Slide 0

Evolutionary Computation

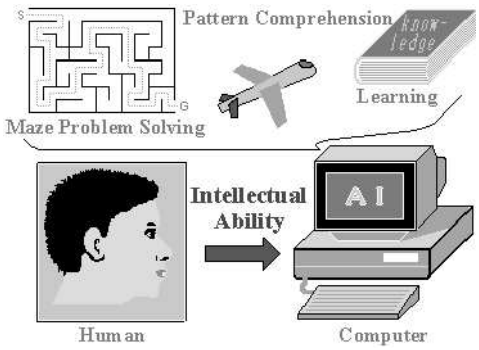
오늘의 교훈

극기하는자 실패도 약이요. 그렇지 못한자 성공도 병이다

What is AI

- logical programming techniques to put human's intellectual activities into a computer
- When new situations occur, they have the ability to correspond to that based on accumulated knowledge.
- AI can put those intellectual activities or abilities of the human into practice by using a computer.

Slide 1



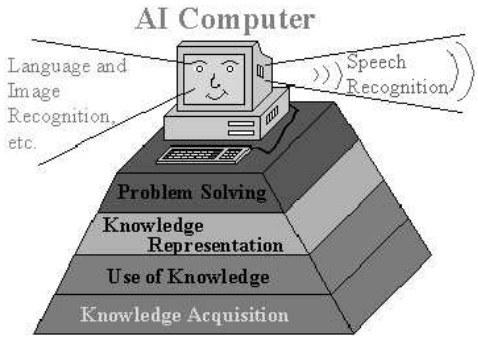
What is AI (계속)

Slide 2

- The goal of AI research is to bring the human's intellectual abilities into practice.
- Therefore, four basic skills are very important in AI, such as:
  - Problem solving
  - Knowledge representation of problems
  - Use of knowledge
  - Knowledge acquisition
  - Furthermore, language, speech, and images are the communication part of knowledge and should be understandable.

What is AI (계속)

Slide 3

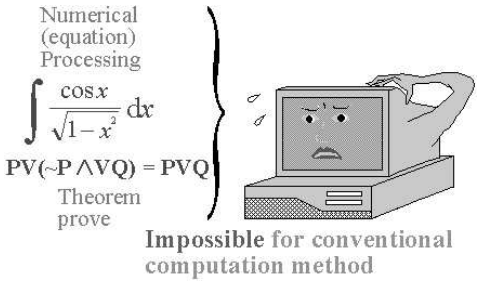


- Problem Solving
  - The object problem for AI is the one which its data are not numerical values but symbols, also its solution algorithms has indefinite properties.
  - For instance, "go out of the maze", "win the chess", "processing the numerical equation", "prove the theorem", etc.
  - It is very difficult for the conventional computation methods to deal with

What is AI (계속)

Slide 4

such problems.

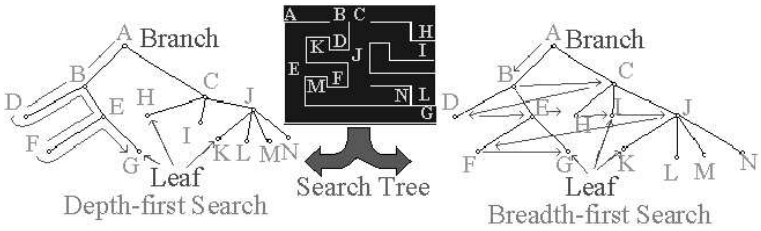


- In most search methods, the problem is represented as a tree and the problem solving is started from its root.
- Moving along the branches until it reaches the leaf that is considered an acceptable solution of the problem.
- Therefore, the objective is to find the optimal route from the root to the leaf.
- There are two basic search methods: depth-first search and breadth-first search.

What is AI (계속)

Slide 5

- Let's look at the maze problem as an example.



- areas
  - speech, image, and natural language recognition
  - autonomous and intelligent control
  - decision making systems

What is AI (계속)

Slide 6

- tools
  - expert systems
  - neural networks
  - fuzzy logic
  - evolutionary computation
- history
  - in 1956, the terminology "AI" first appeared
  - in 1960, AI language "LISP" was invented
  - in 1981, "fifth generation computer" project started in Japan
  - in 1983, AI boom had occured
  - in 1990s, new tools such as NN, fuzzy, and EC have been put into practice

What is Evolutionary Computation

Slide 7

- based on *Neo-Darwinian Paradigm*, i.e., *Natural Selection*
- Darwin's Evolution

  Charles Robert Darwin (1809 - 82) mainly developed the first theory of evolution. Darwin's main point was that evolution occurs as the fittest species are naturally selected to survive and reproduce.
- encompasses methods of simulating evolution on a computer

  computer-based problem solving systems which use computational models of some of the known mechanisms of EVOLUTION as key elements in their design and implementation
- population-based optimization process
- systematic multi-agent search paradigms

What is Evolutionary Computation (계속)

Slide 8

- four main processes
  - Reproduction
    ⟹the creation of a new individual from two parents
  - Crossover
    ⟹a reproduction operator which forms a new chromosome by combining parts of each of two 'parent' chromosomes
  - Mutation
    ⟹a reproduction operator which forms a new chromosome by making (usually small) alterations to the values of genes in a copy of a single, parent chromosome
  - Selection
    ⟹the process by which some INDIVIDUALs in a POPULATION are chosen for REPRODUCTION
    ⟹typically on the basis of favoring individuals with higher FITNESS.

What is Evolutionary Computation (계속)

Slide 9

- three main avenues of research in evolutionary computation
  - genetic algorithms
  - evolutionary programming
  - evolution strategies
- the other research areas
  - genetic programming

GP is the extension of the genetic model of learning into the space of programs. These programs are expressed in GP as parse trees, rather than as lines of code. When these programs are executed, they produce candidate solutions to a problem.

What is Evolutionary Computation (계속)

Slide 10

- artificial life

  Artificial Life (A-Life) is a new technique which aims to find a logical modeling of living objects in life including human, animals, etc. In the research of A-Life, the scientists are working on the behavior and transition of genetics from the beginning of a generation including the process of learning, evolution, etc. These processes and phenomena are common to GA's processes, therefore GAs are used for simulation of A-Life.

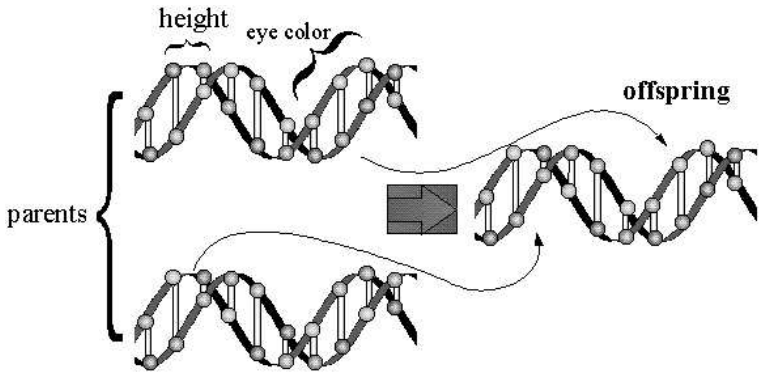
- classifier system (CFS)

  Classifier Systems are added by John Holland to improve GAs. It works like this: The GA begins by assigning strengths to rules (called classifiers), then selects pairs of high strength classifiers to be parents. The system eliminates weaker rules, and the stronger ones compete among themselves.

Fundamental Words

Slide 11

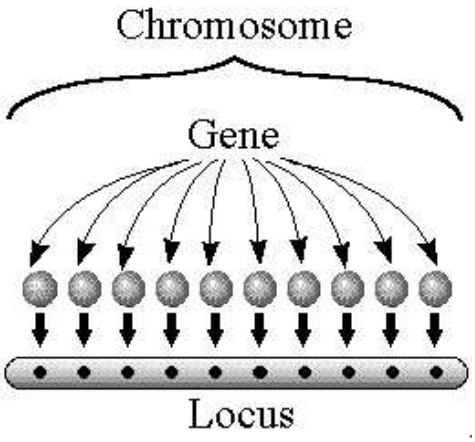
- Gene
  - Gene is a short length of a chromosome which controls a characteristic of an organism.



Fundamental Words (계속)

Slide 12

- Locus
  - Locus is the placement of genes on the double stranded chromosomes.
  - Each place is reserved for a specific gene



- Chromosome (a chain of genes)

Fundamental Words (계속)

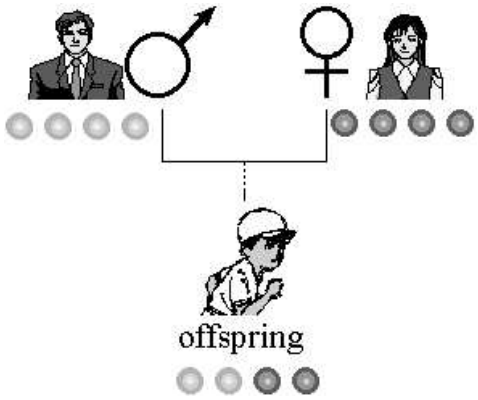
Slide 13

- each living object has a particular number of chromosomes (human beings have 46 chromosomes)
- all chromosomes are in pairs (human being has 23 pairs) in the nucleus of every cell in body

Fundamental Words (계속)

Slide 14

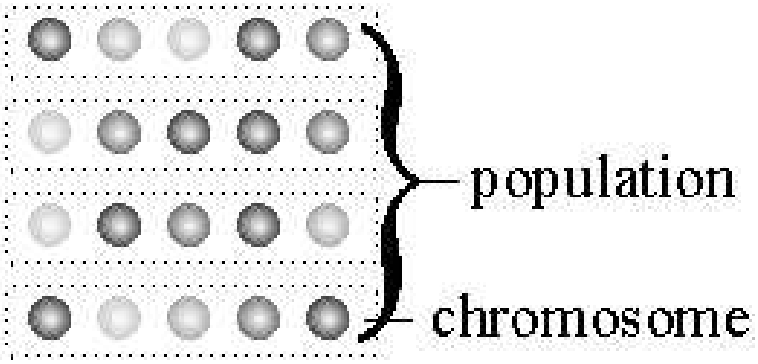
- Offspring
  - The new chromosomes are reproduced by their parents
  - These new chromosomes are called offspring



Fundamental Words (계속)

Slide 15

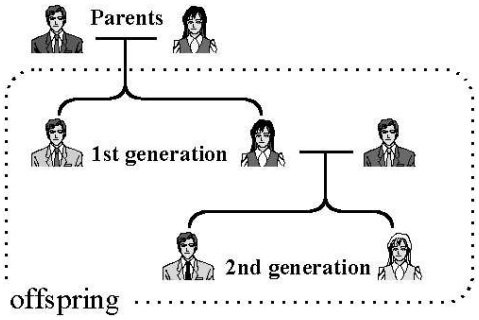
- Population Size
  - Population size is the number of chromosomes in a generation.



Fundamental Words (계속)

Slide 16

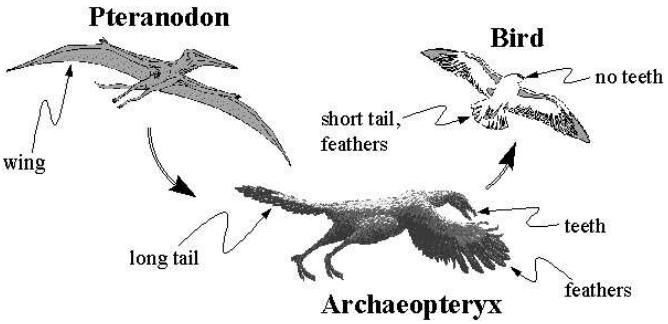
- Generation
  - A single stage in a family
  - the parents from one generation and their offspring the next generation.



Fundamental Words (계속)

Slide 17

- Evolution
  - The changes over many generations
  - different kinds of organisms have arisen from very early forms.



Fundamental Words (계속)

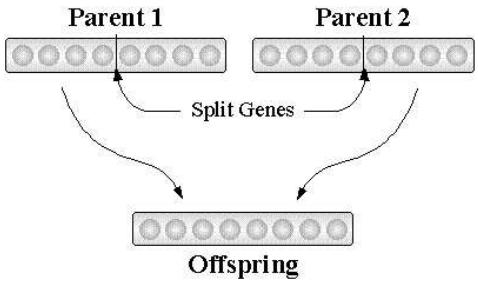
Slide 18

- GA Process
  - GA is based on 4 processes
    - ∗ initialization
      GA starts by choosing a set of possible solutions.
      They are generated in a random way for each independent variable.
    - ∗ fitness evaluation
      This stage involves checking the answers to see how good they are.
    - ∗ mating (cross-over)
      This is the most important stage in GA: two key components
      selection
      decision of which individual solutions should be mated to form the next set
      good individuals have to be more likely selected than bad ones

Fundamental Words (계속)

Slide 19

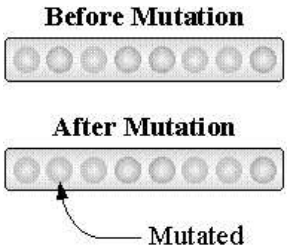
crossover
taking a part of chromosomes from each individual and makes a new solution



Fundamental Words (계속)

Slide 20

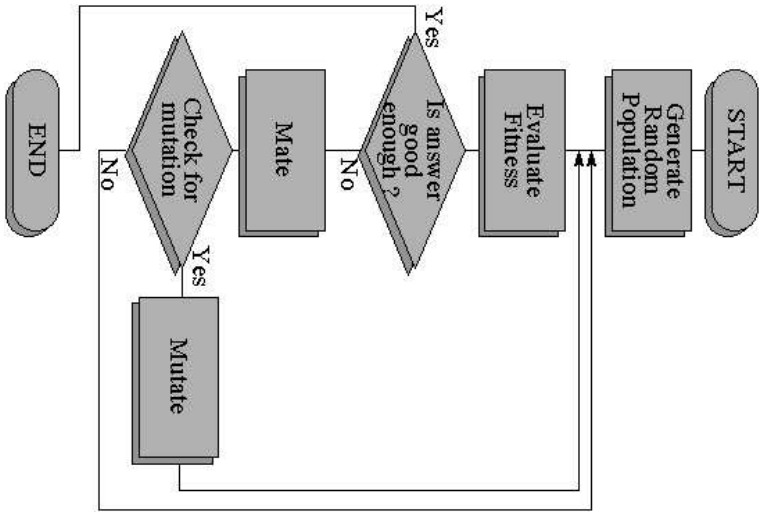
* mutation
  allows a solution to change in a random manner



Fundamental Words (계속)

Slide 21

- Evolutionary Cycle
  - continues the cycles until stopping criteria is reached



What are genetic algorithms

Slide 22

- invented by John Holland in the early 1970s
- techniques for optimization and machine learning
- metaphor of the process of evolution
- solutions are encoded as chromosomes
- multi-directional search based on the mechanics of natural section and natural genetics
  - maintenance a set (population) of solutions

What are genetic algorithms (계속)

Slide 23

- the reasons behind the growing numbers of applications
  - computationally simple, powerful search algorithm
  - no limits of assumptions about the search space
    - continuity
    - existence of derivatives
    - unimodality
    - and other matters
- features of natural evolution
  - evolution operates on chromosomes
  - evolution is not a purposive or directed process
    $\Longrightarrow$application independent methods
  - natural selection
    the better one is, the more likely to survive

What are genetic algorithms (계속)

Slide 24

- the initial inspiration
  - encode problem parameters on chromosome
  - select fitness measure of chromosome
  - construct initial population of chromosomes
  - using genetic operator, make the population to evolve
- five components of a GA
  - an encoding technique—chromosome structure
  - genetic operators—crossover, mutation
  - an evaluation function—fitness value
  - an initialization procedure—creation of population
  - parameter settings—practice and art

Comparison of search and optimization techniques

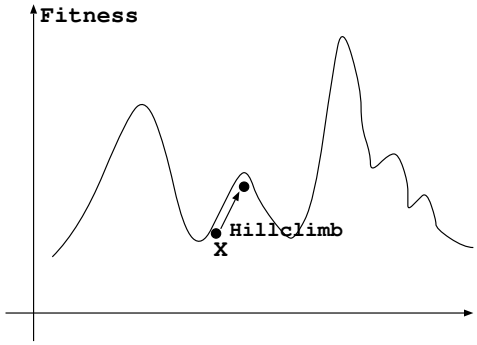
Slide 25

- Three types of search methods
  - calculus-based
  - enumerative
  - random
- Calculus-based search methods (ref. book 1)
  - two types
    - indirect
      1. seek local extrema
      2. by solving the usually nonlinear set of equations
      3. setting the gradient of the objective function equal to zero

Comparison of search and optimization techniques (계속)

Slide 26

* direct
  1. seek local optima
  2. by hopping on the function and moving in a direction related to the local gradient
     $\Longrightarrow$*hillclimbing* $\Longrightarrow$show fig. 1.1
     (a) as a gradient method
     (b) use of gradient to guide the direction of search
     (c) perform well on unimodal functions



Comparison of search and optimization techniques (계속)

Slide 27

- limitations
  * lack of robustness $\Longrightarrow$can fall into local minima
  * depend on the existence of derivatives (well-defined slope values)
  * real world of search is discontinuities and vast multimodal, noisy search spaces
    $\Longrightarrow$show fig. 1.2 and 1.3

- Enumerative search methods
  - lack of efficiency
    * many practical spaces are too large to search
- Random search methods
  - very unintelligent strategy and rarely used
- Guided random search methods
  - simulated annealing
  - evolutionary computation
- Simulated Annealing
  - an analogy the annealing process

Comparison of search and optimization techniques (계속)

Slide 28

- search for a minimum in a more general system.
- major advantage over other methods is an ability to avoid becoming trapped at local minima
  - not only accepts a good solution, but also some changes that accepts a bad solution
  - bad solution are accepted with a probability
    1. $p = exp(-\delta f/T)$
    2. where δf is the increase in f and T is a control parameter, 'temperature' in annealing process

Comparison of search and optimization techniques (계속)

Slide 29

- GAs differ from traditional optimization algorithms in four important respects:
  - using an encoding of the control variables, rather than the variables themselves.
  - searching from one population of solutions to another, rather than from individual to individual.
  - using only objective function information, not derivatives.
  - using probabilistic, not deterministic, transition rules.
- building block hypothesis
  - a small, tightly clustered group of genes which have co-evolved will be likely to give increased fitness to chromosomes

Comparison of search and optimization techniques (계속)

Slide 30

- Exploration and exploitation
  - exploration
    - investigation of new and unknown areas in search space
      →purely random search is good example
      →not guarantee the convergence
  - exploitation
    - make use of knowledge found at points previously visited to help find better points
      →hillclimbing methods is good example
      →not guarantee the global optimum
  - combinations of two strategies
    - can be quite effective, but it is difficult to know where the best balance lies
  - GA combines two strategies at the same time in an optimal way →really?

The goals of optimization

Slide 31

- two parts
  - process of improvement
  - destination or optimum itself
- the most important goal of optimization is improvement
  →to find some good level of performance quickly
  →in complex systems, the optimum itself is much less important

A simple function optimization example

- $f(x) = x^2$ on the integer interval [0,31]
- application steps

step 1 : coding the parameter x as a finite-length string

Slide 32

step 2 : initialization of the population (solutions)

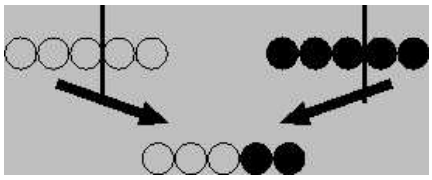
step 3 : evaluate fitness using objective function

step 4 : perform three operators

* reproduction (table 1.1 and fig. 1.7)
  1. selection of individual according to there objective function values (fitness)
  2. one example →roulette wheel selection

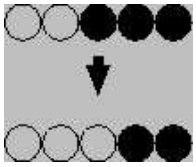
A simple function optimization example (계속)

* crossover (fig. 1.8)



Slide 33

* mutation



MUTATE 1 1 0 0 0

1 1 1 0 0

step 5 : goto step 3 until a sufficient solution is reached

Schema Theorem

Slide 34

- what is schema?
  - a similarity template describing a subset of strings with similarities
  - the meaning of the schema →a pattern matching device
  - binary alphabet $0, 1$ →ternary alphabet $0, 1, *$
  - a schema matches a particular string if at every location in the schema a 1 matches a 1 in the string, a 0 matches a 0, or a * matches either.
  - ex.) 10*11* schema matches $100110, 100111, 101110, 101111$ strings
  - k alphabets of cardinality →$(k+1)^l$ schemata with string length l

Schema Theorem (계속)

Slide 35

- Schema in GA
  - a string population with n members →$n \cdot 2^l$ schemata
  - define the order of a schema H
    - the number of fixed positions
    - ex.) $H = 011 * 1 * *$ →$o(H) = o(011 * 1 * *) = 4$
  - define the length of a schema H
    - the distance between the first and last specific string position
    - ex.) $H = 011 * 1 * *$ →$\delta(H) = 5 - 1 = 4$

Schema Theorem (계속)

Slide 36

- the effect of reproduction
  - Let m examples of a particular schema H at time step t exist
  - then, $\Longrightarrow m = m(H,t)$
  - a string A_i gets selected with probability $p_i = f_i / \sum f_j$
  - at time step $t+1 \rightarrow m(H,t+1) = m(H,t) \cdot n \cdot f(H) / \sum f_j$ where n is population size and $f(H)$ is average fitness of strings representing schema H at time t.
  - Let $\bar{f} = \sum f_j / n$ be the average fitness of the entire population
  - then,

$$m(H,t+1) = m(H,t)\frac{f(H)}{\bar{f}}$$

  - Let a particular schema H remains above average an amount $c\bar{f}$ with c a constant
  - then,

$$m(H,t+1) = m(H,t)\frac{\bar{f}+c\bar{f}}{\bar{f}} = (1+c)\cdot m(H,t)$$

  - At time step t starting at $t=0$, $m(H,t) = m(H,0)\cdot(1+c)^t$

Schema Theorem (계속)

Slide 37

- the effect of crossover
  - at $H_1 = *1****0$ and $H_2 = ***10**$
  - H_1 is less likely to survive than H_2 by crossover
  - destroy probability of H_1 : $p_d = \delta(H_1)/(l-1) = 5/6$
  - survival probability of H_2 : $p_s = 1 - p_d = 5/6$
  - survival probability of H : $p_s = 1 - \delta(H)/(l-1)$
  - with crossover probability p_c :

$$p_s \geq 1 - p_c \cdot \frac{\delta(H)}{l-1}$$

  - as a result

$$m(H,t+1) \geq m(H,t)\cdot\frac{f(H)}{\bar{f}}\left[1 - p_c\cdot\frac{\delta(H)}{l-1}\right]$$

Schema Theorem (계속)

Slide 38

- the effect of mutation
  - survival probability of one 1 or 0 with mutation probability p_m : $(1 - p_m)$
  - survival probability of one schema : $(1 - p_m)^{o(H)}$
  - for small values of $p_m (p_m << 1)$: $(1 - p_m)^{o(H)} \approx 1 - o(H) \cdot p_m$
  - finally, ignoring small cross-product terms

$$m(H, t+1) \geq m(H, t) \cdot \frac{f(H)}{\bar{f}} \left[1 - p_c \cdot \frac{\delta(H)}{l-1} - o(H) p_m \right]$$

- the effect of all operators
  - short, low-order, above-average schemata receive exponentially increasing in next generations

Problems of GAs

Slide 39

- building block hypothesis is presently unclear and depend on application problem
- floating point representations outperform binary encoding for real-value optimization
- premature convergence
  - exponential reproduction of the best chromosomes simply regenerate the current parents
  - further optimization by mutation can be quite slow

Evolution strategy

- adopted by Schwefel and Rechenberg
- basic ES steps

Slide 40

step 1 : an initial population of P parent solutions is selected at random with uniform distribution

step 2 : offspring solutions are created from each parent by adding a Gaussian random variable with zero mean and preselected standard deviation

step 3 : fitness evaluation for P parents and P offspring

step 4 : selection P solutions that possess the best fitness

step 5 : go to step 2 until a sufficient solution is reached

Evolution strategy (계속)

- comma strategy and plus strategy
  - comma strategy
    - μ parents generates λ offspring
    - μ parents are discarded leaving only λ individuals to compete
    - $\Longrightarrow (\mu, \lambda)$ search
  - plus strategy

Slide 41

    - μ parents generates λ offspring
    - all μ and λ individuals compete
    - $\Longrightarrow (\mu + \lambda)$ search
- original ES is $(1+1) - ES$
  - two main drawbacks
    - the constant standard deviation (step size) made slow convergence to optimal solutions
    - point-to-point search may fall into local minima
- Rechenberg defined convergence rate

Evolution strategy (계속)

Slide 42

  - for a quadratic function $F(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$
    where $\mathbf{x}$ is n-dimensional vector of reals, x_i is ith component
  - $\sigma \approx 1.224r/n$
    where r current Euclidean distance from the optimum at n dimensions
  - finally, $x_i^{'} = x_i + N(0, \sigma_i)$
    where $x_i^{'}$ is an offspring and x_i is a parent

- $1/5$ success rule
  - to attain good convergence rate
  - the ratio between successful and all mutations should be come to 1/5
  - if the ratio is above 1/5, then increase σ (broad search), otherwise decrease (narrow search)

Technical history of evolution strategy

Slide 43

- The earliest evolution strategies
  - based on one individual only
  - and only one operator, a mutation
  - an individual is represented by a pair of float-valued vectors ($\mathbf{v} = (\mathbf{x}, \sigma)$)
    where the first vector $\mathbf{x}$ represents a point in the search space and the second vector σ is a vector of standard deviation
  - mutation: $\mathbf{x^{t+1}} = \mathbf{x^t} + \mathbf{N}(\mathbf{0}, \sigma)$ where $\mathbf{N}(\mathbf{0}, \sigma)$ is a vector of independent random Gaussian numbers with a mean of zero and standard deviations σ
  - selection: select $(\mathbf{x^{t+1}}, \sigma)$ as a next parent iff $f(\mathbf{x}^{t+1}) > f(\mathbf{x}^t)$ where f is the objective function
  - $\sigma = (\sigma, \cdots, \sigma)$ remains unchanged during the evolution process
  - this is $(1 + 1) - ES$ strategy

Technical history of evolution strategy (계속)

Slide 44

- Rechenberg proposed a "$1/5$ success rule"
  - σ is changed as follows

$$\sigma^{t+1} = \begin{cases} c_d \cdot \sigma^t & \text{if } \varphi(k) < 1/5 \\ c_i \cdot \sigma^t & \text{if } \varphi(k) > 1/5 \\ \sigma^t & \text{if } \varphi(k) = 1/5 \end{cases}$$

where $\varphi(k)$ is the success ratio of the mutation operator during the last k generations and $c_i > 1, c_d < 1$ are two parameters

Technical history of evolution strategy (계속)

Slide 45

- multimembered evolution strategies
  - all individuals have the same mating probabilities
  - introduction of a recombination operator where two parents,

$$(\mathbf{x}^1, \sigma^1) = ((x_1^1, \cdots, x_n^1), (\sigma_1^1, \cdots, \sigma_n^1))$$

and

$$(\mathbf{x}^2, \sigma^2) = ((x_1^2, \cdots, x_n^2), (\sigma_1^2, \cdots, \sigma_n^2))$$

produce an offspring

$$(\mathbf{x}, \sigma) = ((x_1^{q_1}, \cdots, x_n^{q_n}), (\sigma_1^{q_1}, \cdots, \sigma_n^{q_n}))$$

where $q_i = 1$ or $q_i = 2$ with equal probability for all $i = 1, \cdots, n$

  - $(\mu + 1) - ES$
    - generate a single offspring
    - eliminate the weakest individual

Technical history of evolution strategy (계속)

- $(\mu + \lambda) - ES$s and $(\mu, \lambda) - ES$s
  - $(\mu + \lambda) - ES$
    - ☛ select μ parents from $\mu + \lambda$ individuals
  - $(\mu, \lambda) - ES$
    - ☛ select μ parents from λ individuals where $\lambda > \mu$
  - properties

Slide 46

    - ☛ σ is no longer constant
    - ☛ σ is not changed by deterministic algorithm (like the 1/5 success rule)
    - ☛ σ is incorporated in the individuals and undergoes the evolution process
  - recombination
    - ☛ select two individuals

$$(\mathbf{x}^1, \sigma^1) = ((x_1^1, \cdots, x_n^1), (\sigma_1^1, \cdots, \sigma_n^1))$$

and

$$(\mathbf{x}^2, \sigma^2) = ((x_1^2, \cdots, x_n^2), (\sigma_1^2, \cdots, \sigma_n^2))$$

Technical history of evolution strategy (계속)

- ☛ apply a recombination (crossover) operator
- ☛ two types of crossovers

1. discrete produce an offspring

$$(\mathbf{x}, \sigma) = ((x_1^{q_1}, \cdots, x_n^{q_n}), (\sigma_1^{q_1}, \cdots, \sigma_n^{q_n}))$$

where $q_i = 1$ or $q_i = 2$

2. intermediate produce an offspring

Slide 47

$$(\mathbf{x}, \sigma) = ((x_1^1 + x_1^2)/2, \cdots, (x_n^1 + x_n^2)/2), ((\sigma_1^1 + \sigma_1^2)/2, \cdots, (\sigma_n^1 + \sigma_n^2)/2))$$

  - mutation
    - ☛ $(\mathbf{x}, \sigma)$ generates $(\mathbf{x}', \sigma')$

where

$$\sigma' = \sigma \cdot e^{N(0, \Delta\sigma)}$$

and

$$\mathbf{x}' = \mathbf{x} + \mathbf{N}(\mathbf{0}, \sigma')$$

where $\Delta\sigma$ is a parameter of the method

Evolutionary programming

Slide 48

- originally conceived by Lawrence J. Fogel in 1960
- a stochastic optimization strategy similar to GAs
- emphasis the behavioral link between parents and offspring rather than the genetic link (GAs)
- EP is similar to evolution strategies, although the two approaches developed independently
- application to real-valued function optimization problems
  - operate on the real values themselves rather than any coding of the real values in GAs

Evolutionary programming (계속)

Slide 49

- basic EP steps

step 1 : choose an initial POPULATION of trial N solutions at random

step 2 : Each solution is replicated into a new population by adding a Gaussian random variable with zero mean and preselected standard deviation

step 3 : Each offspring solution is assessed by computing it's fitness.

step 4 : a stochastic tournament is held to determine N solutions

* in $2N$ solutions (parent + offspring)
* for each solutions, select C competitors
* comparison the solution and competitors
* if the fitness of the solution is greater or equal to the opponent, then assign a win
* select N solutions that have the greatest number of wins to be next population

step 5 : go to step 2 until a sufficient solution is reached

Evolutionary programming (계속)

Slide 50

- the main differences between ES and EP
  - Selection
    - EP
      1. EP typically uses stochastic selection via a tournament.
      2. Each trial solution competition against a preselected number of opponents
      3. receives a "win" if it is at least as good as its opponent in each encounter
      4. Selection then eliminates those solutions with the least wins.
    - ES
      1. ES typically uses deterministic selection
      2. the worst solutions are purged from the population based on their function evaluation.

Evolutionary programming (계속)

Slide 51

  - recombination
    - EP
      1. an abstraction of evolution at the level of reproductive populations
      2. thus no recombination mechanisms are typically used because recombination does not occur between species
    - ES
      1. an abstraction of evolution at the level of individual behavior.
      2. When self-adaptive information is incorporated this is purely genetic information (as opposed to phenotypic)
      3. thus some forms of recombination are reasonable and many forms of recombination have been implemented within ES
      4. the effectiveness of such operators depends on the problem

Application areas of EP and ES

Slide 52

- training and designing of neural networks
- system identification, control, and robotics
- pattern recognition problems
- evolutionary and fuzzy systems
- applying evolutionary optimization to machine learning
- designing evolutionary algorithms for implementation on parallel processing machines
- and others