Outline

- Background: Biological Genetics & GA
- Two Examples
- Some Applications
- Online Demos* (if the time allows)
Introduction

What is Genetic Algorithms (GA)?

- A search technique used in computing to find true or approximate solutions to optimization and search problems (Wikipedia).
- Suitable for searching problems with extremely huge search space that are hard or even impossible to be solved by traditional analytical methods or minimum-seeking algorithms (i.e. exhaustive search or calculus based optimization method).
- Inspired by Darwin’s theory of evolution and the study of genetics such as inheritance, mutation, selection, and crossover (also called recombination), etc.
- Originally invented by John Holland from the University of Michigan in the early seventies (1975).
Biological Background

Organism:
- Cell – the basic part of all living organism
- Chromosome – each cell contains the same number of chromosomes
- Gene – a chromosome consists of genes, the traits are encoded in the form of DNAs or genetic codes

Reproduction:
- Inheritance – offspring inherits the traits (genetic code) from parents
- Crossover – the portion of mother’s and father’s chromosomes
- Mutation – a random change may occurs in the offspring’s genetic code

Natural selection & evolution (Darwin’s theory)
- Fitness – how successful is an individual fit for the environment, determined by genetic code.
- Natural selection- higher fitness value means higher survive rate
- Evolution – on the long run, the most recent survived generation intend to have a high fitness
GA v.s Biological Genetics

- GA is the simulation of the natural selection and evolution on a computer
- An intuitive analogy

[Haupt2004]
The GA starts with an initial population that consists of a set of individuals, each individual is represented by a unique binary string or float-point value (chromosome).

Individuals are evaluated (based on their fitness), selected, and mated.

New individuals are generated in the principles of inheritance and variation. Individuals with low fitness are discarded.

The iteration continues until a convergence requirement is fulfilled.

[Haupt2004]
A Basic Example

Problem: find the global maximum point of the following surface (2D function):

\[ f(x, y) = x \sin(4x) + 1.1y \sin(2y) \]

\[ 0 \leq x \leq 10, 0 \leq y \leq 10 \]
GA Steps

- **Chromosome Encoding:** chromosome = \([x, y]\)
- **Fitness Function:** \(fitness = x \cdot \sin(4 \cdot x) + 1.1 \cdot y \cdot \sin(2 \cdot y)\)
- **First Generation:** 16 random individuals (chromosomes)
- **Selection:** 8 most-fit chromosomes survive, others are discarded
- **Pairing:** 4 pairs of parents are randomly selected.
- **Mating & Crossover:** offspring is produced by parents:
  - blending method: \(p_{\text{child}} = b \cdot p_{\text{pa}} + (1-b) \cdot p_{\text{ma}},\)
  - \(p\) is one of the variables (x or y), \(b\) is a random number between [0, 1], same or different \(b\) for each variable.
- **Mutation:** a few of randomly selected variables are replaced by uniform random numbers.
- **New Generation:** made up by survived parents and offspring
- **Iteration:** the above process is iterated until the global maximum is found, or the predefined iteration number is exceeded.
Parameters

- **Elitism:**
  - One or a few best chromosomes are copied directly to the next generation, without changes

- **Crossover probability: between [0, 1]**
  - 0%: all offspring is exact copy of parents (but the new generation is not necessarily the same!)
  - 100%: all offspring is made by crossover

- **Mutation probability: between [0, 1]**
  - 0%: no individual is changed
  - 100%: all individuals are changed (random searching)

- **Population Size:**
  - Too small: the search space is limited
  - Too large: slows down
A Classical Example

- **Traveling Salesman Problem (TSP):**
  - Randomly located cities, the traveling salesman has to minimize the traveling distance while visit all of the cities
  - Essentially an ordering problem: finding an optimal ordering for a sequence of N items
  - Other similar problem: scheduling, routing, resource allocation, assignment

- A “natural” encoding method might be problematic!
  - The simplest case: four cities: 1, 2, 3, 4 – each with a unique bit-string encoding
  - Considering two individuals: 3-2-1-4, and 4-1-2-3
  - A crossover between the parents may produce wrong offspring, i.e. 3-2-2-3!

- “Random-key” encoding method [Forrest1993].

In fact, for each specific GA problem, designing the encoding method is an important issue, so do the crossover and mutation methods.
- selection and pairing methods.
Random-key Method*

- The chromosome is encoded by a bit string, which is divided into $N$ segments of $k$ bits [Forrest1993]
  - $N$ is the same as the cities number
  - $2^k >> N$
- For example: $N = 4$, $k = 4$ (16 bits for each chromosome)
- A randomly generated chromosome: [0101 0011 0001 1001], interpreted as decimal numbers: [5-3-1-9 ]
- City numbers are encoding as the segment position, sequence of traveling route is up to the relative value of each segment.
  - Segment 3 has the smallest value 1, so city 3 is the first.
  - Segment 2 has the second smallest value 3, so city 2 is the second, and so on and so forth
  - Thus, pattern [5-3-1-9] represents the tour [3-2-1-4]
- The offspring’s legality is guaranteed, no matter what “strange” chromosome (bit string) is produced via crossover and/or mutation
- Also called permutation encoding
Online Demos*

- **Global Maximum of 1D Function:**
  
  http://cs.felk.cvut.cz/~xobitko/ga/example_f.html

- **Global Maximum of 2D Function:**
  
  http://cs.felk.cvut.cz/~xobitko/ga/example3d.html

- **TSP:**
  
Examples in Music Tech

- **Granular Synthesis Regulation [Fujinaga1994]**
  - Each grain’s parameters (freq, amp, etc.) mapped to a chromosome’s bit string
  - GA parameters (crossover rate, mutation rate, etc.) control the change in the grain population

- **FM Synthesis Parameters Optimization [Lai2006]**
  - FM parameters (carrier freq, mod freq, amp) mapped to a chromosome’s bit string
  - Spectrum similarity of target sound and synthesized sound is evaluated as fitness

- **Automatic Generation of Sound Synthesis [Garcia2000]**
  - Not only the parameters of synthesis, but the synthesis building block are regarded as genes
  - Generates new topologies and synthesize the target sound
  - Genetic Programming
    - Tree coding
    - Lisp Implementation [Forrest1993]
Examples in Creativity

Music Searching:

Encoding:
- One gene (3 bits) for each beat
- 000 for hold, 001 – 111 mapping to A to G
- Chromosome: [EDCDEEEholdDDDholdEGGhold]
- Encoded as [101 100 011 100 101 101 101 000 100 100 100 000 101 111 111 000]

Fitness Function:
- Objective:
  \[
  \text{cost} = \sum_{n=1}^{48} |\text{answer}[n] - \text{guess}[n]|
  \]
- Subjective: Interactive ranking by ear

Extension of this INTERACTIVE idea:
- GA composing and other genetic art
- New product design
Applications List

General
- Optimization (strategy planning, robot trajectory, sequence scheduling)
- Modeling dynamic systems (ecologic systems, immune systems, genetic systems, social systems)
- Evolving LISP programs (genetic programming)
- Etc.

Music Engineering / Composing
- Synthesis system design
- Synthesis parameters optimization
- Sound synchronization, animation
- Timbre recognition
- Fugue generation
- Jazz solos generation
- Waveform evolving
- Etc.

Many others up to your imagination
References


Questions?

Please ask…