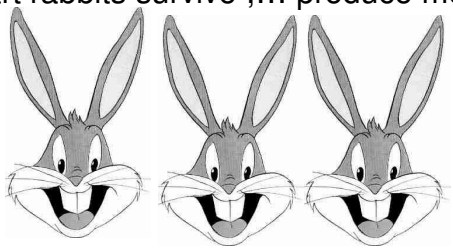
The Genetic Algorithm (Evolutionary Analogy)

- Consider a population of rabbits:
  - some individuals are faster and smarter than others
  - Slower, dumper rabbits are likely to be caught and eaten by foxes

  - Fast, smart rabbits survive ,… produce more rabbits.



Evolutionary Analogy

- The rabbits that survive generate offspring, which start to mix up their genetic material
- Furthermore, nature occasionally throws in a wild properties because genes can mutate
- In this analogy, an individual rabbit represents a solution to the problem(i.e. Single point in the space)
- The foxes represent the problem constraints (solutions that do more well are likely to survive)

Evolutionary Analogy

- Evolution Fundamental Laws: Survival of the fittest.
  - Change in species is due to change in genes over reproduction or/and due to mutation.
  - For selection, we use a fitness function to rank individuals of the population
  - For reproduction, we define a crossover operator which takes state descriptions of individuals and combine them to create new ones
  - For mutation, we can choose individuals in the population and alter part of its state.

The Genetic Algorithm

- Directed search algorithms based on the mechanics of biological evolution
- Developed by John Holland, University of Michigan (1970's)
- To design artificial systems software that retains the robustness of natural systems
- Provide efficient, effective techniques for search problems, optimization and machine learning applications
- Widely-used today in business, scientific and engineering circles

Terminology

- ***Evolutionary Computation (EC)*** refers to computer-based problem solving systems that use computational models of evolutionary process.
- ***Chromosome*** – It is an individual representing a candidate solution of the optimization problem.
- ***Population*** – A set of chromosomes.
- ***gene*** – It is the fundamental building block of the chromosome, each gene in a chromosome represents each variable to be optimized. It is the smallest unit of information.
- **Objective**: To find "a" best possible chromosome for a given problem.

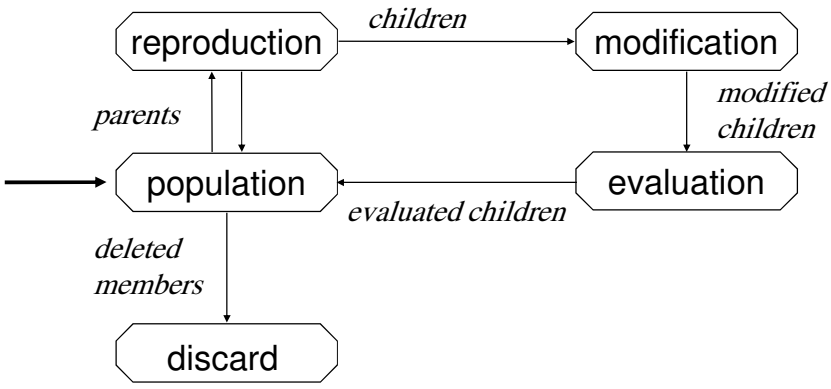
Overview of GAs

- GA emulate genetic evolution.
- A GA has distinct features:
  - A string representation of chromosomes.
  - A selection procedure for initial population and for off-spring creation.
  - A cross-over method and a mutation method.
  - A fitness function.
  - A replacement procedure.

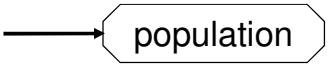
Overview of GAs

- Parameters that affect GA are:
  - initial population
  - size of the population
  - selection process and
  - fitness function

The GA Cycle of Reproduction



Chromosomes

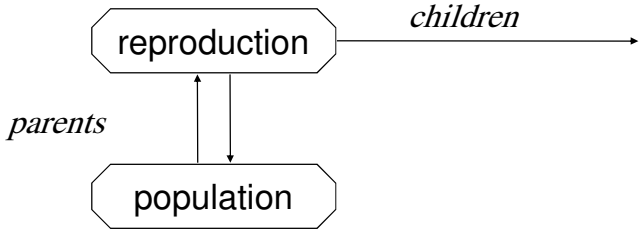


Chromosomes could be:

Bit strings	(0101 ... 1100)
Real numbers	(43.2 -33.1 ... 0.0 89.2)
Permutations of element	(E11 E3 E7 ... E1 E15)
Lists of rules	(R1 R2 R3 ... R22 R23)
Program elements	(genetic programming)

... any data structure ...

Reproduction



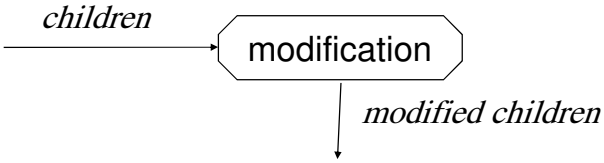
Reproduction is a processes of creating new chromosomes out of chromosomes in the population. Parents are "selected" at each iteration.

Selection Process

- Selection is a procedure of picking parent chromosome to produce off-spring.
- Types of selection:
  - Random Selection – Parents are selected randomly from the population.
  - Proportional Selection – probabilities for picking each chromosome is calculated as:

$$P(\mathbf{x}_i) = f(\mathbf{x}_i)/\Sigma f(\mathbf{x}_j) \quad \text{for all } j$$

Chromosome Modification



- Operator types are:
  - Mutation
  - Crossover (recombination)

Crossover

Cross-over : It is a process of creating one or more new individuals through the combination of genetic material randomly selected from two or parents.

Crossover is a critical feature of genetic algorithms:

- It greatly accelerates search early in evolution of a population
- It leads to effective combination of schemata (subsolutions on different chromosomes)

Cross-over

- Uniform cross-over : where corresponding bit positions are randomly exchanged between two parents.
- One point : random bit is selected and entire sub-string after the bit is swapped.
- Two point : two bits are selected and the sub-string between the bits is swapped.

	Uniform Cross-over	One point Cross-over	Two point Cross-over
Parent1 Parent2	00110110 11011011	00110110 11011011	00110110 11011011
Off-spring1 Off-spring2	01110111 10011010	00111011 11010110	01011010 10110111

Mutation: Local Modification

Before: (1 0 1 1 **0** 1 1 0)

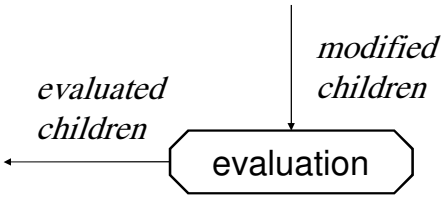
After: (1 0 1 1 **1** 1 1 0)

Before: (1.38 **-69.4** 326.44 0.1)

After: (1.38 **-67.5** 326.44 0.1)

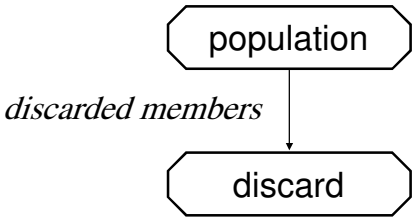
- Causes movement in the search space (local or global)
- Restores lost information to the population
- Prevents falling all solutions in population into a local optimum.

Evaluation



- The evaluator decodes a chromosome and assigns it a fitness measure

Deletion



- ***Generational*** GA:
  entire populations replaced with each iteration
- ***Steady-state*** GA:
  a few members replaced each generation

Evolutionary Algorithm

Let t = 0 be the generation counter;

create and initialize a population *P(0)*;

repeat

Evaluate the fitness, f($\mathbf{x_i}$), for all $\mathbf{x_i}$ belonging to *P(t)*;

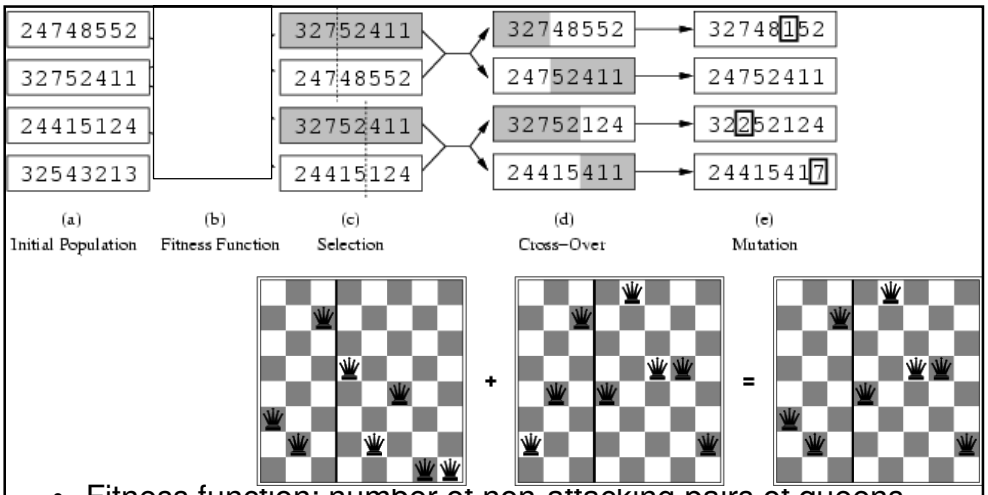
Perform cross-over to produce offspring;

Perform mutation on offspring;

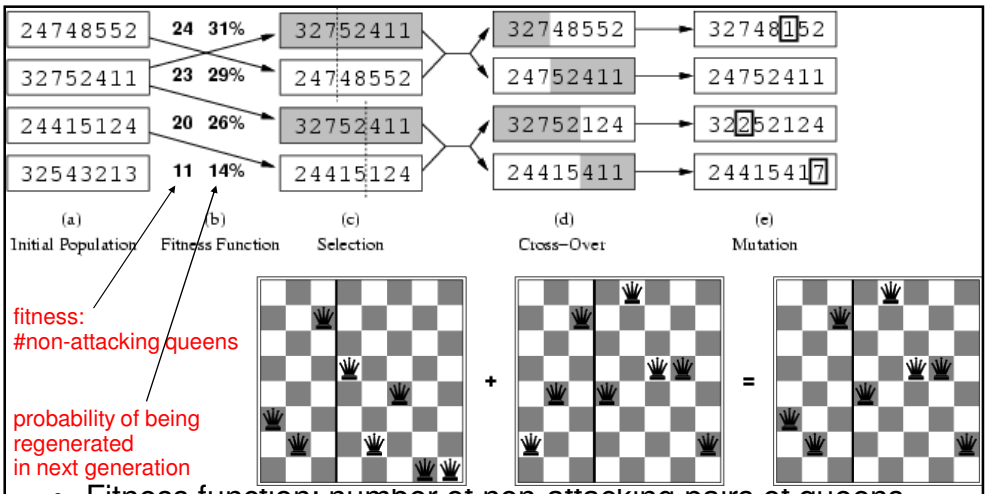
Select population *P(t+1)* of new generation;

Advance to the new generation, *i.e.,* t = t+1;

until *stopping condition is true;*



- Fitness function: number of non-attacking pairs of queens (min = 0, max = 8 × 7/2 = 28)



- Fitness function: number of non-attacking pairs of queens (min = 0, max = 8 × 7/2 = 28)
- P(child) = 24/(24+23+20+11) = 31%
- P(child) = 23/(24+23+20+11) = 29% etc

Creativity in GA

- ✓ GAs can be thought of as a simultaneous, parallel hill climbing search --- The population as a whole is trying to converge to an optimal solution
- ✓ Because solutions can evolve from a variety of factors, very novel solutions can be discovered

A list of AI Search Algorithms

Systematic Search algorithms

- BFS, DFS,...
- A*
  - AO*
  - IDA* (Iterative Deepening)

Local Search Algorithms

- Minimax Search on Game Trees
- Viterbi Search on Probabilistic FSA
- Hill Climbing
- Simulated Annealing
- Gradient Descent
- Stack Based Search
- Genetic Algorithms
- Memetic Algorithms