Artificial Intelligence and Genetic Algorithms

ar·ti·fi·cial in·tel·li·gence

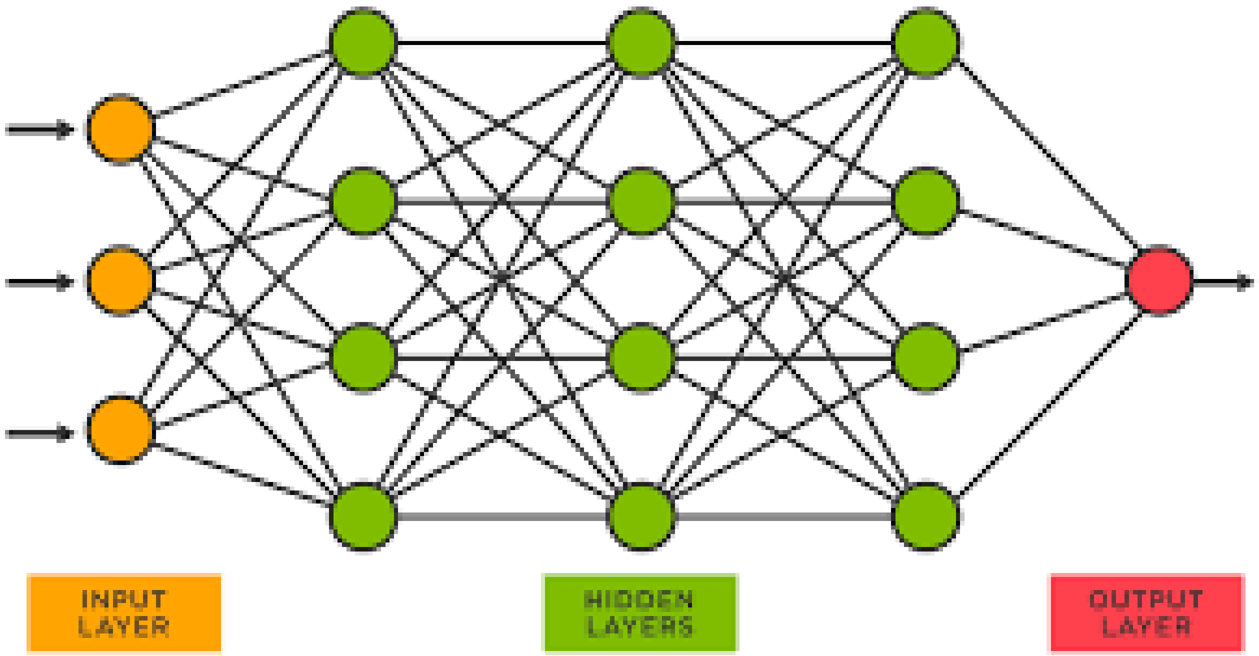
noun

the theory and development of computer systems able to perform tasks that normally require human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.

- Symbolic AI – When access to digital computers became possible in the
  to symbol manipulation.
  - Good Old Fashioned AI –
- Sub-symbolic –
  - Early cybernetics and brain simulation –
  - Behavior based AI –
    - Subsumption architecture –
  - Nouvelle AI –
  - Soft computing –
    - Computational creativity –
    - Machine learning
      - Neural networks –
        - Hybrid neural network –
        - Recurrent neural network –
    - Fuzzy systems –
    - Evolutionary computation, including:
      - Evolutionary algorithms –
        - Genetic algorithm –
      - Brain Emotional Learning Based Intelligent Controller –
      - Swarm intelligence –
        - Ant colony optimization –
- Statistical AI –

Computational Intelligence

Neural Networks



https://ieeetv.ieee.org/conference-highlights/artificial_neural_networks_intro?rf=channels|56|Selected_Videos&

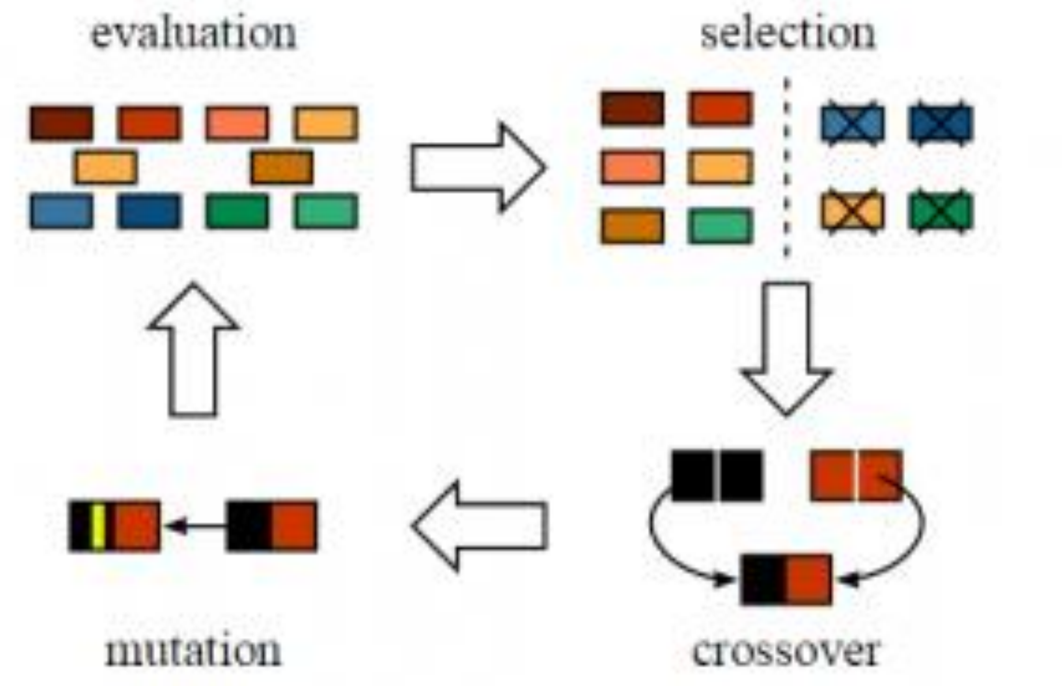
https://www.youtube.com/watch?v=9Yq67CjDqvw

Fuzzy Logic

https://www.youtube.com/watch?v=J_Q5X0nTmrA

https://www.youtube.com/watch?v=wAYCC0r_bN8&t=1s&index=7&list=WL

Genetic Algorithms (Our Focus)



https://www.eejournal.com/article/when-genetic-algorithms-meet-artificial-intelligence/

John Holland – Developer of GAs

http://www2.econ.iastate.edu/tesfatsi/holland.GAIntro.htm

GAs do controlled searches in a Solution space

Consider the problem maximize $z = 64 - x^2 - y^2$

By inspection, we see that the maximum value of z is 64 when x and y are both 0

If we remember calculus, we could use partial derivatives etc to find the optimal solution

But suppose

- We cannot solve the problem by inspection
- We do not remember calculus
- We remember calculus but the partial derivatives are very complicated with

maximize z = 64-x^2 -y^2 by searching

- Continuing with this example, ***suppose***
  - We know that x and y are between 0 and 20
  - We want the answer to be of the form dd.ddddd, that is we want 4 places to the right of the decimal
- A naïve approach would be to search as follows
  - Select all possible values for x so that
    - x is between 0 and 10
    - x has 5 decimal digits
  - Do the same for y
  - Evaluate 64 -x^2 -y^2 and select the best one among all possible values of x and y

The “naive” approach on the previous slide would certainly work

However, how many values would we have to check?

Both x and y are of the form dd.ddddd

For x, there are 10*10*10*10*10*10*10 = 10^7 = 10 million possibilities

There are the same number for y

Checking 2*0 million possibilities may take a long time!!*

We need an algorithm that searches the space in such a way

- An optimal or near optimal solution is found
- Execution time is reduced

Genetic Algorithms – Short Version

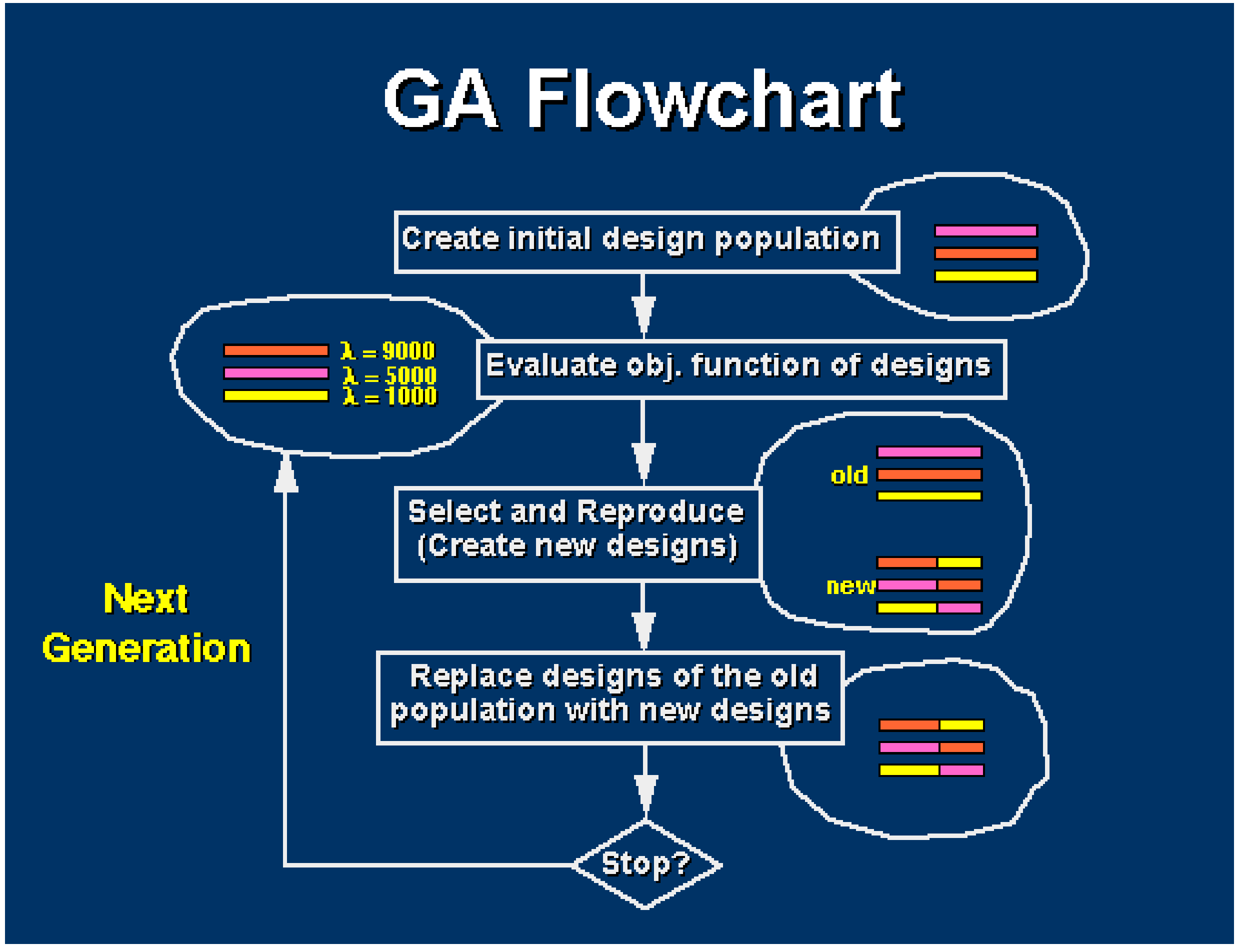
Form a population of organisms

Loop

- Evaluate each organism for its fitness
- Select organisms to mate and have children based on fitness and then form a new generation
- Mutate some of the new generation

End Loop

Report the answer represented by fittest organism after several generations



GA Flowchart
Create initial design population
Evaluate obj. function of designs
λ = 9000
λ = 5000
λ = 1000
Select and Reproduce
(Create new designs)
old
new
Replace designs of the old
population with new designs
Stop?
Next
Generation

GAs – How They Work – Initial Population

- We are given a problem
  - Could be the shortest route problem which you solved by Dijkstra
  - Could be to maximize/minimize a function f(x,y) e.g. 64 - x^2 -y^2
  - Could be to generate room assignments at a school
  - Could be many things!
- We first form a population of size N
  - The members of the population are potential solutions to the problem
  - Each member is called an organism
  - For example {(1,4), (2,7), (7,3), (-2,2), (0,4)} could represent a population for maximizing 64 -x^2 -y^2. Each ordered pair is an organism.

More GAs : Fitness

We evaluate the fitness of each member of the population

We often say we use the Fitness Function to evaluate an organism. In many cases, this requires the GA developer to *apply some art to the science* in order to find a Fitness Function

The fitness could be simply the value of the objective function (for example 64-x^2-y^2)

It could be something else

For example, in our attempt to maximize $64-x^2-y^2$, we have

- fitness(1,4) = 64-1-16 = 47
- fitness(2,7) = 64-4-49 = 11
- fitness(7,3) = 64-49-9 = 6
- fitness(-2,2) = 64-4-4 = 56
- fitness(0,4) = 64-0-16 =48

By inspection, we see that the strongest organism is (-2,2) and the weakest is (7,3)

According to Darwin, the strongest survive!

More GA Introduction - Mating

- We now prepare for members of the population to mate
- In general
  - Two members of the population are chosen to mate. *You* have to decide how that will be done!!
  - We try to insure that strong/fit members of the population get to mate more often than the weak ones
  - We (hope!) that the offspring of the mating are stronger than the parents
  - We allow mutations – random changes in the organism that (hopefully) make an offspring stronger

Mating continued

It is customary to have each mating result in two offspring

The “DNA” of the offspring can be determined is several ways. We will see several examples later on

Two ways of selecting parents are

- Elitism – the strongest member of the population is allowed to mate with everyone else. We see this in nature – such as in deer herds, gorillas, or lions. There are no hard and fast rules but
  - Strongest may mate with all other organisms
  - Strongest may only mate with other strong organisms

GAs Mating Continued

Roulette wheel – organisms are allowed to mate according to the percent of their contribution to the total fitness of the population. Consider the example of $64-x^2-y^2$

Organism	Fitness	% of Total Fitness	Cumulative Perecent
(1,4)	47	27.9%	27.9
(2,7)	11	6.5%	34.4
(7,3)	6	3.6%	38.0
(-2,2)	56	33.3%	71.2
(0,4)	48	28.6%	100.0

Total Fitness = 168

Roulette Wheel continued

Organism	Fitness	% of Total Fitness	Cumulative Perecent
(1,4)	47	27.9%	27.9
(2,7)	11	6.5%	34.4
(7,3)	6	3.6%	38.0
(-2,2)	56	33.3%	71.2
(0,4)	48	28.6%	100.0

To select parents

Generate a random number, r, in [0,1]. Both Java and C can do this

```
If 0<=r <0.279, organism (1,4) is a parent
else if  0.279<=r< 0.344, organism (2,7) is a parent
else if 0.344 <=r <0.38, organism (7,3) is a parent
else if 0.38<=r<0.712 organism (-2,2) is a parent
else organism (0,4) is a parent
```

Summary so far

We have seen that we must <u>construct an initial population</u> and we must be able to <u>evaluate the fitness of each organism</u> in the population

We loop as follows:

- Choose two parents (elitism, roulette wheel, or some other method) based on fitness
- Mate parents and form two children. Put these children in the new generation
- Possibly mutate the children

end loop

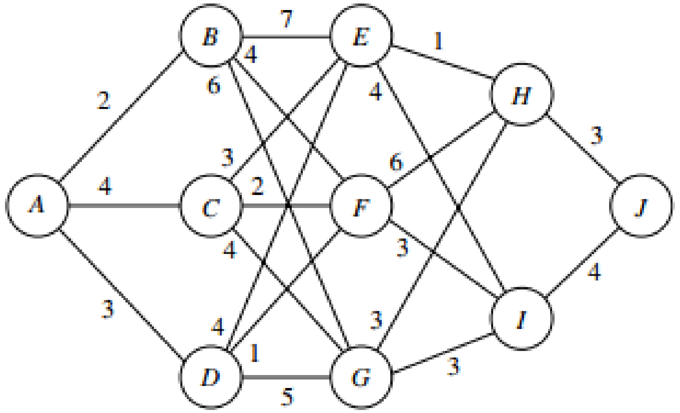
We now discuss representing organisms, mating, and mutation

Representing Organisms – Stage Coach Problem

Consider the Stage Coach problem and suppose we want to solve it with Genetic Algorithms

We might (*might because there are other ways!*) represent organisms as possible routes

For example, we could form the following orgnisms



ACEHJ
ADFHJ
ACGIJ
etc
There are 18 possible organisms

Mating Stage Coach Organisms

We now must find some meaningful way to mate two organisms that are potential solutions to the Stage Coach Problem.

There are no hard and fast rules!!! We must often develop them ourselves

One way is

- Select two parents – $x_1x_2x_3x_4x_5$ and $y_1y_2y_3y_4y_5$ b
- Randomly select an integer – say n -between 2 and 4 inclusive. We don't use positions 1 or 5 since they must be A and J respectively
- Child one gets father's values from 2 to n and mother's values from n+1 to 5
- Child two gets mother's values from 2 to n and father's values from n+1 to 5

More Stage Coach Organism Mating

Suppose that the parents are `ACEHJ` `and` `ADFHJ`

Suppose that we ask Java or C to generate a random integer between 1 and 4 and suppose it generates 2. This random point is called the Crossover Point.

Then the two children are:

ACFHJ and ADEHJ

Next, suppose that the parents are `ACEHJ` `and` `ADFHJ` and the cross over is 2.

Then the two children are ACFHJ and ADEHJ

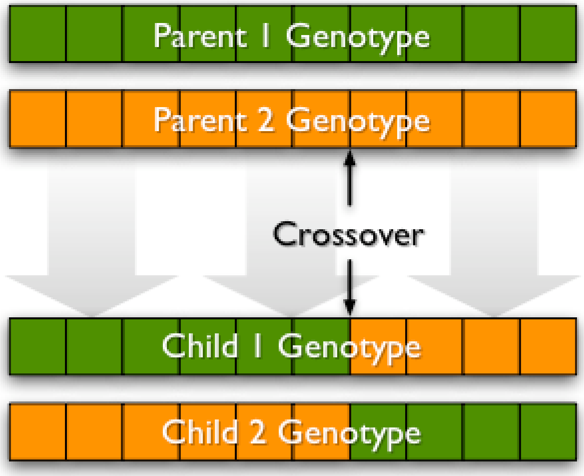
GA mating and biology

The examples on the previous slide can be thought of biologically

Two parents have two children

Each child gets some "gene" or some "DNA" from each parent

The children are similar to but not identical to their parents.



Finding the phrase..

Consider guessing a secret phrase – like in the children's game hangman. We will see if we can envision a computer guessing the phrase by using GAs.

Suppose that the phrase contains n ASCII characters

- “marshall university”
- “presidential election”
- “national football league”

If the phrase has n characters, then organisms could be strings of length n

For example, suppose the string is “australia”

Possible organisms are

```
abcdefghi
zweomaret
iiaovbekp
etc
```

Using lower case letters we see that there are there are 26^n possibilities for the phrase. If we allow blanks, there are 27^n possibilities and if we allow capital letters, there are 53^n possibilities

We could define the fitness of an organism to be the number of places where the organism and the secret phrase agree.

For example, if the secret phrase is “australia” then

- The fitness of “absolutes” is 2 since the first a and the s match
- The fitness of “abstrvxim” is 4 since the a,s,t, and i match

Mating is fairly simple

Select two parents and a crossover point. Form the children like we did in the Stage Coach Problem

If dad = "abcdefghi" and mom = "absolutes" and the crossover point is 4, then the two children are

abcdlutes and absoefghi

If the cross over point is 6, then the children are

abcdeftes and absolughi

The Need For Mutation

In human genetics, we hear the terms mutated and mutant

The root word is mutate and it simply means “to change”

Often we mean that something bad has happened and a child may be born with a serious disease that neither of his parents had

Other times, something good may happen – for example a musical genius may be born into a family which has no musical talent or a child may be born and possess an IQ that far exceeds that of his parents or siblings

Mutation in “Guess the Secret String”

Suppose that we are playing “Guess the String” and that the secret phrase is “far”

Suppose further that the original population was:

{fed, her, gee, sod}

By inspection we see that it is impossible for this population to ever reproduce “far”

Why? There is no organism that has an 'a' in the second position. Consequently no descendant will ever have an 'a' in the second position. Prove it to yourself!!

Mutation on the computer

To rectify the problem discussed on the previous slide, we introduce mutations.

A mutation simply allows us to randomly change some offspring in the hope of improving its fitness

Mutation must be done in a way consistent with your problem. The following is simply an example based on the problem where

- far is the target string
- {fed, her, gee, sod} are the population organisms

Suppose we mate fed and sod using a crossover point of 1.

The children will be fod and sed

Our mutation scheme (as an example!!) is to occasionally randomly select a letter in each child and replace it with a letter randomly chosen from a,b,c,...,z

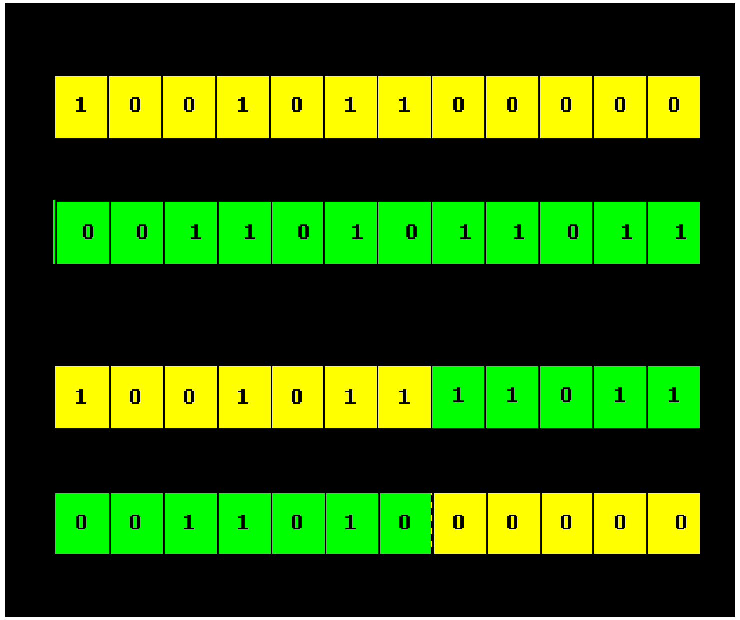
If we are fortunate, we might end up randomly replacing the 'o' in fod with 'a'

Using bit strings to represent organisms

One convenient way to represent organisms is to use bit strings

Each organism is simply a sequence of 0's and 1's

If – and we stress if!!!! - we can do that, mating is



Here, the crossover point is 7.

Note the offspring have "genes" and "DNA" from both parents

Some examples using bit strings

Suppose that our problem involves only one variable, x

Suppose further that we want to represent x as a bit string and we want 3 binary decimal digits to the right of the decimal and 4 to the left.

And suppose further that we need to know if the number is positive or negative

Organisms *could* have the form

bbbbbbbb

implied decimal point

Sign bit. 1 = positive and 0 equals negative

Using the ideas on the previous slide, decode the following

organism = 10110100

Since the last bit is 0, the organism is negative

The decimal value of this organism is

-(1*2^3 +0*2^2 +1*2^1 +1*2^0 +0*1/2 +1*1/4 +0*1/8) =

=-(8+0+2+1 +1/4) = -11.25

organism = 11111111

This is a positive number with value =

8+3+2+1 +1/2 +1/4 +1/8 = 14.875