IV. Evolutionary Computing

A. Genetic Algorithms
Read Flake, ch. 20
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 $t = 0, \ldots, t_{\text{max}}$ 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 $\propto$ relative fitness:
  $$\Pr\{i \text{ mates}\} = \frac{f_i}{\sum_{j=1}^{n} f_j}$$
Crossover: Biological Inspiration

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

(fig. from B&N Thes. Biol.)
GAs: One-point Crossover

parents

offspring
GAs: Two-point Crossover

parents

offspring
GAs: $N$-point Crossover

parents

offspring
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.)
The Red Queen Hypothesis

• **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

“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*
Demonstration of NetLogo Simple Genetic Algorithm

Run NetLogo Simple Genetic Algorithm
Demonstration of GA: Finding Maximum of Fitness Landscape

Run Genetic Algorithms — An Intuitive Introduction by Pascal Glauser
<www.glauserweb.ch/gentore.htm>
Demonstration of GA: Evolving to Generate a Pre-specified Shape (Phenotype)

Run Genetic Algorithm Viewer
<www.rennard.org/alife/english/gavgb.html>
Eaters Seeking Food

• Eaters are FSMs
• Have internal state (memory): 0..15
• Can sense one square ahead
• It can see one of four different things: an Eater, a plant, a blank space, or a wall
• On basis of the above, can change state and:
  – 1. Move forward one square
  – 2. Move backwards one square
  – 3. Turn in place 90 degrees to the left
  – 4. Turn in place 90 degrees to the right
• If lands on a square with food, it eats it
• Genetic strings: $16 \times 4 \times (2 + 4) = 384$ bits
Demonstration of GA: Eaters Seeking Food

http://math.hws.edu/eck/js/genetic-algorithm/GA.html
Morphology Project
by Michael “Flux” Chang

• Senior Independent Study project at UCLA
  – users.design.ucla.edu/~mflux/morphology
• Researched and programmed in 10 weeks
• Programmed in Processing language
  – www.processing.org
Genotype ⇒ Phenotype

• Cells are “grown,” not specified individually
• Each gene specifies information such as:
  – angle
  – distance
  – type of cell
  – how many times to replicate
  – following gene
• Cells connected by “springs”
• Run phenome:
  <users.design.ucla.edu/~mflux/morphology/gallery/sketches/phenome>
Complete Creature

- Neural nets for control (blue)
  - integrate-and-fire neurons
- Muscles (red)
  - Decrease “spring length” when fire
- Sensors (green)
  - fire when exposed to “light”
- Structural elements (grey)
  - anchor other cells together
- Creature embedded in a fluid
- Run <users.design.ucla.edu/~mflux/morphology/gallery/sketches/creature>
Effects of Mutation

• Neural nets for control (blue)
• Muscles (red)
• Sensors (green)
• Structural elements (grey)
• Creature embedded in a fluid
• Run

<users.design.ucla.edu/~mflux/morphology/gallery.sketches/creaturepack>
Evolution

• Population: 150–200
• Nonviable & nonresponsive creatures eliminated
• Fitness based on speed or light-following
• 30% of new pop. are mutated copies of best
• 70% are random
• No crossover
Gallery of Evolved Creatures

- Selected for speed of movement
- Run
  
  <users.design.ucla.edu/~mflux/morphology/gallery.sketches/creaturegallery>
Example: Circle Swimmer
Example: Slug
Karl Sims’ Evolved Creatures
Why Does the GA Work?

The Schema Theorem
Schemata

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

A string belongs to many schemata.

A schema describes many strings.
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 $n = \text{size of population}$

Let $m(S, t) = \text{number of instances of schema } S \text{ at time } t$

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

Let $f(S) = \text{avg fitness of instances of } S \text{ at time } t$

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

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

• We have discovered:
  \[ m(S, t+1) = m(S, t) \cdot \frac{f(S)}{f_{av}} \]
• 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
Effect of Crossover

- Let $\lambda = \text{length of genetic strings}$
- Let $\delta(S) = \text{defining length of schema } S$
- Probability \{crossover destroys } S\}:
  \[ p_d \leq \frac{\delta(S)}{\lambda - 1} \]
- Let $p_c = \text{probability of crossover}$
- Probability schema survives:
  \[ p_s \geq 1 - p_c \frac{\delta(S)}{\lambda - 1} \]
Selection & Crossover Together

\[ m(S,t + 1) \geq m(S,t) \frac{f(S)}{f_{av}} \left[ 1 - p_c \frac{\delta(S)}{\lambda - 1} \right] \]
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 \( p_m \ll 1, (1 - p_m)^{o(S)} \approx 1 - o(S) p_m \)
Schema Theorem: “Fundamental Theorem of GAs”

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

Note: \( (1 - p_c \frac{\delta(S)}{\lambda - 1}) (1 - o(S)p_m) \approx 1 - p_c \frac{\delta(S)}{\lambda - 1} - o(S)p_m \)
The Bandit Problem

• Two-armed bandit:
  – random payoffs with (unknown) means $m_1$, $m_2$
    and variances $\sigma_1^2$, $\sigma_2^2$
  – 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
  – fundamental to invention: 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: population replaced by $s$ copies of top $1/s$
• $s$ quantifies selective pressure
• Estimate $t^* \approx \ln n / \ln s$
Innovation Time

- Innovation time $t_i = \text{average time to get a better individual through crossover & mutation}$
- Let $p_i = \text{probability a single crossover produces a better individual}$
- Number of individuals undergoing crossover = $p_c n$
- Number of probable improvements = $p_i p_c n$
- 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:
  $$Iv = \frac{t^*}{t_i} = p_c p_i \frac{n \ln n}{\ln s} > 1$$
Feasible Region

- $p_c$
- selection pressure
- ln $s$

Boundaries:
- Schema theorem boundary
- Mixing boundary
- Drift boundary
- Cross-competition boundary

Successful genetic algorithm
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…
Additional Bibliography
