**Application of Evolutionary Algorithms for Multi-objective Optimization in VLSI and Embedded Systems**

This book describes how evolutionary algorithms (EA), including genetic algorithms (GA) and particle swarm optimization (PSO) can be utilized for solving multi-objective optimization problems in the area of embedded and VLSI system design. Many complex engineering optimization problems can be modelled as multi-objective formulations. This book provides an introduction to multi-objective optimization using meta-heuristic algorithms, GA and PSO and how they can be applied to problems like hardware/software partitioning in embedded systems, circuit partitioning in VLSI, design of operational amplifiers in analog VLSI, design space exploration in high-level synthesis, delay fault testing in VLSI testing and scheduling in heterogeneous distributed systems. It is shown how, in each case, the various aspects of the EA, namely its representation and operators like crossover, mutation, etc, can be separately formulated to solve these problems. This book is intended for design engineers and researchers in the field of VLSI and embedded system design. The book introduces the multi-objective GA and PSO in a simple and easily understandable way that will appeal to introductory readers.

Application of Evolutionary Optimization Algorithms for Rehabilitation of Water Distribution Networks

* July 2020
* [Journal of Construction Engineering and Management](https://www.researchgate.net/journal/Journal-of-Construction-Engineering-and-Management-0733-9364) 146(7):04020069

DOI:[10.1061/(ASCE)CO.1943-7862.0001856](http://dx.doi.org/10.1061/(ASCE)CO.1943-7862.0001856" \t "_blank)

**Authors:**

[**Nehal Elshaboury**](https://www.researchgate.net/profile/Nehal-Elshaboury-2)

* [Housing and Building National Research Center](https://www.researchgate.net/institution/Housing_and_Building_National_Research_Center)

[**Tarek Attia**](https://www.researchgate.net/profile/Tarek-Attia-2)

* [Housing and Building National Research Center](https://www.researchgate.net/institution/Housing_and_Building_National_Research_Center)

[**Mohamed M. Marzouk**](https://www.researchgate.net/profile/Mohamed-Marzouk-2)

* [Cairo University](https://www.researchgate.net/institution/Cairo-University3)

Request Full-text Paper PDF

Request full-text PDF

To read the full-text of this research, you can request a copy directly from the authors.

[Download citation](https://www.researchgate.net/publication/342603430_Application_of_Evolutionary_Optimization_Algorithms_for_Rehabilitation_of_Water_Distribution_Networks/citation/download)

Copy link

[Citations (22)](https://www.researchgate.net/publication/342603430_Application_of_Evolutionary_Optimization_Algorithms_for_Rehabilitation_of_Water_Distribution_Networks" \l "citations)

[References (42)](https://www.researchgate.net/publication/342603430_Application_of_Evolutionary_Optimization_Algorithms_for_Rehabilitation_of_Water_Distribution_Networks" \l "references)

Abstract

Deteriorated water distribution networks require significant investments to maximize their functionality. The problem is that limited financial resources are allocated for rehabilitation strategies. This deficiency highlights the importance of developing a tool that helps decision makers develop maintenance and replacement management plans. The optimization tool is employed using two evolutionary algorithms: genetic algorithms and particle swarm optimization. The efficacy of the developed model is demonstrated through its application in a case study of Shaker Al-Bahery, Egypt. Furthermore, evaluation metrics are considered to compare the performance of the aforementioned algorithms. The results reveal that the particle swarm optimization exhibited superior results when compared with the genetic algorithms. Moreover, the following two multicriteria decision-making techniques are used to provide a ranking for the near-optimum solutions: multiobjective optimization on the basis of ratio analysis and technique for order preference by similarity to ideal solution. Finally, the Spearman correlation coefficient is utilized to assess the correlation between rankings obtained from different decision-making methods. The results indicate a very strong relation among the aforementioned techniques.

Application of Evolutionary Optimization Algorithms for Rehabilitation of Water Distribution Networks https://www.researchgate.net/publication/342603430\_Application\_of\_Evolutionary\_Optimization\_Algorithms\_for\_Rehabilitation\_of\_Water\_Distribution\_Networks



The 2006 NASA [ST5](https://en.wikipedia.org/wiki/Space_Technology_5) spacecraft antenna. This complicated shape was found by an evolutionary computer design program to create the best radiation pattern. It is known as an [evolved antenna](https://en.wikipedia.org/wiki/Evolved_antenna).

**15 Real-World Applications of Genetic Algorithms**

* **Published by**[**The Editors**](https://www.brainz.org/author/editor/)

**Genetic Algorithm: A heuristic search technique used in computing and Artificial Intelligence to find optimized solutions to search problems using techniques inspired by evolutionary biology: mutation, selection, reproduction [inheritance] and recombination.**

**1. Automotive Design**

**Using Genetic Algorithms [GAs] to both design composite materials and aerodynamic shapes for**[**race cars**](http://www.sae.org/technical/papers/2003-01-1327)**and regular means of transportation (including aviation) can return combinations of best materials and best engineering to provide faster, lighter, more fuel efficient and safer vehicles for all the things we use vehicles for. Rather than spending years in laboratories working with polymers, wind tunnels and balsa wood shapes, the processes can be done much quicker and more efficiently by computer modeling using GA searches to return a range of options human designers can then put together however they please.**

**2. Engineering Design**

**Getting the most out of a range of materials to optimize the structural and operational design of buildings, factories, machines, etc. is a rapidly expanding application of GAs. These are being created for such uses as optimizing the design of heat exchangers, robot gripping arms, satellite booms, building trusses, flywheels, turbines, and just about any other computer-assisted engineering design application. There is work to combine GAs optimizing particular aspects of engineering problems to work together, and some of these can not only solve design problems, but also project them forward to analyze weaknesses and possible point failures in the future so these can be avoided.**

**3. Robotics**

**Robotics involves human designers and engineers trying out all sorts of things in order to create useful machines that can do work for humans. Each robot’s design is dependent on the job or jobs it is intended to do, so there are many different designs out there. GAs can be programmed to search for a range of optimal designs and components for each specific use, or to return results for entirely new types of robots that can perform multiple tasks and have more general application. GA-designed robotics just might get us those nifty multi-purpose, learning robots we’ve been expecting any year now since we watched the Jetsons as kids, who will cook our meals, do our laundry and even clean the bathroom for us!**

**4. Evolvable Hardware**

[**Evolvable hardware**](https://en.wikipedia.org/wiki/Evolvable_hardware)**applications are electronic circuits created by GA computer models that use stochastic (statistically random) operators to evolve new configurations from old ones. As the algorithm does its thing in the running model, eventually a circuit configuration will come along that does what the designer wants. Think of reconfigurable circuits in something like a space robot. It could use a built-in GA library and simulator to re-design itself after something like radiation exposure that messes up its normal configuration, or encounters a novel situation in which it needs a function it doesn’t already have. Such GAs would enable self-adaptation and self-repair.**

**5. Optimized Telecommunications Routing**

**Do you find yourself frustrated by slow LAN performance, inconsistent internet access, a FAX machine that only sends faxes sometimes, your land line’s number of ‘ghost’ phone calls every month? Well, GAs are being developed that will allow for dynamic and anticipatory routing of circuits for**[**telecommunications networks**](https://en.wikipedia.org/wiki/Telecommunications_network)**. These could take notice of your system’s instability and anticipate your re-routing needs. Using more than one GA circuit-search at a time, soon your interpersonal communications problems may really be all in your head rather than in your telecommunications system. Other GAs are being developed to optimize placement and routing of cell towers for best coverage and ease of switching, so your cell phone and blackberry will be thankful for GAs too.**

**6. Joke and Pun Generation**

**Among the linguistic applications of GAs – including a JAPE (automated pun generator) inspired STANDUP program to design communications strategies for people working with children who suffer communications disabilities – are GAs that search for jokes and puns. These come under the heading of**[**“artificial creativity”**](https://en.wikipedia.org/wiki/Artificial_Creativity)**and AI, but could prove very useful to class clowns and wannabe punsters whose public reputations depend upon being funnier than they actually are. These clever GAs will let you input a word you wish to pun or a subject you’d like to joke about, and will return a variety of solutions that just might lead to a lucrative career on the comedy club circuit!**

**7. Biomimetic Invention**

[**Biomimicry**](https://www.brainz.org/15-coolest-cases-biomimicry/index.html)**or biomimetics is the development of technologies inspired by designs in nature. Since GAs are inspired by the mechanisms of biological evolution, it makes sense that they could be used in the process of invention as well. GAs rely primarily on something called implicit parallelism (like to like), using mutation and selection in secondary roles toward a design solution. GA programmers are working on applications that not only analyze the natural designs themselves for a return on how they work, but can also combine natural designs to create something entirely new that can have exciting applications.**

**8. Trip, Traffic and Shipment Routing**

**New applications of a GA known as the**[**“Traveling Salesman Problem”**](http://www.aip.org/isns/reports/2002/060.html)**or TSP can be used to plan the most efficient routes and scheduling for travel planners, traffic routers and even shipping companies. The shortest routes for traveling. The timing to avoid traffic tie-ups and rush hours. Most efficient use of transport for shipping, even to including pickup loads and deliveries along the way. The program can be modeling all this in the background while the human agents do other things, improving productivity as well! Chances are increasing steadily that when you get that trip plan packet from the travel agency, a GA contributed more to it than the agent did.**

**9. Computer Gaming**

**Those who spend some of their time playing computer Sims games (creating their own civilizations and evolving them) will often find themselves playing against**[**sophisticated artificial intelligence GAs**](https://learninggames.wordpress.com/2008/01/04/biologically-inspired-artificial-intelligence-for-computer-games/)**instead of against other human players online. These GAs have been programmed to incorporate the most successful strategies from previous games – the programs ‘learn’ – and usually incorporate data derived from game theory in their design.**[**Game theory**](https://en.wikipedia.org/wiki/Game_Theory)**is useful in most all GA applications for seeking solutions to whatever problems they are applied to, even if the application really is a game.**

**10. Encryption and Code Breaking**

**On the security front, GAs can be used both to**[**create encryption**](http://www.wipo.int/pctdb/en/wo.jsp?IA=US2000040826&DISPLAY=DESC)**for sensitive data as well as to break those codes. Encrypting data, protecting copyrights and breaking competitors’ codes have been important in the computer world ever since there have been computers, so the competition is intense. Every time someone adds more complexity to their encryption algorithms, someone else comes up with a GA that can break the code. It is hoped that one day soon we will have quantum computers that will be able to generate completely indecipherable codes. Of course, by then the ‘other guys’ will have quantum computers too, so it’s a sure bet the spy vs. spy games will go on indefinitely.**

**11. Computer-Aided Molecular Design**

**The de novo design of**[**new chemical molecules**](http://panizzi.shef.ac.uk/cisrg/links/ea_bib.html)**is a burgeoning field of applied chemistry in both industry and medicine. GAs are used to aid in the understanding of protein folding, analyzing the effects of substitutions on those protein functions, and to predict the binding affinities of various designed proteins developed by the pharmaceutical industry for treatment of particular diseases. The same sort of GA optimization and analysis is used for designing industrial chemicals for particular uses, and in both cases GAs can also be useful for predicting possible adverse consequences. This application has and will continue to have great impact on the costs associated with development of new chemicals and drugs.**

**12. Gene Expression Profiling**

**The development of microarray technology for taking ‘snapshots’ of the genes being expressed in a cell or group of cells has been a boon to medical research. GAs have been and are being developed to make**[**analysis of gene expression**](https://bioinformatics.oxfordjournals.org/cgi/content/abstract/17/12/1131)**profiles much quicker and easier. This helps to classify what genes play a part in various diseases, and further can help to identify genetic causes for the development of diseases. Being able to do this work quickly and efficiently will allow researchers to focus on individual patients’ unique genetic and gene expression profiles, enabling the hoped-for “personalized medicine” we’ve been hearing about for several years.**

**13. Optimizing Chemical Kinetic Analysis**

**In the not-so rarified realm of**[**fuels and engines for combustion technologies**](http://www.jstage.jst.go.jp/article/jsmeb/48/4/48_717/_article)**, GAs are proving very useful toward optimizing designs in transportation, aerospace propulsion and electrical generation. By being able to predict ahead of time the chemical kinetics of fuels and the efficiency of engines, more optimal mixtures and designs can be made available quicker to industry and the public. Some computer modeling applications in this area also simulate the effectiveness of lubricants and can pinpoint optimized operational vectors, and may lead to greatly increased efficiency all around well before traditional fuels run out.**

**14. Finance and Investment Strategies**

**In the current unprecedented world economic meltdown one might legitimately wonder if some of those Wall Street gamblers made use of GA-assisted computer modeling of**[**finance and investment strategies**](http://www.wiley.com/WileyCDA/WileyTitle/productCd-0471576794.html)**to funnel the world’s accumulated wealth into what can best be described as dot-dollar black holes. But then again, maybe they were simply all using the same prototype, which hadn’t yet been de-bugged. It is possible that a newer generation of**[**GA-assisted financial forecasting**](http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel5/6342/16979/00782581.pdf?arnumber=782581)**would have avoided the black holes and returned something other than bad debts the taxpayers get to repay. Who knows?**

**15. Marketing and Merchandising**

**We could think the word ‘merchandising’ just the way Mel Brooks said it in the “Space Balls” the movie. Space Balls the toilet paper. Space Balls the lunchbox. Space Balls the flame thrower (the kids love this one)… And laugh because it’s close enough to reality to be funny. So it shouldn’t surprise anyone that GAs are indeed being put to work to help merchandisers to produce products and marketing consultants**[**design advertising and direct solicitation campaigns**](http://www.businessweek.com/innovate/content/dec2008/id20081229_162381.htm)**to sell stuff. Maybe this application of GAs could someday get us out of the financial black hole and get things moving again.**

**Share Post**

**Recent Posts**

**Popular Articles**

**© Designed by DJMiMi**

In [computer science](https://en.wikipedia.org/wiki/Computer_science) and [operations research](https://en.wikipedia.org/wiki/Operations_research), a **genetic algorithm** (**GA**) is a [metaheuristic](https://en.wikipedia.org/wiki/Metaheuristic) inspired by the process of [natural selection](https://en.wikipedia.org/wiki/Natural_selection) that belongs to the larger class of [evolutionary algorithms](https://en.wikipedia.org/wiki/Evolutionary_algorithm) (EA). Genetic algorithms are commonly used to generate high-quality solutions to [optimization](https://en.wikipedia.org/wiki/Optimization_(mathematics)) and [search problems](https://en.wikipedia.org/wiki/Search_algorithm) by relying on biologically inspired operators such as [mutation](https://en.wikipedia.org/wiki/Mutation_(genetic_algorithm)), [crossover](https://en.wikipedia.org/wiki/Crossover_(genetic_algorithm)) and [selection](https://en.wikipedia.org/wiki/Selection_(genetic_algorithm)).[[1]](https://en.wikipedia.org/wiki/Genetic_algorithm#cite_note-FOOTNOTEMitchell19962-1) Some examples of GA applications include optimizing [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning) for better performance, solving [sudoku puzzles](https://en.wikipedia.org/wiki/Sudoku_solving_algorithms" \o "Sudoku solving algorithms),[[2]](https://en.wikipedia.org/wiki/Genetic_algorithm#cite_note-2) [hyperparameter optimization](https://en.wikipedia.org/wiki/Hyperparameter_optimization" \o "Hyperparameter optimization), etc.

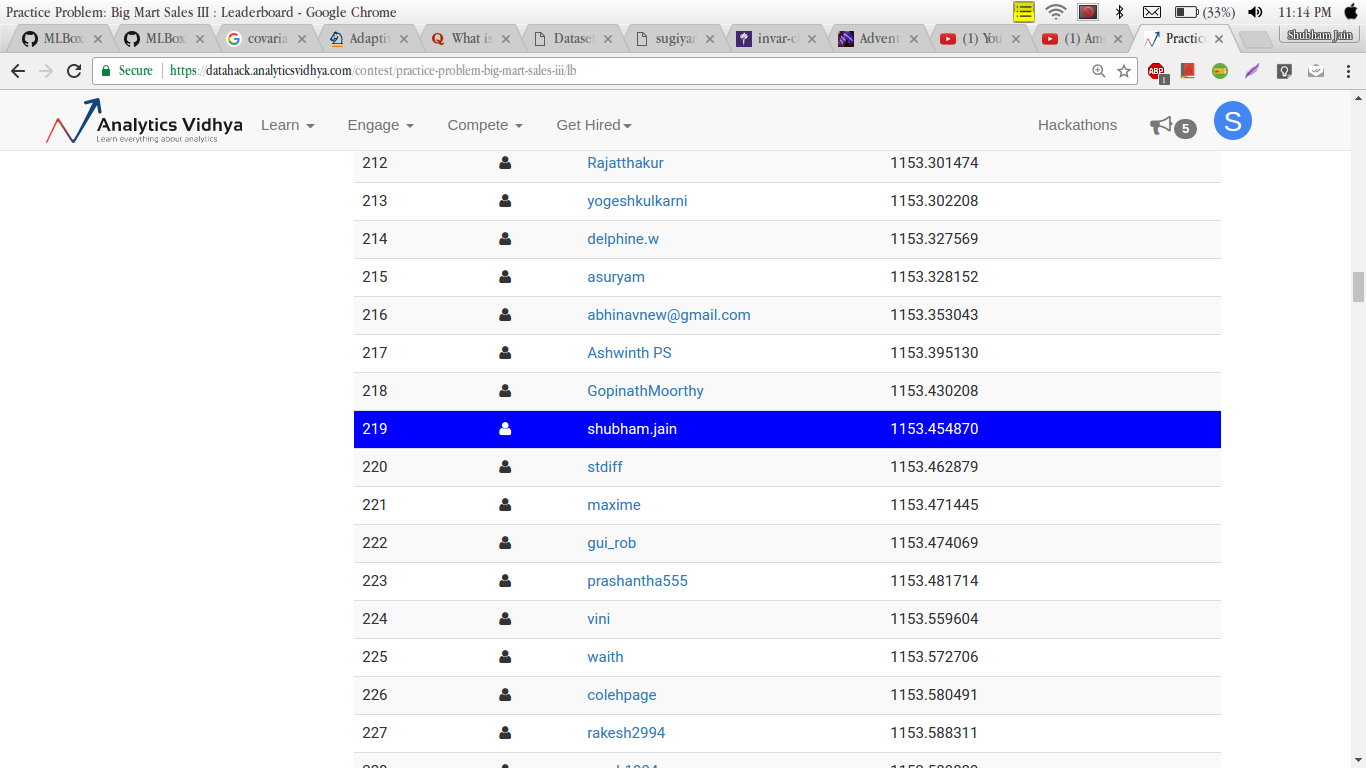
**Introduction to Genetic Algorithm & their application in data science**

[**Shubham.jain Jain**](https://www.analyticsvidhya.com/blog/author/shubham-jain/)**— July 31, 2017**

[Advanced](https://www.analyticsvidhya.com/blog/category/advanced/) [Algorithm](https://www.analyticsvidhya.com/blog/category/algorithm/) [Machine Learning](https://www.analyticsvidhya.com/blog/category/machine-learning/) [Python](https://www.analyticsvidhya.com/blog/category/python-2/) [Structured Data](https://www.analyticsvidhya.com/blog/category/structured-data/)

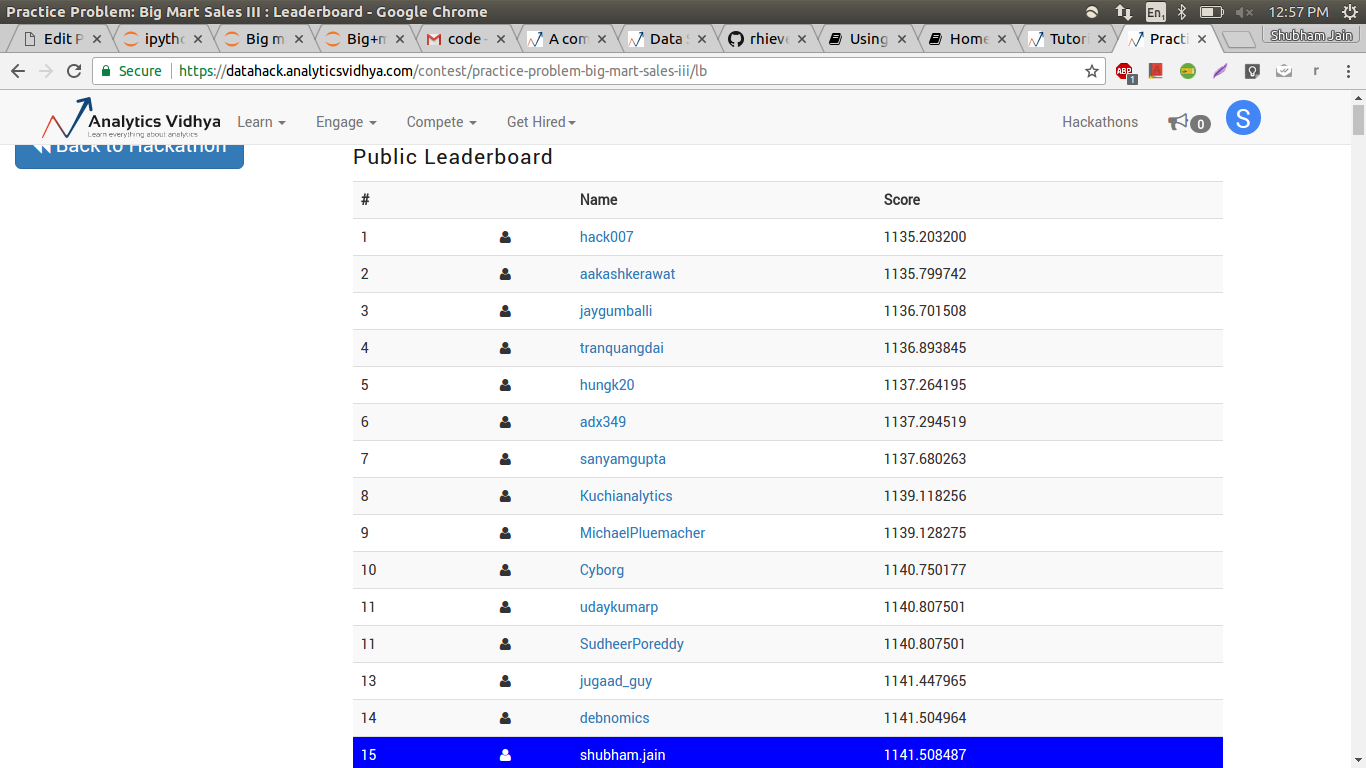
Introduction

Few days back, I started working on a practice problem – Big Mart Sales. After applying some simple models and doing some feature engineering, I landed up on 219th position on the leader board.



Not bad – but I needed something better.

So, I started searching for optimization techniques which could improve my score. It was during this search that I was introduced to genetic algorithms. After applying Genetric algorithm to the practice problem, I ended up taking a considerable leap on the leaderboard.





**Data Science Scholarship Week**

Get 50% Money Back on completion of the 50% Program within 1 year\* + Free Pass to 1729: An AI Event[Enroll Now](https://blackbelt.analyticsvidhya.com/plus?utm_source=blog&utm_medium=bannerbetweenarticle#BBTimmmer)

Yes, a jump from 219th to 15th position just on the basis on genetic algorithm. Isn’t that great? By end of this article, you will be comfortable applying genetic algorithms and can expect similar benefit on the problems you are working on.

Table of Content

1. Intuition behind Genetic Algorithms
2. Biological Inspiration
3. What is Genetic Algorithm?
4. Steps Involved in Genetic Algorithm
   1. Initialisation
   2. Fitness Function
   3. Selection
   4. Crossover
   5. Mutation
5. Application of Genetic Algorithm
   1. Feature Selection
   2. Implementation using TPOT library
6. Application in Real World
7. End Notes

1. Intuition behind Genetic Algorithms

Let’s start with the famous quote by Charles Darwin:

*It is not the strongest of the species that survives, nor the most intelligent , but the one most responsive to change.*

You must be thinking what has this quote got to do with genetic algorithm? Actually, the entire concept of a genetic algorithm is based on the above line.

Let us understand with a basic example:

Let’s take a hypothetical situation where, you are head of a country, and in order to keep your city safe from bad things, you implement a policy like this.

* You select all the good people, and ask them to extend their generation by having their children.
* This repeats for a few generations.
* You will notice that now you have an entire population of good people.

Now, that may not be entirely possible, but this example was just to help you understand the concept. So the basic idea was that we changed the input (i.e. population) such that we get better output (i.e. better country).

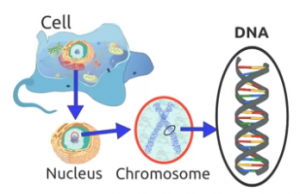
Now, I suppose you have got some intuition that the concept of a genetic algorithm is somewhat related to biology. So let’s us quickly grasp some little concepts, so that we can draw a parallel line between them.

2. Biological Inspiration

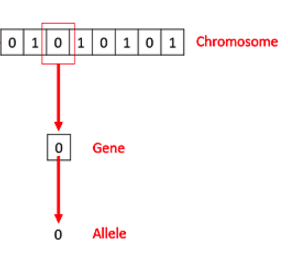
I am sure you would remember:

*Cells are the basic building block of all living things.*

Therefore in each cell, there is the same set of chromosomes. Chromosome are basically the strings of DNA.



Traditionally, these chromosomes are represented in binary as strings of 0’s and 1’s.



Source : [link](https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_fundamentals.htm)

A chromosome consists of genes, commonly referred as blocks of DNA, where each gene encodes a specific trait, for example hair color or eye color.

I wanted you to recall these basics concept of biology before going further. Let’s get back and understand what actually is a genetic algorithm?

3. What is a Genetic Algorithm?

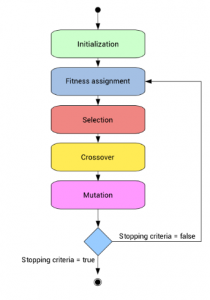
Let’s get back to the example we discussed above and summarize what we did.

1. Firstly, we defined our initial population as our countrymen.
2. We defined a function to classify whether is a person is good or bad.
3. Then we selected good people for mating to produce their off-springs.
4. And finally, these off-springs replace the bad people from the population and this process repeats.

This is how genetic algorithm actually works, which basically tries to mimic the human evolution to some extent.

So to formalize a definition of a genetic algorithm, we can say that it is an optimization technique, which tries to find out such values of input so that we get the best output values or results.

The working of a genetic algorithm is also derived from biology, which is as shown in the image below.



Source: [link](https://www.neuraldesigner.com/blog/genetic_algorithms_for_feature_selection)

So, let us try to understand the steps one by one.

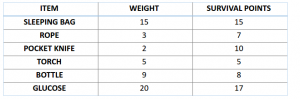
**4. Steps Involved in Genetic Algorithm**

Here, to make things easier, let us understand it by the famous [Knapsack problem](https://en.wikipedia.org/wiki/Knapsack_problem).

If you haven’t come across this problem, let me introduce my version of this problem.

Let’s say, you are going to spend a month in the wilderness. Only thing you are carrying is the backpack which can hold a maximum weight of **30 kg**. Now you have different survival items, each having its own “Survival Points” (which are given for each item in the table). So, your objective is maximise the survival points.

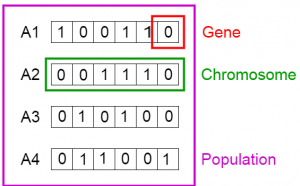
Here is the table giving details about each item.



4.1 Initialisation

To solve this problem using genetic algorithm, our first step would be defining our population. So our population will contain individuals, each having their own set of chromosomes.

We know that, chromosomes are binary strings, where for this problem 1 would mean that the following item is taken and 0 meaning that it is dropped.

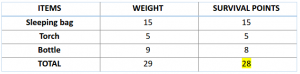


This set of chromosome is considered as our initial population.

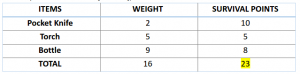
4.2 Fitness Function

Let us calculate fitness points for our first two chromosomes.

For A1 chromosome [100110],



Similarly for A2 chromosome [001110],



So, for this problem, our chromosome will be considered as more fit when it contains more survival points.

Therefore chromosome 1 is more fit than chromosome 2.

4.3 Selection

Now, we can select fit chromosomes from our population which can mate and create their off-springs.

General thought is that we should select the fit chromosomes and allow them to produce off-springs. But that would lead to chromosomes that are more close to one another in a few next generation, and therefore less diversity.

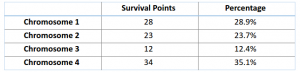
Therefore, we generally use Roulette Wheel Selection method.

Don’t be afraid of name, just take a look at the image below.

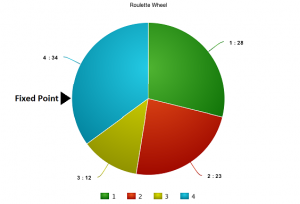


I suppose we all have seen this, either in real or in movies. So, let’s build our roulette wheel.

Consider a wheel, and let’s divide that into m divisions, where m is the number of chromosomes in our populations. The area occupied by each chromosome will be proportional to its fitness value.

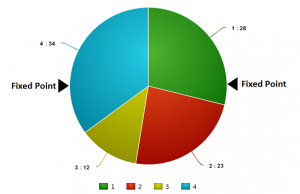


Based on these values, let us create our roulette wheel.



So, now this wheel is rotated and the region of wheel which comes in front of the fixed point is chosen as the parent. For the second parent, the same process is repeated.

Sometimes we mark two fixed point as shown in the figure below.

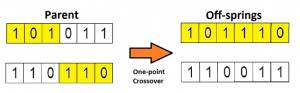


So, in this method we can get both our parents in one go. This method is known as Stochastic Universal Selection method.

4.4 Crossover

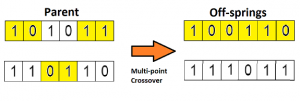
So in this previous step, we have selected parent chromosomes that will produce off-springs. So in biological terms, crossover is nothing but reproduction.

So let us find the crossover of chromosome 1 and 4, which were selected in the previous step. Take a look at the image below.



This is the most basic form of crossover, known as one point crossover. Here we select a random crossover point and the tails of both the chromosomes are swapped to produce a new off-springs.

If you take two crossover point, then it will called as multi point crossover which is as shown below.



4.5 Mutation

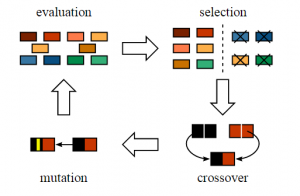
Now if you think in the biological sense, are the children produced have the same traits as their parents? The answer is NO. During their growth, there is some change in the genes of children which makes them different from its parents.

This process is known as mutation, which may be defined as a random tweak in the chromosome, which also promotes the idea of diversity in the population.

A simple method of mutation is shown in the image below.

https://cdn.analyticsvidhya.com/wp-content/uploads/2017/07/22174928/mutation-300x56.png

So the entire process is summarise as shown in the figure.



Source : [link](http://www.jade-cheng.com/au/coalhmm/optimization/)

The off-springs thus produced are again validated using our fitness function, and if considered fit then will replace the less fit chromosomes from the population.

But the question is how we will get to know that we have reached our best possible solution?

So basically there are different termination conditions, which are listed below:

1. There is no improvement in the population for over x iterations.
2. We have already predefined an absolute number of generation for our algorithm.
3. When our fitness function has reached a predefined value.

Now, I suppose you have grasp the basic understanding of the genetic algorithm. So now let us look at some of the application of genetic algorithm in data science.

5. Application of Genetic Algorithm

5.1 Feature Selection

Every time you participate in a data science competition, how do you select features that are important in prediction of the target variable? You always look at the feature importance of some model, and then manually decide the threshold, and select the features which have importance above that threshold.

Is there any better way to deal with this kind of situations? Actually one of the most advanced algorithms for feature selection is genetic algorithm.

The method here is completely same as the one we did with the knapsack problem.

We will again start with the population of chromosome, where each chromosome will be binary string. 1 will denote “inclusion” of feature in model and 0 will denote “exclusion” of feature in the model.

And another difference would be that the fitness function would be changed. The fitness function here will be our accuracy metric of the competition. The more accurate our set of chromosome in predicting value, the more fit it will be.

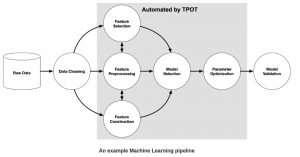
I suppose, you would now be thinking is there any use of such tough tasks. I will not answer this question now, rather let us look at the implementation of it using TPOT library and then you decide this.

5.2 Implementation using TPOT library

So finally, here the comes the part for which you have been waiting from the beginning of this article.

First, let’s take a quick view on the TPOT (Tree-based Pipeline Optimisation Technique) which is build upon scikit-learn library.

A basic pipeline structure is shown in the image below.



So the highlighted grey section in the image above is automated using TPOT. This automation is achieved using genetic algorithm.

So, without going deep into this, let’s directly try to implement it.

For using TPOT library, you first have to install some existing python libraries on which TPOT is build. So let us quickly install them.

# installing DEAP, update\_checker and tqdm

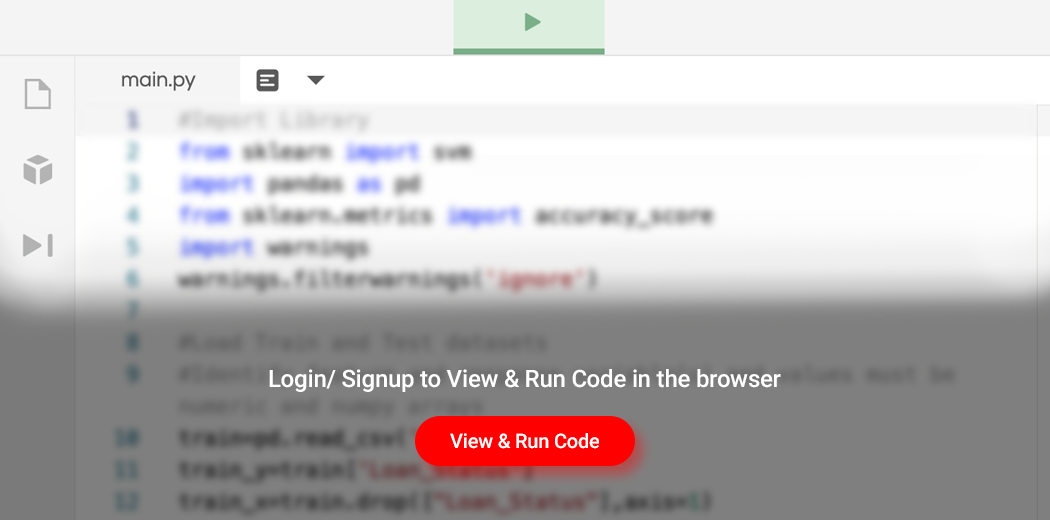
pip **install** deap update\_checker tqdm

# installling TPOT

pip **install** tpot

For the implementation part, here I have used [Big Mart Sales](https://datahack.analyticsvidhya.com/contest/practice-problem-big-mart-sales-iii/) dataset. So quickly download the train and test file.

Now let’s look at its python code.

[](https://id.analyticsvidhya.com/auth/login/?next=https://www.analyticsvidhya.com/blog/2017/07/introduction-to-genetic-algorithm/?&utm_source=coding-window-blog&source=coding-window-blog)



Once this code finishes running, tpot\_exported\_pipeline.py will contain the Python code for the optimised pipeline. We can see that ExtraTreeRegressor worked best for this problem.

## predicting using tpot optimised pipeline

tpot\_pred = tpot.predict(tpot\_test)

sub1 = pd.DataFrame(data=tpot\_pred)

#sub1.index = np.arange(0, len(test)+1)

sub1 = sub1.rename(columns = {'0':'Item\_Outlet\_Sales'})

sub1['Item\_Identifier'] = test['Item\_Identifier']

sub1['Outlet\_Identifier'] = test['Outlet\_Identifier']

sub1.columns = ['Item\_Outlet\_Sales','Item\_Identifier','Outlet\_Identifier']

sub1 = sub1[['Item\_Identifier','Outlet\_Identifier','Item\_Outlet\_Sales']]

sub1.to\_csv('tpot.csv',index=False)

If you submit this csv, you will notice that what I promised in the start has not been fulfilled. Was I lying to make you study all of these?

No, actually there is a simple rule of TPOT library, if you don’t run TPOT for very long, then it may not find the best possible pipeline for your problem.

So, increase the number of generations, grab a cup of coffee and go out for a walk. TPOT will finish your work.

You can also do classification problems with this library. For more, I would suggest you to once check out its [documentation](http://rhiever.github.io/tpot/).

Besides competitions, genetic algorithm also have many applications in the real world.

6. Applications in Real World

Genetic algorithm has many applications in real world. Here I have listed some of the interesting application, but explaining each one of them will require me an extra article.

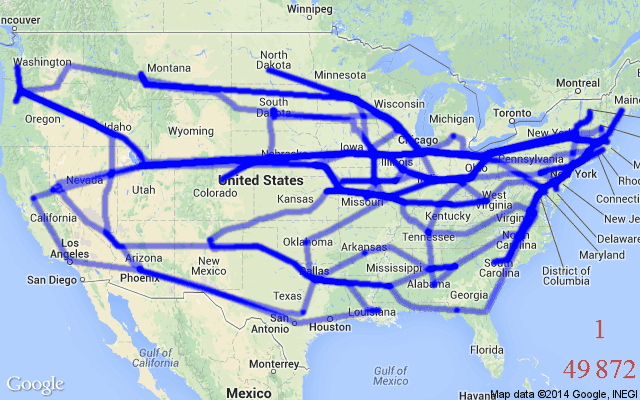
6.1 Engineering Design

Engineering design has relied heavily on computer modeling and simulation to make design cycle process fast and economical. Genetic algorithm has been used to optimize and provide a robust solution.

Resources: [link](http://lib.dr.iastate.edu/cgi/viewcontent.cgi?article=16942&context=rtd)

6.2 Traffic and Shipment Routing (Travelling Salesman Problem)

This is a famous problem and has been efficiently adopted by many sales-based companies as it is time saving and economical. This is also achieved using genetic algorithm.



Source: [link](http://mathgifs.blogspot.com/2014/03/the-traveling-salesman.html)

6.3 Robotics

The use of genetic algorithm in the field of robotics is quite big. Actually, genetic algorithm is being used to create learning robots which will behave as a human and will do tasks like cooking our meal, do our laundry etc.

Resources: [link](https://pdfs.semanticscholar.org/7c8c/faa78795bcba8e72cd56f8b8e3b95c0df20c.pdf)

Now after these I suppose, you must have developed enough curiosity to look out for some more other interesting applications of genetic algorithms. Also you can comment down if you want to share that with us.

7. End Notes

I hope that now you have gain enough understanding about what genetic algorithm is and also how to implement it using TPOT library. But this knowledge is not enough, if you don’t apply it somewhere.

So try to implement it whether in any real world application or in a data science competition. If you face any difficulties, feel free to write on our [discussion](https://discuss.analyticsvidhya.com/) portal.

Did you find this article helpful? Please share your opinions / thoughts in the comments section below.

List of genetic algorithm applications

From Wikipedia, the free encyclopedia

[Jump to navigation](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#mw-head)[Jump to search](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#searchInput)

This is a list of [**genetic algorithm**](https://en.wikipedia.org/wiki/Genetic_algorithm)**(GA) applications**.



**Contents**

* [1Natural Sciences, Mathematics and Computer Science](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#Natural_Sciences,_Mathematics_and_Computer_Science)
* [2Earth Sciences](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#Earth_Sciences)
* [3Finance and Economics](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#Finance_and_Economics)
* [4Social Sciences](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#Social_Sciences)
* [5Industry, Management and Engineering](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#Industry,_Management_and_Engineering)
* [6Biological Sciences and Bioinformatics](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#Biological_Sciences_and_Bioinformatics)
* [7General Applications](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#General_Applications)
* [8Physics](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#Physics)
* [9Other Applications](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#Other_Applications)
* [10References](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#References)

Natural Sciences, Mathematics and Computer Science[[edit](https://en.wikipedia.org/w/index.php?title=List_of_genetic_algorithm_applications&action=edit&section=1" \o "Edit section: Natural Sciences, Mathematics and Computer Science)]

* Bayesian inference links to particle methods in Bayesian statistics and hidden Markov chain models[[1]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-1)[[2]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-hal.inria.fr-2)
* [Artificial creativity](https://en.wikipedia.org/wiki/Computational_creativity)
* Chemical kinetics ([gas](https://archive.today/20121223015305/http:/www.personal.leeds.ac.uk/~fuensm/project.html) and [solid](http://repositories.cdlib.org/postprints/1154) phases)
* Calculation of [bound states](https://en.wikipedia.org/wiki/Bound_state) and [local-density approximations](https://en.wikipedia.org/wiki/Local-density_approximation)
* [Code-breaking](https://en.wikipedia.org/wiki/Code-breaking), using the GA to search large solution spaces of [ciphers](https://en.wikipedia.org/wiki/Cipher) for the one correct decryption.[[3]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-3)
* Computer architecture: using GA to find out weak links in [approximate computing](https://en.wikipedia.org/wiki/Approximate_computing) such as [lookahead](https://en.wikipedia.org/wiki/Combinatorial_search" \l "Lookahead" \o "Combinatorial search).
* Configuration applications, particularly physics applications of optimal molecule configurations for particular systems like C60 ([buckyballs](https://en.wikipedia.org/wiki/Fullerene" \o "Fullerene))
* Construction of [facial composites](https://en.wikipedia.org/wiki/Facial_composite) of suspects by [eyewitnesses](https://en.wikipedia.org/wiki/Witness) in forensic science.[[4]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-4)
* Data Center/Server Farm.[[5]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-5)
* [Distributed computer network](https://en.wikipedia.org/w/index.php?title=Distributed_computer_network&action=edit&redlink=1) [topologies](https://en.wikipedia.org/wiki/Topologies)
* Electronic circuit design, known as [evolvable hardware](https://en.wikipedia.org/wiki/Evolvable_hardware)
* [Feature selection](https://en.wikipedia.org/wiki/Feature_selection) for [Machine Learning](https://en.wikipedia.org/wiki/Machine_learning)[[6]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-6)
* Feynman-Kac models [[7]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-7)[[8]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-8)[[9]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-9)
* File allocation for a [distributed system](https://en.wikipedia.org/wiki/Distributed_system)
* Filtering and signal processing [[10]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-10)[[11]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-11)
* Finding hardware bugs.[[12]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-12)[[13]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-13)
* [Game theory](https://en.wikipedia.org/wiki/Game_theory) equilibrium resolution
* [Genetic Algorithm for Rule Set Production](https://en.wikipedia.org/wiki/Genetic_Algorithm_for_Rule_Set_Production)
* [Scheduling applications](https://en.wikipedia.org/wiki/Genetic_algorithm_scheduling), including [job-shop scheduling](https://en.wikipedia.org/wiki/Job_Shop_Scheduling) and scheduling in [printed circuit board](https://en.wikipedia.org/wiki/Printed_circuit_board) assembly.[[14]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-PCB-14) The objective being to schedule jobs in a [sequence-dependent](https://en.wikipedia.org/w/index.php?title=Sequence-dependent_setup&action=edit&redlink=1) or non-sequence-dependent setup environment in order to maximize the volume of production while minimizing penalties such as tardiness. Satellite communication scheduling for the NASA Deep Space Network was shown to benefit from genetic algorithms.[[15]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-15)
* [Learning robot behavior](https://en.wikipedia.org/wiki/Robot_learning) using genetic algorithms
* Image processing: Dense pixel matching[[16]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-16)
* Learning fuzzy rule base using genetic algorithms
* Molecular structure optimization (chemistry)
* Optimisation of data compression systems, for example using [wavelets](https://en.wikipedia.org/wiki/Wavelet).
* [Power electronics](https://en.wikipedia.org/wiki/Power_electronics) design.[[17]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-17)
* [Traveling salesman problem](https://en.wikipedia.org/wiki/Traveling_salesman_problem) and its applications[[14]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-PCB-14)

Earth Sciences[[edit](https://en.wikipedia.org/w/index.php?title=List_of_genetic_algorithm_applications&action=edit&section=2" \o "Edit section: Earth Sciences)]

* [Climatology](https://en.wikipedia.org/wiki/Climatology): Estimation of [heat flux](https://en.wikipedia.org/wiki/Heat_flux) between the atmosphere and sea ice[[18]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-18)
* [Climatology](https://en.wikipedia.org/wiki/Climatology): Modelling [global temperature](https://en.wikipedia.org/wiki/Temperature_record) changes[[19]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-19)
* Design of [water resource](https://en.wikipedia.org/wiki/Water_resource) systems [[20]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Zhang_&_Babovic_2012-20)
* Groundwater monitoring networks[[21]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-21)

Finance and Economics[[edit](https://en.wikipedia.org/w/index.php?title=List_of_genetic_algorithm_applications&action=edit&section=3" \o "Edit section: Finance and Economics)]

* [Financial mathematics](https://en.wikipedia.org/wiki/Financial_mathematics)[[2]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-hal.inria.fr-2)[[22]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-22)
  + Automated design of sophisticated trading systems in the financial sector; see [Automated trading system](https://en.wikipedia.org/wiki/Automated_trading_system)
  + [Real options valuation](https://en.wikipedia.org/wiki/Real_options_valuation)[[23]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Zhang_&_Babovic-23)
  + [Portfolio optimization](https://en.wikipedia.org/wiki/Portfolio_optimization)[[24]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-24)
* [Genetic algorithm in economics](https://en.wikipedia.org/wiki/Genetic_algorithm_in_economics)
  + Representing rational agents in economic models such as the [cobweb model](https://en.wikipedia.org/wiki/Cobweb_model)
  + the same, in [Agent-based computational economics](https://en.wikipedia.org/wiki/Agent-based_computational_economics) generally

Social Sciences[[edit](https://en.wikipedia.org/w/index.php?title=List_of_genetic_algorithm_applications&action=edit&section=4" \o "Edit section: Social Sciences)]

* Design of [anti-terrorism](https://en.wikipedia.org/wiki/Anti-terrorism) systems [[25]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Buurman,_Zhang_&_Babovic-25)
* Linguistic analysis, including [grammar induction](https://en.wikipedia.org/wiki/Grammar_induction) and other aspects of [Natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing) (NLP) such as word sense disambiguation.

Industry, Management and Engineering[[edit](https://en.wikipedia.org/w/index.php?title=List_of_genetic_algorithm_applications&action=edit&section=5" \o "Edit section: Industry, Management and Engineering)]

* [Audio watermark](https://en.wikipedia.org/wiki/Audio_watermark) insertion/detection
* Airlines revenue management[[26]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-26)
* Automated design of [mechatronic](https://en.wikipedia.org/wiki/Mechatronics) systems using [bond graphs](https://en.wikipedia.org/wiki/Bond_graphs) and [genetic programming](https://en.wikipedia.org/wiki/Genetic_programming) (NSF)
* Automated design of industrial equipment using catalogs of exemplar lever patterns
* [Automated](https://en.wikipedia.org/wiki/Automated) design, including research on [composite material](https://en.wikipedia.org/wiki/Composite_material) design and [multi-objective](https://en.wikipedia.org/w/index.php?title=Multi-objective&action=edit&redlink=1) design of automotive components for [crashworthiness](https://en.wikipedia.org/wiki/Crashworthiness), weight savings, and other characteristics
* [Automated planning](https://en.wikipedia.org/wiki/Automated_planning) of structural inspection[[27]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-EllefsenLepikson2017-27)
* Container loading optimization
* [Control engineering](https://en.wikipedia.org/wiki/Control_engineering),[[28]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-28)[[29]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-29)[[30]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-30)[[31]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Patrascu2015-31)
* [Marketing mix](https://en.wikipedia.org/wiki/Marketing_mix) analysis
* [Mechanical engineering](https://en.wikipedia.org/wiki/Mechanical_engineering)[[32]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-iit_kanpur-32)[[33]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-33)
* Mobile communications infrastructure [optimization](https://en.wikipedia.org/wiki/Optimization_(mathematics)).
* [Plant floor layout](https://en.wikipedia.org/w/index.php?title=Plant_floor_layout&action=edit&redlink=1)
* [Pop music](https://en.wikipedia.org/wiki/Pop_music) record production[[34]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-34)
* [Quality control](https://en.wikipedia.org/wiki/Quality_control_and_genetic_algorithms)
* [Sorting network](https://en.wikipedia.org/wiki/Sorting_network)
* Timetabling problems, such as designing a non-conflicting class timetable for a large university
* [Vehicle routing problem](https://en.wikipedia.org/wiki/Vehicle_routing_problem) [[35]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-35)
* Optimal bearing placement [[36]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-36)
* [Computer-automated design](https://en.wikipedia.org/wiki/Computer-automated_design)[[37]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-37)

Biological Sciences and Bioinformatics[[edit](https://en.wikipedia.org/w/index.php?title=List_of_genetic_algorithm_applications&action=edit&section=6" \o "Edit section: Biological Sciences and Bioinformatics)]

* [Bioinformatics](https://en.wikipedia.org/wiki/Bioinformatics) [Multiple Sequence Alignment](https://en.wikipedia.org/wiki/Multiple_Sequence_Alignment)[[38]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Gondro-38)[[39]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Notredame-39)[[40]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-40)
* [Bioinformatics](https://en.wikipedia.org/wiki/Bioinformatics): [RNA](https://en.wikipedia.org/wiki/RNA) structure prediction[[41]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Batenburg-41)
* [Bioinformatics](https://en.wikipedia.org/wiki/Bioinformatics): [Motif Discovery](https://en.wikipedia.org/w/index.php?title=Motif_Discovery&action=edit&redlink=1)[[42]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-42)
* Biology and [computational chemistry](https://en.wikipedia.org/wiki/Computational_chemistry)[[43]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-43)[[44]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-44)
* Building [phylogenetic trees](https://en.wikipedia.org/wiki/Phylogenetic_tree).[[45]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Hill-45)
* [Gene expression profiling](https://en.wikipedia.org/wiki/Expression_profiling) analysis.[[46]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-To-46)
* [Medicine](https://en.wikipedia.org/wiki/Medicine): [Clinical decision support](https://en.wikipedia.org/wiki/Clinical_decision_support) in ophthalmology[[47]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-47) and oncology[[48]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-48)
* [Computational Neuroscience](https://en.wikipedia.org/wiki/Computational_Neuroscience): finding values for the maximal conductances of ion channels in biophysically detailed neuron models[[49]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-49)
* [Protein folding](https://en.wikipedia.org/wiki/Protein_folding) and protein/[ligand docking](https://en.wikipedia.org/wiki/Ligand_docking)[[50]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Willet-50)[[51]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-51)
* Selection of optimal mathematical model to describe biological systems
* [Operon](https://en.wikipedia.org/wiki/Operon) prediction.[[52]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Wang-52)

General Applications[[edit](https://en.wikipedia.org/w/index.php?title=List_of_genetic_algorithm_applications&action=edit&section=7" \o "Edit section: General Applications)]

* [Neural Networks](https://en.wikipedia.org/wiki/Neural_network); particularly [recurrent neural networks](https://en.wikipedia.org/wiki/Recurrent_neural_networks)[[53]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-53)
* Training [artificial neural networks](https://en.wikipedia.org/wiki/Artificial_neural_networks) when pre-classified training examples are not readily obtainable ([neuroevolution](https://en.wikipedia.org/wiki/Neuroevolution" \o "Neuroevolution))

Physics[[edit](https://en.wikipedia.org/w/index.php?title=List_of_genetic_algorithm_applications&action=edit&section=8" \o "Edit section: Physics)]

* Optimization of beam dynamics in [accelerator physics](https://en.wikipedia.org/wiki/Accelerator_physics).[[54]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-54)
* Design of [particle accelerator](https://en.wikipedia.org/wiki/Particle_accelerator) beamlines [[55]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-55)

Other Applications[[edit](https://en.wikipedia.org/w/index.php?title=List_of_genetic_algorithm_applications&action=edit&section=9" \o "Edit section: Other Applications)]

* Clustering, using genetic algorithms to optimize a wide range of different fit-functions.[[*dead link*](https://en.wikipedia.org/wiki/Wikipedia:Link_rot)][[56]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-56)
* [Multidimensional systems](https://en.wikipedia.org/wiki/Multidimensional_systems)
* Multimodal Optimization [[57]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-57)[[58]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-58)[[59]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-59)
* Multiple criteria production scheduling[[60]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-Bagchi-60)
* Multiple population [topologies](https://en.wikipedia.org/wiki/Topologies) and interchange [methodologies](https://en.wikipedia.org/wiki/Methodologies)
* [Mutation testing](https://en.wikipedia.org/wiki/Mutation_testing)
* [Parallelization](https://en.wikipedia.org/wiki/Parallelization) of GAs/GPs including use of [hierarchical decomposition](https://en.wikipedia.org/w/index.php?title=Hierarchical_decomposition&action=edit&redlink=1) of [problem domains](https://en.wikipedia.org/w/index.php?title=Problem_domains&action=edit&redlink=1) and design spaces [nesting of irregular shapes](https://en.wikipedia.org/wiki/Nesting_problem_(geometric_optimization)) using [feature matching](https://en.wikipedia.org/wiki/Feature_matching) and GAs.
* Rare event analysis [[61]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-61)[[62]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-62)
* Solving the machine-component grouping problem required for [cellular manufacturing](https://en.wikipedia.org/wiki/Cellular_manufacturing) systems
* Stochastic optimization [[63]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-63)
* [Tactical asset](https://en.wikipedia.org/w/index.php?title=Tactical_asset&action=edit&redlink=1) allocation and [international equity](https://en.wikipedia.org/w/index.php?title=International_equity&action=edit&redlink=1) strategies
* Wireless sensor/ad-hoc networks.[[64]](https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications#cite_note-64)

Application of ant colony and immune combined optimization algorithm in path planning of unmanned craft <https://aip.scitation.org/doi/10.1063/5.0077858>

[ScientificWorldJournal.](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3818646/) 2013; 2013: 528069.

Published online 2013 Oct 10. doi: [10.1155/2013/528069](https://doi.org/10.1155%2F2013%2F528069)

PMCID: PMC3818646

PMID: [24222739](https://pubmed.ncbi.nlm.nih.gov/24222739)

Swarm Intelligence and Its Applications

[Yudong Zhang](https://pubmed.ncbi.nlm.nih.gov/?term=Zhang%20Y%5BAuthor%5D),1 ,\* [Praveen Agarwal](https://pubmed.ncbi.nlm.nih.gov/?term=Agarwal%20P%5BAuthor%5D),2[Vishal Bhatnagar](https://pubmed.ncbi.nlm.nih.gov/?term=Bhatnagar%20V%5BAuthor%5D),3[Saeed Balochian](https://pubmed.ncbi.nlm.nih.gov/?term=Balochian%20S%5BAuthor%5D),4and [Jie Yan](https://pubmed.ncbi.nlm.nih.gov/?term=Yan%20J%5BAuthor%5D)5

[Author information](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3818646/) [Article notes](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3818646/) [Copyright and License information](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3818646/) [Disclaimer](https://www.ncbi.nlm.nih.gov/pmc/about/disclaimer/)

This article has been [cited by](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3818646/citedby/) other articles in PMC.

Swarm intelligence (SI) is the collective behavior of decentralized, self-organized systems, natural or artificial. SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment. The inspiration often comes from nature, especially biological systems. The agents follow very simple rules, and although there is no centralized control structure dictating how individual agents should behave, local, and to a certain degree random, interactions between such agents lead to the emergence of “intelligent” global behavior, unknown to the individual agents. Natural examples of SI include ant colonies, bird flocking, animal herding, bacterial growth, and fish schooling.

Research in SI started in the late 1980s. Besides the applications to conventional optimization problems, SI can be employed in library materials acquisition, communications, medical dataset classification, dynamic control, heating system planning, moving objects tracking, and prediction. Indeed, SI can be applied to a variety of fields in fundamental research, engineering, industries, and social sciences.

The main objective of this special issue is to provide the readers with a collection of high quality research articles that address the broad challenges in application aspects of swarm intelligence and reflect the emerging trends in state-of-the-art algorithms.

The special issue received 42 high quality submissions from different countries all over the world. All submitted papers followed the same standard (peer-reviewed by at least three independent reviewers) as applied to regular submissions to “this journal”. Due to the limited space, 15 papers were finally included. The primary guideline was to demonstrate the wide scope of SI algorithms and applications in various aspects. Besides, mathematically oriented papers with promising potential in practical problems were also included.

The paper authored by Y.-L. Wu et al. (National Chiao Tung University and Ming Chuan University) presents an integer programming model of the studied problem by considering how to select materials in order to maximize the average preference and the budget execution rate under some practical restrictions including departmental budget and limitation of the number of materials in each category and each language. They propose a discrete particle swarm optimization (DPSO) with scout particles, design an initialization algorithm and a penalty function to cope with the constraints, and employ the scout particles to enhance the exploration within the solution space.

In the paper by Z. Yin et al. (Harbin Institute of Technology), they propose an efficient multiuser detector based on a suboptimal code mapping multiuser detector and artificial bee colony algorithm (SCM-ABC-MUD) and implement the proposed algorithm in direct-sequence ultrawideband (DS-UWB) systems under the additive white Gaussian noise (AWGN) channel.

M. S. Uzer et al. (Selçuk University) offer a hybrid approach that uses the artificial bee colony (ABC) algorithm for feature selection and support vector machines for classification. For the diagnosis of hepatitis, liver disorders, and diabetes datasets from the UCI database, the proposed system reached classification accuracies of 94.92%, 74.81%, and 79.29%, respectively.

Another paper is by M. Karakose (Fırat University) and U. Cigdem (Gaziosmanpaşa University). It proposes a new approach for improvement of DNA computing with adaptive parameters towards the desired goal using quantum-behaved particle swarm optimization (QPSO). Experimental results obtained with MATLAB and FPGA demonstrate ability to provide effective optimization, considerable convergence speed, and high accuracy according to DNA computing algorithm.

In the paper by Y. Celik (Karamanoglu Mehmetbey University) and E. Ulker (Selcuk University), their research proposes an improved marriage in honey bees optimization (IMBO) by adding Levy flight algorithm for queen mating flight and neighboring for worker drone improving. The IMBO algorithm's performance and its success are tested on the well-known six unconstrained test functions and compared with other metaheuristic optimization algorithms.

M. Baygin (Ardahan University) and M. Karakose (Fırat University) study a new approach of immune system-based optimal estimate for dynamic control of group elevator systems. The method is mainly based on estimation of optimal way by optimizing all calls with genetic, immune system and DNA computing algorithms, and it is evaluated with a fuzzy system. With dynamic and adaptive control approach in this study, a significant progress on group elevator control systems has been achieved in terms of time and energy efficiency according to traditional methods.

The paper by M. Karakose (Fırat University) proposes a reinforcement-learning based artificial immune classifier. The proposed new approach has many contributions according to other methods in the literature such as effectiveness, less memory cell, high accuracy, speed, and data adaptability. Some benchmark data and remote image data are used for experimental results. The comparative results with supervised/unsupervised based artificial immune system, negative selection classifier, and resource limited artificial immune classifier are given to demonstrate the effectiveness of the proposed new method.

In their paper, T. J. Choi et al. (Sungkyunkwan University) and (Daegu Gyeongbuk Institute of Science and Technology) present an adaptive parameter control DE algorithm. The control parameters of each individual are adapted based on the average of successfully evolved individuals' parameter values using the Cauchy distribution. The experimental results show that their proposed algorithm is more robust than the standard DE algorithm and several state-of-the-art adaptive DE algorithms in solving various unimodal and multimodal problems.

In the paper by R.-J. Ma et al. (Southwest Jiaotong University and CSR Qishuyan Institute Co., Ltd.), the authors present an integral mathematical model and particle swarm optimization (PSO) algorithm based on the life cycle cost (LCC) approach for the heating system planning (HSP) problem. The results show that the improved particle swarm optimization (IPSO) algorithm can more preferably solve the HSP problem than PSO algorithm.

In the paper by M. Tang et al. (National University of Defense Technology and Université Pierre et Marie Curie), they report that the flocking has some negative effects on the human, as the infectious disease H7N9 will easily be transmitted from the denser flocking birds to the human. Their paper focuses on the H7N9 virus transmission in the flocking birds and from the flocking birds to the human. Some interesting results have been shown: (1) only some simple rules could result in an emergence such as the flocking; (2) the minimum distance between birds could affect H7N9 virus transmission in the flocking birds and even affect the virus transmissions from the flocking birds to the human.

Y. Wang et al. (China University of Petroleum) present a memory-based multiagent coevolution algorithm for robust tracking the moving objects. Each agent can remember, retrieve, or forget the appearance of the object through its own memory system by its own experience. Experimental results show that their proposed method can deal with large appearance changes and heavy occlusions when tracking a moving object.

The paper by Q. Ni and J. Deng (Southeast University and Soochow University) analyzes the performance of PSO with the proposed random topologies and explores the relationship between population topology and the performance of PSO from the perspective of graph theory characteristics in population topologies. Further, in a relatively new PSO variant which named logistic dynamic particle optimization, an extensive simulation study is presented to discuss the effectiveness of the random topology and the design strategies of population topology.

Y. Zhou and H. Zheng (Guangxi University for Nationalities, Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis) propose a novel complex valued cuckoo search algorithm. They use complex-valued encoding to expand the information of nest individuals and denote the gene of individuals by plurality. The value of independent variables for objective function is determined by modules, and a sign of them is determined by angles. The position of nest is divided into real part gene and imaginary gene. Six typical functions are tested, and the usefulness of the proposed algorithm is verified.

The paper by R. Alwee et al. (Universiti Teknologi Malaysia) introduces a hybrid model that combines support vector regression (SVR) and autoregressive integrated moving average (ARIMA) to be applied in crime rates forecasting. Particle swarm optimization is used to estimate the parameters of the SVR and ARIMA models. The experimental results show that their proposed hybrid model is able to produce more accurate forecasting results as compared to the individual models.

Finally, K. S. Lim et al. (Universiti Teknologi Malaysia, Universiti Malaysia Pahang, and University of Malaya) describe an improved Vector Evaluated Particle Swarm Optimization algorithm by incorporating the nondominated solutions as the guidance for a swarm rather than using the best solution from another swarm. The results suggest that the improved Vector Evaluated Particle Swarm Optimization algorithm has impressive performance compared with the conventional Vector Evaluated Particle Swarm Optimization algorithm.

[Go to:](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3818646/)

Aknowledgments

We would like to express their gratitude to all of the authors for their contributions, and the reviewers for their effort providing valuable comments and feedback. We hope this special issue offers a comprehensive and timely view of the area of applications of swarm intelligence and that it will offer stimulation for further research.

*Yudong Zhang* *Yudong Zhang*

*Praveen Agarwal* *Praveen Agarwal*

*Vishal Bhatnagar* *Vishal Bhatnagar*

*Saeed Balochian* *Saeed Balochian*

*Jie Yan* *Jie Yan*

Articles from The Scientific World Journal are provided here courtesy of **Hindawi Limited**

**Genetic Algorithms**

* Difficulty Level : [Medium](https://www.geeksforgeeks.org/medium/)
* Last Updated : 27 Apr, 2022

Genetic Algorithms(GAs) are adaptive heuristic search algorithms that belong to the larger part of evolutionary algorithms. Genetic algorithms are based on the ideas of natural selection and genetics. These are intelligent exploitation of random search provided with historical data to direct the search into the region of better performance in solution space. **They are commonly used to generate high-quality solutions for optimization problems and search problems.**

**Genetic algorithms simulate the process of natural selection** which means those species who can adapt to changes in their environment are able to survive and reproduce and go to next generation. In simple words, they simulate “survival of the fittest” among individual of consecutive generation for solving a problem. **Each generation consist of a population of individuals** and each individual represents a point in search space and possible solution. Each individual is represented as a string of character/integer/float/bits. This string is analogous to the Chromosome.

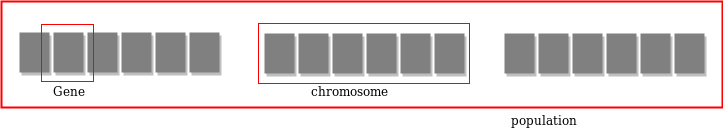
**Foundation of Genetic Algorithms**

Genetic algorithms are based on an analogy with genetic structure and behaviour of chromosomes of the population. Following is the foundation of GAs based on this analogy –

1. Individual in population compete for resources and mate
2. Those individuals who are successful (fittest) then mate to create more offspring than others
3. Genes from “fittest” parent propagate throughout the generation, that is sometimes parents create offspring which is better than either parent.
4. Thus each successive generation is more suited for their environment.

**Search space**

The population of individuals are maintained within search space. Each individual represents a solution in search space for given problem. Each individual is coded as a finite length vector (analogous to chromosome) of components. These variable components are analogous to Genes. Thus a chromosome (individual) is composed of several genes (variable components).



**Fitness Score**

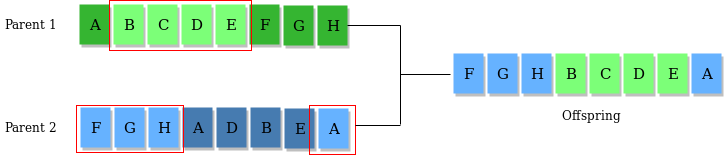
A Fitness Score is given to each individual which **shows the ability of an individual to “compete”**. The individual having optimal fitness score (or near optimal) are sought.

The GAs maintains the population of n individuals (chromosome/solutions) along with their fitness scores.The individuals having better fitness scores are given more chance to reproduce than others. The individuals with better fitness scores are selected who mate and produce **better offspring** by combining chromosomes of parents. The population size is static so the room has to be created for new arrivals. So, some individuals die and get replaced by new arrivals eventually creating new generation when all the mating opportunity of the old population is exhausted. It is hoped that over successive generations better solutions will arrive while least fit die.

Each new generation has on average more “better genes” than the individual (solution) of previous generations. Thus each new generations have better **“partial solutions”** than previous generations. Once the offspring produced having no significant difference from offspring produced by previous populations, the population is converged. The algorithm is said to be converged to a set of solutions for the problem.

**Operators of Genetic Algorithms**

Once the initial generation is created, the algorithm evolves the generation using following operators –   
**1) Selection Operator:** The idea is to give preference to the individuals with good fitness scores and allow them to pass their genes to successive generations.   
**2) Crossover Operator:** This represents mating between individuals. Two individuals are selected using selection operator and crossover sites are chosen randomly. Then the genes at these crossover sites are exchanged thus creating a completely new individual (offspring). For example –



**3) Mutation Operator:** The key idea is to insert random genes in offspring to maintain the diversity in the population to avoid premature convergence. For example – 



The whole algorithm can be summarized as –

1) Randomly initialize populations p

2) Determine fitness of population

3) Until convergence repeat:

a) Select parents from population

b) Crossover and generate new population

c) Perform mutation on new population

d) Calculate fitness for new population

**Example problem and solution using Genetic Algorithms**

Given a target string, the goal is to produce target string starting from a random string of the same length. In the following implementation, following analogies are made –

* Characters A-Z, a-z, 0-9, and other special symbols are considered as genes
* A string generated by these characters is considered as chromosome/solution/Individual

**Fitness score** is the number of characters which differ from characters in target string at a particular index. So individual having lower fitness value is given more preference.

As we can see from the output, our algorithm sometimes stuck at a local optimum solution, this can be further improved by updating fitness score calculation algorithm or by tweaking mutation and crossover operators.

**Why use Genetic Algorithms**

* They are Robust
* Provide optimisation over large space state.
* Unlike traditional AI, they do not break on slight change in input or presence of noise

**Application of Genetic Algorithms**

Genetic algorithms have many applications, some of them are –

* Recurrent Neural Network
* Mutation testing
* Code breaking
* Filtering and signal processing
* Learning fuzzy rule base etc

**References**   
<https://en.wikipedia.org/wiki/List_of_genetic_algorithm_applications>   
<https://en.wikipedia.org/wiki/Genetic_algorithm>   
<https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol1/hmw/article1.html>  
This article is contributed by [**Atul Kumar**](https://www.linkedin.com/in/atul-kumar-733b32136/). If you like GeeksforGeeks and would like to contribute, you can also write an article using [write.geeksforgeeks.org](http://www.write.geeksforgeeks.org/) or mail your article to review-team@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.  
Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

**Introduction to Particle Swarm Optimization(PSO)**

* Difficulty Level : [Hard](https://www.geeksforgeeks.org/hard/)
* Last Updated : 14 Jan, 2019

**Background of Particle Swarm Optimization**

Particle Swarm Optimization characterized into the domain of [**Artificial Intelligence**](https://www.geeksforgeeks.org/artificial-intelligence-an-introduction/). The term *‘Artificial Intelligence’* or ‘*Artificial Life*‘ refers to the theory of simulating human behavior through computation. It involves designing such computer systems which are able to execute tasks which require human intelligence. For eg, earlier only humans had the power to recognize the speech of a person. But now, speech recognition is a common feature of any digital device. This has become possible through artificial intelligence. Other examples of human intelligence may include decision making, language translation, and visual perception etc. There are various techniques which make it possible. These techniques to implement artificial intelligence into computers are popularly known as *approaches of artificial intelligence’*.

***These techniques are designed basis two categories*:**

* The first study includes how biological phenomena can be studies using computation.
* The second one shows how biological phenomena can help understand computation problems. While *studying the PSO Technique, we deal with in the second category*.

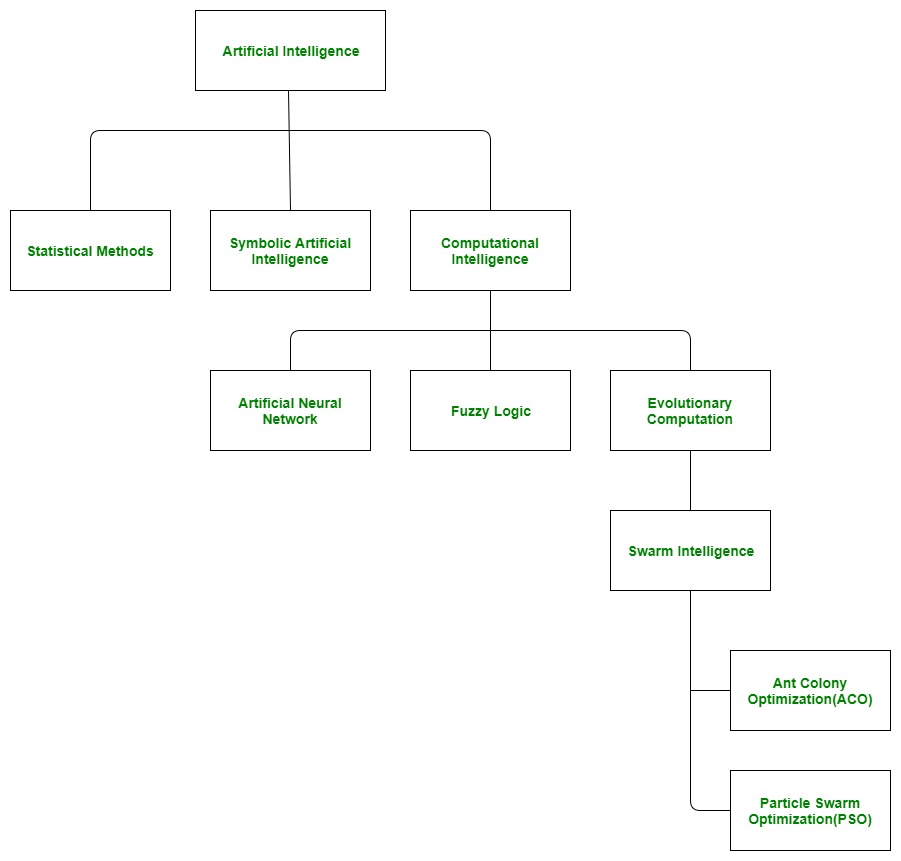
***There are three approaches of the artificial intelligence:***

* Statistical Methods
* Symbolic Artificial Intelligence
* Computational Intelligence

***Computational Intelligence can be implemented using either of the three methods:***

* Artificial Neural Network
* Fuzzy Logic
* Evolutionary Computation

**Note:** Under Evolutionary Computation, are the Swarm Intelligence Techniques which include Particle Swarm Optimization.



**Concept of Particle Swarm Optimization**

As described earlier, Swarm Intelligence is a branch of Artificial Intelligence where we observe nature and try to learn how different biological phenomena can be imitated in a computer system to optimize the scheduling algorithms. In swarm intelligence, we focus on the collective behavior of simple organisms and their interaction with the environment.

***There are two types of Optimization algorithms in Swarm Intelligence:***

* The first one is *Ant Colony Optimization(ACO)*. Here the algorithm is based on the collective behavior of ants in their colony.
* The second technique is *Particle Swarm Optimization(PSO).*

In PSO, the focus in on a group of birds. This group of birds is referred to as a ‘*swarm*‘. Let’s try to understand the Particle Swarm Optimization from the following scenario.

**Example:** Suppose there is a swarm (a group of birds). Now, all the birds are hungry and are searching for food. These hungry birds can be correlated with the tasks in a computation system which are hungry for resources. Now, in the locality of these birds, there is only one food particle. This food particle can be correlated with a resource. As we know, tasks are many, resources are limited. So this has become a similar condition as in a certain computation environment. Now, the birds don’t know where the food particle is hidden or located. In such a scenario, how the algorithm to find the food particle should be designed. If every bird will try to find the food on its own, it may cause havoc and may consume a large amount of time. Thus on careful observation of this swarm, it was realized that though the birds don’t know where the food particle is located, they do know their distance from it. Thus the best approach to finding that food particle is to follow the birds which are nearest to the food particle. This behavior of birds is simulated in the computation environment and the algorithm so designed is termed as Particle Swarm Optimization Algorithm.

**Note:** This same behavior is also executed by a fish school. Thus Particle Swarm Optimization Technique is said to be inspired by a swarm of birds or a school of fish. Thus, this algorithm is also called a *population-based stochastic algorithm* and was developed by Dr. Russell C. Eberhart and Dr. James Kennedy in the year 1995.  
This is the overall concept of what a particle swarm optimization is, and on what biological phenomena, its working is based upon.

**Introduction to Swarm Intelligence**

* Difficulty Level : [Easy](https://www.geeksforgeeks.org/easy/)
* Last Updated : 15 May, 2021

**Swarm Intelligence (S.I.)** was introduced by **Gerardo Beni and Jing Wang in the year 1989**. S.I. simply means using the knowledge of collective objects (people, insects, etc.) together and then reaching the optimized solution for a given problem. **“*Swarm” means a group of objects (people, insects***,***etc.)***. In other words, let’s say we give a problem statement to a single person and tell him or her to go through this problem and then give the solution, then this means that we will consider the solution of that particular person only, but the problem is that the solution given by that person may not be the best solution or maybe, that solution is not good for others. So to avoid that, what we do is we give that problem to a certain amount of people together (swarm) and ask them to reach the best solution possible for that problem, and then computing all the responses together to reach the best solution possible, so here we are using the knowledge of the group as a whole to reach to the best solution or optimized solution for that problem and that solution will be good for all of them individually too, so that is the idea behind swarm intelligence.

***Note****: In swarm intelligence, each individual (object) in the group is independent of others, each individual is responsible for their own contribution to solve that problem regardless of what others are doing.*

Let’s take an example ***to prove that the collective knowledge of objects is better than any individual object***.

**Example**

Let’s say we have a jar containing 500 marbles in that. The question is without touching the jar a person needs to predict how many marbles are in that jar. Suppose we take only one response from a person and it predicts that according to him the jar contains 400 marbles. So by this result, we can conclude that this estimation of that person is not very bad since the ***difference (error) is of 100*** only, but this might not be the best solution, we can optimize this even more. So now what we will do is instead of taking response from only one person we will be taking response from 10 people let’s say. Let ‘P’ denote a person therefore the responses are as follows:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | P10 |
| 400 | 450 | 550 | 600 | 480 | 390 | 520 | 490 | 510 | 450 |

So after collecting the responses from 10 different individuals we can take the average of their responses.

***Average:***

*(400 + 450 + 550 + 600 + 480 + 390 + 520 + 490 + 510 + 450) / 10*

*Average = 4840 / 10*

***= 484 (marbles in the jar)***

Now from this, we can say that from the collective predictions from 10 different persons we have reached a more optimal answer that is 484 marbles in the jar. We are very close to the actual result of 500 marbles in the jar, here in this case the ***difference (error) reduces to only 16 marbles as compared to the previous error which was 100***. So that is the main idea behind swarm intelligence, that is to use the collective knowledge of objects.

**Why Swarm Intelligence?**

The answer to this is simple now and we proved this in our previous marble example, which is collecting the answers (responses) from different objects individually and then computing all responses as a whole to a solution that best fits our given problem. So here, with this approach, we are having a more optimized solution for a given problem and that is the reason why swarm intelligence came into the picture, because of this reason we can use it in different scenarios of life e.g. Forecasting, Which policy is good for the business, etc. So simply we are using the **‘Brain of Brains’**  to reach the solution for a given problem. If we will observe in our surroundings (nature) then we will be able to find many examples of swarm intelligence like ***‘ant colony’, ‘swarm of bees’, ’flock of birds’*** etc. and in reality, also the idea of swarm intelligence was taken from nature only. Some are explained below:

**Ant Colony**

If we will observe closely then the ants also follow the principle of swarm intelligence, for example, to build the home they collect mud particles from the surroundings and individually have a responsibility to build their home. They communicate through signals and pheromones (ants use this for tracing other ants) and regardless of what other ants are doing, an individual ant is responsible for only its own contribution to build the home. Similarly when they search for food then at first they search individually for food leaving the pheromones behind and once they find the food source then that ant communicates with other ants and then other ants can trace it and follow that path to reach the source of food instead of just randomly searching food in different locations every time.

**Swarm of Bees**

Bees also use the same principle for their survival that is when they search for the place like where they can build their hive, then the task of each bee is to consider several parameters that the hive which will be built should be on good height to avoid predators, should be near water resource, should be near pollens (flowers to collect nectar), etc. then they use their collective research and finally a place is decided that where the hive would be built considering all those parameters and they reach the best solution for that problem.

**Artificial Swarm Intelligence (ASI)**

***It is also known as Human Swarm***. Here also the idea is same that we randomly make some of the persons participating in a real-time system and tell them to find the solution for that particular problem individually, the final solution is then computed after taking responses from each participant and the final solution is presented which is more optimized as compared to the solution taking from only one participant.

**Applications of Swarm Intelligence**

* Used in military services.
* NASA is generating the idea to use swarm intelligence for planetary mapping.
* Used in Data Mining.
* M. Anthony Lewis and George A. Bekey presented the idea that with the help of swarm intelligence we can control nanobots in our body to kill cancer tumors.
* Used in business to reach better financial decisions etc.

In [computer science](https://en.wikipedia.org/wiki/Computer_science), **evolutionary computation** is a family of [algorithms](https://en.wikipedia.org/wiki/Algorithm) for [global optimization](https://en.wikipedia.org/wiki/Global_optimization) inspired by [biological evolution](https://en.wikipedia.org/wiki/Biological_evolution), and the subfield of [artificial intelligence](https://en.wikipedia.org/wiki/Artificial_intelligence) and [soft computing](https://en.wikipedia.org/wiki/Soft_computing) studying these algorithms. In technical terms, they are a family of population-based [trial and error](https://en.wikipedia.org/wiki/Trial_and_error) problem solvers with a [metaheuristic](https://en.wikipedia.org/wiki/Metaheuristic) or [stochastic optimization](https://en.wikipedia.org/wiki/Stochastic_optimization) character.

In evolutionary computation, an initial set of candidate solutions is generated and iteratively updated. Each new generation is produced by stochastically removing less desired solutions, and introducing small random changes. In biological terminology, a [population](https://en.wikipedia.org/wiki/Population) of solutions is subjected to [natural selection](https://en.wikipedia.org/wiki/Natural_selection) (or [artificial selection](https://en.wikipedia.org/wiki/Artificial_selection)) and [mutation](https://en.wikipedia.org/wiki/Mutation). As a result, the population will gradually [evolve](https://en.wikipedia.org/wiki/Evolution) to increase in [fitness](https://en.wikipedia.org/wiki/Fitness_(biology)), in this case the chosen [fitness function](https://en.wikipedia.org/wiki/Fitness_function) of the algorithm.

Evolutionary computation techniques can produce highly optimized solutions in a wide range of problem settings, making them popular in [computer science](https://en.wikipedia.org/wiki/Computer_science). Many variants and extensions exist, suited to more specific families of problems and data structures. Evolutionary computation is also sometimes used in [evolutionary biology](https://en.wikipedia.org/wiki/Evolutionary_biology) as an *in silico* experimental procedure to study common aspects of general evolutionary processes.

Evolutionary algorithms[[edit](https://en.wikipedia.org/w/index.php?title=Evolutionary_computation&action=edit&section=3" \o "Edit section: Evolutionary algorithms)]

*Main article:*[*Evolutionary algorithm*](https://en.wikipedia.org/wiki/Evolutionary_algorithm)

[Evolutionary algorithms](https://en.wikipedia.org/wiki/Evolutionary_algorithms) form a subset of evolutionary computation in that they generally only involve techniques implementing mechanisms inspired by [biological evolution](https://en.wikipedia.org/wiki/Biological_evolution) such as [reproduction](https://en.wikipedia.org/wiki/Reproduction), [mutation](https://en.wikipedia.org/wiki/Mutation), [recombination](https://en.wikipedia.org/wiki/Genetic_recombination), [natural selection](https://en.wikipedia.org/wiki/Natural_selection) and [survival of the fittest](https://en.wikipedia.org/wiki/Survival_of_the_fittest). [Candidate solutions](https://en.wikipedia.org/wiki/Candidate_solutions) to the optimization problem play the role of individuals in a population, and the [cost function](https://en.wikipedia.org/wiki/Loss_function) determines the environment within which the solutions "live" (see also [fitness function](https://en.wikipedia.org/wiki/Fitness_function)). [Evolution](https://en.wikipedia.org/wiki/Evolution) of the population then takes place after the repeated application of the above operators.

In this process, there are two main forces that form the basis of evolutionary systems: **Recombination** **mutation** and **crossover** create the necessary diversity and thereby facilitate novelty, while **selection** acts as a force increasing quality.

Many aspects of such an evolutionary process are [stochastic](https://en.wikipedia.org/wiki/Stochastic). Changed pieces of information due to recombination and mutation are randomly chosen. On the other hand, selection operators can be either deterministic, or stochastic. In the latter case, individuals with a higher [fitness](https://en.wikipedia.org/wiki/Fitness_function) have a higher chance to be selected than individuals with a lower [fitness](https://en.wikipedia.org/wiki/Fitness_function), but typically even the weak individuals have a chance to become a parent or to survive.

Evolutionary algorithms and biology[[edit](https://en.wikipedia.org/w/index.php?title=Evolutionary_computation&action=edit&section=4" \o "Edit section: Evolutionary algorithms and biology)]

*Main article:*[*Evolutionary algorithm*](https://en.wikipedia.org/wiki/Evolutionary_algorithm)

[Genetic algorithms](https://en.wikipedia.org/wiki/Genetic_algorithms) deliver methods to model [biological systems](https://en.wikipedia.org/wiki/Biological_systems) and [systems biology](https://en.wikipedia.org/wiki/Systems_biology) that are linked to the theory of [dynamical systems](https://en.wikipedia.org/wiki/Dynamical_systems), since they are used to predict the future states of the system. This is just a vivid (but perhaps misleading) way of drawing attention to the orderly, well-controlled and highly structured character of development in biology.

However, the use of algorithms and informatics, in particular of [computational theory](https://en.wikipedia.org/wiki/Computational_theory), beyond the analogy to dynamical systems, is also relevant to understand evolution itself.

This view has the merit of recognizing that there is no central control of development; organisms develop as a result of local interactions within and between cells. The most promising ideas about program-development parallels seem to us to be ones that point to an apparently close analogy between processes within cells, and the low-level operation of modern computers.[[11]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-11) Thus, biological systems are like computational machines that process input information to compute next states, such that biological systems are closer to a computation than classical dynamical system.[[12]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-12)

Furthermore, following concepts from [computational theory](https://en.wikipedia.org/wiki/Computational_theory), micro processes in biological organisms are fundamentally incomplete and undecidable ([completeness (logic)](https://en.wikipedia.org/wiki/Completeness_(logic))), implying that “there is more than a crude metaphor behind the analogy between cells and computers.[[13]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-13)

The analogy to computation extends also to the relationship between [inheritance systems](https://en.wikipedia.org/wiki/Inheritance_systems) and biological structure, which is often thought to reveal one of the most pressing problems in explaining the origins of life.

*Evolutionary automata*[[14]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-ldr11-14)[[15]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-ldr13-15)[[16]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-ldr14-16), a generalization of *Evolutionary Turing machines*[[17]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-ldr15-17)[[18]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-ldr16-18), have been introduced in order to investigate more precisely properties of biological and evolutionary computation. In particular, they allow to obtain new results on expressiveness of evolutionary computation[[16]](https://en.wikipedia.org/wiki/Evolutionary_computation" \l "cite_note-ldr14-16)[[19]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-ldr17-19). This confirms the initial result about undecidability of natural evolution and evolutionary algorithms and processes. *Evolutionary finite automata*, the simplest subclass of Evolutionary automata working in *terminal mode* can accept arbitrary languages over a given alphabet, including non-recursively enumerable (e.g., diagonalization language) and recursively enumerable but not recursive languages (e.g., language of the universal Turing machine)[[20]](https://en.wikipedia.org/wiki/Evolutionary_computation#cite_note-ldr18-20).