Genetic Algorithms

Population Management

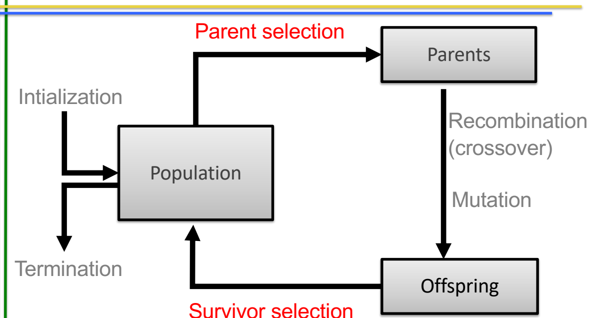
New Reading (all students)

- Later this week, we will discuss methods to maintain population diversity
- Reading:
  - Extinction Events Can Accelerate Evolution, Joel Lehman and Risto Miikkulainen. PLOS| one, 2015.

Fitness, Selection and Population Management

- Selection is the second fundamental force for evolutionary systems
  - What is the first fundamental force?
    - variation
- Components exist of:
  - Population management models
  - Selection operators
  - Preserving diversity

Scheme of an EA: General scheme of EAs



Population Management Models: Introduction

- μ–λ method:
  - μ: population size
  - λ: number of individuals to replace
- Generational model
  - $\lambda = \mu$: all parents replaced by children each generation
  - Typically, μ children created, though could be more
- Steady-state model
  - $\lambda < \mu$: some parents remain
  - λ can be as small as 1
  - What happens if λ is 0?
- Generation Gap
  - The proportion of the population replaced: λ/μ

Population Management Models: Fitness based competition

- Selection can occur in two places:
  - Selection from current generation to take part in mating (parent selection)
  - Selection from parents + offspring to go into next generation (survivor selection)
- Selection operators are representation-independent
  - They depend on the individual's fitness (and sometimes secondary measures)

Parent Selection: Fitness-Proportionate Selection

- Probability for individual *i* to be selected for mating in a population size *μ* with FPS is: $P_{FPS}(i) = f_i \Big/ \sum_{j=1}^{\mu} f_j$
- Problems include
  - Highly fit members can rapidly take over if rest of population is much less fit: Premature Convergence
  - At end of runs: fitnesses are similar, loss of selection pressure
  - Highly susceptible to fitness function translation (shifting)
- Scaling can fix last two problems
  - Windowing: $f'(i) = f(i) - \beta_g$ *for generation g* where *β* is worst fitness in this (last *k*) generations
  - Sigma Scaling: $f'(i) = \max(f(i) - (\bar{f} - c \bullet \sigma_f), 0)$

    where *c* is a constant, usually 2

Problem: Function translation

Individual	Fitness for function *f*	Selection Prob for *f*	Fitness for *f* + 10	Sel prob for *f* + 10	Fitness for *f* + 100	Sel prob for *f* + 100
A	1	0.1	11	0.275	101	0.326
B	4	0.4	14	0.35	104	0.335
C	5	0.5	15	0.375	105	0.339
Sum	10	1.0	40	1.0	310	1.0

Definition: Selection Pressure

- Degree of emphasis on selecting fitter individuals
  - High selection pressure: higher probability of choosing fitter members
  - Low selection pressure: lower probability of choosing fitter members
- Formal definition: probability of choosing best member over probability of choosing average member.
- How would you characterize selection pressure = 1?

Parent Selection: Rank-based Selection

- Attempt to remove problems of FPS by basing selection probabilities on *relative* rather than *absolute* fitness
- Rank population according to fitness and then base selection probabilities on rank (fittest has rank μ*-1* and worst rank 0)
- This imposes a sorting overhead on the algorithm, but this is usually negligible compared to the fitness evaluation time
- Ranking schemes not sensitive to fitness function translation

Rank-based Selection: Linear Ranking

$$P_{lin-rank}(i) = \frac{(2-s)}{\mu} + \frac{2i(s-1)}{\mu(\mu-1)}$$

- Parameterised by factor *s:* 1< *s* ≤ 2
  - measures advantage of best individual
- Simple 3 member example

Individual	Fitness for function *f*	Rank	Sel prob FPS	Sel prob LR (s = 2)	Sel prob LR (s = 1.5)
A	1	0	0.1	0.0	0.167
B	4	1	0.4	0.33	0.33
C	5	2	0.5	0.67	0.5
Sum	10		1.0	1.0	1.0

Rank-based selection: Exponential Ranking

- Linear Ranking is limited in selection pressure

$$P_{exp-rank}(i) = \frac{c^{\mu-i}}{\sum_{j=1}^{\mu} c^{\mu-j}}$$

- Denominator normalizes probabilities to ensure the sum is 1.0
- Note: $\sum_{j=1}^{\mu} c^{\mu-j} = \frac{c^{\mu}-1}{c-1}$
- So: $P_{exp-rank}(i) = \frac{c-1}{c^{\mu}-1} c^{\mu-i}, \qquad i \in \{1, \ldots, \mu\}$
- $0 < c < 1$, closer to 1 yields lower exponentiality

Rank-based selection: Exponential Ranking

Individual	Rank	Sel prob LR (s = 2)	Sel prob LR (s = 1.5)	Sel prob ER (c = 1/e)	Sel prob ER (c = 0.1)	Sel prob ER (c = 0.8)
A	1	0.000816	0.010408	3.314 e-22	9.000 e-50	3.568 e-06
B	5	0.004082	0.012041	1.809 e-20	9.000 e-46	8.711 e-06
C	10	0.008163	0.014082	2.685 e-18	9.000 e-41	2.658 e-05
D	20	0.016326	0.018163	5.915 e -14	9.000 e-31	2.476 e-04
E	50	0.040816	0.030408	0.63212	0.9	0.200003
Sum (of all 50)		1.0	1.0	1.0	1.0	1.0

Parent Selection: Tournament Selection

- All methods above rely on global population statistics
  - Could be a bottleneck esp. with very large population or on parallel architecture
  - Relies on presence of external fitness function which might not exist: *e.g.* evolving game players, evolutionary art
- Idea for a procedure using only local fitness information:
  - Pick *k* members uniformly at random then select the best one from these
  - Repeat to select more individuals

Parent Selection: Tournament Selection

- Probability of selecting member *i* will depend on:
  - Rank of *i*
  - Size of sample *k*
    - higher *k* increases selection pressure
  - Whether contestants are picked with replacement
    - Picking without replacement increases selection pressure
      - Without replacement, least fit k-1 individuals can never win a tournament
      - With replacement, even the least fit individual has probability $(1/\mu)^k$ of being selected (all tournament participants are that member)
  - Whether fittest contestant always wins (deterministic) or wins with probability *p (stochastic)*

Parent Selection: Uniform

$$P_{uniform}(i) = \frac{1}{\mu}$$

- Parents are selected by uniform random distribution whenever an operator needs one/some
- Uniform parent selection is unbiased - every individual has the same probability to be selected
- When working with extremely large populations, over-selection can be used.
  - Population ranked and divided into 2 groups: top *x* % in one group
  - *k* % of parents chosen from top group, remaining from other
    - Typical value for *k* is 80

Survivor Selection

- Managing the process of reducing the working memory of the EA from a set of μ parents and λ offspring to a set of μ individuals forming the next generation
- The parent selection mechanisms can also be used for selecting survivors
- Survivor selection can be divided into two approaches:
  - Age-Based Selection
    - Fitness is not taken into account
    - In SSGA can implement as "delete-random" (not recommended) or as first-in-first-out (a.k.a. delete-oldest)
  - Fitness-Based Replacement

Survivor Selection: Fitness-based replacement

- Elitism
  - Always keep at least one copy of the fittest solution so far
  - Widely used in both population models (GGA, SSGA)
- GENITOR: a.k.a. "delete-worst"
  - From Whitley's original Steady-State algorithm (he also used linear ranking for parent selection)
  - Rapid takeover: use with large populations (slows takeover)
- Round-robin tournament
  - Tournament competitors are: P(t): μ parents and P'(t): μ offspring
  - Pairwise competitions in round-robin format:
    - Each solution x from P(t) $\cup$ P'(t) is evaluated against q other randomly chosen solutions
    - For each comparison, a "win" is assigned if x is better than its opponent
    - The μ solutions with the greatest number of wins are retained for the next generation
  - Parameter q allows tuning selection pressure
  - Typically q = 10, but can be as large as μ - 1

Survivor Selection: Fitness-based replacement

- (μ,λ)-selection
  - based on the set of children only ($\lambda > \mu$)
  - choose best μ
- (μ+λ)-selection
  - based on the set of parents and children
  - choose best μ
- Often (μ,λ)-selection is preferred for:
  - Better in leaving local optima
  - Better in following moving optima
- Historically, $\lambda \approx 7 \cdot \mu$ was a good setting. More recently, $\lambda \approx 3 \cdot \mu$ is more popular

Selection Pressure – a different view

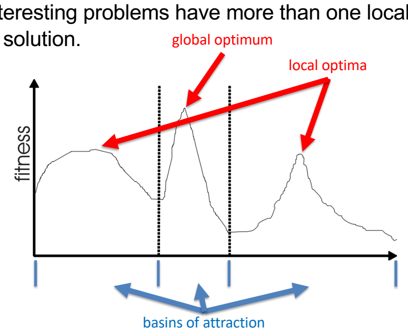
- Takeover time τ^* is a measure to quantify selection pressure
- The number of generations it takes until the application of selection completely fills the population with copies of the best individual
- For (μ,λ)-selection Goldberg and Deb showed:

$$\tau^* = \frac{\ln \lambda}{\ln(\lambda / \mu)}$$

- For proportional selection in a GA with $\mu = \lambda$, the takeover time is: $\lambda \ln \lambda$ (about 460 for pop size = 100)

Multimodality

Most interesting problems have more than one locally optimal solution.



Multimodality: Genetic Drift

- Finite population with global mixing and selection eventually convergence around one optimum
- Why?
  - Suppose population evenly divided between 2 optima
  - Eventually, due to random nature of variation operators, population is likely to skew slightly toward one optimum
  - This increases likelihood of choosing parents favoring that optimum
  - Then it is likely that over time population converges in that direction
- Not always desirable: might want to identify several possible peaks; sub-optimum can be more attractive
  - Common in evolved design problems – human judgements such as aesthetics might be important

Definition: Niche

- A *niche* is a subpopulation located in some area of the search space.
  - Niching can help ensure diversity
  - Also important when multiple optima should be represented in the population

Approaches for Preserving Diversity: Introduction

- Explicit vs implicit
- Implicit approaches:
  - Impose an equivalent of geographical separation
    - neighborhoods
  - Impose an equivalent of speciation
    - Species (subpopulations) that restrict mating
- Explicit approaches
  - Make similar individuals compete for resources (fitness)
  - Make similar individuals compete with each other for survival

Approaches for Preserving Diversity: Introduction

Different spaces:
- Genotype space
  - Set of representable solutions
- Phenotype space
  - The end result
  - Neighborhood structure may bear little relation to genotype space
- Algorithmic space
  - Equivalent of the geographical space on which life on earth has evolved
  - Structuring the population of candidate solutions
    - Across multiple cores, for example

Explicit Approaches for Preserving Diversity: Fitness Sharing

- Reduces fitness of individuals within a given niche by scaling ("sharing") their fitness in proportion to the size of the niche
- Controls number of members in niche since individuals allocated to niches in proportion to the niche fitness
- need to set the size of the niche σ_{share} in either genotype or phenotype space
- run EA as normal but after each generation set

$$f'(i) = \frac{f(i)}{\sum_{j=1}^{\mu} sh(d(i,j))} \qquad sh(d) = \begin{cases} 1 - d/\sigma & d \leq \sigma \\ 0 & otherwise \end{cases}$$

Explicit Approaches for Preserving Diversity: Fitness Sharing

- d is distance between two members – in same space (phenotype or genotype) as σ_{share}
- Note: if we used sh(d) = 1 for d < σ_{share} then the sum that reduces the fitness would simply count the number of neighbours, i.e., individuals closer than σ_{share}
- Using 1 – d/ σ_{share} instead of 1 implies that we count distant neighbours less
- Can change "shape" of sharing function by introducing α:
  - $(1 - d/\sigma_{share})^{\alpha}$
  - α = 1: linear
  - α > 1: affects decrease with distance

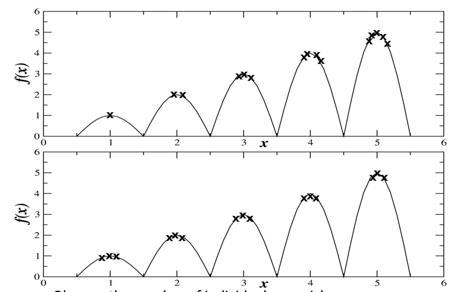
Explicit Approaches for Preserving Diversity: Crowding

- Attempts to distribute individuals evenly amongst niches
- relies on the assumption that offspring will tend to be close to parents
- uses a distance metric in either phenotype or genotype space
- randomly shuffle and pair parents
- produce 2 offspring per pair of parents

Explicit Approaches for Preserving Diversity: Crowding

- Set up competitions between parents and children
  - number the two p's (parents)and the two o's (offspring) based on competition in which they participate
  - Arrange competitions such that intercompetition distances are minimized:
    - if $d(p_1,o_1) + d(p_2,o_2) < d(p_1,o_2) + d(p_2,o_1)$
    - then let o_1 compete with p_1 and o_2 compete with p_2
- Winners move on to next generation
- Reduces likelihood that niches lose members since competition between parent and child that are similar

Explicit Approaches for Preserving Diversity: Crowding or Fitness sharing?



Fitness sharing

Crowding

Observe the number of individuals per niche

Explicit Approaches for Preserving Diversity: Extinction

- Key idea: reboot the population with significant infusion of new members
- Periodically kill large portion of the population and rebuild
- Many implementation choices that impact performance

Explicit Approaches for Preserving Diversity: Extinction

Design choices:

- What triggers an extinction event?
  - Typically at fixed intervals
  - Other options?
- How much of population to kill?
  - Typically 50% - 75%
- How to rebuild population?
  - Typically random members
  - Sometimes members created from survivors are used as well
- How quickly to rebuild population?
  - Always instantaneous
  - Does it have to be that way?

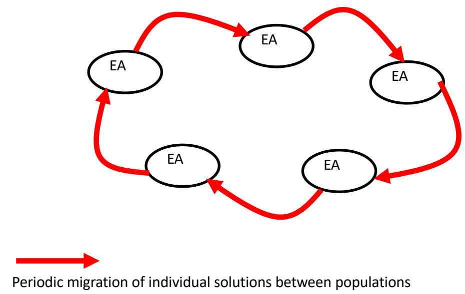
Implicit Approaches for Preserving Diversity: Automatic Speciation

- Only mate with genotypically / phenotypically similar members

  or

- Add bits (tags) to problem representation
  - that are initially randomly set
  - subject to recombination and mutation
  - when selecting partner for recombination, only pick members with a good match
  - Initially, similar tags do not imply similar solutions but within a small number of generations, they correlate
- Speciation does not guarantee diversity but does increase likelihood of diverse population

Implicit Approaches for Preserving Diversity: "Island" Model Parallel EAs



Implicit Approaches for Preserving Diversity: "Island" Model Parallel EAs

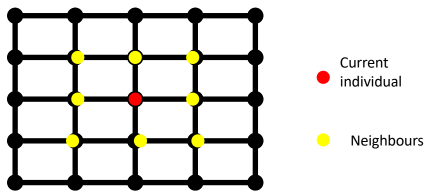
- Run multiple, independent populations in parallel
- After a (usually fixed) number of generations (an ***Epoch***), exchange individuals between populations
- Repeat until ending criteria met (optimal solution, max time, max generations, etc)
- Partially inspired by parallel/clustered systems
  - On such systems, typically one population per core

Island Model: Parameters

- How often to exchange individuals ?
  - too quick: all sub-populations converge to same solution
  - too slow: wastes time
  - most authors use range~ 25-150 generations
  - can do it adaptively (stop each pop when no improvement for (say) 25 generations)
- How many, which individuals to exchange ?
  - usually ~2-5, but depends on population size.
  - Copied vs moved
  - Martin *et al* found that better to exchange randomly selected individuals than best
- Operators can differ between the sub-populations

Implicit Approaches for Preserving Diversity: Cellular EAs

- Impose spatial structure (usually grid) in 1 pop



Implicit Approaches for Preserving Diversity: Cellular EAs

- Consider each individual to exist on a point on a (usually rectangular toroid) grid
- Selection (hence recombination) and replacement happen using concept of a neighborhood *a.k.a.* ***deme***
- Leads to different parts of grid searching different parts of space, good solutions diffuse across grid over a number of gens

Implicit Approaches for Preserving Diversity: Cellular EAs

- Assume rectangular grid so each individual has 8 immediate neighbors
- Equivalent of 1 generation is:
  - pick individual in pop at random
  - pick one of its neighbours using roulette wheel
  - crossover to produce 1 child, mutate
  - replace individual if fitter
  - cycle through population until done