Data mining, machine learning, and uncertainty reasoning

林偉川

Genetic Algorithm (GA)

- A powerful alternative to the traditional heuristic search techniques
- GAs search by simulating evolution, starting from an initial set of solutions or hypotheses, and generating successive "generations" of solutions.
- Genetic Algorithms (GAs) are adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection and genetic.

2

Genetic Algorithm

- The basic concept of GAs is designed to simulate processes in natural system necessary for evolution, specifically those that follow the principles first laid down by Charles Darwin of survival of the fittest.
- As such they represent an intelligent exploitation of a random search within a defined search space to solve a problem.

3

Genetic Algorithm

- Each trait(特徵) is coded by some combination of DNA (there are four bases, A (Adenine鹽基), C (Cytosine胞嘧啶), T (Thymine胸腺嘧啶) and G (Guanine鹹).
- Like an alphabet in a language, meaningful combinations of the bases produce specific instructions to the cell

4

Genetic Algorithm

- Some basic terms are used in GA:
  - Specimens ➔ samples
  - Chromosomes(染色體), ➔ a long, complicated thread of DNA

5

Genetic Algorithm

- Mutation ➔some change in the original one
  - **mutation** also changes some traits. Sometimes an error may occur during copying of chromosomes
  - The parent cell may have -A-C-G-C-T- but an accident may occur and changes the new cell to -A-C-T-C-T-.
  - Much like a typist copying a book, sometimes a few mistakes are made. Usually this results in a nonsensical word and the cell does not survive. But over millions of years, sometimes the accidental mistake produces a more beautiful phrase for the book, thus producing a better species

6

Genetic Algorithm

- Sexual Reproduction
  - Changes occur during reproduction. The chromosomes from the parents exchange randomly by a process called crossover. Therefore, the offspring exhibit some traits of the father and some traits of the mother.
- Recombination (**crossover**)

7

3 fundamental principles of the GA

- Survival of the fittest
  - The strongest specimens have the highest chance to survive and reproduce, whereas the weak ones are likely to die before they reach the reproduction stage
- Sexual reproduction
  - The specimens find the partners they consider to be the best ones thus further contributing to the survival-of-the-fittest principle. They recombine their genetic information, thus creating new specimens with somewhat different characteristics

8

3 fundamental principles of the GA

- Mutation
  - Cause random and relatively rare, changes on the genetic information
- Search technique for a solution ➔ is a big problem
- How to facilitate the solution of a problem by means of a GA?

9

Genetic Algorithm

- Many of the real world problems involved finding optimal parameters, which might prove difficult for traditional methods but ideal for GAs.
- However, because of its outstanding performance in optimization, GAs have been wrongly regarded as a function optimiser.
- In fact, there are many ways to view genetic algorithms. Perhaps most users come to GAs looking for a problem solver, but this is a restrictive view.

10

Genetic Algorithm

We will examine GAs as a number of different things:

- GAs as problem solvers
- GAs as challenging technical puzzle
- GAs as basis for competent machine learning
- GAs as computational model of innovation and creativity
- GAs as computational model of other innovating systems
- GAs as guiding philosophy

11

Genetic Algorithm

- GAs were introduced as a computational analogy of adaptive systems.
- They are modeled loosely on the principles of the evolution via natural selection (the fittest survived and the unfit died out), employing a population of individuals that undergo selection in the presence of variation-inducing operators such as mutation and recombination
- Though it might not find the best solution. More often than not, it would come up with a partially optimal solution

12

2 questions of the GA

- 2 questions must be answered:
  - How to encode the search space in chromosomes?
  - How to define the fitness function that play the role of evaluation function in heuristic search?

13

2 questions of the GA

- For implementation, the chromosomes are represented by bit strings
- Each bit can stand for a binary attribute, the presence of a multi-valued attribute, or the presence of a predicate
- The fitness function, measuring the survival chance of the specimen, can be defined as the accuracy of the description derived from the chromosomes

14

Use a standard GA

- We want to represent a rule-based system. Given a rule such as "If color=red and size=small and shape=round then object=apple“
- The chromosomes are represented by bit strings
- Each bit can stand for a binary attribute, the presence of a multi-valued attribute, the presence of a predicate

15

Use a standard GA

- Describe it as a bit string by first assuming each of the attributes can take on a fixed set of possible values.
- Say color={red, green, blue}(3 bits), size={small, big} (2 bits), shape={square, round} (2 bits), and fruit={orange, apple, banana, pear} (4 bits).
- We could represent the value for each attribute as a substring of length equal to the number of possible values of that attribute.

16

Use a standard GA

- For example, color=red could be represented by 100, color=green by 010, and color=blue by 001.
- We can represent color=red or blue by 101, and any color (i.e., a "don't care") by 111.
- Rule "If color=red and size=small and shape=round then object=apple“ might then look like: 100 10 01 0100.

17

Use a standard GA

- A set of rules is then represented by concatenating together each rule's 11-bit string

18

Reproduction methods

2 basic methods of reproduction

- **Mutation**
  - Randomly change one or more digits in the string representing an individual.
  - For example, the individual 1-2-3 may be changed to 1-3-3 or 3-2-3, giving two new offspring.
  - adjustable parameters
    - How often to do mutation?
    - How many digits to change?
    - How big a change to make?

19

Reproduction methods

- **Crossover**
  - Randomly pick one or more **pairs of individuals** as parents and randomly swap segments of the parents.
  - The individuals 1-3-3 and 3-2-3 may be chosen as parents. Suppose we select a crossover point after the first digit, then the above will generate two offspring: 3-3-3 and 1-2-3.
  - As another example, given two parents 1011010 and 1100010, if the crossover point is between the third and fourth digits, then the two offspring are 1010010 and 1101010.

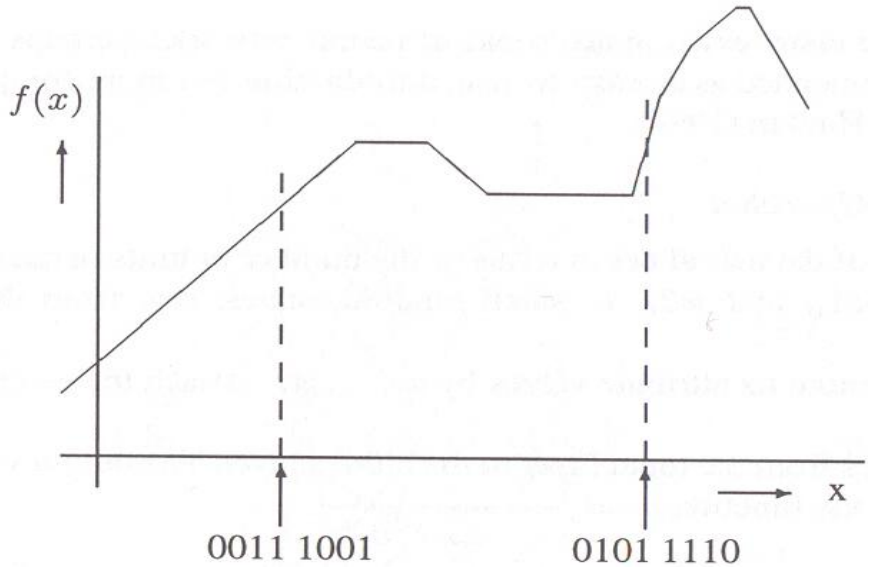
20

Reproduction methods

- This method is called *1-point crossover*. Similarly, we could define *2-point crossover*, which would select two points in each individual defining three intervals; the middle intervals are swapped to produce the two offspring.
- The rate of crossover, the number of parent pairs, the number of crossover points, and the positions of the crossover points are adjustable parameters.

21

Evaluation function of 2 binary specimens



Fitness Function

- The most fit solution in the final generation is the one that maximizes or minimizes the objective (fitness) function; this solution can be thought of as the GA has recommended choice.
- Therefore with single objective problems the user of GA is assisted in the choice phase of decision processing.

23

Fitness Function

- A simple selection method is each individual, *i*, has the probability
  Fitness(i) / sum_over_all_individuals_j Fitness(j),
  where Fitness(i) is the fitness function value for individual *i*.
- This method is sometimes called **fitness proportionate selection**.
- Other selection methods have also been used, e.g., **rank selection**, which sorts all the individuals by fitness and the probability that an individual will be selected is proportional to its rank in this sorted list.

24

Fitness Function

- One potential problem that can be associated with the selection method is called **crowding**.
- Crowding occurs when the individuals that are most fit quickly reproduce so that a large percentage of the entire population looks very similar.
- This reduces diversity in the population and may hinder the long-run progress of the algorithm.

25

Example of fitness function

- Fitness function is defined as f(x)=1/(x+1), where x is the number represented in the chromosome in binary form (e.g. '111'=7)
- The maximum value of f(x) will be reached for the string '000000' ➔ x=0, f(x)=1

26

Example of fitness function

- The first chance is given by a random number generator ensuring that the strongest specimens can be replicated more than once in the space of survivors while the weakest specimens die out ➔ reproduction
- In the next step, each survivor chooses a mating partner and exchanges with it part of their genetic information which is modeled as the exchange of random substrings ➔ recombination

27

Example of fitness function

- Choose the best and the second best specimens and exchange information of substring ➔ only tails of random length
- After this step, a new generation of stronger specimens comes into being
- The value of the fitness function indicate that its maximum as well as the average value increased

28

Example of fitness function

- The mutation operator is modeled quite straightforwardly: with a very small likelihood, a bit is flip-flopped to its opposite value
- The likelihood constant is usually adjusted so that in one generation no more than just a few (0-5) mutations appear

29

Example of fitness function

- The old generation contains 4 numbers: 37, 11, 20 and 7. The maximum value of the fitness function is reached for f(7)=1/8=0.125, so chromosome representing x=7 has the highest chance to survival
- The number x=37 has the smallest fitness function, f(37)=1/38=0.027 has a negligent chance to survive

30

Genetic search for the evaluation function

Old gen.	x	1/(x+1)	survivor	New gen.	x	1/(x+1)
(1)100101	37	0.026	(4)000111	000011	3	0.25 ✓
(2)001011	11	0.083	(2)001011	001111	15	0.063
(3)010100	20	0.048	(4)000111	000100	4	0.2
(4)000111	7	0.125✓	(3)010100	010111	23	0.042

31

Genetic search for the evaluation function

Old gen.	x	1/(x+1)	survivor	New gen.	x	1/(x+1)
(1)000011	3	0.25 ✓	(1)000011	000000	0	1 ✓
(2)001111	15	0.063	(3)000100	000111	7	0.125
(3)000100	4	0.2	(1)000011	000011	3	0.25
(4)010111	23	0.042	(2)001111	001111	15	0.063

32

Genetic Algorithm

1. Define the initial population as a set of binary strings generated randomly or by some pre-defined mechanism
2. Replicate the specimens in the population into the set of survivors by a mechanism that ensures that specimens with a high value of fitness function have higher chances of survival (and can be replicated more than once)

33

Genetic Algorithm

3. For each survivor, find a mate with which it exchanges part of the information encoded in the binary strings. With a very low frequency, a single bit is flip-flopped to model random mutations
4. If the fitness function has not increased throughout several cycles, stop. Otherwise go to step 2

34

Genetic Algorithm

- A fitness function is used to evaluate individuals, and reproductive success varies with fitness.
- The Algorithms
  1. Randomly generate an initial population M(0)
  2. Compute and save the fitness u(m) for each individual m in the current population M(t)
  3. Define selection probabilities p(m) for each individual m in M(t) so that p(m) is proportional to u(m)
  4. Generate M(t+1) by probabilistically selecting individuals from M(t) to produce offspring via genetic operators
  5. Repeat step 2 until satisfying solution is obtained

35

Genetic Algorithm

Proc GA(Fitness, theta, n, r, m)

// Fitness is the fitness function for ranking individuals

// theta is the fitness threshold, which is used to determine when to halt

// n is the population size in each generation (e.g., 100)

// r is the fraction of the population generated by crossover (e.g., 0.6) see slice 19 How often!!!

// m is the mutation rate (e.g., 0.001)

// initial generation is generated randomly

P := generate n individuals at random

36

Genetic Algorithm

while max Fitness(hi) < theta **do** // **slice 23,24**

i // define the next generation S (also of size n)

Reproduction step:

Probabilistically select (1-r)n individuals of P and add them to S intact, where the probability of selecting individual h_i is

Prob(h_i) = Fitness(h_i) / SUM Fitness(h_j)

37

Crossover step:

Genetic Algorithm

Mutate step:

Choose m% of S and randomly invert one bit in each P := S

end_while

Find *b* such that Fitness(b) = max Fitness(h_i)

return(b)

end_proc

38

HomeWork

- Find a problem solves by GA and write down your report which contains:
  - The problem description
  - The proposed GA encoding mechanism
  - The fitness function
  - The mating and reproduction method
  - The mutation rate
  - Your hand-written example by applying the above procedures
  - The conclusion

39