Reading

- CS 420/594: Read Flake, ch. 22 (Neural Networks and Learning)
- CS 594: Read Bar-Yam, sec. 2.3 (Feedforward Networks)

Genetic Algorithms

- Developed by John Holland in ‘60s
- Did not become popular until late ‘80s
- A simplified model of genetics and evolution by natural selection
- Most widely applied to optimization problems (maximize “fitness”)

Assumptions

- Existence of fitness function to quantify merit of potential solutions
  - this “fitness” is what the GA will maximize
- A mapping from bit-strings to potential solutions
  - best if each possible string generates a legal potential solution
  - choice of mapping is important
  - can use strings over other finite alphabets

Outline of Simplified GA

1. Random initial population $P(0)$
2. Repeat for $t = 0, \ldots, t_{max}$ or until converges:
   a) create empty population $P(t + 1)$
   b) repeat until $P(t + 1)$ is full:
      1) select two individuals from $P(t)$ based on fitness
      2) optionally mate & replace with offspring
      3) optionally mutate offspring
      4) add two individuals to $P(t + 1)$

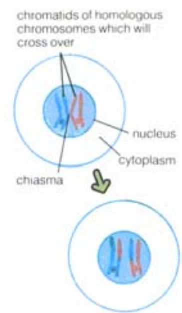
Fitness-Biased Selection

- Want the more “fit” to be more likely to reproduce
  - always selecting the best ⇒ premature convergence
  - probabilistic selection ⇒ better exploration
- Roulette-wheel selection: probability ∝ relative fitness:

$$\Pr\{i \text{ mates}\} = \frac{f_i}{\sum_{j=1}^{n} f_j}$$

11/21/04 5

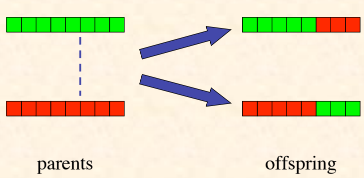
Crossover: Biological Inspiration



- Occurs during meiosis, when haploid gametes are formed
- Randomly mixes genes from two parents
- Creates genetic variation in gametes

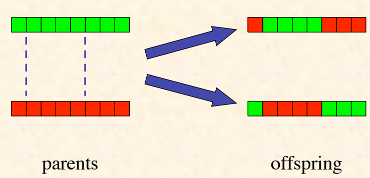
11/21/04 (fig. from *B&N Thes. Biol.*) 6

GAs: One-point Crossover



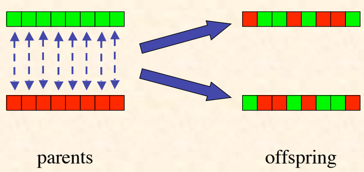
11/21/04 7

GAs: Two-point Crossover



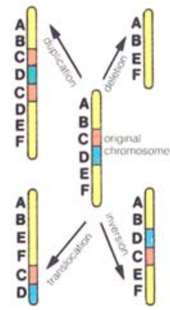
11/21/04 8

GAs: *N*-point Crossover



Mutation: Biological Inspiration

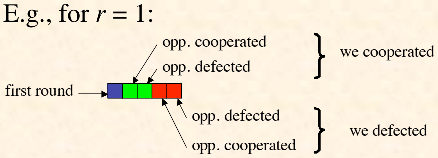
- Chromosome mutation ⇒
- Gene mutation: alteration of the DNA in a gene
  - inspiration for mutation in GAs
- In typical GA each bit has a low probability of changing
- Some GAs models rearrange bits



(fig. from *B&N Thes. Biol.*)

Example: GA for IPD

- Genetic strings encode strategy
  - for first round
  - based on self's & opponent's action on r previous rounds
  - hence $2^{2r} + 1$ bits
- E.g., for $r = 1$:



Typical Result

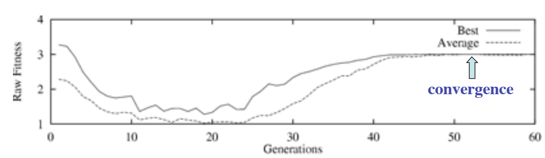


Figure 20.4 Average and best raw fitness scores for the IPD-playing GA

The Red Queen Hypothesis

"Now, *here*, you see, it takes all the running *you* can do, to keep in the same place."
— *Through the Looking-Glass and What Alice Found There*

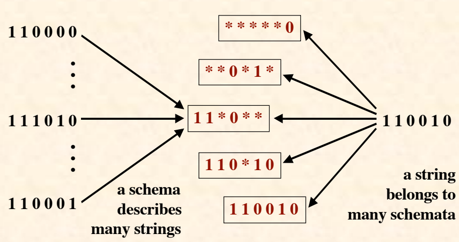
- *Observation*: a species probability of extinction is independent of time it has existed
- *Hypothesis*: species continually adapt to each other
- Extinction occurs with insufficient variability for further adaptation

Why Does the GA Work?

The Schema Theorem

Schemata

A schema is a description of certain patterns of bits in a genetic string



The Fitness of Schemata

- The schemata are the building blocks of solutions
- We would like to know the average fitness of all possible strings belonging to a schema
- We cannot, but the strings in a population that belong to a schema give an estimate of the fitness of that schema
- Each string in a population is giving information about all the schemata to which it belongs (implicit parallelism)

Effect of Selection

Let n = size of population

Let $m(S,t)$ = number of instances of schema S at time t

String i gets picked with probability $\frac{f_i}{\sum_j f_j}$

Let $f(S)$ = avg fitness of instances of S at time t

So expected $m(S,t+1) = m(S,t) \cdot n \cdot \frac{f(S)}{\sum_j f_j}$

Since $f_{av} = \frac{\sum_j f_j}{n}$, $m(S,t+1) = m(S,t)\frac{f(S)}{f_{av}}$

11/21/04 17

Exponential Growth

- We have discovered: $m(S, t+1) = m(S, t) \cdot f(S) / f_{av}$
- Suppose $f(S) = f_{av} (1 + c)$
- Then $m(S, t) = m(S, 0) (1 + c)^t$
- That is, exponential growth in above-average schemata

11/21/04 18

Effect of Crossover

1 … 0*
|←δ→|

- Let λ = length of genetic strings
- Let $\delta(S)$ = defining length of schema S
- Probability {crossover destroys S}: $p_d \le \delta(S) / (\lambda - 1)$
- Let p_c = probability of crossover
- Probability schema survives:

$$p_s \ge 1 - p_c \frac{\delta(S)}{\lambda - 1}$$

11/21/04 19

Selection & Crossover Together

$$m(S,t+1) \ge m(S,t)\frac{f(S)}{f_{av}}\left[1 - p_c \frac{\delta(S)}{\lambda - 1}\right]$$

11/21/04 20

Effect of Mutation

- Let p_m = probability of mutation
- So $1 - p_m$ = probability an allele survives
- Let $o(S)$ = number of fixed positions in S
- The probability they all survive is $(1 - p_m)^{o(S)}$
- If $p_m << 1$, $(1 - p_m)^{o(S)} \approx 1 - o(S)\, p_m$

Schema Theorem: "Fundamental Theorem of GAs"

$$m(S,t+1) \geq m(S,t)\frac{f(S)}{f_{av}}\left[1 - p_c\frac{\delta(S)}{\lambda - 1} - o(S)p_m\right]$$

The Bandit Problem

- Two-armed bandit:
  - random payoffs with (unknown) means m_1, m_2 and variances σ_1, σ_2
  - optimal strategy: allocate exponentially greater number of trials to apparently better lever
- k-armed bandit: similar analysis applies
- Analogous to allocation of population to schemata
- Suggests GA may allocate trials optimally

Goldberg's Analysis of Competent & Efficient GAs

Paradox of GAs

- Individually uninteresting operators:
  - selection, recombination, mutation
- Selection + mutation ⇒ continual improvement
- Selection + recombination ⇒ innovation
  - generation vs.evaluation
- Fundamental intuition of GAs: the three work well together

Race Between Selection & Innovation: Takeover Time

- Takeover time t^* = average time for most fit to take over population
- Transaction selection: top $1/s$ replaced by s copies
  - s quantifies selective pressure
- Estimate $t^* \approx \ln n / \ln s$

Innovation Time

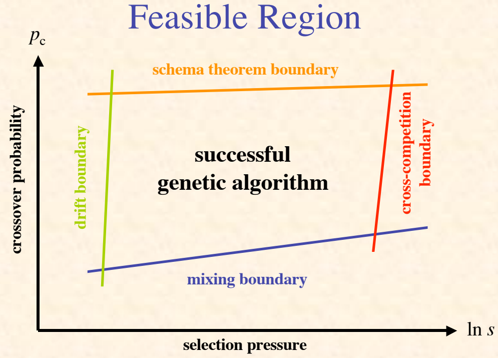
- Innovation time t_i = average time to get a better individual through crossover & mutation
- Let p_i = probability a single crossover produces a better individual
- Number of individuals undergoing crossover = $p_c\, n$
- Probability of improvement = $p_i\, p_c\, n$
- Estimate: $t_i \approx 1 / (p_c\, p_i\, n)$

Steady State Innovation

- Bad: $t^* < t_i$
  - because once you have takeover, crossover does no good
- Good: $t_i < t^*$
  - because each time a better individual is produced, the t^* clock resets
  - *steady state innovation*
- Innovation number:

$$\text{Iv} = \frac{t^*}{t_i} = p_c p_i \frac{n \ln n}{\ln s} > 1$$

Feasible Region



11/21/04 29

Other Algorithms Inspired by Genetics and Evolution

- Evolutionary Programming
  - natural representation, no crossover,time-varying continuous mutation
- Evolutionary Strategies
  - similar, but with a kind of recombination
- Genetic Programming
  - like GA, but program trees instead of strings
- Classifier Systems
  - GA + rules + bids/payments
- and many variants & combinations…

11/21/04 30

Additional Bibliography

1. Goldberg, D.E. *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. Kluwer, 2002.
2. Milner, R. *The Encyclopedia of Evolution*. Facts on File, 1990.

11/21/04 31