Genetic Algorithms

Philippe Pasquier
Office 565 (floor 14)
pasquier@sfu.ca

SFU

Philippe Pasquier, January 2008

Introduction

- **Genetic algorithms (GAs) provide an approach to machine learning that is loosely based on simulated evolution:**
  - **Motivated by an analogy to biological evolution**
  - **Starting with a population of random hypothesis (or individuals), GAs generate successor hypothesis by repeatedly recombining and mutating parts of the best currently known hypothesis**
- **The popularity of GAs is motivated by:**
  - **Evolution is known to be a successful and robust method for adaptation within biological systems**
  - **GAs can search for hypothesis in which the impact of parts on the overall hypothesis fitness may be difficult to model.**
  - **The basic idea is very simple (and yet quite convincing and powerful)**

Genetic Algorithms

- The problem:
  - We want to create an entity (target function) $h \in H$
  - This entity must fulfill precise specifications given by a fitness function: $finess : H \rightarrow \mathbb{R}$
  - The fitness function assigns a score to any given hypothesis
- The method:
  - Generate a random population of hypothesis (100 to several thousand)
  - Evaluate each of them according to the fitness function
  - A new population is formed by probabilistically selecting the most fit hypothesis from the current population
  - Some of these hypotheses are carried to the next generation population intact while others are used for creating new offspring individuals by applying some genetic operators such as crossover and mutation

Genetic Algorithms

- The ingredients:
  - Representation of instances (hypotheses or individuals)
  - A well-defined fitness function
  - Genetic operators:
    - Cross-over
    - Mutation
    - Any other operation that preserves the integrity of the offsprings
  - Note that here, biological evolution is used as a guide rather than a constraint and in practice many other operators have been used

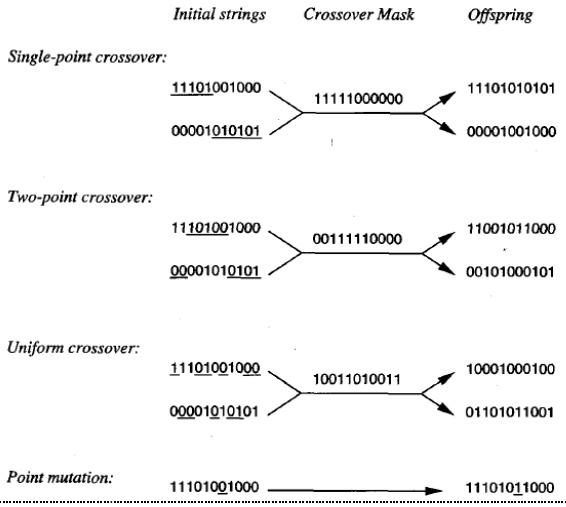
Representing hypothesis

- Hypotheses in GAs are often represented using bit strings (simplify the definition of operators)
- For example, a set of if-then rules can easily be represented that way:
  - Outlook={Sunny, Overcast, Rainy}
  - Three bits are used, one per value:
    - 010 stand for Overcast
    - 110 stand for Sunny or Overcast
    - 111 stand for either
  - Wind={Weak, Medium, Strong}
  - How to represent “Rainy Outlook and Strong Wind”?
    - 001001 (we concatenate the corresponding strings)

Representing hypothesis

- Rules' pre and pos-conditions are represented in a similar fashion and are concatenated:
  - Action={WorkonProject, Windsurf, Tan}
  - 011111100, What does this rule means?
- Each attribute (a dimension of the hypothesis space is represented) which yield to a fixed length bit-string representation of rules
- An hypothesis (or individual) is the concatenation of a variable number of rules.
- Various other representations are possible. In any case, it is important that any string corresponds to a valid hypothesis.

Genetic operators



The fitness function

- The fitness function is what specify the solution (target function) searched for

- GAs move the design problem from:
  - Knowing how to design the solution to a problem to
  - Knowing how to evaluate potential solutions

- If for example, the problem is a classification problem, and the hypothesis space is made of sets of rules, then the target function has one component that scores the classification according to a set of "training examples"
- For example: the percentage of examples properly classified

Simple Genetic Algorithm

1) p: number of hypothesis to be included in the population
   r: fraction of the population to be replaced by crossover
   m: mutation rate
2) Initialise population: $P_0 \leftarrow p$ hypothesis generated at random
3) Evaluate: While $\max_h Fitness(h) < Fitness_{threshold}$ do
   4) Select: select $(1-r)\,p$ members of P_t to add to P_{t+1}
      each element is selected with a probability:
   5) Crossover: Probabilistically select $\frac{r*p}{2}$ pairs of hypothesis, which offsrpings are added to P_{t+1}
   6) Mutate: mutate m percents of P_{t+1} with uniform probability
   7) Update: $P_t \leftarrow P_{t+1}$
8) Result: return the hypothesis with the highest fitness

$$Pr(h_i) = \frac{Fitness(h_i)}{\sum_{j=1}^{p} Fitness(h_j)}$$

Genetic algorithms

- **It uses fitness proportionate selection (also called roulette wheel selection)**
- **Initiated in the 1970's by, J. Holland, K. DeJong, D. Goldberg. Holland's original GA is now known as the simple genetic algorithm (SGA)**
- **The ending criteria can be:**
  - **A fitness threshold to be reached**
  - **A fixed number of generation**
  - **No more improvement (in terms of fitness)**
  - **Time, patience, memory or funding!**

Variations on genetic operators

- Numerous variations of GAs are possible:
  - Crossovers: For example, in GABIL by Dejonc and Al (1993), the following crossover variation was used:
    - Two crossover points are chosen in one parent at random
    - The distances to the leftmost (rightmost) closest rule boundary are calculated
    - Two crossover points that respect these distances are chosen in the second parent
    - The two resulting offspring are created
    - Parent A: 001100010 000|11100|0 111000111
    - Parent B: 010|101010 101010101 10011001|1
    - Off1: 001100010 000101010 101010101 10011001
    - Off1: 010111001
  - This variation allows varying the number of rules in each hypothesis

Variations on genetic operators

- Amongst other variations:
  - New operators:
    - Add_alternative: add a 1 randomly in conditions of selected rules
    - Drop_condition: turn all the bits of one feature/attribute to 1 (thus eliminating it since any value for that feature will select the rule). This operator corresponds to a generalisation.
    - Some bit can be used to encode which operator can be used on a given rule:
      - 010010111 01 111010010 10
  - That illustrates how GAs can in principle be used to evolve their own hypothesis search methods

Back to the fitness function

- The fitness function can be:
  - Explicit: a well-defined function or procedure that score any instance according to
    - Its performance
    - Its distance to the optimal
    - Its distance to the specified solution
  - Implicit: the system is embedded in an environment that is doing the selection
    - Simulation of ecosystem
    - Synthetic environment (e.g. Eden)
  - Interactive (human computation):
    - Humans are manually selecting parents, 1, 2 or more (e.g. Karl Sims)
    - Humans are interacting with the system so as to determine/constrain the selection (e.g. GenJam)

Variations on the fitness function

- Variations on the fitness function include:
  - Tournament selection: two hypotheses are chosen at random and the fittest is kept with probability p (gives a more diverse population then proportional selection)
  - Rank selection: first sort the population by fitness, then select individual with probability depending on their rank (as opposed to their fitness value)
  - Reproduction specialization: allowing only similar individual to reproduce will encourage the creation of clusters, or sub-species in the population

Variations on the fitness function

- More variations on the fitness function:
  - Spatially distribute individuals and allow them to only reproduce with nearby ones
  - Fitness sharing: the fitness is reduced by the presence of similar individual
- Crowding: occurs when one highly fit individual reproduces so much that copies or very similar individuals take over a large fraction of the population
  - It reduces diversity and thus slows down the search
  - Solutions: some of the many variations of the fitness function can be applied to try to remedy crowding.

Properties of GA

- A GA is doing a search through the space of hypothesis to find the hypothesis that optimise (maximise) the fitness function
- In that sense, it relates to the other types of learning we have been looking at: Hill climbing, gradient descent, ...
- However, we have seen that these techniques were having trouble with local optima (because they explore only one region at a time and only do little jump
- GAs are doing various sorts of jump!
- GAs are doing the search in “parralel”:
  - The multiple strings in a population are samples the search space in multiple regions
  - Notably, the rate at which a given region is sampled corresponds directly to its elevation (related to the probability of finding a good solution in that vicinity).

"Genetic engineers don't make new genes, they rearrange existing ones."

Thomas E. Lovejoy