schwefel recommends the $(\mu,?)$-ES be preferred over the $(\mu+1)$-ES, recent evidence suggests that the latter performs as well or better than the former in practical applications.

The $(\mu+1)$-ES and $(\mu+1)$-ES algorithms also implement self-adaptation by subjecting the evolution parameters (the standard deviation of the mutations) to evolve themselves. While this will lead to short periods of recession, it avoids stagnation periods where the same individuals dominate the entire population.

In addition to the previous step-size adaptation scheme, another scheme that is also state-of-the-art called de-randomized mutation step size control. Hansen and Ostermeier [19] developed this method where the path of a population is observed over multiple generations and the scheme develops a correlation between the increased fitness of a population and the direction of mutation.

While the selection of mutation rate has been extensively explored, their success and limitations have only been observed through empirical methods. Since all the self-adaptation schemes can be caught at local maxima in certain situations, Rudolph [38] began development of a method to model the mutation of ES using Markov Chains. He has since proved [39] that the "1/5 Success Rule" does not guarantee convergence on the global maximum of a numerical optimization problem.

In the last decade, components of different evolutionary methods have been mixed to create hybrid evolutionary algorithms. One such hybrid between GA and ES is the self-adaptation genetic algorithm. Back, who has contributed to self-adaption in both GA and ES, and expanded by Smith and Fogarty [46], added mutation rate to the chromosomes of individuals in GA that allowed the

mutation rate to evolve at the same rate as the variables in the chromosome.

Another area of focus in the past decade is the development of multi-objective evolutionary algorithms. Most of the implementations have been based upon Genetic Algorithms but a few have been implemented using Evolution Strategies. In these problems, it is possible for multiple solutions to exist that are known as *Pareto*-optimal solutions. The first multi-objective evolution algorithm was introduced by Schaffer [42] in 1985 and referred to as the *Vector Evaluated Genetic Algorithm (VEGA)*. In this method, the mating pool is divided into parts in which each part is evaluated by a single objective fitness function. Although this algorithm has limitations, it is typically used as a reference.

Hajela and Lin introduced [20] the *Aggregation by Variable Objective Weighting* in which the total fitness of each chromosome is calculated by summing each objective's fitness function scaled by a weight value that is also subjected to evolution. Most multi-objective algorithms are Paereto-based including the *Niched Pareto Genetic Algorithm (NPGA)*, introduced by Horn and Nafpliotis [25], where tournament selection is used to select individuals. This algorithm is often used as a reference in publications. Although most multi-objective schemes are based upon GA, Knowles and Corne[26] in 1999 developed a (1,1)-ES that is capable of multi-objective optimization. This algorithm is quite different from the before mentioned GA based schemes since this algorithm is confined only to local search but it is a true Pareto optimizer.

III. THEORY

Evolutionary computing is a family of stochastic search techniques that mimic the natural evolution proposed by Charles Darwin in 1858. In the realm of search techniques the following classification indicates the position of evolutionary algorithms:

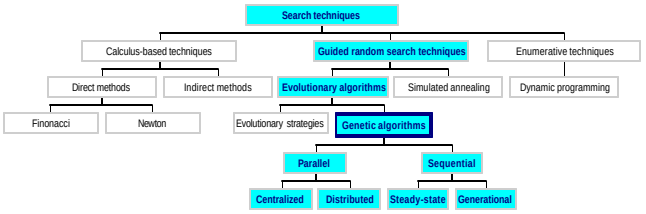


Fig 1. Search techniques

If we consider intelligence as a kind of capability of an entity to adapt itself to ever changing environment, we could consider evolutionary algorithms as a subdivision of soft computing:

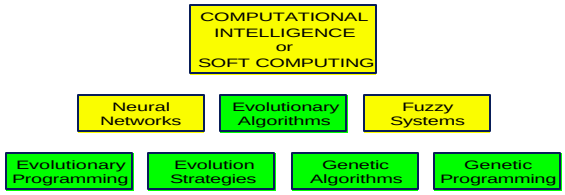


Fig 2. Artificial Intelligence techniques

These algorithms are made of the several iterations of basic Evolution Cycle:

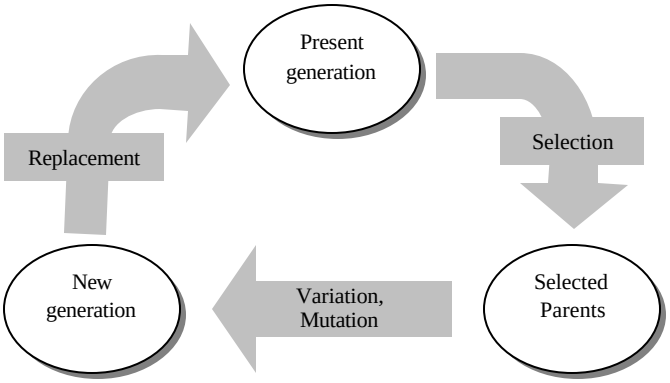


Fig 3. Basic Evolution Cycle

Different variations of Evolutionary Computing incorporate the same basic cycle with different presentations' model or specific combinations of Variation, Mutation, Selection, and Replacement methods. The interesting point in implementation is the balance between two opposite operations. In one hand the Selection operation intends to reduce diversity of population (set of possible solutions) and on the other hand the Variation and Mutation operators try to increase diversity of population. This fact leads to the convergence rate and quality of solution.

As an optimization algorithm, Evolutionary Algorithms should be analyzed to find answers to fare questions like:

- Rate of convergence
- Quality of evolved solution
- Computational requirements

So far, no general analysis framework has been proposed to analyze the general form of evolutionary algorithms, but some specific variations or implementations could be focused along two lines of investigations: theoretical and empirical. The theoretical approach attempts to discover mathematical truths of algorithms that will hold in a reasonably broad domain of applications. However, the empirical approach attempts to assess the performance of an implementation in specific domain of application. Both methods have advantages and disadvantages, and in practice they should be used as complementary means to design and tune the specific instance of an algorithm.

A. Genetic Algorithm

The canonical form of GA is in the following steps:

1. Define the objective function (environment).

2. Present he possible solutions (phenotype) as binary strings (genotype or chromosome). All the optimization parameters should be placed somewhere inside chromosome structure e.g. if the objective function is to be optimized for three parameters x, y and z one possible structure could be as follows:

$$\text{ObjectiveFunction}: F(x,y,z)$$

$$c_i = \underbrace{11101010}_{x}\ \underbrace{11100011}_{y}\ \underbrace{00110111}_{z}$$

3. Generate a random population of specific size. The population size affects the efficiency and performance of GA [1], [3]. GA does poorly for very small size of populations and very large population size impacts performance of the algorithm. For typical applications, the suggested range is between 10-160 chromosomes [1].
Initial population of m chromosome: $c_1, c_2, ..., c_m$

4. Evaluate the fitness of every solution over the objective function. There are many methods to evaluate fitness and assign a real number to each chromosome, the most popular one is called *proportional selection method:*

5.
$$Fitness\ (x_i) = \frac{F(\ phenotype\ (x_i))}{\sum_i F(\ phenotype\ (x_i))}$$

c_i	Binary string	Decoded integer	Fitness	Fitness ratio
c_1	1100	12	36	16.5
c_2	0100	4	44	20.2
c_3	0001	1	14	6.4
c_4	1110	14	14	6.4
c_5	0111	7	56	25.7
c_6	1001	9	54	24.8
Sum			218	100

Table 1. Example fitness for $F(x) = 15x - x^2$, size of Population is 6 and chromosome length is 4.

6. Select a pair of chromosomes for mating by a random selection method e.g. roulette wheel (there are other selection algorithms like tournament selection and rank based selection). For previous example we simulate the following roulette wheel:

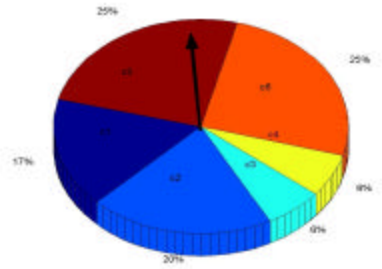
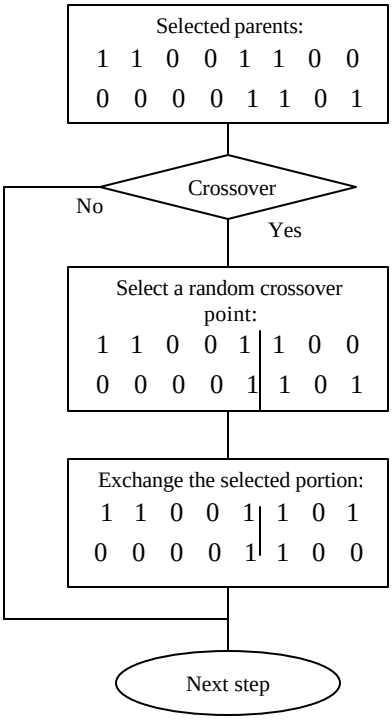


Fig 4. Roulette Wheel selection

6. Apply crossover operation on the selected pair if they have been chosen for crossover (based on probability of crossover p_c). The most applied crossover operation is *single point crossover:*



Based on the probability of bit mutation p_m flip the correspondent bit if selected for mutation. At this point we finished the process of producing a pair of offspring form two selected parents.

7. Repeat steps 5 and 6 until the production of next generation exceeds size of previous generation.
8. Replace the parent population with new generation.
9. Go through steps 4 to 8 until the termination criteria met.

Example: For $-1 \le x, y \le 1$ find the maximum of:

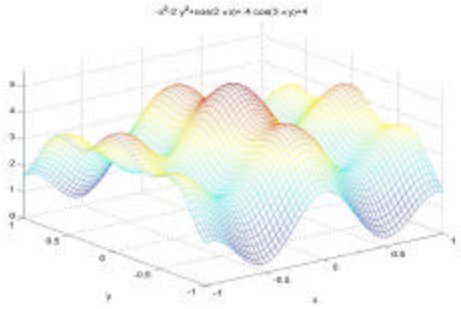


Fig 5.

$$F(x,y) = -x^2 - 2y^2 + \cos(3px) + 0.3\cos(4py) + 4$$

The next step is to find the chromosome structure:

$$s_i = [\underbrace{1\ 0\ 0\ 1\ 0\ 1\ 1\ 1}_{x}\ \underbrace{0\ 0\ 1\ 1\ 1\ 0\ 1\ 1}_{y}]$$

Initial population of 50 random chromosomes:

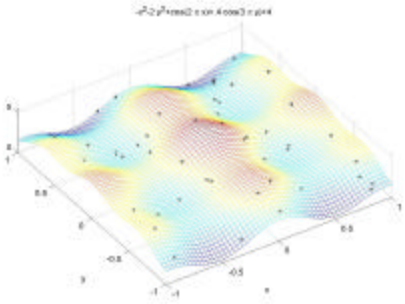


Fig 6. Initial population

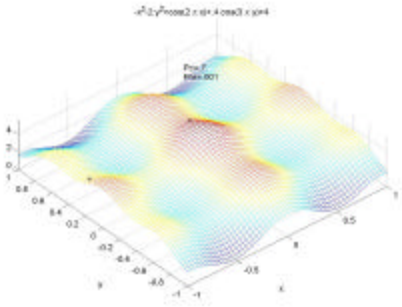


Fig 7. After 150 generation

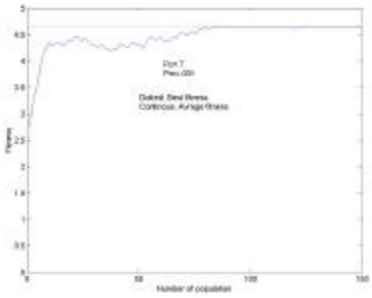


Fig 8. Evolution of average fitness

B. Mathematical Analysis of GA

In this section, we will review the mathematical foundation theory of GA. Several people tried to assess GA behavior mathematically. The most popular work has been done by Holland [4] based on Schema Theory. Fogel criticized Holland's Schema approach and tried to model GA with Markov Chains [1]. Others [7] attempt to apply other less known approaches. In spite of many significant researches it is still controversial and an open research problem. In this paper, we will briefly look at key implications and criticism of Schema theory.

Schema: It is Greek word (plural: schemata) that means form, and in our discussion it is a string that contain '0', '1' and '*' characters. As a simple example, '0**1' is a schema that represents the following variations (instances):

0001, 0011, 0101, 0111

Schema is useful for analysis of GA, because we can categorize points in different parts of the search space by different schemas. This concept can be described by following figure:

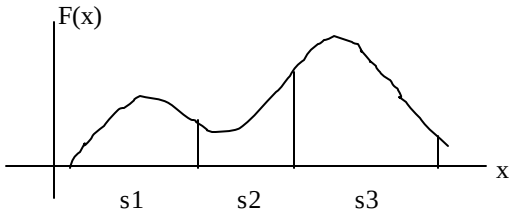


Fig 9. Division of Search Space by Schema

In this figure the search space in different parts of the objective function have been categorized by three schemata s1, s2 and s3. If we generate a population of chromosomes they will have instances of s1, s2 or s3.

Defining length and order of schema: The distance between outermost non '*' bits is called defining length. For example, the defining length of '*0*1*10*' is 5. Order of a schema is the number of non '*' bits. For example the order of '*0*1*10*' is 4.

Schema Theorem: In canonical form of GA with single point crossover probability p_c and mutation probability of p_m the expected number of instances of schema H, (M_H) with length l, defining length l_d, order $O(H)$ and the average fitness $\hat{f}_H(t)$ in next generation satisfies the following inequality:

$$M_H(t+1) \geq \frac{\hat{f}_H(t)}{\hat{f}(t)} M_H(t)\left[1 - p_c\left(\frac{l_d}{l-1}\right)\right](1-p_m)^{O(H)}$$ Average fitness of current population: $\hat{f}(t)$

The complete proof could be found in [2] and [3].

Holland concluded that if we just consider the destructive effects of crossover and mutation on schema structure, the above inequality will turn to equality, so in next generations number of instances of schemata with higher average fitness will increase exponentially and the number instances of less fit schemata will decrease exponentially:

$$M_H(t) \propto M_H(0)\left[\frac{\hat{f}_H(t)}{\hat{f}(t)}\right]^t$$

K-Armed Bandit Theorem: Given a number of competing payoff sources with unknown rates of payoff, a nearly optimal adaptive strategy is to allocate an exponentially increasing number of trials to the one with the best observed performance. This strategy achieves a balance between the conflicting goals of exploiting the current payoff estimate and improving the payoff estimate (exploring new facts) in order to avoid premature decisions.

Putting together Holland concluded that GA with proportional selection method would achieve optimal processing of sampled schemata by the randomly selected population of chromosomes. On the other hand the GA achieves optimal Monte Carlo search.

Criticisms of schema theory: The main criticism of Holland's inference is about the assumption that neglects the constructive effects of crossover and mutation. In one hand, these assumptions do not look like reasonable simplifications, on the other hand without those assumptions schema theorem would not lead to any valid implication about changes of population fitness over evolution cycles. Besides that, this theorem does not answer many key questions about convergence, convergence rate and implementation guidelines like the size of population.

Schema theorem is not a very strong framework to provide satisfactory analytical basis, and as a result, many attempts have been done to offer a more robust mathematical understanding of GA. One of the most valuable works introduced by Vose and Liepins [47] is based on Markov Chain frameworks. By considering the GA as a finite state Markov Chain they proved the asymptotical convergence of GA with probability one.

C. Evolution Strategies

Another variation of evolutionary algorithms suggested by Rechenberg [5] in 1975 and initially applied for parameter optimization. The main difference between ES and GA are in the presentation of population and the types of evolution operators. In ES, instead of binary strings we use real values to present parameters of optimization. Also contrary to GA that incorporates both crossover and mutation, ES just use mutation. Regarding these differences, it seems that ES are easier to implement and might be faster than GA. Of course, *no free launch theorem [1]* states that there is no globally best optimization algorithm, and each algorithm will be efficient for specific application domains.

The basic implementation of evolution strategies was *two membered* $(1+1)-ES$, i.e. one parent generates one offspring and the best of two is selected and the other eliminated. In this paper, we will explain this basic form and then introduce some later extensions.

1. Choose a single parent vector that contains m parameters $X = (x_1, x_2, ..., x_m)$. Each parameter is chosen with random process and satisfies the constraints of problem.
2. Create a new offspring by mutation. To achieve the mutation in this method, add a random vector of size X with normal distribution (mean zero and variance s):

   $$X' = X + N(0, s)$$

   From the mathematical analysis for two sample cost functions, Rechenberg [5] suggested the following heuristic rule for adjusting s :

   1/5 success rule: *The ratio of successful mutation to all mutations should be 1/5. If it is greater than 1/5, increase the variance; if it is less, decrease the mutation variance.*

3. Compare the solutions for X and X'. Choose the best member for the next generation.
4. Repeat steps 2 and 3 until a satisfactory solution is found or the computation time is exhausted.

Joachim Born [5] proved, for the regular optimization problems [5] this algorithm converges to global optimum with probability:

Theorem: For a regular optimization [5] problem with cost function f and global optimum $f^* > -\infty$:

$$\Pr ob\left\{\lim_{t \to \infty} f(x^t) = f^*\right\} = 1$$

As we could see in $(1+1)-ES$, we never used the concept of population in search and it is a 'point to point' search, therefore it can likely be entrapped in local maxima (although we will showed asymptotically, it converges to global maximum with probability 1). In order to improve the algorithm to use the concept of population and decrease entrapment risk, Rechenberg suggested $(m+1)-ES$ algorithm. The two general form of $(m+l)-ES$ and $(m,l)-ES$ suggested to improve ES for both parallel processing behave better with respect to local optima. The other improvement suggested in these new versions was to

change mutation variance adaptively from population to population. In other words, add a kind of learning to the search algorithm. These new general algorithms could be implemented in following steps:

1. Choose m parent vectors that contain m parameters $X = (x_1, x_2, ..., x_m)$. Each parameter is chosen through a random process and satisfies the constraints of problem.

2. Create l new offspring ($m \prec l$) by recombining m parents and mutation like step 2 in $(1+1)ES$.

   Comment: There are five types of recombination operators:

   1. No recombination: Select one parent randomly and let $x_i'' = x_i$.
   2. Discrete: Select two parents a and b randomly and let $x_i' = x_{i,a}$ or $x_i' = x_{i,b}$ with equal probability.
   3. Intermediate: Select two parents a and b randomly and let $x_i' = \frac{1}{2}(x_{i,a} + x_{i,b})$.
   4. Global Discrete: Select a new pair of a_i and b_i parents for every parameter x_i and let $x_i' = (x_{a_i,1}$ or $x_{b_i,2})$ with equal probability.
   5. Global Intermediate: Select a new pair of a_i and b_i parents for every parameter x_i and let $x_i' = \frac{1}{2}(x_{a_i,1} + x_{b_i,2})$.

   Generate the offspring population with following algorithm:
   Current population: P^t
   Intermediate offspring: $X'(x_1', x_2', ..., x_m')$
   Mutation operator: M
   Step size meta-control: Δs
   Recombination Operator: R

$$(X', s') = R(P^t)$$
$$(X'', s'') = M[(X', s')]$$
$$s'' = s'.Exp(N(0, \Delta s))$$
$$X'' = X' + N(0, s'')$$

   Note that the mutation operator is applied for both parameters and correspondent variance.

3. Select m most fit solutions for next generation:

   For $(m + l) - ES$: Select the next generation from $(m + l)$ population of all parents and offspring. For $(m, l) - ES$: Select the next generation from l population of offspring.

4. Repeat steps 2 through 3 until satisfactory solution found or the computation time exhausted.

Example: We simulated $(1+5) - ES$ to find the maximum of $F(x, y) = -x^2 - 2y^2 + 4$.

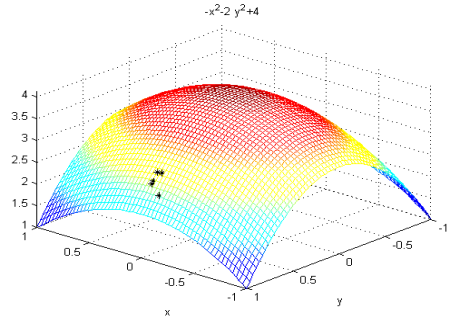


Fig 10. First generation of offspring

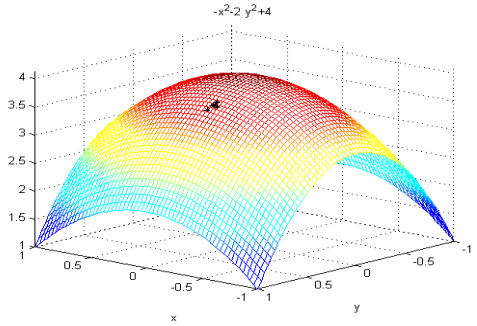


Fig 11. Generation 10

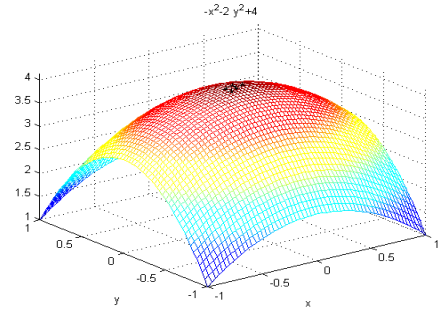


Fig 12. Generation 30

In this simple example, we could see hill climbing capability of a simple ES.

IV. APPLICATIONS

Evolutionary Computing methods – GA and ES – have applied to solve wide range of engineering problems, such as manufacturing scheduling, image processing, robotics, power electronics, VLSI design, CAD design, chemistry, signal processing and physics; in most application area, the average annual growth of GA papers has been approximately 40% during last twenty years [56, 54, 59, 55, 57, 53]. In chemistry and physics, GA has been used various topics, such as protein folding, antennas, and

macromolecules [53]. In Computer Aided Design area, layout design, optimizations, and shape design are main topics [54]. In electronics and VLSI design area, layout, VLSI design, and testing are major research topics [55]. In manufacturing area, process planning, scheduling, and processing control are main research topics [56]. In optics and image processing research area, pattern recognition and filters are major topics [53]. In power control area, motor control, optimization, scheduling, and economic dispatch are main research topics [57]. In robotics area, robot control, mobile robot path planning, and motion planning are major research topics.

A. GA Application

1) Job-Shop Scheduling problem

Madureia et al. suggested GA for the resolution of real world scheduling problems, and proposed a coordination mechanism [29]. Because of frequently changing dynamic environments, providing efficient production management and timely delivery are one of the hard to solve problems. Scheduling is to allocate a set of machines to perform a set of jobs within a certain time period, and the goal of scheduling is to find a appropriate allocation – schedule – which maximize certain performance measure. For the implementation issues, the solutions are encoded by natural representation, and the order crossover operator is used. They used the inversion mechanism as mutation operator. Finally, Madurea et al. solved dynamic scheduling problem using a set of static scheduling by GA, and they showed the feasibility of GA in Job-Shop scheduling problem.

2) Real-time system task managements

Sandstrom et al. applied GA for assigning task priorities and offsets to guarantees that real time timing constraints [41]. Assigning timing constraint to task is not trivial problem in real-time system. They showed how timing constraints be mapped to attributes of periodic tasks running on standard pre-emptive RTOS (Real-Time Operating System) such as VxWorks and QNX. They used GA because of the GA's ability to generate a result that satisfies a subset of the timing constraints in cases where it is impossible to fulfill all constraints. GA, the mechanism of natural selection, gradually improves individuals –timing constraints assignment - in a population. They have tested on a many test cases, and showed good result.

3) Robot Path planning with give map by GA

Zein-Sabatto and Ramakrishnan applied GA for multiple robot path planning [52]. They generated optimal path for multiple robot using GA, and for multiple robot environment, they had to consider the robot size and location of known obstacles in the environment and topological elevations of the environment, because the robots are in 3D environment. In their problem setup, all the obstacles are known; they concentrated on global planning. The path planning for multiple robot caused more challenges, such as minimize total traveling distance and energy consumed by each robot. They employed GA for this research because GA provides a robust search in complex spaces, and is less expensive then other search algorithms. They used elevation changes and energy changes for their fitness function. Finally, they showed efficiency and robustness of GA on multi-robot path planning in 3D environment.

4) Sensor-based Robot Path Planning

Yasuda and Takai applied GA for sensor-based mobile robot path planning under unstructured environment in real-time [50]. After finding obstacles, the planning module generates a short and safe path to goal with obstacle avoidance, which is a sequence of control vectors of orientation. With GA, a path is represented as a set of orientation vectors with equal distance. Thus, the final path is the composition of polygonal lines (sum of vectors). To minimize the length, the change of orientation is restricted to 5 values from –45 deg to 45 deg. For fitness function, they used distance parameters between goals, obstacles. They used the combination of roulette and elite selection, one-point cross over. They tried to make their system simple to operate in real-time environment.

5) Image Processing by GA

Gong and Yang applied GA for stereo image processing [18]. Stereovision system generates disparity map; the disparity map should be smooth and detail. They used intensity-based approach. They increased the accuracy of the disparity map by removing the mismatches from occlusions and false targets. They formalized stereo matching as optimization problem; GA optimized the compatibility between corresponding points and continuity of the disparity map. First, the 3D disparity is populated with dissimilarity values based on the source images; a fitness function is defined based on the Markov Random Field to test a disparity map. GA extracts best population from disparity map. Color image segmentation, graft crossover was applied; elitist strategy is applied for selection. Their experiment showed that GA out performed existing methods.

B. ES Application

1) Parameter estimation by ES

In system parameter estimation, there have been many researches using the maximum likelihood (ML), the maximum a posterior (MAP), or the least squares (LS). However, ES - as a stochastic search - can be applied to system parameter estimation. Hatanaka et al. applied ES for multiple estimates of the system parameters, and showed numerical examples [24]. In system parameter estimation, adaptability and robustness are important. Adaptability is adaptiveness to the system dynamics; robustness is to robust to outliers. Hatanaka et al. applied ES to the parameter estimation of autoregressive (AR) model, and they used $(\mu + \lambda)$-ES selection. Finally, they showed the out-performance of ES over recursive least square and recursive weighted least squares methods; they emphasized the adaptability and robustness of ES over other methods.

2) Image processing and Computer Vision system

ES can also be applied to image analysis applications. Louchet applied ES to stereo image analysis [28]. Because of the image data, ES in image analysis suffered from heavy computation complexity to manage population. Thus, Louchet split the problem into several independent and simpler primitives; he used 3D points: one of the simplest primitives. The main idea of Louchet is to evolve a population of 3D points using a fitness function. The fitness function evaluates the similarity of the 3-D points of each stereo image. He used deterministic selection operator: a ranking process based on fitness values. They searched extensively of the search space using mutation operator: a quasi-Gaussian noise added to each chromosome with a fixed standard deviation. He also claimed real-time properties of ES because ES are adaptive - cope with modifications of the fitness function during the algorithm's run - and speed of ES is heavily dependent on the computational complexity of the fitness function - the easier the fitness function the faster the speed of ES is. Each chromosome is a data structure for representing task assignment. They use $(\mu + \lambda)$ -ES.

3) Task scheduling by ES

Greenwood et al. applied ES for task scheduling in multiprocessor systems, and he illustrated the scheduling of a digital signal-processing algorithm on a two processor distributed system [15]. Multiprocessor scheduling is to assign a set of task onto a multiprocessor system to minimize overall scheduling length, and this is one of the NP-complete problems. ES showed shorter scheduling time than other method.

4) Mobile manipulator path planning by ES

Watanabe et al. applied ES for omni-directional mobile manipulator path planning [49]. For B-spline, choosing appropriate data points and end points is most important. Thus, Watanabe et al. suggested automatic selection of those points using various cost function: motion smoothness, movable range of joint, singular operation, and falling down. This path planning method is also useful for the path generation with time constraints.

5) Car automation using ES

Ostertag et al. applied ES for airbag release optimization; they presented a tuning method for airbag release [31]. A quality function and a modified ES are introduced. Airbag release optimization is difficult problem because there are many different crash situations and misfiring of airbag causes dangers and high repair costs. Quality function is defined for optimal performance; however, it includes erroneous trigger decisions and timings. Because of the characteristic of the quality function, it is difficult to apply general methods: gradient descent or hill climbing. In most of their experimental test, close to the optimal solution were obtained.

V. CONCLUSIONS

In this survey paper, we introduced two variations of the Evolutionary Algorithms: Genetic Algorithms (GA) and Evolution Strategies (ES). Both of them are efficient stochastic optimal search method to solve complex and non-linear problems. The idea for both methods originated from natural evolution consisting of generation, selection, and mutation. Although the methods of GA and ES are similar, they have different techniques for implementation; for example, in GA we need to encode and decode our population (solution candidate); however, in ES we can use real vector data for our population. Finally, both methods are applied to a variety of engineering problems from chemistry to robotics, and we introduced some of these applications.

VI. REFERENCES

[1] David B. Fogel, "Evolutionary Computing", IEEE Press 2002.

[2] Darrel Whitely, A Genetic Algorithm Tutorial, Co mputer Science Department, Colorado University.

[3] Michael Negnevitsky, "Artificial Intekkigence", Addison Wesley 2002.

[4] Bill P. Buckles and Frederick E.Petry, "Genetic Algorithms" IEEE Press 1992.

[5] Thimas Back, Frank Hoffmcister, Hans-Paul Schwefel, "A survey of Evolution Strategies",University of Dortmund, Department of Computer Science XI

[6] Gunter Rudilph, "Convergence of Evolutionary Algorithms in General Search Spaces", JCD Informatik Centrum Dortmund.

[7] Adam Prugel-Bennett, Alex Rogers, "Modeling GA Dynamics", Oct 1999.

[8] Back, T. "Self-Adaption in Genetic Algorithms", *in Proceedings of the 1st European Conference on Aritificial Life*, F.J. Varela and P. Bourgine, Eds. Cambridge, MA: MIT Press, pg 263-271, 1992.

[9] Back, T., Schwefel, H.P. "Evolutionary computation: an overview", *Evolutionary Computation, 1996., Proceedings of IEEE Internation Conference on, pp 20-29, 1996.*

[10] Bremermann, H.J. " The evolution of intelligence. The nervous system as a model of its environment", *Technical report, no.1, contract no. 477(17),* Dept. Mathematics, Univ. Washington, Seattle, July, 1958.

[11] Fogel, D.B., Ghozeil, A. "Schema processing under proportional selection in the presence of random effects". *IEEE Transactions on Evolutationary Computation*, Vol 1, Iss. 4, pp 290-293, 1997.

[12] Fogel, D.B., Anderson, R.W. "Revisiting Bremermann's genetic algorithm. I. Simultaneous mutation of all parameters". *Evolutionary Computation, 2000. Procedings of the 2000 Congress on,* Volume: 2, 2000.

[13] D. K. Gehlhaar and D. B. Fogel, "Tuning evolutionary programming for conformationally flexible molecular docking," in *Proc. 5th Annu. Conf. on Evolutionary Programming.* Cambridge, MA: MIT Press, 1996, pp. 419-429.

[14] Greene, W.A., "Dynamic load-balancing via a genetic algorithm",.*Tools with Artificial Intelligence, Proceedings of the 13th International Conference on*, Page(s): 121 –128, 2001.

[15] Greenwood, G.W.; Gupta, A.; McSweeney, K., "Scheduling tasks in multiprocessor systems using evolutionary strategies", *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on* , vol.1, pp345 –349, 1994.

[16] J. J. Grefenstette. "Deception considered harmful". In L. D. Whitley, ed., *Foundations of Genetic Algorithms 2.* Morgan Kaufmann, 1993.

[17] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.

[18] Minglun G. and Yee-Hong Y., "Multi-resolution stereo matching using genetic algorithm", *Stereo and Multi-Baseline Vision, 2001. (SMBV 2001). Proceedings. IEEE Workshop on* , pp 21 –29, 2001.

[19] Hansen, N., Ostermeier, A. "Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation". *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on, 1996, pp312-317, 1996.*

[20] Hajela, P., Lin, C.Y. "Genetic search strategies in multicriterion optimal design", in *Structural Optimization,* vol 4, pp. 99-107, 1992.

[21] J. H. Holland. "Outline for a logical theory of adaptive systems". *Journal of the Association for Computing Machinery,* 3, 1962, pp. 297-314.

[22] J. H. Holland. *Hierarchical descriptions of universal spaces and adaptive systems* (Technical Report ORA Projects 01252 and 08226. Ann Arbor: University of Michigan, Deparment of Computer and Communication Sciences, 1968.

[23] J. H. Holland. *Adaptation in natural and artificial systems.* Ann Arbor: The University of Michigan Press, 1975.

[24] Hatanaka, T.; Uosaki, K.; Tanaka, H.; Yamada, Y. "System parameter estimation by evolutionary strategy", *SICE '96. Proceedings of the 35th SICE Annual Conference. International Session Papers*,Page(s): 1045 – 1048, 1996.

[25]Horn, J., Nafpliotis, N. *Multiobjective optimization using the niched pareto genetic algorithm, IlliGAL Report 93005,* Illinois Genetic Algorithms Lab., Univ. Illiniois, Urbana-Champagn, July 1993.

[26] Knowles, J., Corne, D. "The Pareto archived evolution strategy: a new baseline algorithm for Pareto multiobjective optimization", *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on,* Vol 1, pp98-105, 1999.

[27] Kotani, M., Ochi, M., Ozawa, S., Akazawa, K. "Evolutionary discriminant functions using genetic algorithms with variable-length chromosome". *Neural Networks, 2001. Proceedings. IJCNN '01. Inernational Joint Conference on,* Vol 1, Pg 761-766, 2001.

[28] Louchet, J. "Stereo analysis using individual evolution strategy" ,*Pattern Recognition, 2000. Proceedings. 15th International Conference on* , Volume: 1 , Page(s): 908 –911,2000.

[29] Madureira, A.; Ramos, C.; do Carmo Silva, S.,"A Coordination Mechanism for Real World Scheduling Problems using Genetic Algorithms", *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on* , Volume: 1, pp 175 –180,, 2002.

[30] Nix, E.A., Vose, M.D. "Modelling genetic algorithms with Markov Chains", *Ann. Math Artif. Intell,* Vol 5, pp79-88, 1992.

[31] Ostertag, M.; Nock, E.; Kiencke, U. "Optimization of airbag release algorithms using evolutionary strategies", *Control Applications, 1995., Proceedings of the 4th IEEE Conference on* , pp275 –280, 1995.

[32] E. Pettit and K.M. Swigger, "An analysis of genetic-based pattern tracking" in Pric. National Conf. on AI, AAAO '83, pp. 327-332, 1983.

[33] Poli, R. "Why the schema theorem is correct also in the presence of stochastic effects". *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on* , Volume: 1 , 2000, Page(s): 487 -492 vol.1

[34] Radcliffe, Nicholas J. " Schema Processing". In: *Handbook of Evolutionary Computation* (T. Baack, D.B. Fogel and Z. Michalewicz, eds.) pp. B.2.5-1.10, Oxford University Press, 1997.

[35] I. Rechenberg. Cybernetic solution path of an experimental problem. Royal Aircraft Establishment, Library translation NO. 1122, Farnborough, Hants., UK, August 1965.

[36] I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, Stuttgart, 1973.

[37] Rogers, A. and Prügel-Bennett, A. Modelling the Dynamics of a Steady State Genetic Algorithm. Foundations of Genetic Algorithms - 5 p.57-68, 1999.

[38] Rudolph, G. "Convergence of evolutionary algorithms in general search spaces", *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on,* pp 50-54, 1996.

[39] Rudolph, G." Self-adative mutations may lead to premature convergence", *Evolutionary Computation, IEEE Transactions on,* Vol 5, Issue 4, pp 410-414, Aug. 2001.

[40]Prugel-Bennett, A., and Shaprio, J.L. 1994. "An analysis of genetic algorithms using statistical mechanics". *Physical Review Letters 72*, no. 9: 1305-1309.

[41] Sandstrom, K.and Norstrom, C , "Managing complex temporal requirements in real-time control systems",. *Engineering of Computer-Based Systems, 2002. Proceedings. Ninth Annual IEEE International Conference and Workshop,* Page(s): 103 –109, 2002.

[42] Schaffer, J.D. "Multiple-Objective optimization using genetic algorithm". *Proc. Of the First Int. Conf. on Genetic Algorithms,* pp. 93-100, 1985.

[43] H.-P. Schwefel. *Evolutionsstrategie und Numerische Optimierung.* Dissertation, Technische Universitat Berlin, May 1975.

[44] H.-P. Schwefel. *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*, volume 26 of *Interdisciplinary Systems Research.* Birkhauser, Basel, 1977.

[45] H.-P. Schwefel. *Numerical Optimization of Computer Models.* Wiley, Chichester, 1981.

[46] Smith, J. and Fogarty, T.C. "Self adaptation of mutation rates in a steady state genetic algorithm", *in Proc. 3rd IEEE Conf. on Evolutionary Computation.* Piscataway, NJ: IEEE Press, pp 318-323, 1996.

[47] Vose, M. D., and Liepins, G. E. "Punctuated equilibria in genetic search". *Complex Systems.* 5:31-44, 1991.

[48] Whitley, D., "A Genetic Algorithm Tutorial", Computer Science Department, Colorado State University, whiteky@cs.colostate.edu

[49] Watanabe, K.; Kiguchi, K.; Izumi, K.; Kunitake, Y., "Path planning for an omnidirectional mobile manipulator by evolutionary computation", *Knowledge-Based Intelligent Information Engineering Systems, 1999. Third International Conference,* Page(s): 135 –140, 1999.

[50] Yasuda, G. and Takai, H. "Sensor-based path planning and intelligent steering control of nonholonomic mobile robots", *Industrial Electronics Society, 2001. IECON '01. The 27th Annual Conference of the IEEE,* Volume: 1, 2001 Page(s): 317 -322 vol.1, 2001.

[51] Zitzler, E. and Thiele, L. "Multiobjective evolutionary algorithms: ac comparartive case study and the strength Pareto approach", *Evolutionary Computation, IEEE Transactions on,* Vol. 3, Issue 4, pp 257-271, 1999.

[52] AlgorithmsZein-Sabatto, S.and Ramakrishnan, R., "Multiple Path Planning for a Group of Mobile Robots in 3D Environment using Genetic" *SoutheastCon, 2002*. Proceedings IEEE, pp 359 –363, 2002.

[53] Jarmo T. Alander , An indexed bibliography of genetic algorithm with Chemistry and Physics, http://citeseer.nj.nec.com

[54] Jarmo T. Alander ,An indexed bibliography of genetic algorithm with Computer Aided Design , http://citeseer.nj.nec.com

[55] Jarmo T. Alander ,An indexed bibliography of genetic algorithm with Electronics and VLSI Design and Testing, http://citeseer.nj.nec.com

[56] Jarmo T. Alander ,An indexed bibliography of genetic algorithm with Manufacturing, http://citeseer.nj.nec.com

[57] Jarmo T. Alander ,An indexed bibliography of genetic algorithm with Power Engineering, http://citeseer.nj.nec.com

[58] Jarmo T. Alander ,An indexed bibliography of genetic algorithm with Optics and Image Processing, http://citeseer.nj.nec.com

[59] Jarmo T. Alander ,An indexed bibliography of genetic algorithm with Robotics, http://citeseer.nj.nec.com