SWINBURNE UNIVERSITY OF TECHNOLOGY

COS30018 – Week 7:
Evolutionary Computing/Algorithms (EC/EAs)

1

Nature-inspired computing

- Nature has always served as a source of inspiration for engineers and scientists
- The best problem solver known in nature is:
  - the (human) brain that created “the wheel, New York, wars and so on” (after Douglas Adams’ Hitch-Hikers Guide)
  - the evolution mechanism that created the human brain (after Darwin’s Origin of Species)
- Answer 1 → neurocomputing
  - Week 6
- Answer 2 → evolutionary computing
  - Today + Week 8

2 / 39

2

Contents

- Motivations/applicable situations
- Basics of Evolutionary Computing (EC) Metaphor
- Basic scheme of an EA
- Basic Components:
  - Representation / Evaluation / Population / Parent Selection / Recombination / Mutation / Survivor Selection / Termination
- Examples : eight queens / knapsack
- Typical behaviours of EAs
- EC in context of global optimisation

3

Motivation

- Searching some search spaces with traditional search methods would be intractable. This is often true when states/candidate solutions have a large number of successors.
  - Example: Designing the surface of an aircraft.

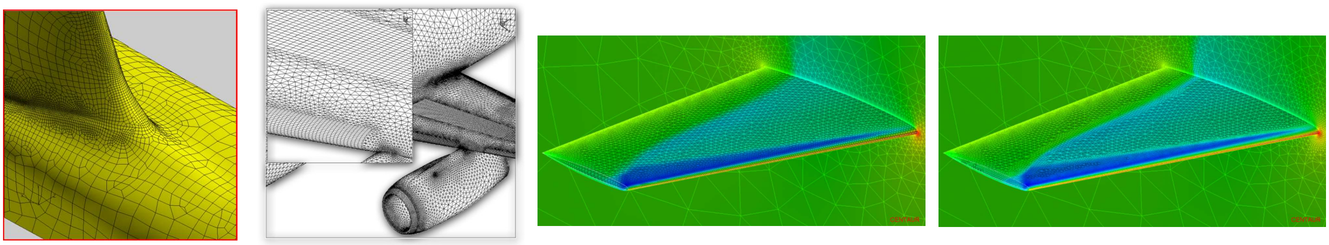


Image source: https://home.centaursoft.com

4

4

Evolutionary Computing (EC) – Main Idea

- Adaptation is Intelligence

(Nature)

Survival of the Fittest (aka. "natural selection")

Darwin/Wallace's theory: ***Evolution*** through *natural selection* of the *fittest* **individuals**

A process going through **multiple generations**

EC: How to use this idea for **Optimization**?

5

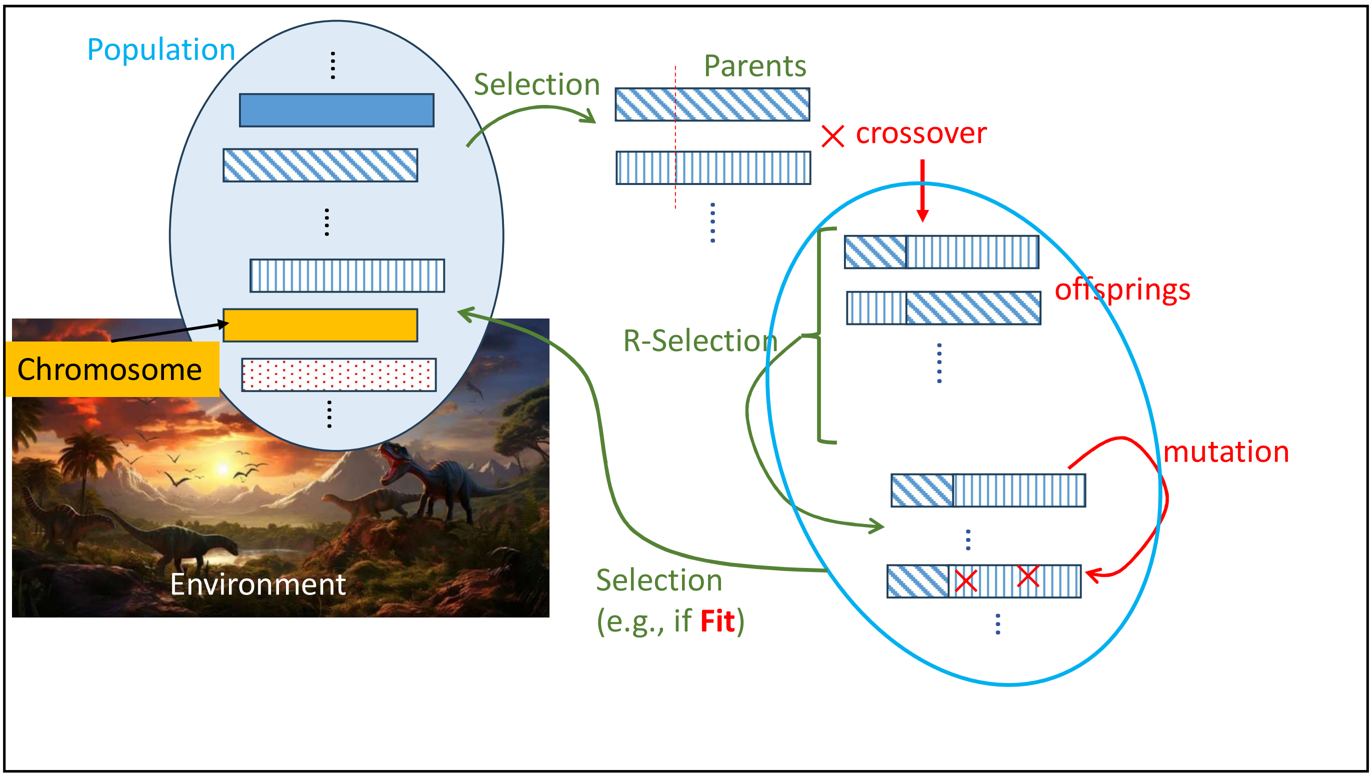
The Main EC Metaphor

natural selection	**EVOLUTION**		**PROBLEM SOLVING (or, OPTIMIZATION)**	evolutionary algorithm
	Environment	⟷	Problem	
	Individual	⟷	Candidate Solution	
	Fitness	⟷	Quality	

Fitness → chances for survival and reproduction

Quality → chance for an existing solution to survive and seed new solutions

6



7

Basics of EC metaphor

- A population of individuals exists in an environment with limited resources
- ***Competition*** for those resources causes selection of those ***fitter*** individuals that are better adapted to the environment
- These individuals act as seeds for the generation of new individuals through recombination and mutation
- The new individuals have their fitness evaluated and compete (possibly also with parents) for survival.
- Over time ***Natural selection*** causes a rise in the fitness of the population

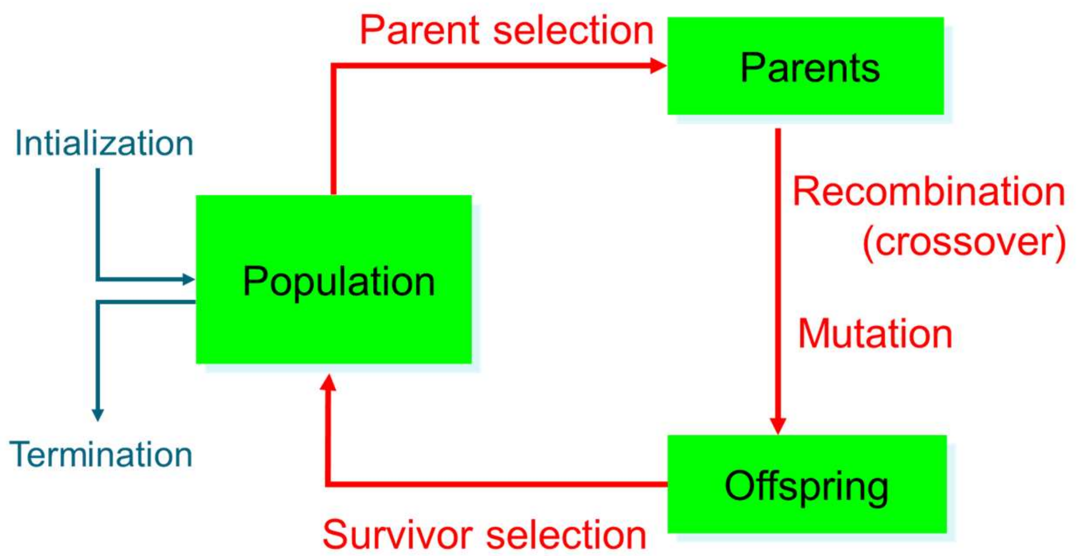
8

Basics of EC metaphor

- EAs fall into the category of "generate and test" algorithms
- They are stochastic, population-based algorithms
- Variation operators (recombination and mutation) create the necessary **diversity** and thereby facilitate novelty
- Selection reduces diversity and acts as a force pushing quality

9

General Scheme of EAs



10

Pseudo-code for typical EA

```
BEGIN
  INITIALISE population with random candidate solutions;
  EVALUATE each candidate;
  REPEAT UNTIL ( TERMINATION CONDITION is satisfied ) DO
    1 SELECT parents;
    2 RECOMBINE pairs of parents;
    3 MUTATE the resulting offspring;
    4 EVALUATE new candidates;
    5 SELECT individuals for the next generation;
  OD
END
```

11

What are the different types of EAs

- Historically different flavours of EAs have been associated with different representations
  - Binary strings : **Genetic Algorithms**
  - Real-valued vectors : **Evolution Strategies**
  - Finite state Machines: **Evolutionary Programming**
  - LISP trees: **Genetic Programming**
- These differences are largely irrelevant, best strategy
  - choose representation to suit problem
  - choose variation operators to suit representation
- Selection operators only use fitness and so are independent of representation

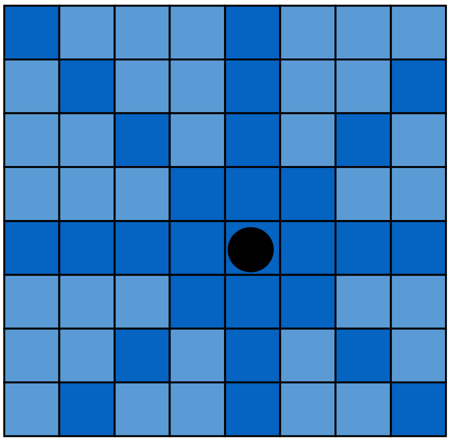
12

Examples of EAs – Genetic Algorithms (GAs)

- Advantages:
  - Easy to code
  - Can provide multiple solutions
  - Simple ways to avoid local minima/maxima (not guarantee)
  - Can be parallelized
- Disadvantages:
  - They can be slow
  - Can be hard to design a good fitness function
  - Can be hard to represent solutions of the problem as GA chromosomes

13

Example: the 8 queens problem



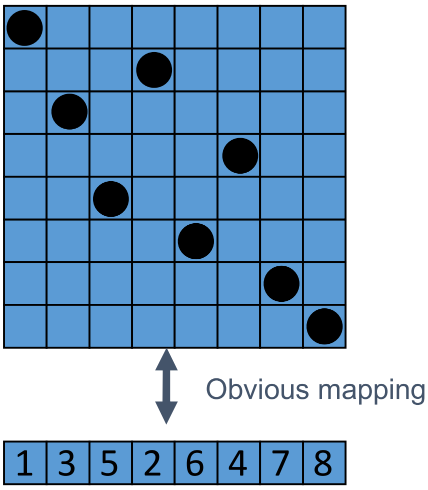
Place 8 queens on an 8x8 chessboard in such a way that they cannot attack each other

14

The 8 queens problem: Representation

a board configuration

a permutation of
the numbers 1 - 8



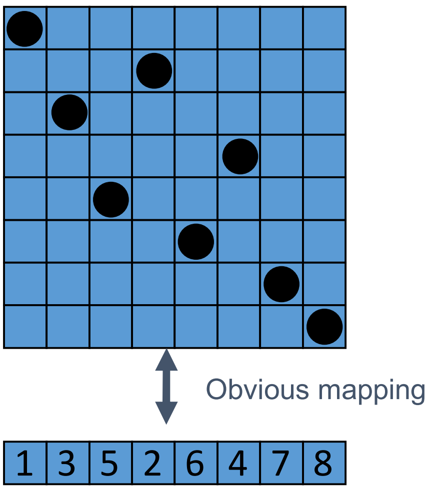
15

The 8 queens problem: Representation

Suitable representation?
GENETIC ALGORITHM

a board configuration

a permutation of
the numbers 1 - 8



16

Evaluation (Fitness) Function

- Represents the requirements that the population should adapt to
- a.k.a. *quality* function or *objective* function
- Assigns a single real-valued fitness to each phenotype which forms the basis for selection
  - So the more discrimination (different values) the better
- Typically we talk about fitness being maximised
  - Some problems may be best posed as minimisation problems, but conversion is trivial

17

8 Queens Problem: Fitness evaluation

- Penalty of one queen:
  the number of queens she can attack.
- Penalty of a configuration:
  the sum of the penalties of all queens.
- Note: penalty is to be minimized
- Fitness of a configuration:
  inverse penalty to be maximized

18

Population

- Holds (representations of) possible solutions
- Selection operators usually take whole population into account i.e., reproductive probabilities are *relative* to *current* generation
- Diversity of a population refers to the number of different **fitnesses** and/or **individuals/chromosomes** present (note: not the same thing)

19

Parent Selection Mechanism

- Assigns variable probabilities of individuals acting as parents depending on their fitnesses
- Usually probabilistic
  - high quality solutions more likely to become parents than low quality
  - but not guaranteed
  - even worst in current population usually has non-zero probability of becoming a parent
- This *stochastic* nature can aid escape from local optima

20

Variation Operators

- Role is to generate new candidate solutions
- Usually divided into two types according to their arity (number of inputs):
  - Arity = 1 (aka. unary operators): mutation
  - Arity > 1 : Recombination operators
  - Arity = 2 (aka. binary operators): typically called crossover
- There has been much debate about relative importance of recombination and mutation
  - Nowadays most EAs use both
  - Choice of particular variation operators is representation dependant

21

Recombination

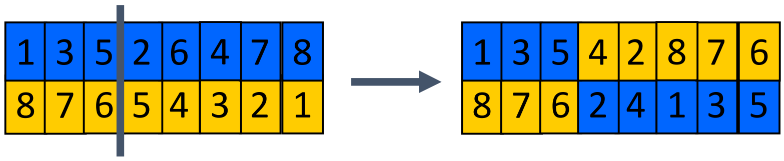
- Merges information from parents into offspring
- Choice of what information to merge is stochastic
- Most offspring may be worse, or the same as the parents
- Hope is that some are better by combining elements of genotypes that lead to good traits
- Principle has been used for millennia by breeders of plants and livestock

22

The 8 queens problem: Recombination

Combining two permutations into two new permutations:

- choose random crossover point
- copy first parts into children
- create second part by inserting values from other parent:
  - in the order they appear there
  - beginning after crossover point
  - skipping values already in child



23

Mutation

- Acts on one genotype and delivers another
- Element of randomness is essential and differentiates it from other unary heuristic operators
- Importance ascribed depends on representation and dialect:
  - Binary GAs – background operator responsible for preserving and introducing diversity
  - EP for FSM’s/ continuous variables – only search operator
  - GP – hardly used
- May guarantee connectedness of search space and hence convergence proofs

24

The 8 queens problem: Mutation

Small variation in one permutation, e.g.:

•swapping values of two randomly chosen positions,

1	3	5	2	6	4	7	8

→

1	3	7	2	6	4	5	8

25

Survivor Selection

- a.k.a. ***replacement***
- Most EAs use fixed population size so need a way of going from (parents + offspring) to next generation
- Often deterministic
  - Fitness based : e.g., rank parents+offspring and take best
  - Age based: make as many offspring as (reproduced) parents and delete all those parents
- Sometimes do combination (elitism)

26

The 8 queens problem: Selection

- Parent selection:
  - Pick randomly 5 parents and take best two to undergo crossover
- Survivor selection (replacement)
  - When inserting a new child into the population, choose an existing member to replace by:
  - sorting the whole population by decreasing fitness
  - enumerating this list from high to low
  - replacing the first with a fitness lower than the given child

27

Initialisation / Termination

- Initialisation usually done at random,
  - Need to ensure even spread and mixture of possible values
  - Can include existing solutions, or use problem-specific heuristics, to "seed" the population
- Termination condition checked every generation
  - Reaching some (known/hoped for) fitness
  - Reaching some maximum allowed number of generations
  - Reaching some minimum level of diversity
  - Reaching some specified number of generations without fitness improvement

28

8 Queens Problem: Summary

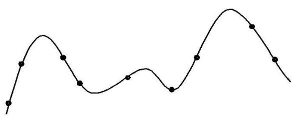
Representation	Permutations
Recombination	"Cut-and-crossfill" crossover
Recombination probability	100%
Mutation	Swap
Mutation probability	80%
Parent selection	Best 2 out of random 5
Survival selection	Replace worst
Population size	100
Number of Offspring	2
Initialisation	Random
Termination condition	Solution or 10,000 fitness evaluation

Note that this is ***only one possible***
set of choices of operators and parameters

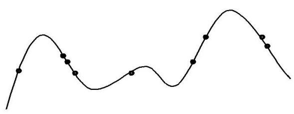
29

Typical behaviour of an EA

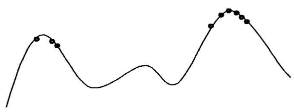
Phases in optimising on a 1-dimensional fitness landscape



Early phase:
quasi-random population distribution



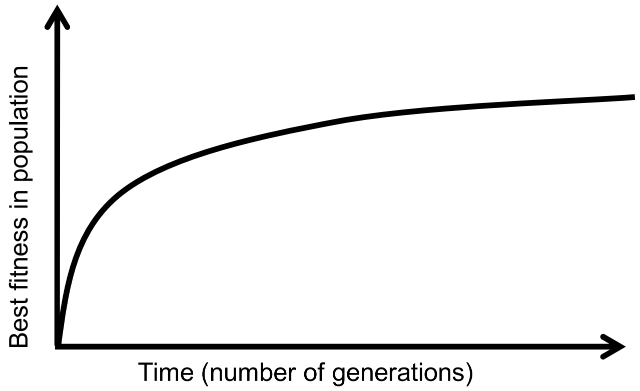
Mid-phase:
population arranged around/on hills



Late phase:
population concentrated on high hills

30

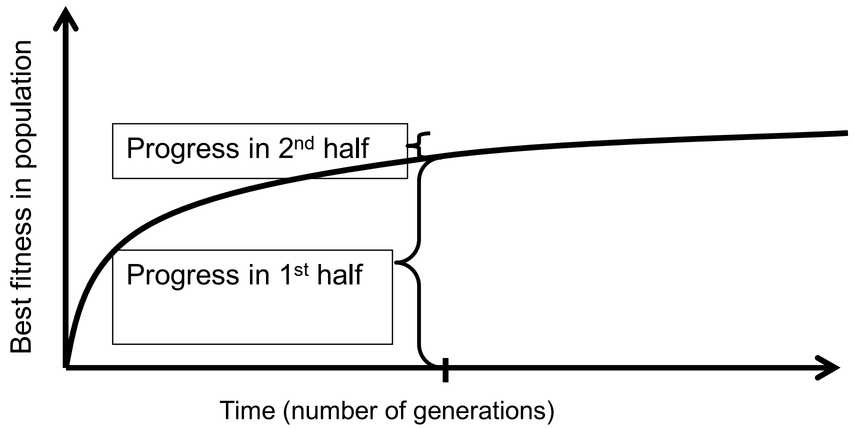
Typical run: progression of fitness



Typical run of an EA shows so-called “anytime behavior”

31

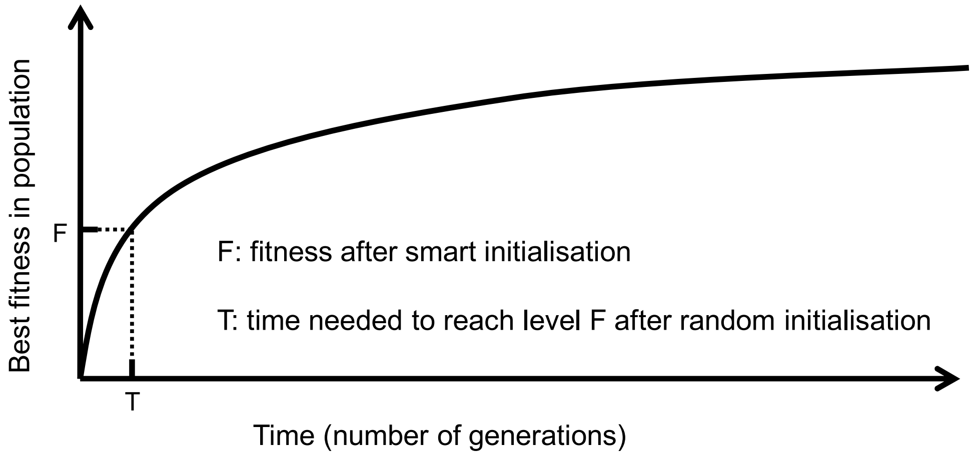
Are long runs beneficial?



- Answer:
  - it depends how much you want the last bit of progress
  - it may be better to do more shorter runs

32

Is it worth expending effort on smart initialisation?



- Answer : it depends:
  - possibly, if good solutions/methods exist.
  - care is needed

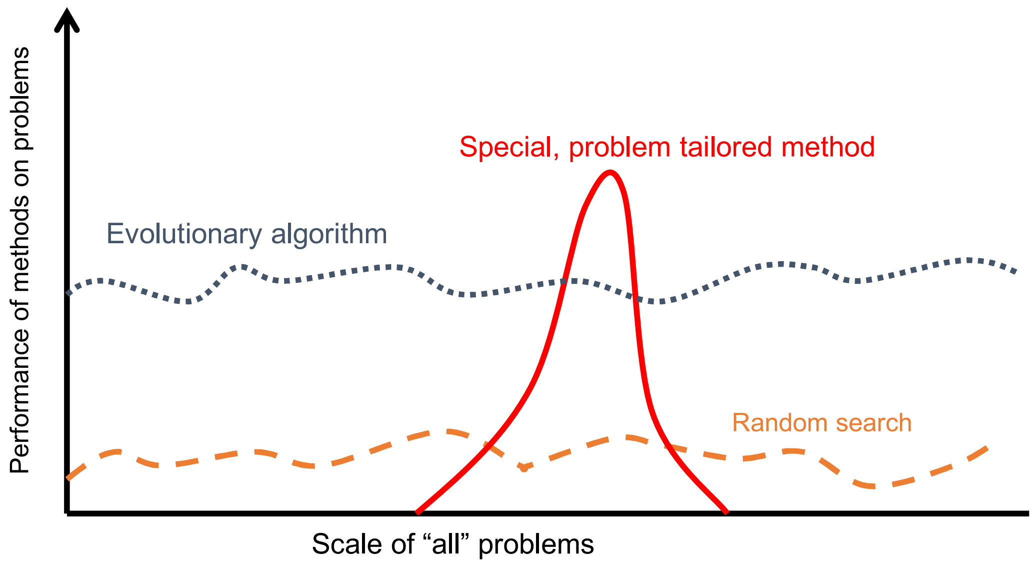
33

Evolutionary Algorithms in Context

- There are many views on the use of EAs as robust problem solving tools
- For most problems a problem-specific tool may:
  - perform better than a generic search algorithm on most instances,
  - have limited utility,
  - not do well on all instances
- Goal is to provide robust tools that provide:
  - evenly good performance
  - over a range of problems and instances

34

EAs as problem solvers: Goldberg's 1989 view



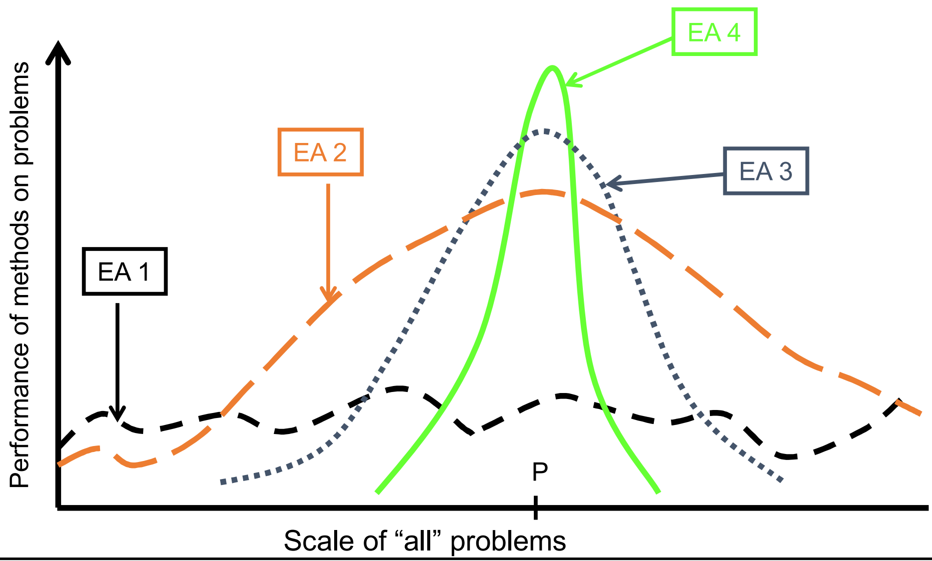
35

EAs and domain knowledge

- Trend in the 90's:
  adding problem specific knowledge to EAs
  (special variation operators, repair, etc)
- Result: EA performance curve "deformation":
  - better on problems of the given type
  - worse on problems different from given type
  - amount of added knowledge is variable
- Recent theory suggests the search for an "all-purpose" algorithm may be fruitless

36

Michalewicz' 1996 view



37

EC and Global Optimisation

- Global Optimisation: search for finding best solution x^* out of some fixed set S
- Deterministic approaches
  - e.g. box decomposition (branch and bound etc)
  - Guarantee to find x^*, but may run in super-polynomial time
- Heuristic Approaches (generate and test)
  - rules for deciding which $x \in S$ to generate next
  - no guarantees that best solutions found are globally optimal

38

Applicable situations

- Often used for optimization (scheduling, design, etc.) problems, though can be used for many other things as well, as we'll see a bit later.
  - Good problem for EAs: Scheduling air traffic
  - Bad problems for EA: Finding large primes (why?), 2D pathfinding (why?)

39

39

Applicable situations

- EAs work best when the "fitness landscape" is continuous (in some dimensions). This is also true of standard search, e.g. A*.
  - Intuitively, this just means that we can find a heuristic that gives a rough idea of how close a candidate is to being a solution.

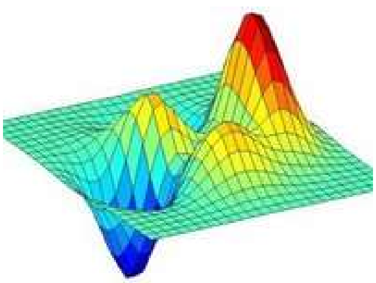


Image source: scholarpedia.org

40

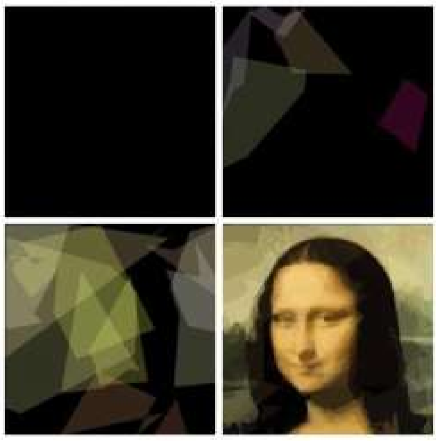
40

Examples - EA in the wild

- Image compression – evolving the Mona Lisa

Generation 1

Generation 301



Generation 2716

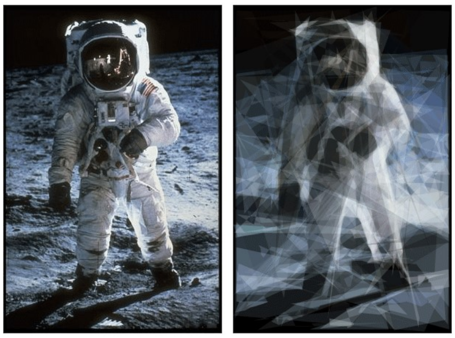
Generation 904314

http://rogeralsing.com/2008/12/07/genetic-programming-evolution-of-mona-lisa/

Evolving the Mona Lisa

- Uses only 50 polygons of 6 vertices each.
- Population size of 1, no crossover – parent compared with child, and superior image kept.
- Assuming each polygon has 4 bytes for color (RGBA) and 2 bytes for each of 6 vertices, this image only requires 800 bytes.
- However, compression time is prohibitive and storage is cheaper than processing time. ☹

More evolved images



http://rogeralsing.com/2008/12/07/genetic-programming-evolution-of-mona-lisa/

43