POLITECNICO DI MILANO

Lecture 12: Extended Compact Genetic Algorithms

Genetic Algorithms and Other Evolutionary Techniques

Prof. Pier Luca Lanzi

References

- David E. Goldberg "The Design of Innovation" Springer 2002
- Prof. David E. Goldberg, GE531 "Genetic Algorithms in Search, Optimization, and Machine Learning" University of Illinois at Urbana Champaign
- Martin Pelikan "Hierarchical Bayesian optimization algorithm: Toward a new generation of evolutionary algorithms", Springer 2005
- Prof. Martin Pelikan CS 5320 "Introduction to Evolutionary Computation", University of Missouri, St. Louis, MO

Motivations

- Variation operators can be replaced by building and sampling probabilistic models like in UMDA or PBIL.
- Probability vector is easy to learn and sample.
- Probability vector works great for simple problems like OneMax, where single-bit statistics lead towards the optimum.
- Probability vector even outperforms simple GAs for those simple problems because it provides more effective mixing.

But probability vector fails for problems with misleading single-bit statistics (deceptive bits).

UMDA on OneMax and Concatenated Trap

Why Failure?

- Onemax:
  - Optimum in 111…1
  - 1 outperforms 0 on average.
- Traps: optimum in 11111, but
  - f(0****) = 2
  - f(1****) = 1.375
- So single bits are misleading.

Linkage Learning in EDAs

- Challenge
  - Can we design probabilistic models that would allow us to solve additively separable problems of bounded order and other similar problems?

- Comments
  - This is closely related to linkage learning.
  - Need to modify probability vector to allow linkage learning.
  - Remember messy GA?

Two Steps

- Step 1: Show that it can be done
  - Try to find a probabilistic model that actually works for traps and other additively separable problems.
- Step 2: Do it
  - Find a way to actually learn such a model so that we solve traps and other additively separable problems and similar problems.

Step 1: Good Model for Trap-5

For each trap subfunction, store the proportions of different combinations of 5 bits in these 5 bits.

Example

- For n = 10 bits where a trap is applied to the first 5 bits and the last 5 bits, we would store two arrays
- One array stores the proportions of all combinations of 5 bits in the first 5 positions (i.e., proportions of 00000, 00001, …)
- The other array stores the proportions of all combinations of 5 bits in the last 5 positions.
- We have n/5 tables with 32 proportions/probabilities each.

Good Model for Trap-5

Good model for 15-bit trap of order 5

Good Model for Trap-5

- Learning
  - Go through the population, and for each trap subfunction, compute the proportions of all 32 combinations of 5 bits in the corresponding string positions.

- Sampling
  - Go through all trap subfunctions, and for each subfunction, generate the 5 bits in the corresponding positions with the probabilities computed from the selected solutions.
  - Generates 5 bits at a time.
  - Can be implemented using the roulette-wheel selection of fitness-proportionate selection.

Good Model for Concatenated Trap-5

Why does it work?

- For single bits (probability vector), 0 outperforms 1 (average fitness) and, as a result, it fails.
- For 5 bits, 11111 is the global optimum of each trap subfunction, so it performs better than any other combination of 5 bits in the corresponding string positions.
- So, 11111 should win over other competing schemata.
- That is what happens.
- We get convergence in O(n log n) evaluations just like for onemax and UMDA!

Step 2: How to Do It?
Extended Compact Genetic Algorithm

- Extended compact genetic algorithm (ECGA) of Harik (1999)
- Uses models that consider subsets of string positions.
- Consider the good model for traps as an example.
- Different subsets can be of different size.
- Each subset can contain any positions (no contiguous blocks)
- A subset of size k will require 2k probabilities/proportions to learn.
- ECGA learns both that subsets we need to consider as well as the probabilities for all combinations of values for each subset.
- ECGA models are called marginal product models (MPMs).

Example of an ECGA Model

A subproblem (BB) of order k requires 2^k proportions

Extended Compact Genetic Algorithm (ECGA)


```
ECGA(n,N,f,pm,pc)
  P = generate(n,N);
  while (!done())
     fv = evaluate(P,f,n,N);
     S = selection(P,fv,n,N);

     M = build_MPM_model(S,n,N);
     O = sample_MPM_model(M,n,N);

     P = replacement(O,P,n,N);
```

Learning ECGA Model

There are two tasks for learning an MPM model:

- Learn what subsets (BBs) to consider.
- Learn the probabilities/proportions for each subset.

ECGA

Learning Probabilities

- Two inputs
  - Population of selected solutions.
  - Subsets of string positions (BBs).
- Task: Learn all the proportions for the given BBs and population.
- For every subset of string positions (BB), do the following:
  - Denote the number of string positions in this subset by k.
  - Allocate an array for 2k positions (k is the number of string positions in this BB) to store the proportions.
  - Compute the proportion of each of the 2k instances of this BB from the selected population of solutions.
  - Store all the proportions in the allocated array.

Learning Probabilities: Example

- Inputs
  - Subsets (BBs): [1 3] [2] [4 5]
  - Population:

    01100
    00101
    11111
    01011

- Learned proportions
  - BB [1 3]: p(00)=0.25; p(01)=0.50; p(10)=0.00; p(11)=0.25;
  - BB [2]: p(0)=0.25; p(1)=0.75;
  - BB [4 5]: p(00)=0.25; p(01)=0.25; p(10)=0.00; p(11)=0.50;

Learning Subsets (Structural Learning, Task 1)

- Use greedy algorithm
- Start with each string position being a separate BB (like the probability vector).
- Repeat until we cannot improve model anymore:
  - Consider merging any pair of BBs and evaluate how much each merge of two BBs improves the model.
  - Merge the best pair of BBs (improving the model the most).

Learning Subsets (Structural Learning, Task 1)

How to Evaluate Model Structure?

- What we have
  - We know how to learn proportions for given BB decomposition (MPM).
  - We know how to learn MPM (where the BBs are) if we knew how to compare quality different MPMs.

- What we want
  - We need to know how to compare two alternative MPMs.
  - Example: [1 3] [2] [4] [5] vs. [1 3] [2] [4 5]
  - What information do we have? Selected set of solutions.

Occam's Razor

- One should not increase, beyond what is necessary, the number of entities required to explain anything.
- Other words: Simplest model is the best.
- Other ways to argue: Bayesian statistics.
- But let's believe this for now.
- We'll use a specific measure based on this principle: Bayesian Information Criterion or BIC (Schwarz, 1978).

Bayesian Information Criterion (BIC)

Two Components

$$\mathrm{BIC}(M) = -\sum_{i=1}^{N} \log_2 p(X_i|M) + N_{\text{params}} \times \frac{\log_2 N}{2}$$

number of bits to encode the population given the model.

number of bits to encode all probabilities/proportions.

Bayesian Information Criterion

- BIC(M) = number of bits to encode both the population as well as the model parameters.
- Population component represents accuracy.
- Parameter component represents pressure toward simple models with fewer parameters.

BIC Components

- Probability of Xi given M is given by

$$p(X_i|M) = \prod_{j=1}^{m} p(S_j|M)$$

- where m is the number of subsets (BBs) encoded by M, and p (Sj |M) denotes the proportion/probability of the jth subset of bits given by model M.

- Observation (from information theory)
- $-\sum_{i=1}^{N} \log_2 p(X_i|M)$ component of BIC represents the number of bits required to encode the entire population of selected solutions with the knowledge of the model M.

Generating New Solutions

- Go one subset of bits after another (based on the model).
- For each subset, generate bit combinations based on the observed proportions.
- Each BB has the same probability as its proportion.
- This can be done with the roulette wheel generator.
- Recall how we did this for the good model for traps.

Summary

- ECGA Solves Additively Separable Problems
- ECGA can find BBs (linkage learning).
- ECGA can mix BBs (by sampling model).
- ECGA solves decomposable problems fast.
- Requires about O(n1.5 log n) evaluations (regardless k).
- The challenge we discussed at the beginning is met by ECGA.
- Very important result, but also done with other EDAs and other methods.