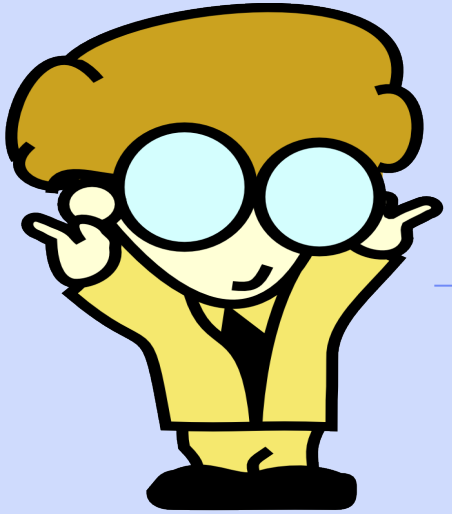
Optimization & Genetic Algorithms

By Brad Morantz PhD

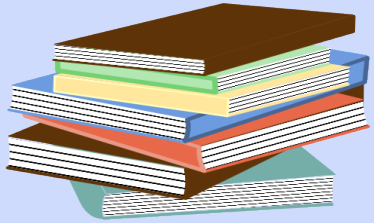


Suppose . . .

- You were a shoe salesperson, and had to call on stores in 3 states, but stores only open certain times, etc.

- You owned a candy factory, had limited cash flow and resources, and wanted to maximize profit

- You were taking 5 classes, had too much homework, and needed to keep good grades



What are the common factors?

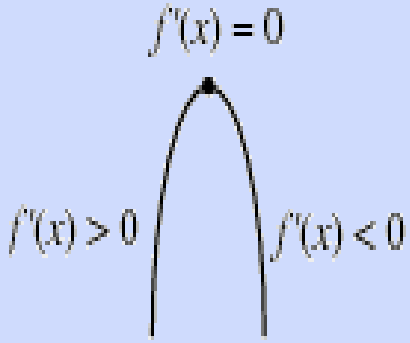
- Very complex problem
- Very hard, if not impossible, to find solution
- Need solution that will find optimal path
- Or NEAR optimal path
- The more variables, the more dimensions in the solution space
- There can be constraints

What is Optimization

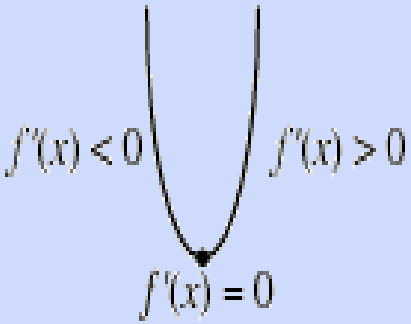
- Finding the *BEST* solution
  - Closely related to finding the worst
- The *minimum*
  - e.g. the minimum MSE (mean square error)
- The *maximum*
  - e.g. the maximum profit
- The *optimum*
  - e.g. the shortest path to intercept

Mathematically

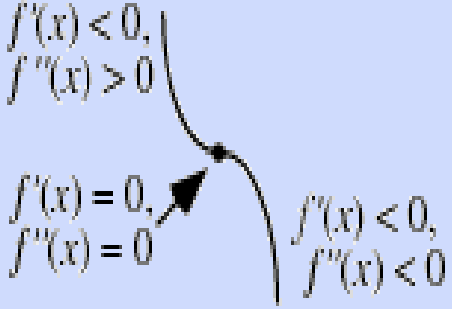
- Where first derivative is Zero



maximum



minimum



inflection point

Definition of Optimization

- The procedure or procedures used to make a system or design as effective or functional as possible, especially the mathematical techniques involved
- In mathematics: trying to find maxima and minima of a function
- In computing: the process of modifying a system to make some aspect of it work more efficiently or use fewer resources
- Process: improving the efficiency

How to solve?

- Mathematical techniques
  - Linear programming
    - Simplex, Nelder-Mead, etc.
  - Steepest Descent or Generalized Reduced Gradient (Waren & Lasdon)
  - Near optimal mathematical solutions of discrete systems (Romanovsky)
- Good Guessing (do not underestimate)
- Genetic Algorithms, simulated annealing, or various search algorithms (A*, greedy, etc.)

Requirements

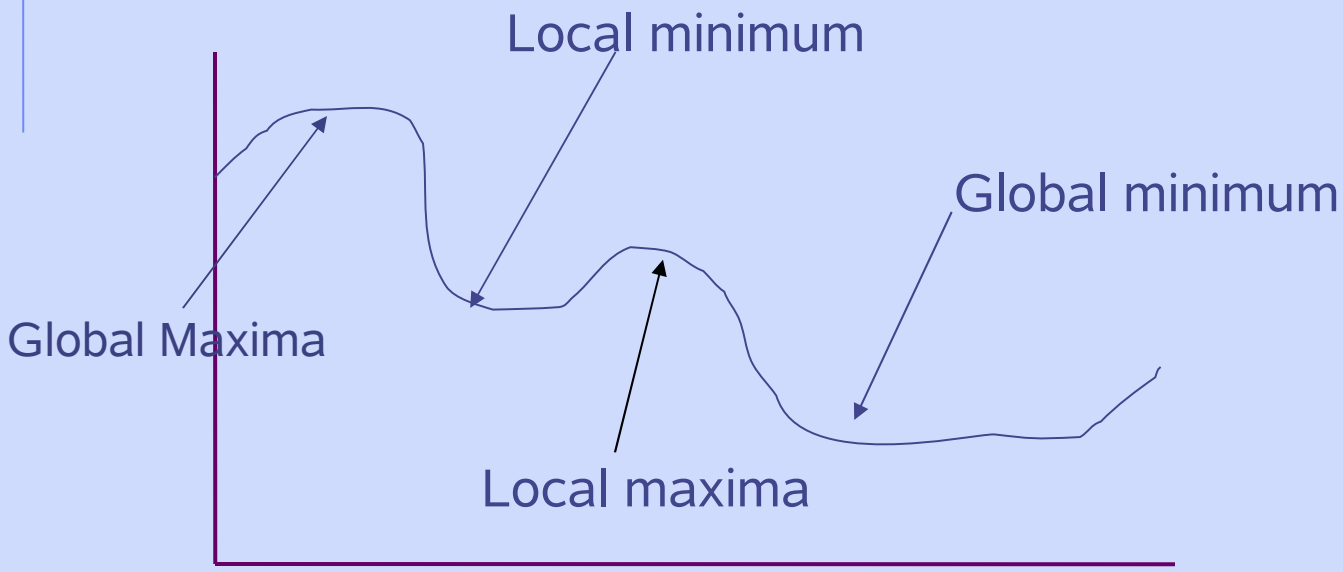
- Linear Programming
  - Linear relationship
  - Objective function
- Descent type
  - First derivative
  - Gradient matrix
  - Objective function
- Genetic algorithm
  - Fitness Function

Quality of Solution

- Mathematical can guarantee the optimal solution
- GA only guarantees NEAR optimal solution
- Must be careful not to get trapped on local minima (or maxima)

Local Minima

- Major problem, can get trapped in plateau when need to find global minima



Solutions to Local Minima

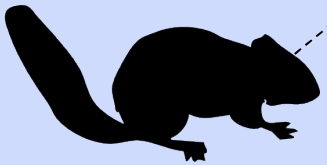
- Momentum
  - Try to jump out of valley
- Random numbers & multiple solutions
  - Use random numbers, vary the seed, and compare a number of solutions
- Mutation in GA's
  - Introduces new features and possibilities

Processing Time

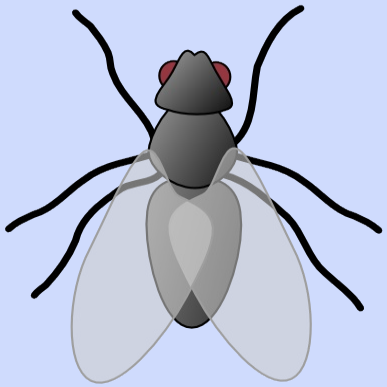
- Traveling salesman problem
  - Over 1 year w/traditional methods on Pentium 4
  - under 10 minutes w/GA on Pentium 4
- Mathematical faster on small problems
  - Solve mathematics
- GA faster on large problems
  - Large overhead setting up system
  - Very applicable to parallel processing systems

Biology Example

- Giant Eagle
  - Flies Very high & fast
  - If does not see prey, dies from hunger
- Eagle with great vision
  - Finds prey and eats
  - Has offspring with great vision
- Survival of the fittest



Entomology Example



- We have many flies
- DDT is produced
- Many flies die
- Flies that don't die reproduce
- Now have many flies that DDT does not kill

Genetic Algorithms

- Search Procedure
  - Modeled on natural selection
  - Also called Evolutionary Computing
  - Keep and mate the most fit
  - Let the least fit die off
- In biology called:
  - Survival of the fittest
  - Adaptation & mutation
  - Natural Selection

Genetic Algorithms

- Problem is defined
- Each potential solution is a chromosome
- Generate population of chromosomes
- Weak ones die
- Strong ones reproduce
- Each successive generation gets stronger

Terminology

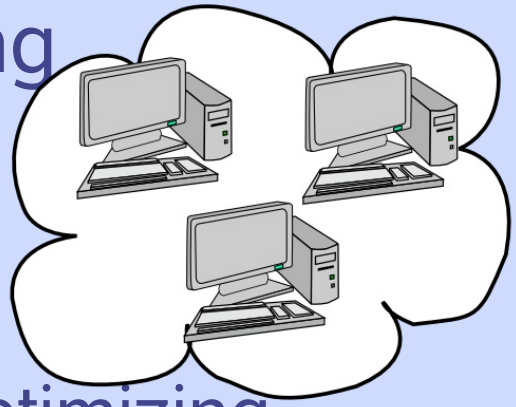
- Each solution is a chromosome or organism
- Chromosome is a vector of genes & alleles
- Set of chromosomes is a population
- Successive populations are generations
- A chromosome is evaluated by a fitness function
- A random change in a gene is a mutation
  - Just random, no cause

Schema Theorem

- John Holland
  - University of Michigan
  - 1975
- Organisms of greater fitness appear with exponentially greater frequency as chromosomes are replaced
- Apply to search

John Koza

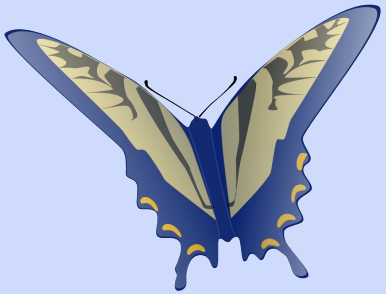
- Studied under John Holland
- Father of Genetic programming
- Made invention machine
  - 1000 networked computers
  - For optimizing many things
  - Machine created program for optimizing factories
    - One of first patents for IP (intellectual property) for a non-human
    - Virtually no human guidance



Chromosome

- Abstract representation of potential solution
- Denotes intersection of hyperplanes in search space
- Typically a binary string
- Initial population is used to seed process
  - Expert supplied
  - (pseudo) Random number generator

Chromosome



Thrust	Weight	Length	Diameter	Payload	Burn Time	Guidance	Velocity
0 0 1	0 1 1	1	0 0	0	1 1	1	0 1

Example Chromosome

- Each allele is typically binary, one of two states
- Some genes are made of several alleles
- A genotype can be pleiotropic, where one gene affects multiple phenotypes

Fitness Function

- Evaluates each chromosome
- Evaluates system performance with solution prescribed by organism
- Calculates figure of merit for each
  - Assigns a metric to performance in area of interest
- Ordering values will show which are best performers or most robust

Fitness Function

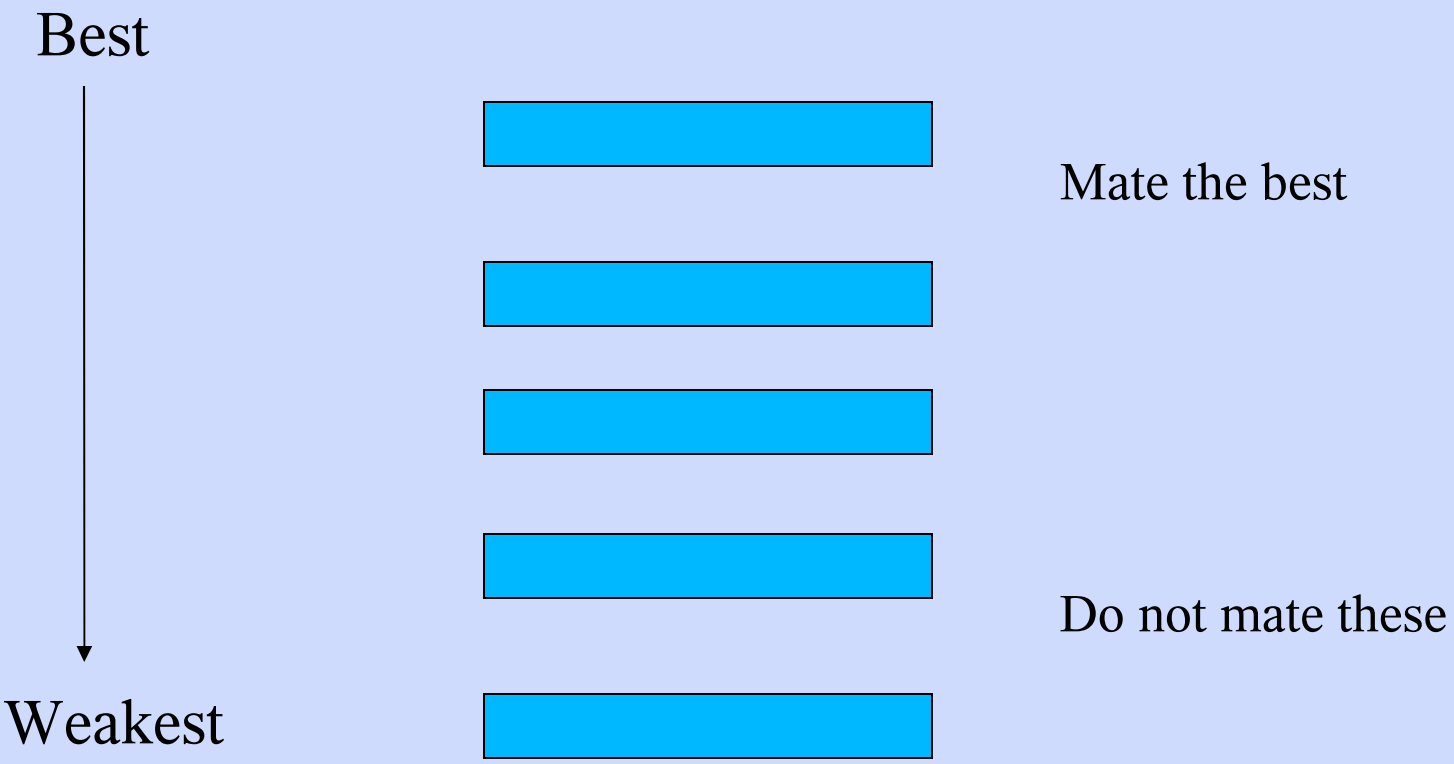
- Formula, model, or neural network

Can even be black box

Simple Selection

- Put all chromosomes into order by figure of merit
- Discard the half of population with lowest values
- Mate the remaining chromosomes

Selection



Stochastic Selection

- Assign probability to each organism
  - Directly proportional to merit of fitness
- Generate cumulative probability table
- Use random number generator to select parents
  - Most robust organisms propagate more readily
- Mate
- Delete worst 50% of performers

Mating

- Crossover point is where chromosome is split
- Front part of one parent is put with back of other
- Creates 2 new children

This is one of many possible crossover methods

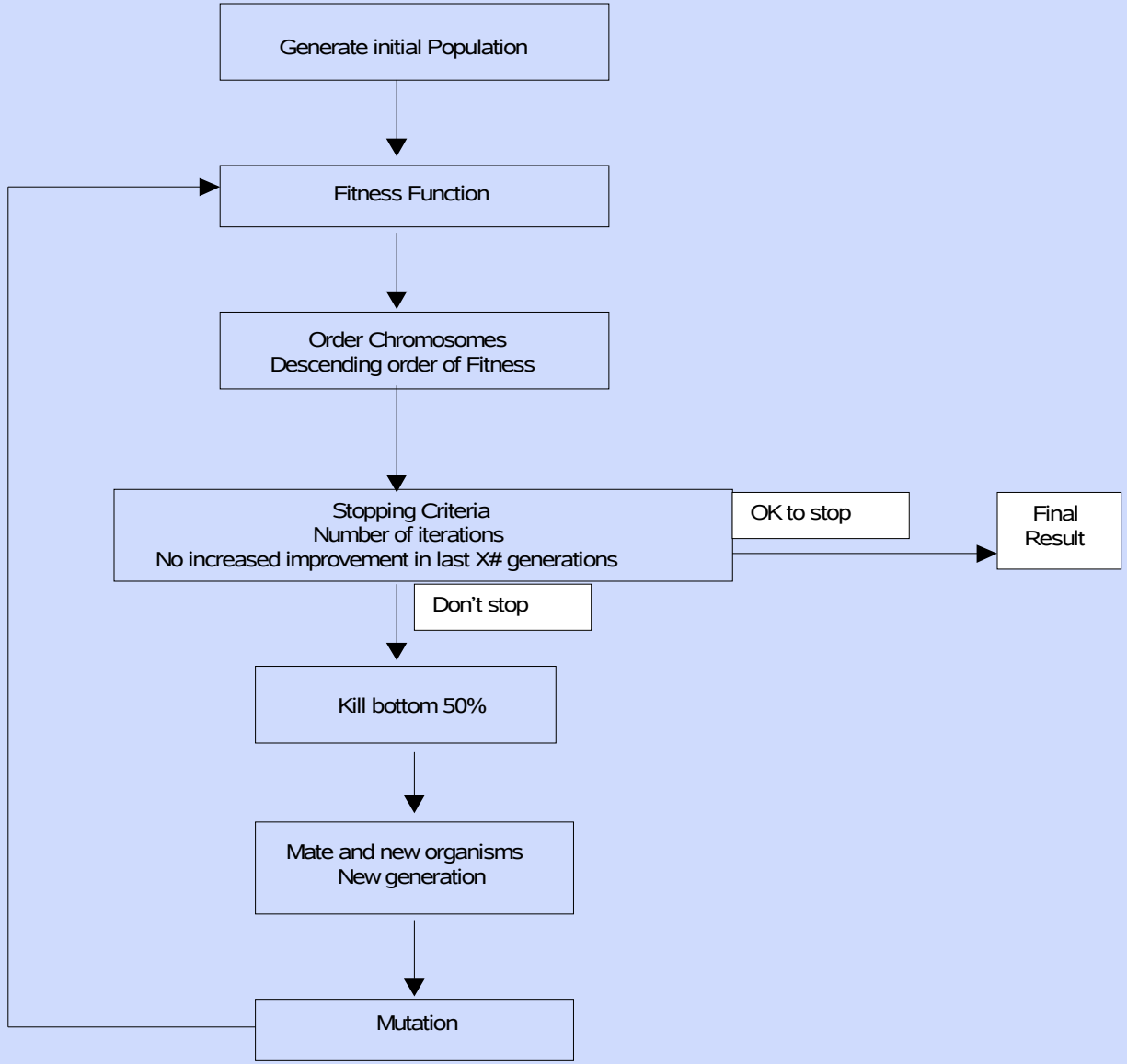


Figure 1 Typical Genetic Algorithm Block Diagram

The Search

- Search space is exploited and explored
- Based upon previous iteration, not random
- Many examined, parallel operation
- Hill climbing (Russell & Norvig)
  - Small changes made
  - Most fit selected for next step

Pin the Tail on the Donkey

- Person is searching for optimal space to pin the tail
- Person is searching
- Other people are giving information:
  - Hot
  - Cold
  - Getting warmer
  - Etc.

Mutation

- Expands the search space
- Introduces new features
- Random process
- Accomplished by randomly changing a gene
- Empirically shown that optimum rate is 1.5% to 3%

Mutation

Mutation

- This is just a piece of a chromosome
- Do not necessarily use the second spot
- This random change introduces new information

Stopping Criteria

- Number of generations/iterations
  - Prevents endless loop
- Achieved desired level of performance
- Successive iterations failing to yield any additional improvement
  - Less than some delta
  - Do a few more generations just in case of plateau & local minima

Computational Considerations

- Inherently a very parallel process
  - Would be a good fit on Fortran 95/2003
  - Speedup would be significant w/ parallel processing
  - Multiprocessing environment could bring into near real time
- Does not have large memory requirement of some mathematical techniques (not a real problem in today's environment of cheap memory)

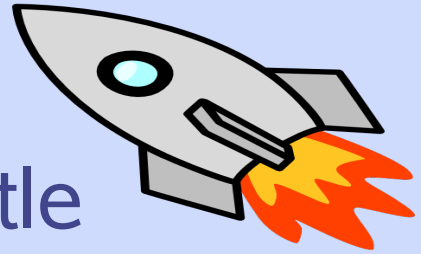
Application Example

- Want to develop optimal protein
- Do NOT have model for what makes a protein optimal
- Train a neural network on a variety of proteins
  - Give better proteins higher score
- Use this as the fitness function in a GA
- Results were very good

Interesting Application

- Adaptive testing
  - For complex systems that are too large for exhaustive testing or would take too long
  - Process:
    - Use volume filling test matrix for initial test
    - Create fitness function that rewards for poor performance
    - Use exploited search (GA) to find failure points or poorest performing spots

Adaptive Test Example



- Next generation space shuttle
  - Operating system is huge
  - Exhaustive test would take > 2 years
- Run volume filling test matrix testing
- Create fitness/cost function to reward poor performance
- Use exploited search to find places where system malfunctions

Information sources

- IEEE Evolutionary Computing Society
- AAAI www.aaai.org
- Books:
  - *Genetic Algorithms* by Buckles & Petry IEEE press
  - *Optimization*, Waren, Lasdon, & Rom
  - *Artificial Intelligence,* Russell & Norvig
- IEEE Computational Intelligence Society www.ieee-cis.org

Contact Information

- Brad Morantz
- bradscientist@machine-cognition.com
- www.machine-cognition.com
- www.numbersforlife.com
- www.cognitive-decisions.com

Questions?