9

Improving performance

Cedric Gondro

The Centre for Genetic Analysis and Applications
University of New England
The best of the rest

- Multicriteria optimization
- Parallelization
- Self-evolving parameters
- Hybrid evolutionary algorithms
Multicriteria optimization

- Independent problems
  - Scaling

- Dependent problems
  - Scaling
  - Co-evolution (concurrency)
  - Integration of objective functions in a single fitness function

- Real multicriteria problems
  - Trade offs - cannot eat and have the cake at the same time!
  - But can decide how much to eat and how much to keep!
No unique solution -> a set of equivalent solutions

Pareto optimal
- An improvement in any criterion will result in loss of fitness of other criteria
- Set of all possible solutions at the Pareto optimal
- Tip: calculate and keep the Pareto set for downstream decisions
Pareto or frontier?
Most common: weighting

- Self-evolving
  - accounts for how far can you go in each direction
  - can reach higher fitness than achievable with a predetermined fitness

- Manual
  - user controlled

Simple scaling function

\[
f = -1 \times \sum_i^n \frac{\sum_j^m (x_{ij} - y_{ij})^2}{\sigma^2 x_i}
\]
EAs are embarrassingly parallel
• Ideal for multi cores and low cost clusters
• Main difficulty is to get the processors talking
• Main bottleneck is the network lag

Simplest parallelization
• Run the same job independently on different machines

More exciting
• Master-slave model
• Island model
Self-evolving parameters

- Can be tricky to manually adjust the parameters

- Evolve problem and EA parameters concurrently
  - DE inherently evolves parameters
  - Common in ES and EP
  - Less common in GA and GP

- Simple strategy (GA/GP)
  - Evolve the EA parameters using ‘real’ fitness gain as fitness
Effect of parameters on fitness
Different classes of EAs
- Explore the best properties of each
- Current trend is to blur the line

EAs with other optimizers
- e.g. hill climbing, gradient search…

EAs with other techniques
- e.g. expert systems, neural networks, self organizing maps…
Initialize random values for variables within a set of constraints
Do until (termination criterion)
{
  Iteration i
  
  **GEP**
  Initialize random population of models
  Replace chromosome 0 with best model
  Do until GEPGeneration = GEPMaxGenerations
  
  
  Select
  Crossover
  Mutate
  Evaluate
  Replace
  Generation++

  } 
  If (GEP Best Model Improves Fitness)
    Replace model with best model from GEP
  Else Keep original model

  **Bloat Reduction Method**
  **DE**
  Use Best Model to optimize variables
  Initialize random population of variables within constraints
  Replace chromosome 0 with best variables
  Do until DEGeneration = DEMaxGenerations
  
  
  Select
  Crossover
  Mutate
  Evaluate
  Replace
  Generation++

  } 
  If (DE Best Values Improve Fitness)
    Replace variables with best values from DE
  Else Keep original variables
  i++
}
Advantages of the HDE-GEP

- Uses each method within its specific domain
- Allows the use of a different fitness function for structure discovery and parameterization
- Increases variability of the population, avoiding premature convergence
- Greatly improves model discovery