Genetic Algorithms

Outline

Part One:

Overview of genetic algorithms

Theoretical properties

Implementation issues

Sources

Induction, John Holland, Keith Holyoak, Richard Nisbett and Paul Thagard, Univ. Michigan, 1986.

Genetic Algorithms and Simulated Annealing, Lawrence Davis, ed., Pitman Publishing, 1987.

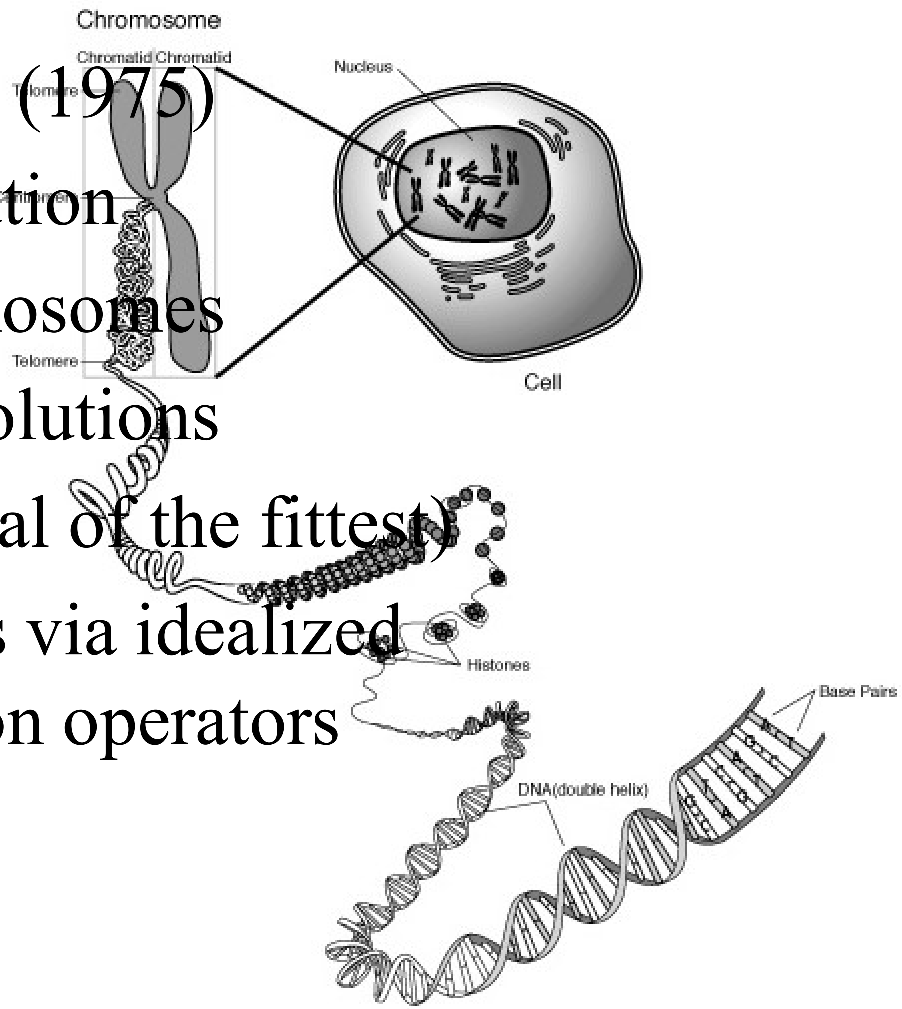
Genetic Algorithms in Optimization, Search and Machine Learning, David Goldberg, Addison Wesley, 1989.

Handbook of Genetic Algorithms, Lawrence Davis, Van Nostrand Reinhold, 1990.

- Tutorial
- Applications
- Optional Diskette with two GA Systems

Main Ideas

- Developed by John Holland (1975)
- Inspired by biological evolution
- Solutions encoded as chromosomes
- Maintains a population of solutions
- Biased reproduction (survival of the fittest)
- Generation of new solutions via idealized recombination and mutation operators



http://www.accessexcellence.org/AB/GG/chromosome.html

When to Use Genetic Algorithms

Problem requires good, but not optimal solution

Acceptable performance measure is available

Feasible to test many potential solutions

Acceptable representation is available

When to Use Genetic Algorithms

Size and complexity of search space preclude traditional approaches:

- Analytic solution
- Exhaustive search
- Hill-climbing
- Random search

Subdivisions of the Field

- Basic genetic algorithms
- Applied genetic algorithms
- Classifier systems
  - A class of parallel rule-based systems
  - GAs used to discover new rules
  - See (Holland et al, 1986)

Components of a Genetic Algorithm

An encoding technique (“chromosome structure”)

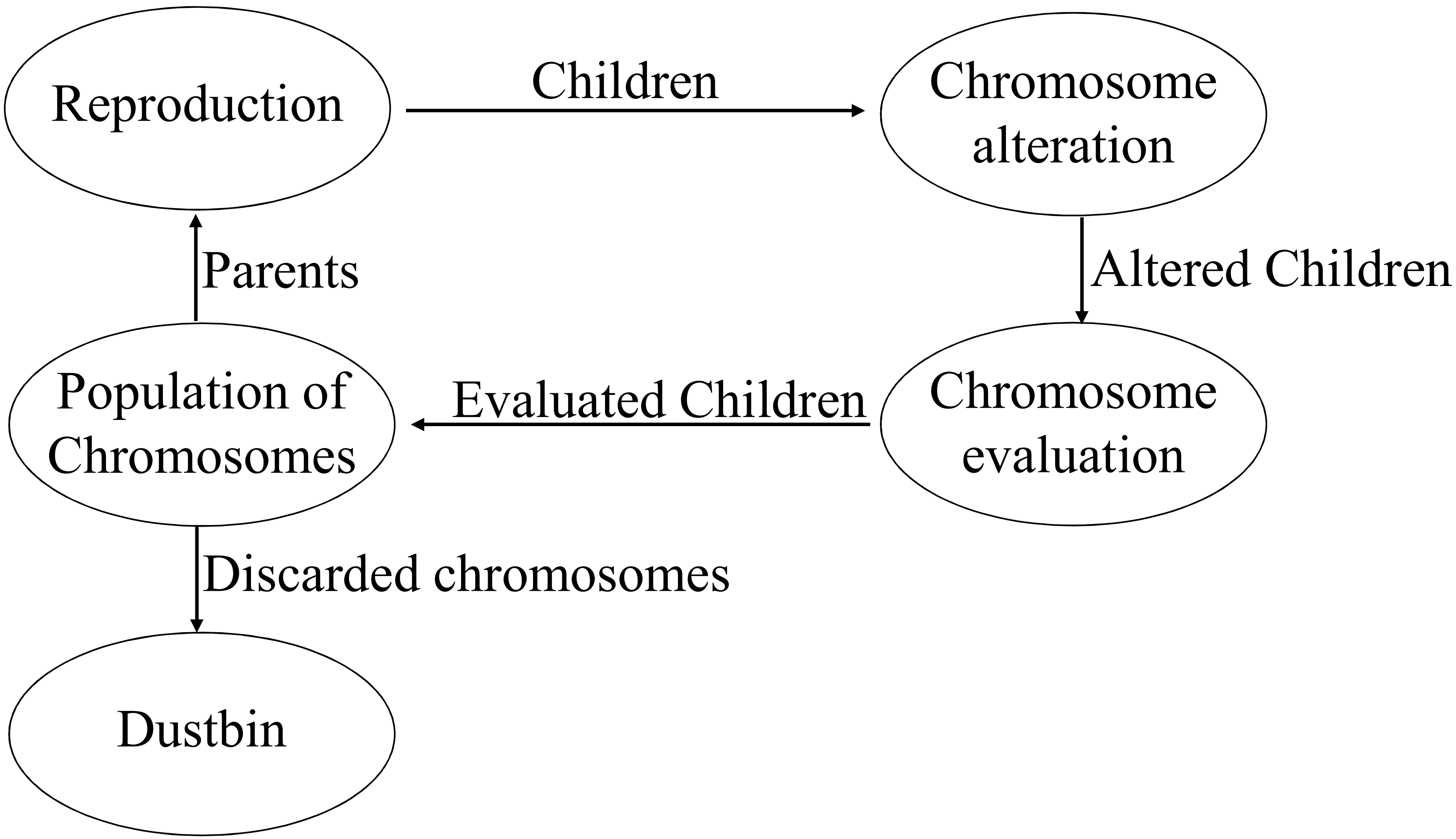
An evaluation function (“the environment”)

An initialization procedure (“the creation”)

Genetic operators (“mutation, crossover, etc.”)

Parameter settings (practice and art)

How a Genetic Algorithm Works

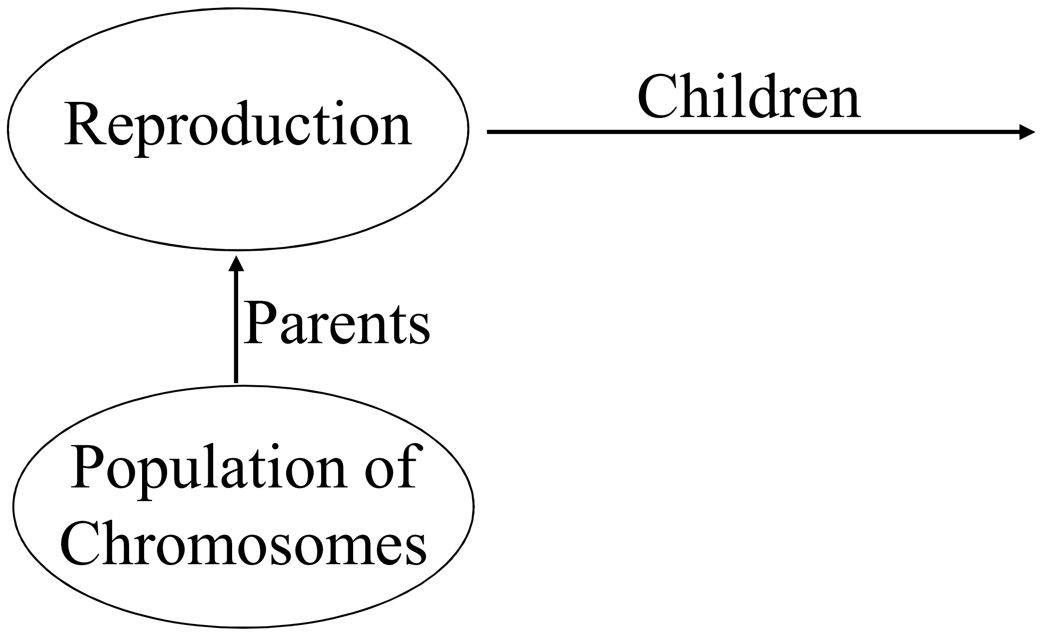


The Population

Chromosome might be:

- Bit strings (0110…1011)
- Real number lists (23.6 –2.45 … 3.1 0.0)
- Permutations of elements (E8 E3 … E1 E7)
- List of rules (R1 R2 R3 … R32)
- Etc.

Reproduction



Parents are selected with probability biased toward chromosomes with better evaluations

Selection of Parents

Population	f(x)	C(x)	children
chromosome1	10	0.25	0
chromosome2	100	2.50	3
chromosome3	25	0.625	1
chromosome4	50	1.25	1
chromosome5	40	1.00	1
chromosome6	15	0.375	0

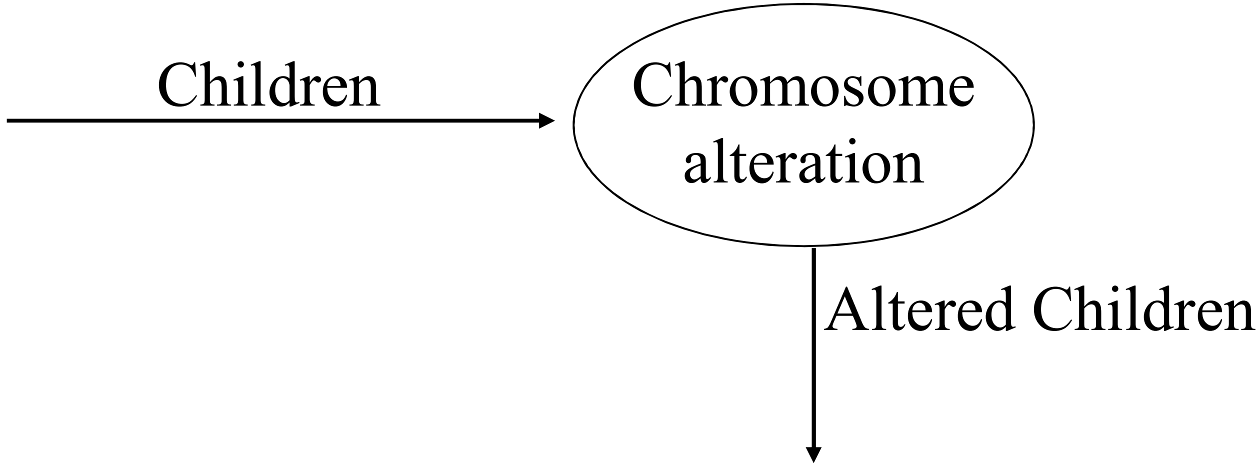
f(x)= fitness

C(x)= Expected number of children

$$C(x) = \frac{f(x)}{f(Pop)}$$

f(Pop) = 240/6 = 40

Chromosome Alteration



Alterations are stochastically triggered.

Crossover operators cause exchange of genetic material between two parents.

Mutation operator cause local alteration in a single chromosome.

Crossover

(0 1 1 0 1 1 0)

(1 1 0 1 1 0 1)

(0 1 0 1 1 1 0)

(1 1 1 0 1 0 1)

Probably most important feature of genetic algorithms.

Greatly accelerates search early in the evolution of a population.

Leads to effective combination of partial solutions on different chromosomes.

Mutation

(0 1 1 0 1 1 0)

(0 1 0 0 1 1 0)

(2.67 –49.3 128.68 33.1)

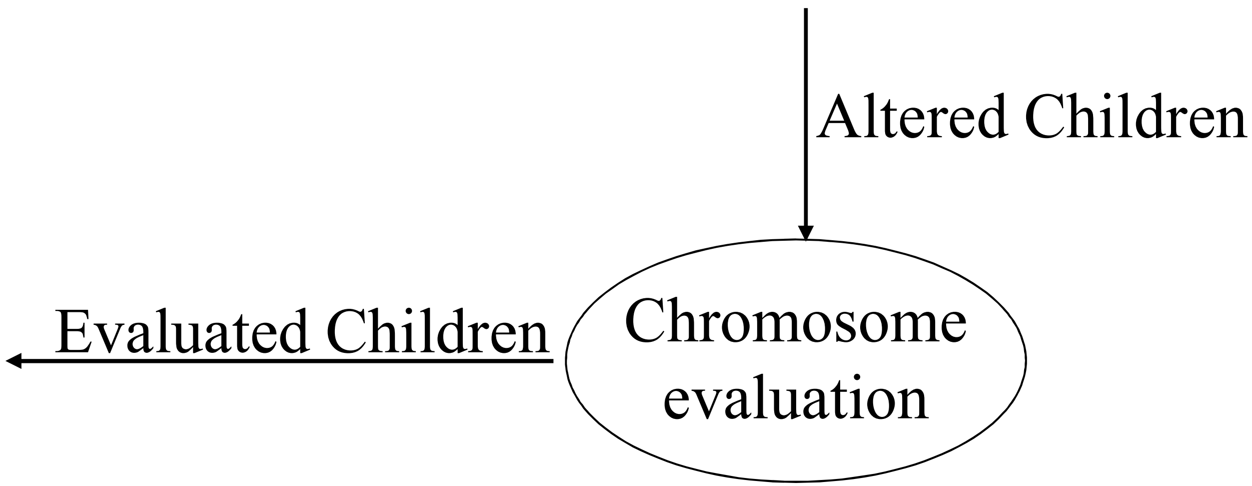
(2.67 –47.6 128.68 33.1)

Mutation causes movement in the search space

Movement can be global or local

Restores lost information to the population

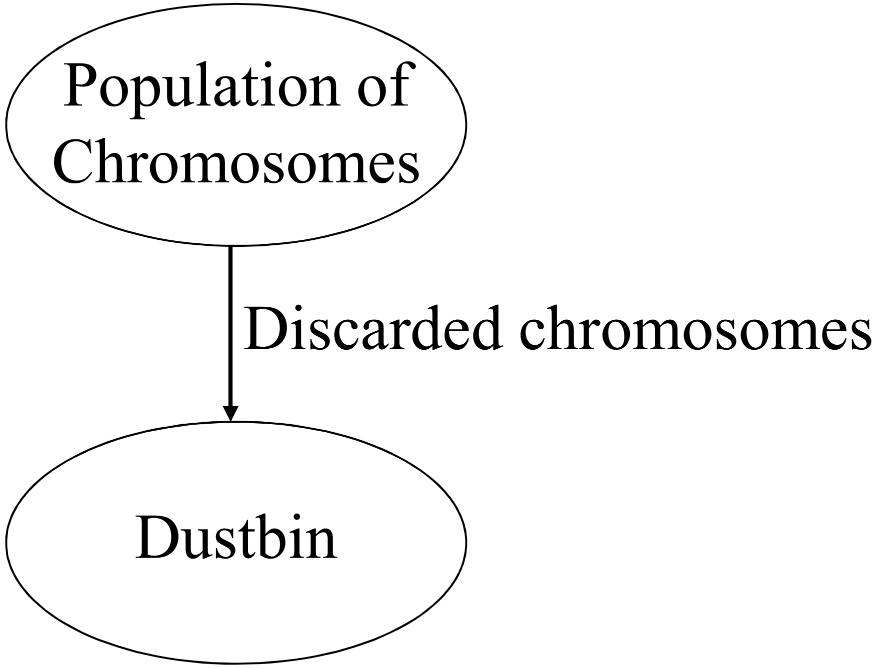
Evaluation



The evaluator decodes a chromosomes and assigns it a fitness measure.

The evaluator is the only link between a (pure) genetic algorithm and the problem it is solving.

Deletion

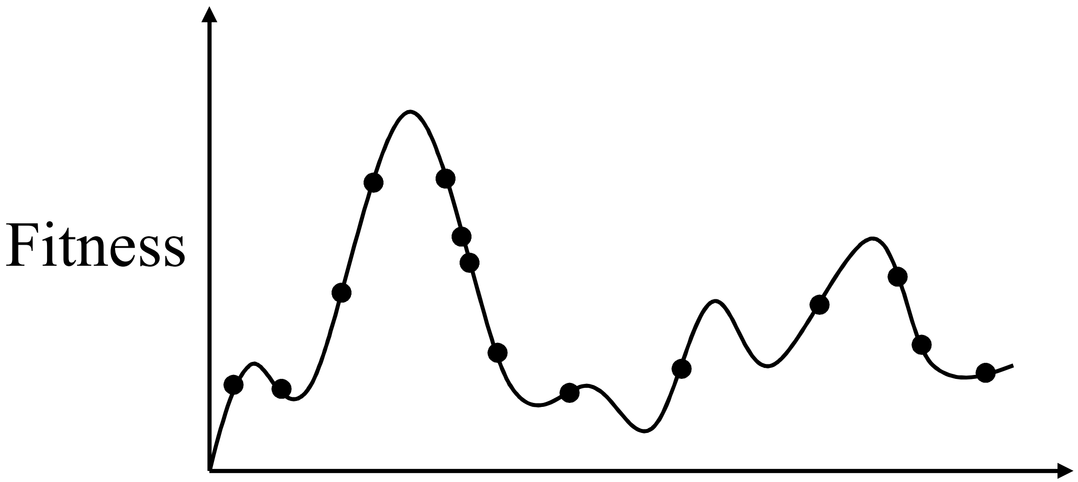


Two approaches:

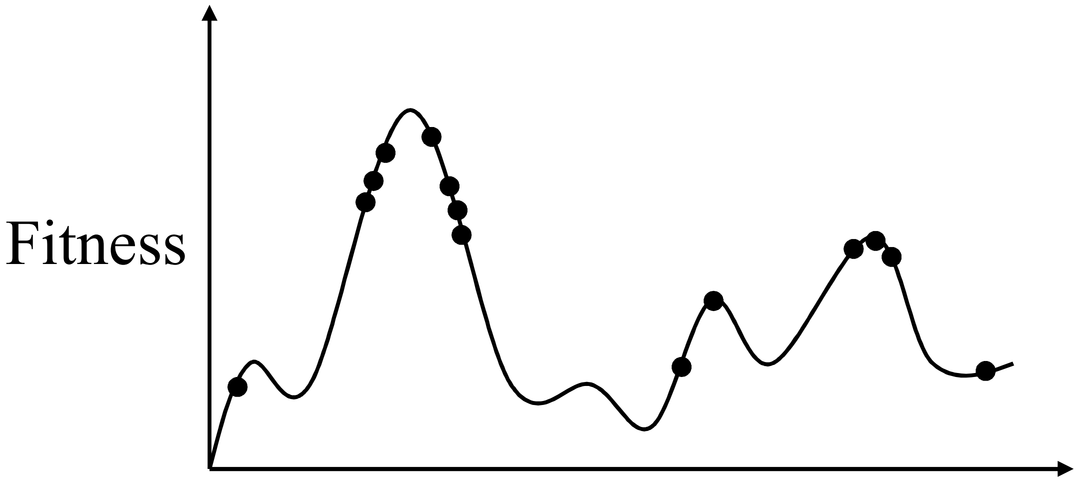
Generational: Entire population is replaced at each iteration.

Steady state: A few members are discarded at each iteration.

An Abstract Example



Distribution of individuals in Generation 1



Distribution of individuals in Generation 10

Principles Underlying Genetic Algorithms

Focus of attention on schemas (patterns in representation space)

Implicit parallelism

Can use noisy evaluation function

Some problems are deceptive

Suitable for parallel processing

Focus on Schemas

Examples:

	P(t)	f(x)	P(t+1)	f(x)
x1:	011010	1	011010	1
x2:	100111	0	011000	1
x3:	110010	0	000110	3
x4:	011000	1	000110	3
x5:	000110	3	000110	3
x6:	000111	1	000111	1
x7:	110110	0	101001	2
x8:	101001	2	101001	2

Selection Rule: The number of children is proportional to a chromosome's relative performance.

What is the effect on the patterns in the population?

Implicit Parallelism

Theorem: The number of representatives from any *schema* S increases in proportion to the observed relative performance of S.

Let S = 0#####

Let N(S,t) be number of elements of S at t.

Then f(S,t) = (1+1+3+1)/4 = 1.5

So, N(S,t+1) = 1.5 * N(S,t)

A large number of schema are processed without explicit computation of utilities.

Noisy Evaluation Functions

- The genetic algorithm estimates the value of key building blocks from evaluation of individual chromosomes.
- The estimated value may be accurate even if the individual evaluations are very noisy (See Fitzpatrick and Grefenstette, Machine Learning Journal, 1988).
- Good for Monte Carlo evaluation functions
  - Image processing
  - Machine learning (SAMUEL)

Deceptive Problems for Genetic Algorithms

Some (problem, representation) pairs may fail

Misleading building blocks

Classes of deceptive problems have been defined and analyzed. (See Goldberg, 1989)

In practice, a change of representation can often reduce deceptiveness

Comparison with Simulated Annealing

SA	GA
1 structure	population
perturbation	mutation
acceptance rule	selection rule
temperature	selection pressure
Boltzmann distribution	implicit parallelism
?	recombination

Both methods based on analogies with natural systems

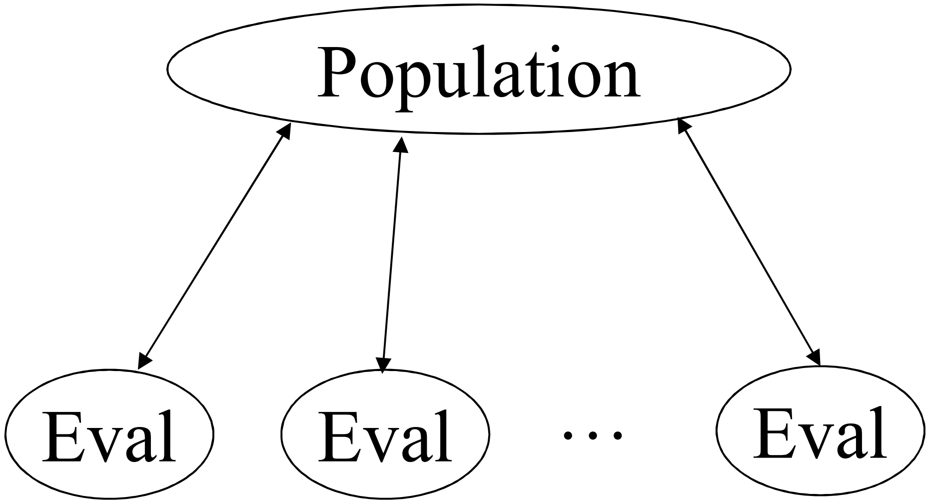
GA more likely to handle discontinuous, noisy functions

Comparison with Simulated Annealing

Selection Pressure

Selection pressure can be defined as the level of selectivity of the GA, that is its tendency to select only the best individuals in the current generation, either as parents or for survival. On one hand, an excessive selection pressure can have a negative impact on the genetic **diversity** and thus lead the algorithm towards a local rather than the global optimum. On the other hand, insufficient selection pressure will slow down **convergence** (Goldberg and Deb, 1991). For these reasons most GAs avoid taking only and exclusively the fittest of the population for reproduction and discarding all others. Such a strategy maintains a higher genetic diversity in a population because even the less fit individuals have a chance to reproduce. This enables the algorithm to better explore the search space and possibly find excellent solutions in unexpected regions, instead of concentrating on a narrow area. The actual amount of selection pressure depends on the implemented method.

Suitable for Parallel Processing



Evaluation can proceed in parallel

Evaluation is usually the main bottleneck

Implementation Issues

Representation

Evaluation function

- Scaling methods

Population size

Alternative crossover operators

Representation

Numeric parameters are most easily represented.

- bit strings, lists of reals.

Combinatorial representations have been developed (Davis, Whitley).

GAs have been applied to non-linear chromosomes.

- LISP expressions (Koza)
- Rule sets (Grefenstette)

Evaluation Function Design

- Evaluation function should be informative and have regularities
  - Avoid completely chaotic and “needle-in-haystack” functions
- Must reward desired behavior
  - GAs are opportunistic
- Evaluation may be stochastic or noisy
- Need not be low-dimensional, continuous, differentiable, or unimodal

Population Size

- Must provide reasonable diversity in initial stage
- Avoid small populations (< 30 chromosomes)
  - stochastic effects take over
  - premature convergence
  - inadequate information for implicit parallel search.
- Typical size: 50 – 200 chromosomes

Reproductive Alternatives

Proportional selection

$$C(x) = f(x) / f(Pop)$$

where C(x) is the expected number of children of x

f(x) is the fitness of x

f(Pop) is the mean fitness of the population

Selection Alternatives

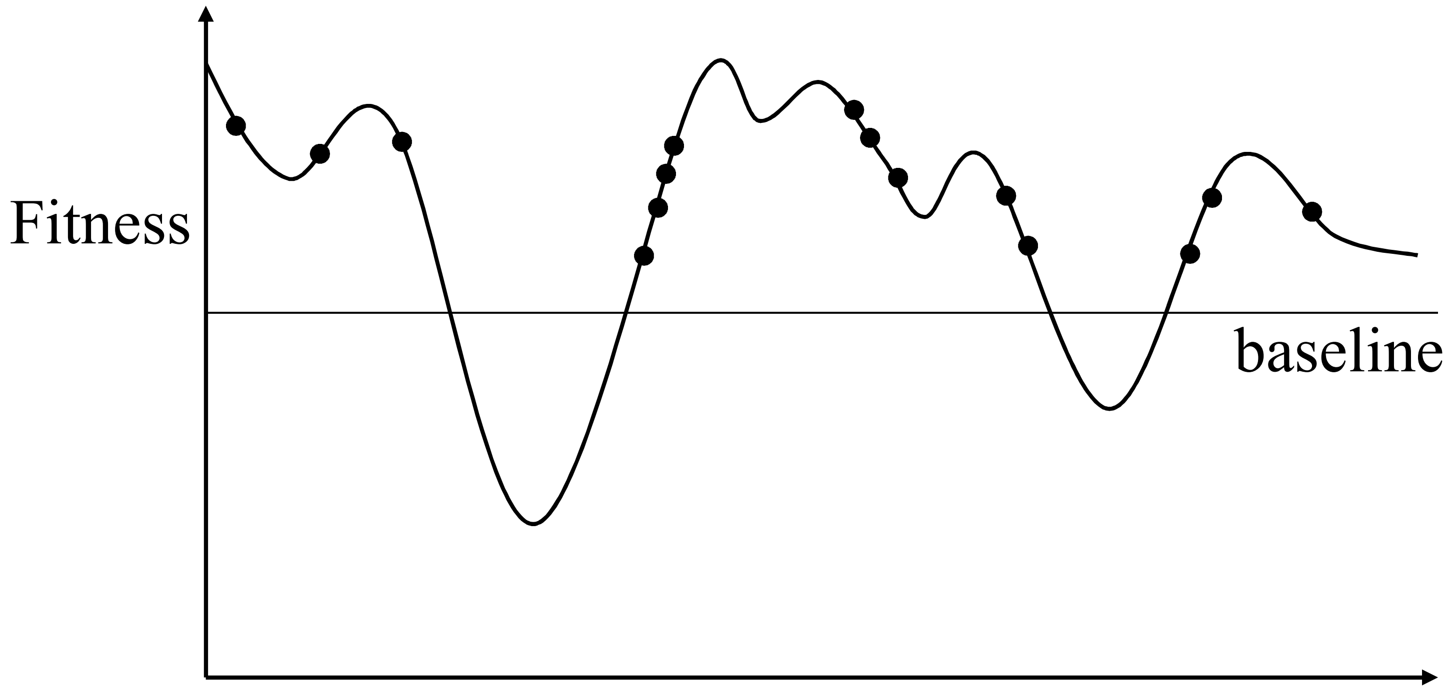
Rank-based selection

$$C(x) = a + b * rank(x)$$

where rank(x) is x’s relative position within population, 0 for worst, 1 for best.

Selection based on local competition

Scaling



Raise the baseline as search proceeds to keep selective pressure high

Alternative Crossover Operators

1-point, 2-point widely used

Uniform: take each "gene" from each parent with probability p, (1-p)

(3.5 12.4 2.0 3.0 10.5 12.3)

(6.0 33.3 5.0 6.0 20.0 55.2)

(3.5 33.3 5.0 3.0 10.5 55.2)

(6.0 12.4 2.0 6.0 20.0 12.3)

Many specialized versions for other representations

Applications of Genetic Algorithms

Parametric Design of Aircraft

Routing in Telecommunications Networks

Robot Path Planning

Nonlinear Dynamical Systems and Models of International Security

Learning Rules for Reactive Systems

Synthesis of Neural Network Architectures

Optimization of Air-Injected Hydrocyclone

Multiple Fault Diagnosis

Conformational Analysis of DNA

Sonar Information Processing

Engineering Design Optimization

Scheduling

Summary of Genetic Algorithms

- General purpose search method requiring little domain knowledge
- Implicit parallel search of solution space
- Idealized recombination operators combine good partial solutions
- Population of candidate solutions provides robust search in complex spaces where other search methods fail.