Evolutionary Algorithms
– simulated evolution

Thomas Stidsen
tks@imm.dtu.dk

DTU-Management
Technical University of Denmark

Outline1

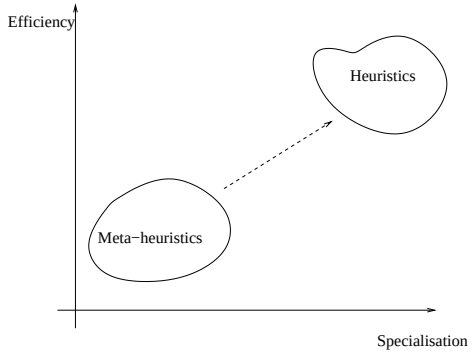
- Heuristics and Meta-heuristics
- General information on EA's
- The standard EA
- EA for Facility Location
- EA disected
- Different EA types
- Advanced topics
- Why and why not use EA
- Parameters

Specialised heuristics vs. metaheuristics

This part of the course is termed: Metaheuristics, what is a metaheuristic ?

- A **general** skeleton to an algorithm applicable to a wide range of optimisation problems.
- A meta-heuristic often "evolve" into a more specialised heuristic when it is specialised to a particular problem
- Approximation algorithms and very large neighbourhood searches are very specialised techniques which have a more limited scope

Specialised heuristics vs. metaheuristics



General Information

Evolutionary Algorithms is **not** just Genetic Algorithms.

- EA tries to mimic the Darvinian evolution
- EA is significantly different from SA and TABU and thus complements these.
- EA is difficult to use:
  - Many parameters
  - Parameters are hidden and depends non-linearly on each other.

History

- Invented independently three different places, in the 60'ies:
  - Michigan: John Holland, Genetic Algorithms.
  - Germany: Schwefel and Rechenberg, Evolution Strategies.
  - California: Edward Fogel: Evolutionary Programming.
- First dedicated conference: ICGA-85.
- Popularised by D. Goldberg, 1989 (Genetic Algorithms).

The Standard EA

Initialise the entire population
repeat
    Select promising individuals from current population
    Reproduce selected individuals
    Mutate reproduced individuals
    Evaluate created individuals
    Create new/updated population
until finished()
return best from population

Application of EA

When applying EA's to a certain problem we need to:

- Define problem dependent parts
- Select type of EA to use
- Select parameters for EA

Problem dependent parts

This is where the modelling of a real world problem begins, three different parts needs to be specified:

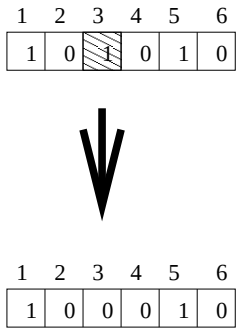
- Evaluation
- Crossover and Mutation operators (and Representation)

Let's look at the simple and classic problem ...

Representation: A Bit vector

Very simple: Boolean vector, corresponding to e.g. binary variables ...

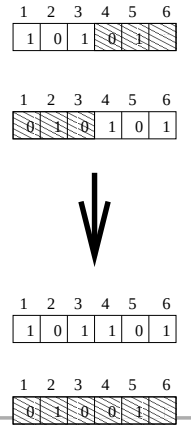
Mutation



Exactly like the neighborhood

Crossover

The simple 1-point crossover:



Operators

The bit-mutation operator and the 1-point crossover are general non-problem dependent operators. Generally these are not the best operators for specific problems. What could be the alternative ?

- Problem dependent crossovers ???
- PMX crossover to Christmas Lunch problem

Reproduction

What is a good crossover ? Difficult to quantify but:

- Avoid epistasis: Related genes should be “close”.
- Avoid collateral mutation, keep the genetic material.
- New genetic material should be created by the mutation operator.
- Choose the best material (a little dangerous).

Creating good reproduction operators is an art not a science.

Optimisation algorithm choice

Choosing the right Evolutionary Algorithm for the optimization problem then becomes the next issue, the components are:

- **Population**, several current solutions instead of just one.
- **Selection**, which individuals should the next population consist of.
- **Stopping Criteria**, when are we finished ?

The Components II: Population

- The population is necessary for the reproduction.
- Makes the algorithm robust, but increases the need for computer power and memory.
- The trade off: Prob. of getting a good solution vs. time.
- The size determines this trade off and is problem dependent.
- Important measure: Convergense, is the degree of similarity of the solutions in the population.

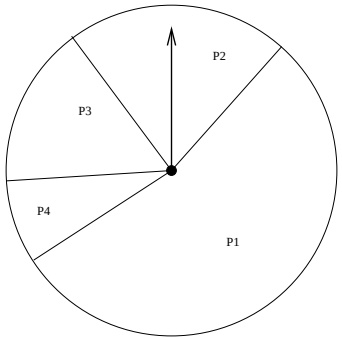
The crossover **requires** a non-converged population to work

The Components III: Selection

A much overlooked component which is very important.

- Takes one population to a reduced population.
- Can be compared to the temperature of the Simulated Annealing.
- Three important types:
  - Proportional selection, the classic method, but problematic ...
  - Tournament selection, the simplest method.
  - Stochastic Universal Sampling after Linear Fitness ranking.

Selection: Proportional Selection



Selection: Proportional Selection

This classical method is the basis for most of the theory, but there are two problems:

- Premature convergence: A good initial individual may quickly spread into the entire population.
- Slow finishing: In the end, the individuals usually have almost identical fitness values

For these reasons, proportional selection is never used ...

Selection: Tournament selection

A simple and efficient approach: Select two different individuals randomly and choose the best:

- Avoids the premature convergence and slow finishing.
- Easy to use in parallel

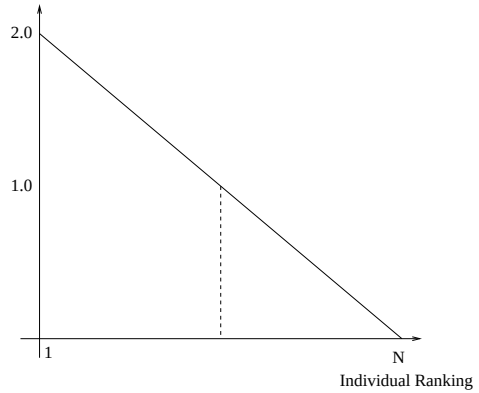
Random fluctuations may cause problems

Selection: LFR-SUS

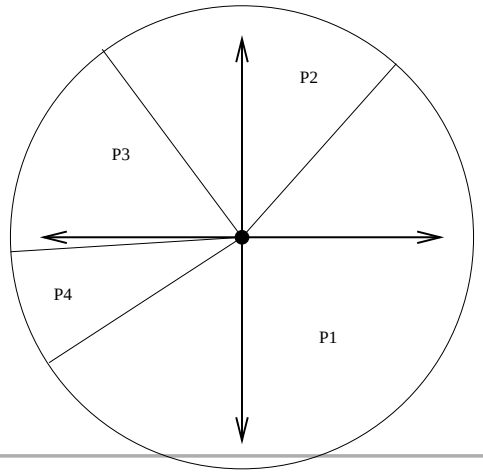
LFR-SUS: Linear Fitness Ranking with Stochastic Universal Sampling. Works in several stages:

- Sort the entire population
- Assign selection value linearly
- Use stochastic universal sampling to copy individuals

LFR



SUS



Selection: LFR-SUS

- The most “fair” method, with the least noise in the sampling of new individuals.
- More computational demanding, but that’s usually not a problem
- Difficult to parralellise

The Components IIII: Reproduction

The biggest difference compared to other algorithms: Create a new individual/solution by combining material from two previous solution:

- Intuitively: Create new solutions by combining existing solutions in a new way.
- Very problemdependent.
- Standard types: 1-point, 2-point.
- Traditionally considered the most important component.
- If you cannot create a good crossover, don't use EA.

The Components IIII: Mutation

Traditionally a background operator.

- Corresponds to the traditional neighbourhood update: $\mathbf{s}_m \in \mathcal{N}(\mathbf{s}_l)$.
- Possibility for guided mutations.

Two recomandable EA's

Two good algorithms worth considering are:

- Generational EA with LFT-SUS (bacteria evolution)
- Steady State EA with tournament selection (elephant evolution)

Depending on the problem, these are excellent algorithms.

Generational EA

$P \leftarrow RandomIndividuals()$
repeat
  $P' := LFR - SUS(P)$
  $P'' \leftarrow \{\}$
  while $P' \neq \{\}$
    $p_1, p_2 := RandomRemove2(P')$
    $o_1, o_2 := Crossover(p_1, p_2)$
    $P'' \leftarrow o_1', o_2' := Mutate(o_1, o_2)$
  endwhile
  $P := P''$
until finished()
return $Best(P)$

Steady State EA

$P \leftarrow RandomIndividuals()$
repeat
$p_1, p_2, p_3, p_4 := RandomSelect4dif(P)$
$o_1, o_2 := Crossover(Best(p_1, p_2), Best(p_3, p_4))$
$o_1', o_2' := Mutate(o_1, o_2)$
$r_1, r_2, r_3, r_4 := RandomSelect4different(P)$
$Replace(P, Worst(r_1, r_2), o_1)$
$Replace(P, Worst(r_3, r_4), o_2)$
until finished()

Convergence ?

What did I mean by converged ? Depending on the representation, we may look on the similarity of the different individuals:

- Binary strings: Hamming distance
- Permutation ???
- Real valued arrays: Euclidian distance
- Special structures: ???

Advanced Topics

There are **MANY** variations of EA's, just to name a few:

- Island EA.
- Hill climber inclusion: Baldwin and Lamarcian evolution.
- Parallel implementations.
- Multi-criteria optimisation.
- Co-evolutionary EA.
- Automatic operator probability adjustment.

EA Theory

As for the other metaheuristics, the EA theory is very limited:

- The Schemata theorem, in your notes.
- Price'es theorem. Comes from population biology, and has been transferred to EA's by Lee Altenberg. Very complicated mathematics.
- Goldberg's work on population sizing.

In practice none of theories are useable. Note, contrary to SA there is no convergence theory.

The first theory book ...

Is the book: “Genetic Algorithms - Principles and Perspectives”, C.R. Reeves & J. E. Rowe.

- The first (to my knowledge) theory **book** on evolutionary algorithms.
- The lack of theory is acknowledged ...
- ... but not ignored !

Many bad articles ...

“The number of papers devoted to GA’s is enormous - and increasing - but sadly, too many of them have titles like: **A New Genetic Algorithm for the Multi-Resource Travelling Gravedigger Problem with Variable Coffin Size**”, “Genetic Algorithms - Principles and Perspectives”, C.R. Reeves & J. E. Rowe.

When to use EA ?

When should you consider EA ?

- If the problem is REALLY hard:
  - If the problem is very multimodal.
  - If the problem is a multi criteria problem.
  - If the standard neighbourhood is bad ...
- If you have a good reproduction operator.
- If your problem is multi-objective

When not use EA

Don’t use EA if:

- Your problem is too simple.
- Your (single solution) neighbourhood operator is efficient
- Your reproduction cannot produce good individuals.
- Time is critical.

Parameters

There are more parameters in EA than in SA and most TABU algorithms. Further some parameters are **hidden**:

- Population size.
- Selection pressure, Important but hidden parameter.
- Operator probability: Which operator should the algorithm use ?
- Stopping criteria.
- Generation gap.

These are just the basic parameters, many others may show up ...

Tips and tricks

A few general advices:

- The most important: Get the representation and reproduction operator right.
- Use a large population initially.
- Be carefull about the selection pressure, use a low selection pressure initially.
- Tune your evaluation function, this is where the bulk of the work is performed by the computer.
- Watch the convergens.