Data mining, machine learning, and uncertainty reasoning

林偉川

Who can benefit from GA

- Nearly everyone can gain benefits from Genetic Algorithms, once he can encode solutions of a given problem to chromosomes in GA, and compare the relative performance (fitness) of solutions.
- An effective GA representation and meaningful fitness evaluation are the keys of the success in GA applications.

2

Who can benefit from GA

- The appeal of GAs comes from their simplicity and elegance as robust search algorithms as well as from their power to discover good solutions rapidly for difficult high-dimensional problems.

3

Who can benefit from GA

- GAs are useful and efficient when
  - The search space is large, complex or poorly understood.
  - Domain knowledge is scarce or expert knowledge is difficult to encode to narrow the search space.
  - No mathematical analysis is available.
  - Traditional search methods fail.

4

Who can benefit from GA

- GAs have been used for problem-solving and for modelling. GAs are applied to many scientific, engineering problems, in business and entertainment, including:
  - Optimization: GAs have been used in a wide variety of optimisation tasks, including numerical optimisation, and combinatorial optimisation problems such as traveling salesman problem (TSP), circuit design [Louis 1993] , job shop scheduling [Goldstein 1991] and video & sound quality optimisation.

5

GA on optimization and planning

- TSP is NP-hard (NP stands for Non-deterministic Polynomial time) - it is generally believed cannot be solved in polynomial time.
- The TSP is constrained:
  - The salesman can only be in a city at any time.
  - Cities have to be visited once and only once.

6

GA on optimization and planning

- The TSP is interesting not only from a theoretical point of view, many practical applications can be modeled as a traveling salesman problem or as variants of it, for example, pen movement of a plotter, drilling of Printed Circuit Boards (PCB), real-world routing of school buses, airlines, delivery trucks and postal carriers.

7

GA on optimization and planning

- In the last two decades an enormous progress has been made with respect to solving TSP to optimality
- To solve for the most economical way for a traveling salesman to tour 5 cities the researcher can take a straightforward method, having the computer calculate the lengths of all 120 different routes to find the shortest one. (5!)

8

GA on optimization and planning

- This calculation could be performed in a fraction of a second. However, using the same method to figure the optimal route for a 100 cities could take billions of years. How can a more efficient solution be reached? (100!)
- Finding an optimal route becomes more challenging as the number of cities involved increases.
- Many other groups have created algorithms that can reach approximate (within 2%) solutions for this problem, calculating approximate optimal routes for 1000 cities in a few minutes.

9

Use a standard GA

- The following problems have to be solved:
  - A binary representation for tours is found such that it can be easily translated into a chromosome.
  - An appropriate fitness function is designed, taking the constraints into account.
  - The fitness function, measuring the survival chance of the specimen, can be defined as the accuracy of the description derived from the chromosome

10

GA on optimization and planning

- Improved versions of those techniques were used here to get upper bounds. The difficult part was to drive up the lower bound and prove optimality. Calculations for the 3038 cities problem (3038-city problem) required one and one-half years of computer time.
- This progress can be solved by increasing hardware power of computers.

11

Use a standard GA

- Non-permutation matrices represent unrealistic solutions, i.e. , the GA can generate some chromosomes that do not represent valid solutions. This happens:
  - in the random initialization step of the GA.
  - as a result of genetic operators (mutation and crossover).

12

Use a standard GA

- Permutation matrices are used. Two tours including the same cities in the same order but with different starting points or different directions are represented by different matrices and hence by different chromosomes, for example:
  - tour (23541) = tour (12354)
- Like a chromosome, the genetic structure of an individual is described using a fixed, finite alphabet. In GAs, the alphabet {0, 1} is usually used. This string is interpreted as a solution to the problem we are trying to solve

13

Use a standard GA

- An proper fitness function is obtained using penalty-function method to enforce the constraints.
- However, the ordinary genetic operators generate too many invalid solutions, leading to poor results. Alternative solutions to TSP require new representations (Position Dependent Representations) and new genetic operators.

14

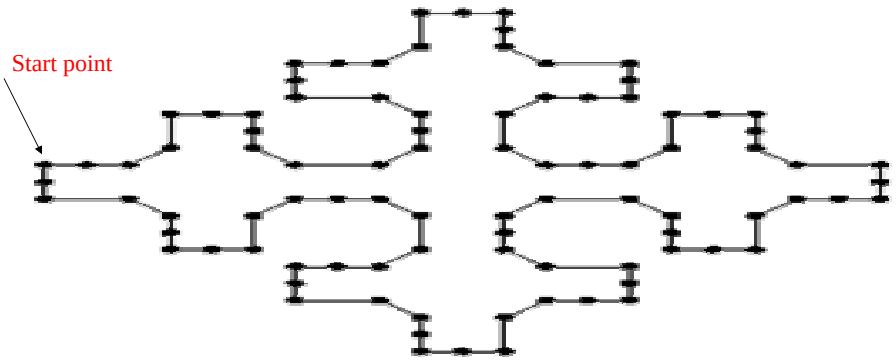
Use a standard GA

- We want to find the optimal quantity of the 3 major ingredients in a recipe (sugar, wine, and sesame oil).
- We can use the alphabet {1, 2, 3 ..., 9} denoting the number of ounces of each ingredient. Some possible solutions are 1-1-1, 2-1-4, and 3-3-1.
- The TSP is the problem of finding the optimal path to traverse, e.g. 10 cities. The salesperson may start in any city. A solution is a permutation of the 10 cities: 1-4-2-3-6-7-9-8-5-10.

15

TSP example

- The picture below shows an instance of the Euclidean, Planar TSP and the optimal curve among the set of cities which named MNPeano Order 2.



16

GA on optimization and planning

- Researchers have tracked TSPs to study bimolecular pathways, to route a computer networks‘ parallel processing, to advance cryptography (密碼學), to determine the order of thousands of exposures needed in X-ray crystallography (結晶學) and to determine routes searching for forest fires (which is a multiple-salesman problem partitioned into single TSPs).

17

GA on optimization and planning

- The goal of TSP is to devise a travel plan (a tour) which minimizes the total distance travel.
  ➔ Hamilton Cycle Problem
- When GAs applied to very large problems, they fail in two aspects:
  - They scale rather poorly (in terms of time complexity) as the number of cities increases.
  - The solution quality degrades rapidly.

18

Who can benefit from GA

- Automatic Programming: GAs have been used to evolve computer programs for specific tasks, and to design other computational structures, for example, cellular automata and sorting networks.

19

Who can benefit from GA

- Machine and robot learning: GAs have been used for many machine- learning applications, including classification and prediction, and protein structure prediction. GAs have also been used to design neural networks, to evolve rules for learning classifier systems or symbolic production systems, and to design and control robots.
- Economic models: GAs have been used to model processes of innovation (創新), the development of bidding strategies, and the emergence of economic markets.

20

Who can benefit from GA

- Ecological (生態) models: GAs have been used to model ecological phenomena such as biological arms races, host-parasite (宿主) co-evolutions, symbiosis (共生) and resource flow in ecologies.
- Population genetics models: GAs have been used to study questions in population genetics, such as "under what conditions will a gene for recombination be evolutionarily viable?"

21

GA Applications -- Finance

- GA are payoff (收益) driven. Payoffs can be improvements in predictive power or returns over a benchmark. There is an excellent match between the tool and the problems addressed.
- GA are inherently quantitative, and well-suited to parameter optimization (unlike most symbolic machine learning techniques).

22

GA Applications -- Finance

- GA are robust, allowing a wide variety of extensions and constraints that cannot be accommodated in traditional methods."
- Models for tactical (策略性) asset allocation and international equity strategies have been improved with the use of GAs.
- They report an 82% improvement in cumulative portfolio (投資組合) value over a passive benchmark model and a 48% improvement over a non-GA model designed to improve over the passive benchmark.

23

GA Applications -- Production/Operation

- Genetic Algorithm has been used to schedule jobs in a sequence dependent setup environment for a minimal total tardiness (延遲).
- All jobs are scheduled on a single machine; each job has a processing time and a due date.
- The setup time of each job is dependent upon the job which immediately precedes it. The GA is able to find good, but not necessarily optimal schedules, fairly quickly.

24

GA Applications -- Production/Operation

- GA is also used to schedule jobs in non-sequence dependent setup environment.
- The jobs are scheduled on one machine with the objective of minimizing the total weighted penalty for earliness or tardiness from the jobs' due dates. However, this does not guarantee that it will generate optimal solutions for all schedules.

25

GA Applications -- Decision Making

- Applying the well established decision processing phase model, GA appear to be very well suited for supporting the design and choice phases of decision making.
- In solving a single objective problem, GA designs many solutions until no further improvement (no increase in fitness) can be achieved or some predetermined number of generations have evolved or when the allotted (指定的) processing time is complete.

26

GA Applications -- Decision Making

- GA is developed for solving the machine-component grouping problem required for cellular manufacturing systems. GA provides a collection of satisfactory solutions for a two objective environment (minimizing cell load variation and volume of inter cell movement), allowing the decision maker to select the best alternative. ➔單元式製造 (指在一條生產線或一個機器設備單元內，由生產線或生產單元的操作工，生產多種產品或零件的生產製造過程)

27

GA Applications -- Decision Making

- When solving multi-objective problems, GA gives out many satisfactory solutions in terms of the objectives, and then allows the decision maker to select the best alternative.
- Therefore GAs assist with the design phase of decision processing with multi-objective problems.

28

GA Applications -- Decision Making

- GAs can be of great assistance for examining alternatives since they are designed to evaluate existing potential solutions as well to generate new (and better) solutions for evaluation. Thus GAs can improve the quality of decision making.

29

Conclusion of GA

- It has been shown that the GA perform better in finding areas of interest even in a complex, real-world scene.
- GA are adaptive to their environments, and as such this type of method is appealing to the vision community who must often work in a changing environment.

30

Conclusion of GA

- GAs are very helpful when the developer does not have precise domain expertise, because GAs possess the ability to explore and learn from their domain.
- Genetic Algorithms are easy to apply to a wide range of problems, from optimization problems like the TSP, to inductive concept learning, scheduling, and layout problems.

31

Conclusion of GA

- If only mutation is used, the algorithm is very slow.
- Crossover makes the algorithm significantly faster.
- Timing improvement could be done by utilizing the implicit parallelization of multiple independent generations evolving at the same time.
- The results can be very good on some problems, and rather poor on others.

32

Conclusion of GA

- Several improvements must be made in order that GAs could be more generally applicable.
- Grey coding the field would greatly improve the mutation operation while combing segmentation with recognition so that the interested object could be evaluated at once.
- GA is a kind of **hill-climbing** search; more specifically it is very similar to a **randomized beam search**.

33

Conclusion of GA

- As with all hill-climbing algorithms, there is a problem of local maxima.
- Local maxima in a genetic problem are those individuals that get stuck with a pretty good, but not optimal, fitness measure.

34

Conclusion of GA

- Mutation is a random process, so it is possible that we may have a sudden large mutation of individuals
- One significant difference between GAs and hill-climbing is that, it is generally a good idea in GAs to fill the local maxima up with individuals.
- GAs have less problems with local maxima than back-propagation neural networks.

35

Ant Colony System

- Ants are a classic example of social insects, which work together for the good of the colony(聚居地)
- A colony of ants finds new food sources by sending out foragers(掠奪者) who explore the surroundings more or less at random.
- If it finds food, a forager will return to the colony, laying a pheromone(費洛蒙) trail as it goes - a trail that other ants can follow back to the food.

36

Ant Colony System

- 演算法原理：螞蟻可以由蟻穴到食物目的地找到一條最短路線，它們用的不是視覺，而是在走過的地方會殘留一種分泌物，當以後的螞蟻經過時，就有較高的機率選擇pheromone濃度高的方向，因此隨著時間增長，漸漸螞蟻會走同一路線(亦即最短路線)由蟻穴到食物目的地來回，利用這種自然界的原理已有效率地解一些尋優問題(Optimization Problems)。

37

Ant Colony System(ACS)

- 研究方向：對於ACS 主題，我們可以找出一些適當的 OR/IE 問題，應用 ACS 原理來設計演算法，並與其他方法做比較評估，當然我們採用ACS 不一定要沿襲以往的方式，主要是採用群體合作的架構，但可考慮不同機制去導引”解”的搜尋方向，我們也希望多瞭解一些自然界的現象(如蜜蜂、海豚等的群體生活)，以啟發我們設計新的演算法的架構。

38

Ant Colony System

- Many researchers have focused their attention on a new class of algorithms called Meta-heuristics.
- Meta-heuristics are rather general algorithmic frameworks which can be applied to several different optimization problems. Meta-heuristics are often inspired by natural processes and one of the most recent nature-inspired meta-heuristic is ant colony optimization.(大量啟發式方法)

39

Ant Colony System

- An ACO algorithm is based on the result of low-level interaction among many cooperating simple agents that are not explicitly aware of their cooperative behavior. Each simple agent is called ant and the ACO algorithm (a distributed algorithm) is based on a set of ants working independently and cooperating in a common problem solving activity.

40

Ant Colony System

- The ant colony optimization is a population-based approach in which a set of artificial ants cooperate to build solutions using an indirect form of communication mediated by deposited pheromone.

41

Ant Colony System

- Ant Colony Optimization (ACO) is a relatively new meta-heuristic proposed by Dr. M. Dorigo for hard combinatorial optimization problems.
- This meta-heuristic belongs to the class of heuristics derived from nature, which includes Evolutionary Algorithms, Neural Networks, Simulated Annealing.
- http://iridia.ulb.ac.be/~mdorigo/ACO/ACO.html
- http://chern.ie.nthu.edu.tw/Ant_Algorithms.htm
- http://uk.geocities.com/markcsinclair/aco.html

(good animation!!!)

42

Ant Colony System

- To solve the NP- Hard problems such as TSP, vehicle routing, Job-shop Scheduling, Graph Coloring, the Multiple Knapsack Problem (MKP), the Set Covering Problem (SCP), and the Maximum Independent Set Problem (MISP), Packet-switched Communication Network, Sequential Ordering, Shortest Common Super-Sequence, Load Balancing in Communication Networks … etc.

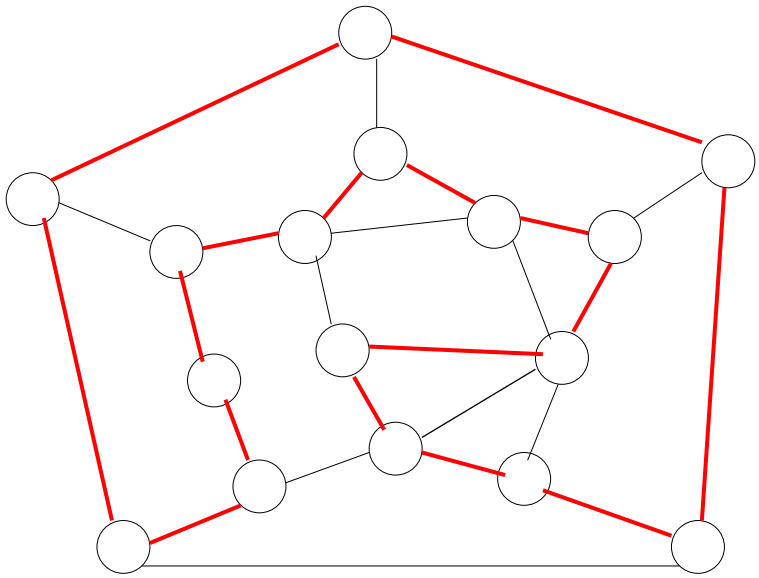
43

Ant Colony System

- Given a graph G(V,E), the problem of deciding whether G is Hamiltonian, i.e., whether or not there is a simple cycle in E spanning all vertices in V. This problem is known to be NP-complete, hence cannot be solved in polynomial time in |V| unless P=NP.

44

Hamiltonian



45

Ant Colony System

- Solve the Hamiltonian cycle problem by a new ant-inspired approach, based on repeated covering of the graph.
- This is based on a process in which an ant traverses the graph by moving from vertex to vertex along the edges, occasionally leaving traces in the vertices, and deciding on the next step according to the level of traces in the surrounding neighborhood.

46

Ant Colony System

- It shows that Hamiltonian cycles are limited cycles of the process, and investigate the average time needed by the ant process to recognize a Hamiltonian graph, on the basis of simulations made over large samples of random graphs with varying structure and density.
- See the sample chapter of ACO algorithm from Dr. M. Dorigo!!

47

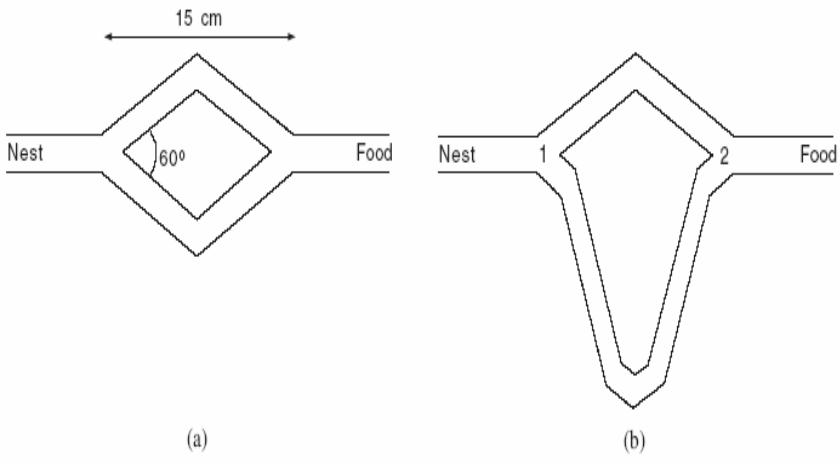
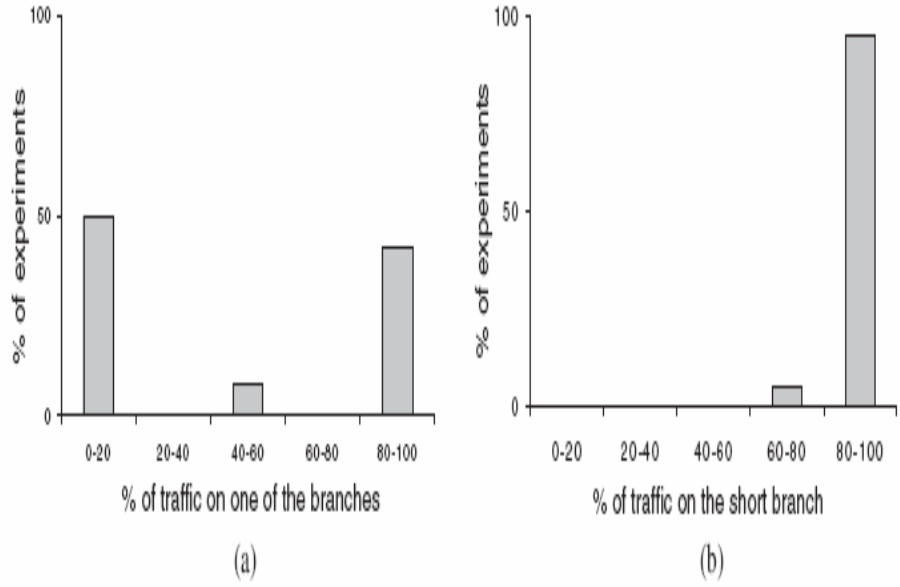


Figure 1.1
Experimental setup for the double bridge experiment. (a) Branches have equal length. (b) Branches have different length. Modified from Goss et al. (1989).

Ant Colony System

- The figure as shown on the next slice, (a) Results for the case in which the two branches have the same length (r =1); in this case the ants use one branch or the other in approximately the same number of trials.
- (b) Results for the case in which one branch is twice as long as the other (r = 2); here in all the trials the great majority of ants chose the short branch.

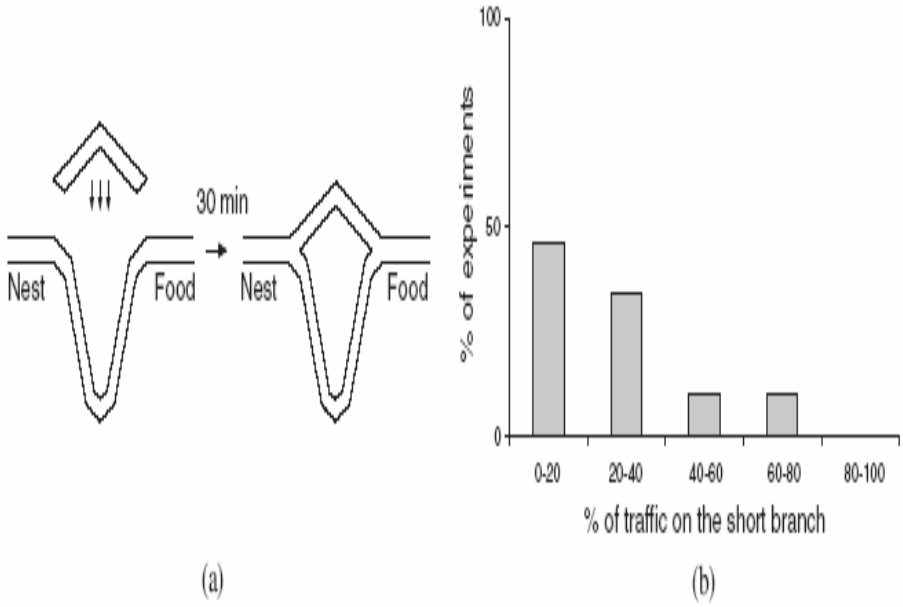
49



Ant Colony System

- The figure (a) as listed on next shows that initially only the long branch was offered to the colony.
- After 30 minutes, when a stable pheromone trail has formed on the only available branch, a new shorter branch is added. (a) The initial experimental setup and the new situation after 30 minutes, when the short branch was added.
- (b) In the great majority of the experiments, once the short branch is added the ants continue to use the long branch is gradually disappeared.

51



Ant Colony System

- Given the lengths l_s and l_l (in cm) of the short and of the long branch, an ant per second cross the bridge in each direction at a constant speed of v cm/s, depositing one unit of pheromone on the branch, it chooses the short branch will traverse it in $t_s=l_s/v$ seconds, while an ant choosing the long branch will use $r*t_s$ seconds, where $r= l_l / l_s$.

53

Ant Colony System

- The probability $p_{ia}(T)$ that an ant arriving at decision point $i \in \{1, 2\}$ (see Figure 1.1b) selects branch $a \in \{s, l\}$, where s and l denote the short and long branch respectively, at instant t is set to be a function of the total amount of pheromone $\varphi_{is}(T)$ on the branch, which is proportional to the number of ants that used the branch until time t. For example, the probability $p_{is}(T)$ of choosing the short branch is given by

$$p_{is}(t) = \frac{(t_s + \varphi_{is}(t))^{\alpha}}{(t_s + \varphi_{is}(t))^{\alpha} + (t_s + \varphi_{il}(t))^{\alpha}},$$

54

Ant Colony System

where the functional form of equation (1.1), as well as the value $\alpha=2$, was derived from experiments on trail-following (Deneubourg et al., 1990); $p_{il}(T)$ is computed similarly, with $p_{is}(T) + p_{il}(T) = 1$.

55