Genetic Algorithms

because it sounds cool!

Probably the Most Popular Heuristic Algorithm

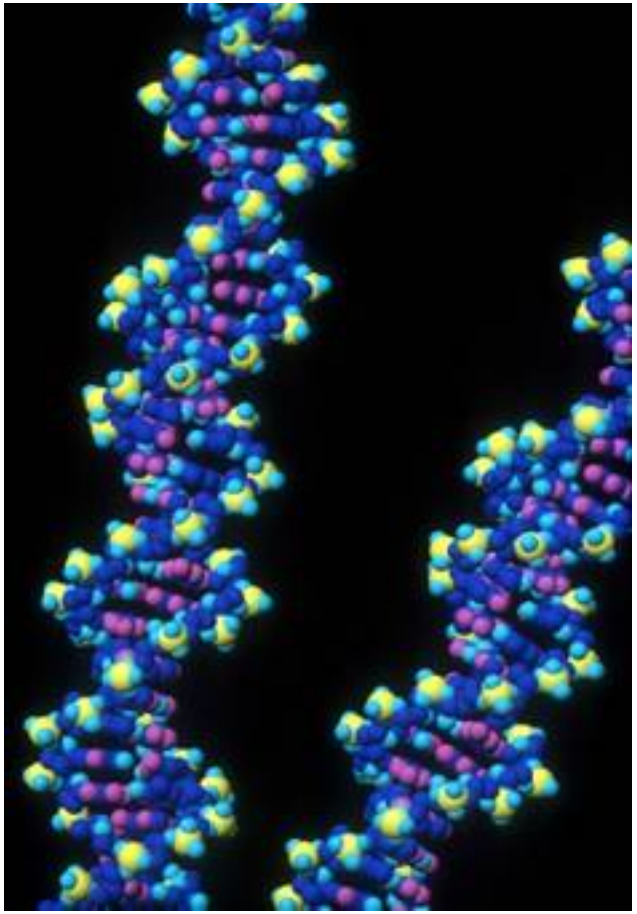
(but not necessarily the best for some applications)

Handout 9-9-11

Genetics and Genetic Algorithms

Nuclei

~~Nuclaic~~ acids carrying genetic information



We are all interested in our own genetics so perhaps that is one of the reasons Genetic Algorithms are so popular.

Genetic Algorithms

- The heuristic search method Genetic Algorithms is inspired by the mechanism Nature uses to "improve" populations.

  → much the way Simulated Annealing is inspired by the annealing process in materials science

- Among animals, each parent has genes.

- The genes of a child are directly related to the genes of the parents.

- The ability of an organism to survive (fitness) depends directly on the child's genes.

- In the algorithm analogy, "genes" are decision variables.

Effect of Genetic Alteration

Plant on the left is unaltered and the right has been genetically altered so the plants have different genes. actually the implementation of the genes are different.

Both plants are the same age. Algorithm analogy is that right side is a “good” solution and left side is a “bad” solution.

Population Genetics Suggests Heuristic

- "Survival of the fittest" means that children whose genes give high fitness are more likely to survive and produce more children.

- In this way over time, genes with characteristics beneficial for survival appear in an increasing fraction of the population and hence "fitness" of the population improves.

- It is an analogy of this principle that is the basis for the heuristic called "Genetic Algorithm".

Definition

- Gene- basic genetic element
- Chromosone- a collection of genes
- Allele- The values a gene can take (e.g. we usually have alleles = 0 or 1).
- Examples: Let S be a binary string of length 4 so
  - Chromosome example are 1010, 0110, etc.
  - Genes are the bits in the string
  - Alleles are 0 or 1
- For most heuristic optimization cases one chromosome = “genotype” = “phenotype”
  - (The actual biological distinction is more complex.)
- ”

Definitions (continued)

- For genetic algorithms, the objective function is called “Fitness” which we will call F(S) (not Cost(S)).

- We want to maximize the fitness F(S)

↳ if you have a minimization problem, flip the sign of the output of the cost function

- An iteration of the GA algorithm is called a “generation

Biggest Difference Between Genetic Algorithm and SA

- Simulated annealing carries the single "curS" from one iteration to the next.

↳ current S

- Genetic algorithms carry many solutions from one iteration to the next.

- These many solutions are called a "population"

- You can think of a population as a vector of current S's.
- This allows you to carry many solutions to improve robustness of the algorithm.

Introduction to Genetic Algorithms (cont.)

- ***The primary operations in Genetic Algorithms are***

- **Crossover** (this involves switching locations of substrings within a binary string) ⇒ swapping genes with other solutions

- **Mutation** (this involves changing the value of one value (e.g. one gene) in the binary string based on a probability distribution)

⇒ forced permutation to explore the search space

Characteristics of Genetic Algorithms (GA) :

- They require an effective representation of the decision variables in the form of **a chromosome** (encoded **string**). For "binary coded GA", this is a string of 0's and 1's.

- In each iteration (generation) there are **multiple members of population (CurSi, I=1,…,M)**

- The method is stochastic (i.e. they search using probabilistic functions)

- The method does not know **when to stop**.Usually set maxtime.

Characteristics of Genetic Algorithms (continued)

- (Like other heuristics) the algorithm only requires the **objective function value (S)**. (It doesn't require derivatives, continuity, etc.)

→ essentially the cost(s) we are used to, also called f(s)

- However, knowledge of problem structure can possibly be incorporated into the encoding of the string or into the reproductive process to improve performance.

→ string structure can effect outcome due to mutation and crossover

- Initially we will discuss decision variables that are binary strings (e.g. 0110010) for GA

Basic structure of genetic algorithms

- 1. Randomly generate an initial "population". ($CurS_j$) for all j= 1,…,M.
- 2. Compute $F(CurS_j)$ for all j= 1,…,M. (Objective Function)
- 3. Generate "children" which are the members of the population in the next generation. This is done by applying crossover and mutation, which are influenced by the values of the F(CurSj). Every child has two parents.
- 4.The population of children becomes a population of parents and go to step 2 unless a stopping criterion has been satisfied.

Crossover (Roulette)

- In order to generate N children you will perform the following:
  - Randomly select one parent with a probability that is proportional to its fitness $F(S_i)$.
  - Then randomly select a second parent also with a probability that is proportional to the fitness of the parent.
  - For one point crossover: pick a crossover point. This can be done randomly although there may be some restrictions discussed later.
  - Then switch the portions of the binary strings that occur before the crossover point between the parents.

Generating Children with *Crossover*

- The simple cross over procedure is
- Parent 1 0 1 1 0 0 | 0 0 1
- Parent 2 1 0 0 10 | 101

Offspring 1: 0 1 1 0 0 | 1 0 1

Offspring 2: 1 0 0 1 0 | 0 0 1

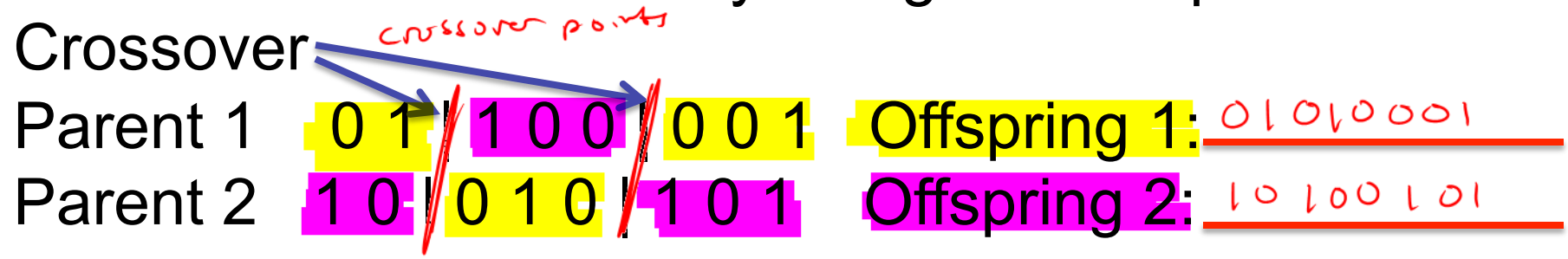
one-point

Crossover point

Note: in some cases you may choose to have only one offspring.

Multipoint Crossover

Another method for binary strings is Multipoint Crossover



Where multiple crossover points are randomly generated. (Two crossover points are in the example above.)

⇒ note you could generate more children here, but we are assuming we don't

⇒ really this is just another way of representing the statement "swap a specific subset of bits"

⇒ we'd want to put related things next to each other to capture correlation in this binary string

Mutation

- Mutation produces incremental random changes in the offspring by randomly changing allele values. Typically a mutation "flips" a value (e.g. 0 to 1) when S is a binary string.
  → chosen at random, usually uniform probability
- Mutation produces new characteristics.
- Without mutation, there can be some values you can never reach.

⇒ This helps you explore the search space

Maximize ~~F(S) = S3 – 60 S2 + 900 S + 100~~

(Same problem solved with Simulated Annealing.)

$F(s) = s^3 - 60s^2 + 900s + 100$

- S is a scalar integer represented in binary form
- The decision vector is a binary string with 5 elements.

- The total number of times the objective function is evaluated is 10 (same as for SA example)

- Table 1.12 (next slide) is initial population, which is 5 randomly generated strings) (note P(select) is proportional to fitness F(x))

Basic structure of genetic algorithms (Repeated Slide)

- 1. Randomly generate an initial "population". ($CurS_j$) for all j= 1,…,M. M=5 initially
- 2. Compute fitness $F(CurS_j)$ for all j= 1,…,M. (Objective Function)
- 3. Generate "children" which are the members of the population in the next generation. This is done by applying crossover and mutation, which are influenced by the values of the F(CurSj). Every child has two parents.
- 4.The population of children becomes a population of parents and go to step 2 unless a stopping criterion has been satisfied.

- In the next slide there is a probability of selection of a parent.

- We will discuss later how this probability is selected.

Initial Population of N=5

Table 1.12 Five random strings

fitness

Probability of Selection

Randomly Generated

No.	String	x	$f(x)$	$P[select]$
1	10011	19	2399	0.206
2	00101	5	3225	0.277
3	11010	26	516	0.044
4	10101	21	1801	0.155
5	01110	14	3684	0.317
Average fitness			2325	

Selection of Parents and Crossover and Mutation positions (Assumes selection based on probability weighted selection. Weight for parents depends on their fitness.

from table 1.12

- Parent 1: 10011
- Parent 2: 00101
- Crossover point 4
- Mutation postion 4

All 5 Children (Offspring) created after first iteration (generation) with Population of 5

Table 1.13 A typical experiment.

take only the first child

fitness

Step	Parent 1	Parent 2	Crossover point	Mutation?	Offspring String	Offspring $f(x)$
1	1	2	4	NNNYN	10001	2973
2	5	3	2	NNNNN	01010	4100
3	5	2	3	NNNNN	01101	3857
4	4	2	1	NYNNN	11101	129
5	2	5	4	NNNNN	00100	2804
	Average fitness					2773

converged very quickly

How did this do in comparison to Simulated Annealing? Would you expect GA to work this well in general with few evaluations of f(x)?

Maximize Cost (S) = $S^3 - 60\,S^2 + 900\,S + 100$.
S is a scalar integer represented in binary form

Initial temperature is 500, S_o=1011, Cost (S_o)=2399

Review: Simulated Annealing Approach to Same Problem

Table 1.5 Second Attempt: T=500

T	Changed bit	Snew string	J(Snew) $	delta Δf	accept move?	SO new string	J(SO)
500	1	0 0 0 1 1	2287	112	Y	0 0 0 1 1	2287
450	3	0 0 1 1 1	3803	< 0	Y	0 0 1 1 1	3803
405	5	0 0 1 1 0	3556	247	Y	0 0 1 1 0	3556
364.5	2	0 1 1 1 0	3684	< 0	Y	0 1 1 1 0	3684
328.0	4	0 1 1 0 0	3988	< 0	Y	0 1 1 0 0	3988
295.2	3	0 1 0 0 0	3972	16	Y	0 1 0 0 0	3972
265.7	4	0 1 0 1 0	4100	< 0	Y	0 1 0 1 0 **	optimal 4100
239.1	5	0 1 0 1 1	4071	29	Y	0 1 0 1 1	4071
215.2	1	1 1 0 1 1	343	3728	N	0 1 0 1 1	343

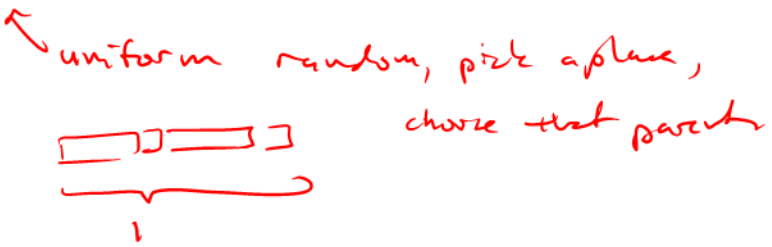
from Simulated Annealing handout with with 7 evaluations

Selection Methods Picking Parents for Crossover

- The method used in the previous table is **Roulette.** Compute the p_i, the probability of parent S_i being selected is

P[select]= probability of an individual *i* being selected as a parent. P[select for S_i]= P_i.

$$P_i = \frac{F(S_i)}{\sum_{j=1}^{N} F(S_j)} \Rightarrow \sum_{j=1}^{N} P_j = 1$$



All Children (Offspring) created after first iteration (generation) (Repeated Slide)

Table 1.13 A typical experiment.

Step	Parent 1	Parent 2	Crossover point	Mutation?	Offspring String	Offspring $f(x)$
1	1	2	4	NNNYN	10001	2973
2	5	3	2	NNNNN	01010	4100
3	5	2	3	NNNNN	01101	3857
4	4	2	1	NYNNN	11101	129
5	2	5	4	NNNNN	00100	2804
	Average fitness					2773

In this example, parents are picked at random by "Roulette" method.

Alternative Method for Crossover is **Tournament** Selection

- 1. Let all parents have an equal probability of being selected (e.g. $p_i = 1/M$, where M is number of individuals in the population).
- 2. Pick two parents S_i and S_k at random.
- 3. Select the parent with the higher fitness to be the first parent.
- 4. Repeat the process starting with step 1 to select the second parent (if 2 offspring/ parent)

Next you select a crossover point and mutation to generate the new child.

Closer to the way parents are selected for some species.