Evolutionary Computation

R.S.Forsyth@lboro.ac.uk

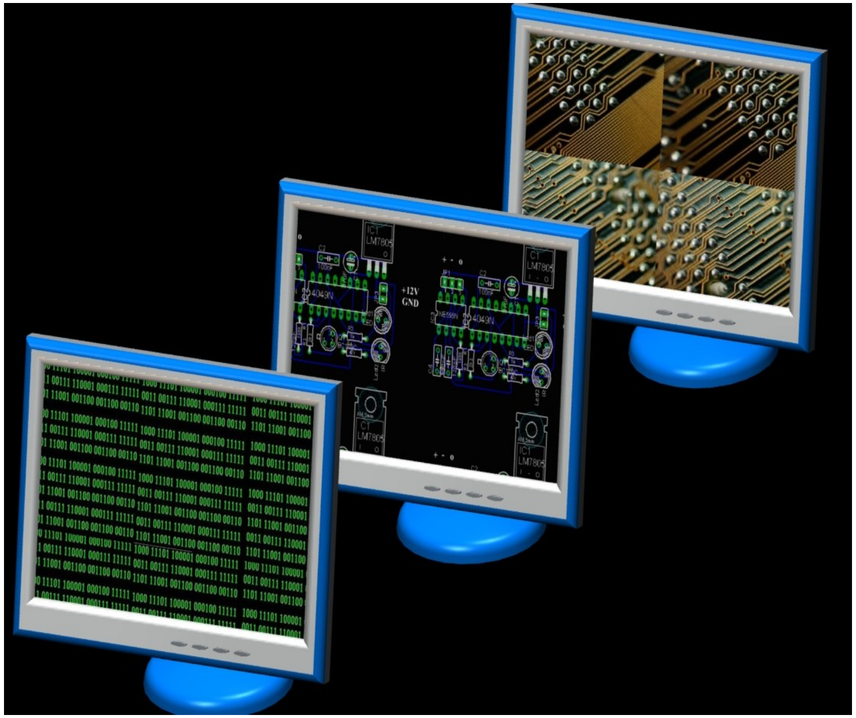
COC131

March 2009

Outline

- (1) Tour of fundamental concepts
- (2) Example implementation
- (3) Plus a few bits & pieces

Basic idea

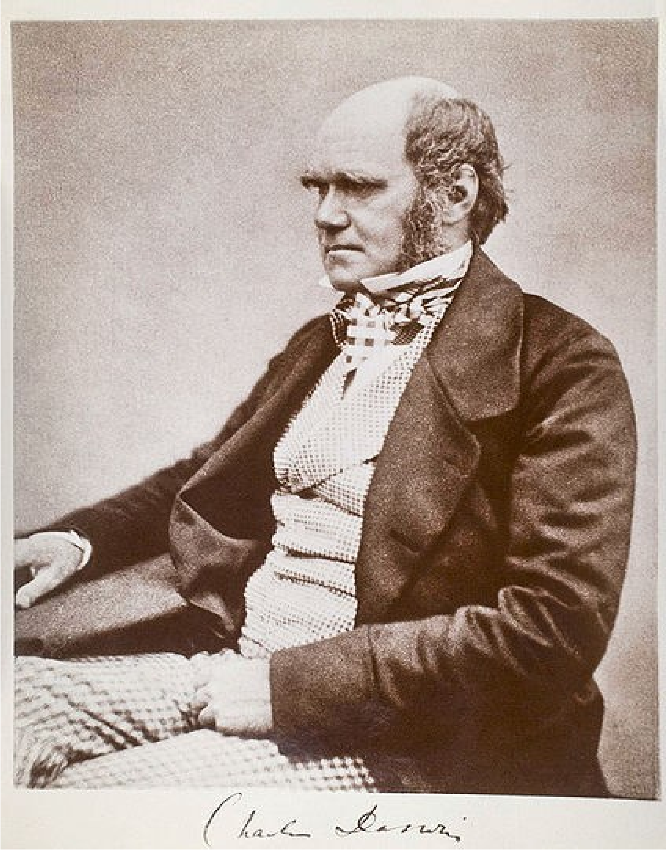


4 billion years of field testing can't be bad. (Can it?)

Like most neat computing ideas, Turing thought of it first

- Turing identified a third approach to machine intelligence in his 1948 paper entitled “Intelligent Machinery” (Turing 1948, page 12; Ince 1992, page 127; Meltzer and Michie 1969, page 23), saying:
- “There is the *genetical or evolutionary* search by which a combination of genes is looked for, the criterion being the survival value.”

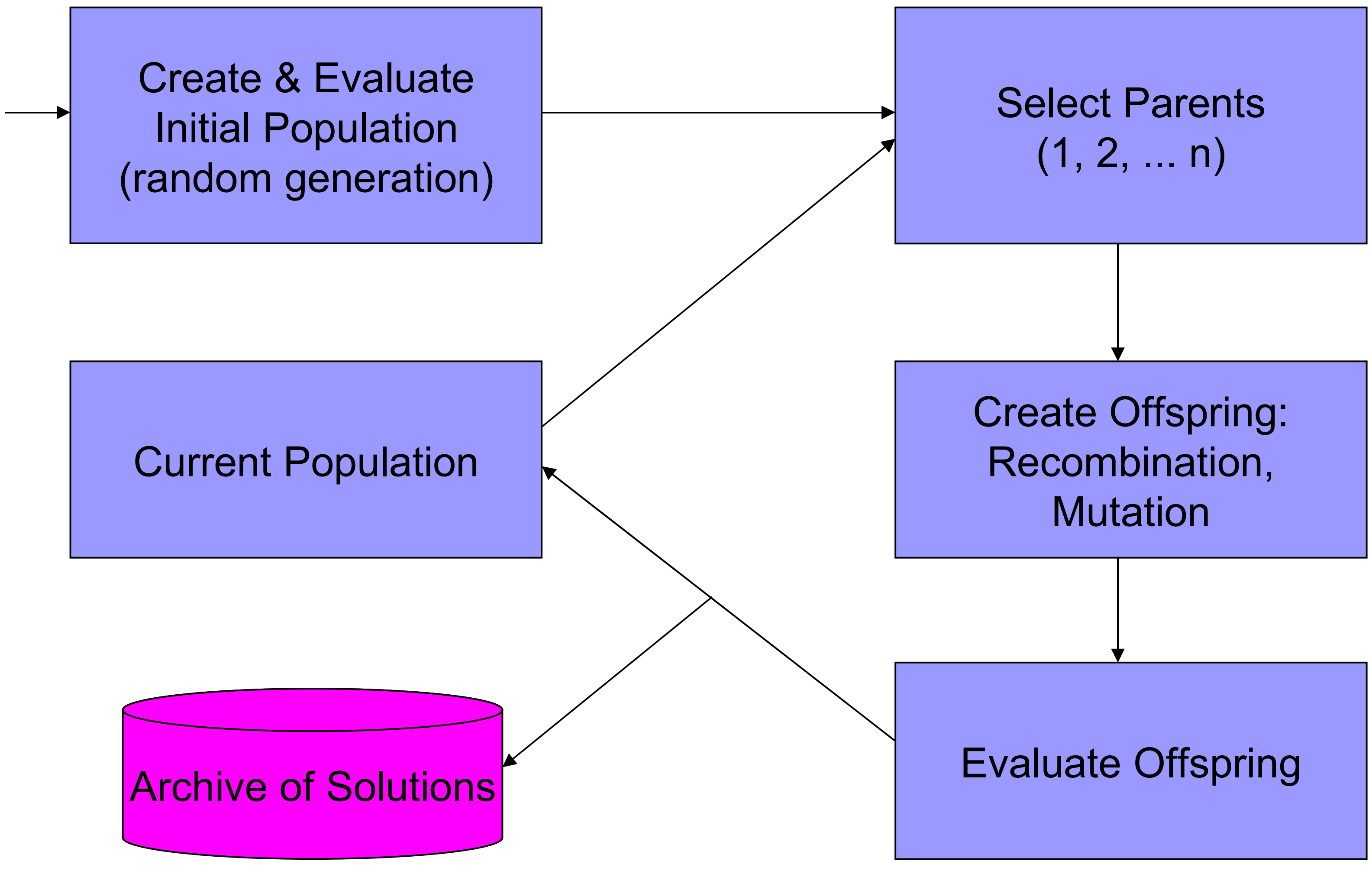
Though of course Darwin laid the foundations



Evolutionary Computing, major "species" ("genera", "families" ?)

- Evolution Strategy (ES)
  - Ingo Rechenberg, Germany
- Genetic Algorithms (GA)
  - John Holland, USA
- Genetic Programming (GP)
  - John Koza, USA
- Evolutionary Programming (EP)
  - Lawrence/David Fogel, USA

Basic Evolutionary Computing Cycle



Crossover operators

- Point crossover :

a b c d e f g h i j
0 1 2 3 4 5 6 7 8 9
a b c d 4 5 6 7 8 9

- Uniform crossover :

a b c d e f g h i j
0 1 2 3 4 5 6 7 8 9
0 b 2 3 4 5 g 7 i 9

Mutation operators

- Depends on problem representation :
  - flip a bit, e.g. 0->1, 1->0
  - add/subtract small random value to a floating-point number, e.g. 12.34 -> 12.21
  - change a symbol, e.g. * -> +
  - swap 2 elements, e.g. "lots" -> "lost"
    - (sometimes treated as separate operator, inversion)
- Has to be "small" change in some sense
  - explores "neighbouring" solutions

Selection

- Warning! Don't use "fitness-proportional selection"
  - (aka "Roulette wheel selection")
- Whitely, D.L. (1989).
  - The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best
  - Proceedings of the 3rd International Conference on Genetic Algorithms
  - Morgan Kaufmann Publishers Inc.

Generational versus incremental procedures

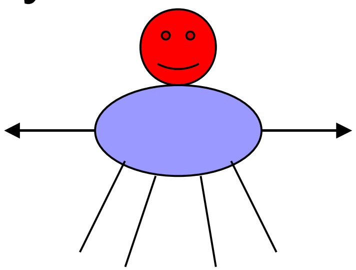
- Generational :
  - like Mayflies or 17-year cicadas
  - entire population replaced on each cycle
- Incremental :
  - like most plants, vertebrates etc.
  - some parental survival (often majority)
- N.B. Computational effort should be measured by number of offspring created
  - not number of generations

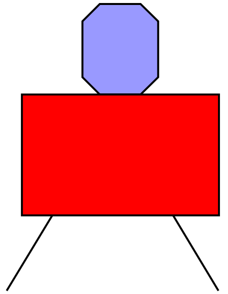
From genotype to phenotype

- Genome contains info on how to build body, e.g.:
  - 2, 1, 1, 1, 0, 4, 2
  - 0, 0, 0, 0, 1, 2, 0

- Genes:
  - Eyes, Smile, Roundbody, Redhead, Redbody, Legs, Arms

- "Body"





Then environment evaluates phenotype

- Fitness function gives a score, e.g.
  - network connectivity with simulated traffic
  - wing shape in simulated wind tunnel
  - investment strategy applied to past price series
  - timetable compared to constraints
  - classification rule-set applied to training data

Key implementation ingredients

- Genome representation :
  - should be easy to chop into bits and splice bits together
  - Basic GA uses binary strings
  - ES often uses floating-point vectors
  - GP uses tree structured representation
- Fitness function :
  - Problem-dependent, not always obvious

A CACE study: IOGA revisited

- Background:
  - 1-NNC a simple & robust classification technique (aka IBL)
    - Just find "nearest" case in training data to current instance & assign its category label as predicted class
      - requires a distance function (more details later)
  - But:
    - no compression, just memorization
    - rather slow classification phase
    - fails to deal with redundant features
    - doesn't help insight

Enhancing basic 1-NNC

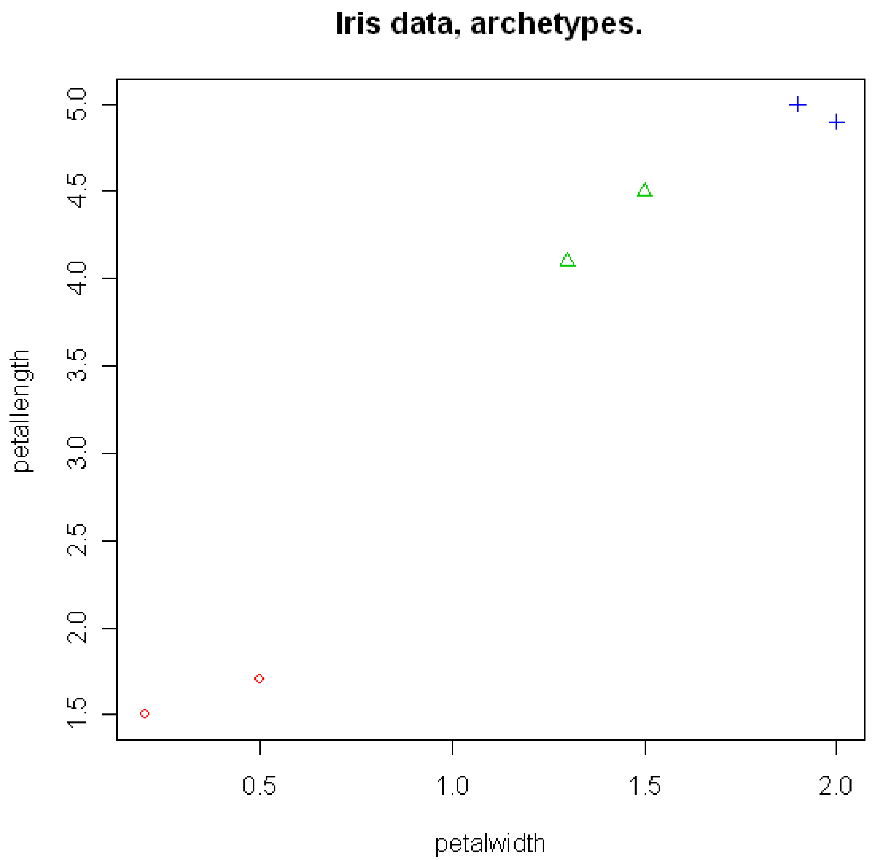
- Many improvements proposed
  - E.g. removing redundant features
  - E.g. removing redundant instances
- But not both at once (till 1995)
  - Ideally suited to genetic representation!

Reviewing some basic concepts

- Typical classifier trained on "flat-file" training data:
  - data matrix (R rows, C columns)
    - cases/instances, attributes/features
  - 1 column gives known category label
  - (Weka uses arff representation)
    - attribute-relation file format
- Hence concept of "feature space"

Example of feature space

- petallength petalwidth typecode
- 1.7 0.5 1
- 1.5 0.2 1
- 4.5 1.5 2
- 4.1 1.3 2
- 4.9 2.0 3
- 5.0 1.9 3



IOGA/EASE representation scheme

- Bitstring of length R + V
  - R = number of rows (instances)
  - V = number of variables (features)
- First R bits:
  - 1 means keep this case, 0 means ignore
- Last V bits:
  - 1 means use this feature, 0 means ignore
- N.B. leave-1-out mode:
  - no case allowed to be its own nearest neighbour

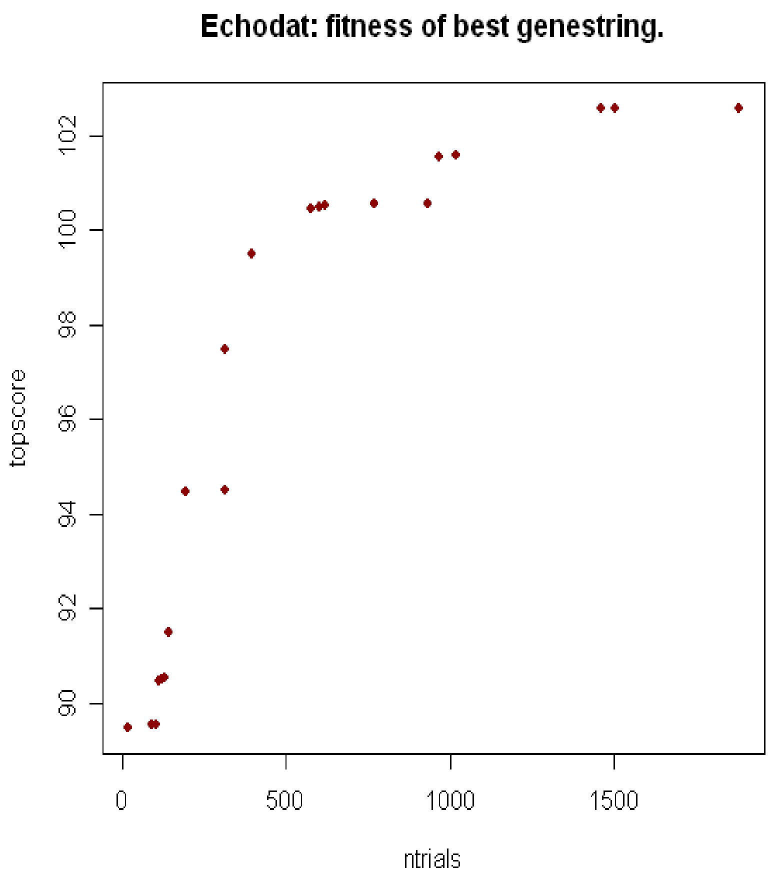
EASE fitness function

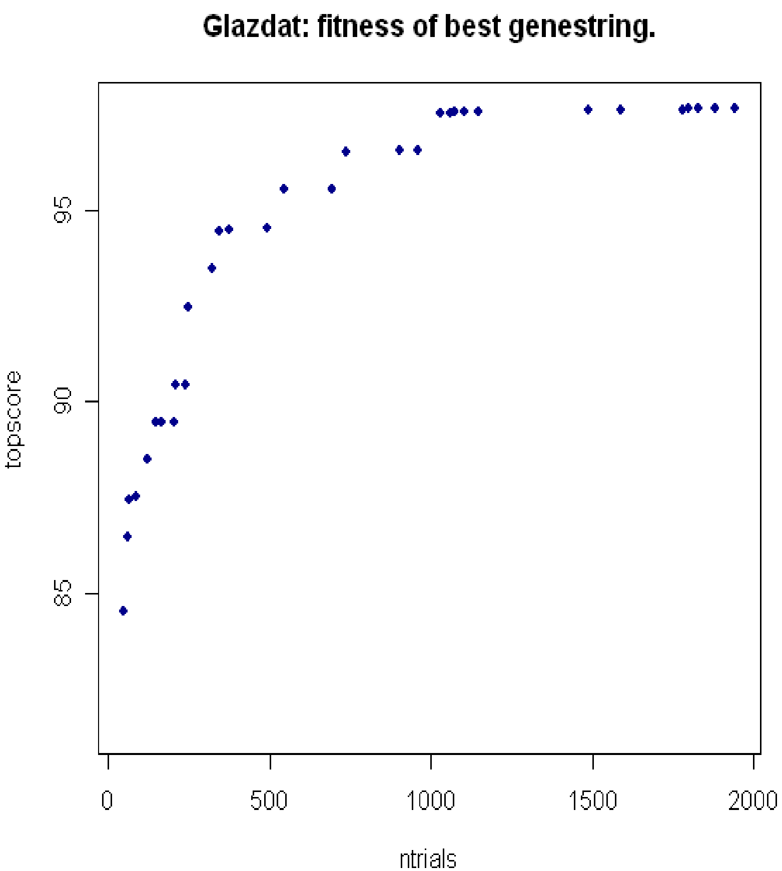
- Based on leave-1-out classification score:
  - F = K – B/(R+V)
    - F = fitness
    - K = number of correct classifications
    - B = number of bits set to 1 in genestring
    - R = cases, V = features
- Bias towards brevity (B/(R+V)) :
  - essentially just a tie-breaker
  - "Ockham's Razor" ?

Applied to four datasets

- Echo (sonar data, in UCI)
  - cases 107/101, vars 60, classes 2
- Glaz (glass data, z-scores, from UCI)
  - cases 111/103, vars 9, classes 6
- Iris (Iris data, in UCI)
  - cases 77/73, vars 4, classes 3
- Zoobase (animal data, in UCI)
  - cases 54/47, vars 17, classes 7

Examples of fitness progression





EASE + CACE

- Evolutionary Archetype Search Engine
  - uses evolutionary algorithm to generate archetypes

- Closest Archetype Classification Engine
  - uses archetype file from EASE to classify (holdout sample) cases
  - applies nearest-neighbour technique
    - ("city-block" distance metric in results presented here)

Accuracy comparisons

Dataset	1-NNC holdout success %	CACE holdout success % (median of 3)
echodat	76.24	79.21
glazdat	65.05	62.14
irisdat	95.89	98.63
zoobase	93.62	91.49
mean =	82.70	82.87

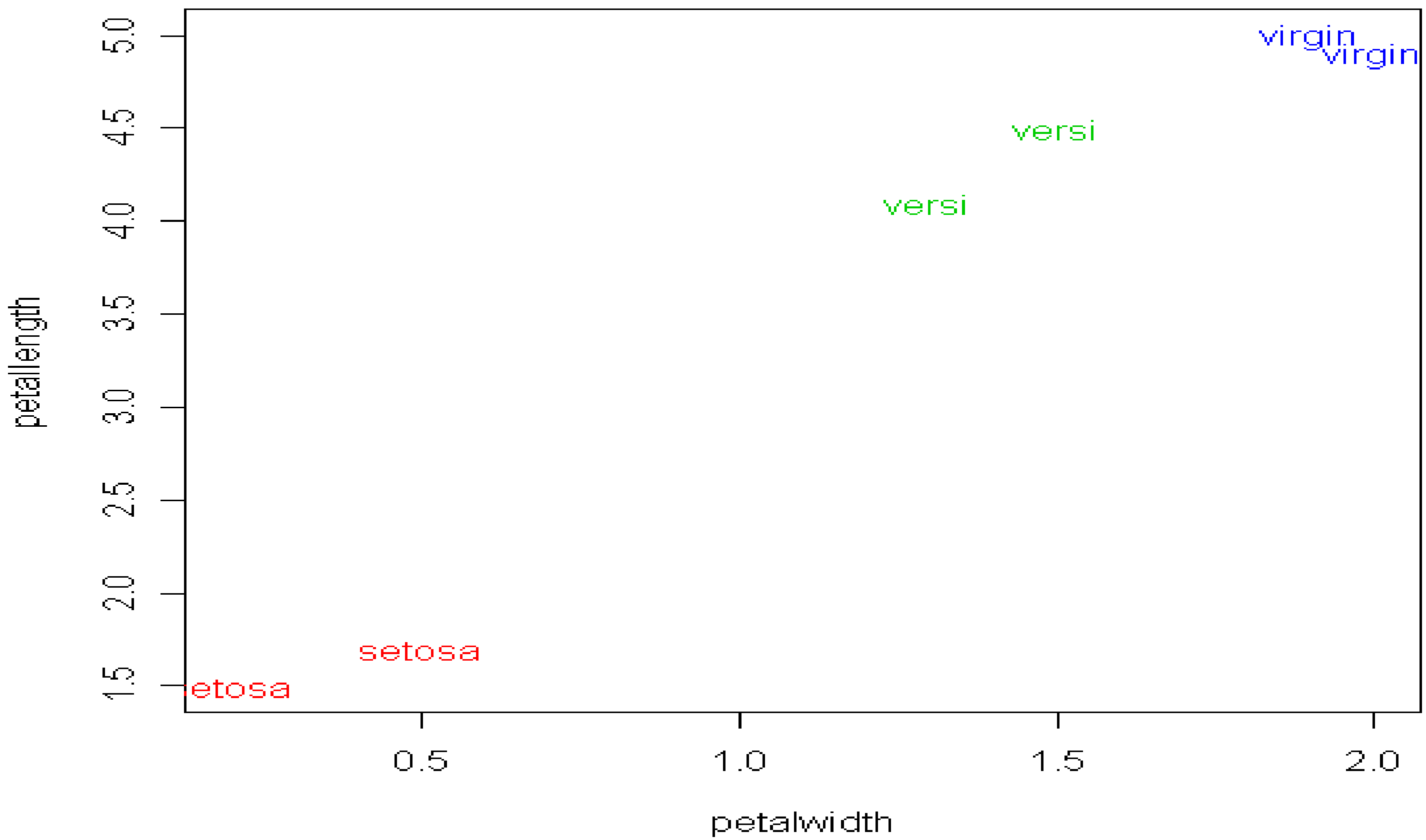
Size comparisons

Data	training rows	training cols	archetype rows	archetype cols	Scaling
echo	107	60	51	18	6.99
glaz	111	9	35	6	4.76
iris	77	4	6	3	17.11
zoobase	54	17	16	5	11.48
					10.08

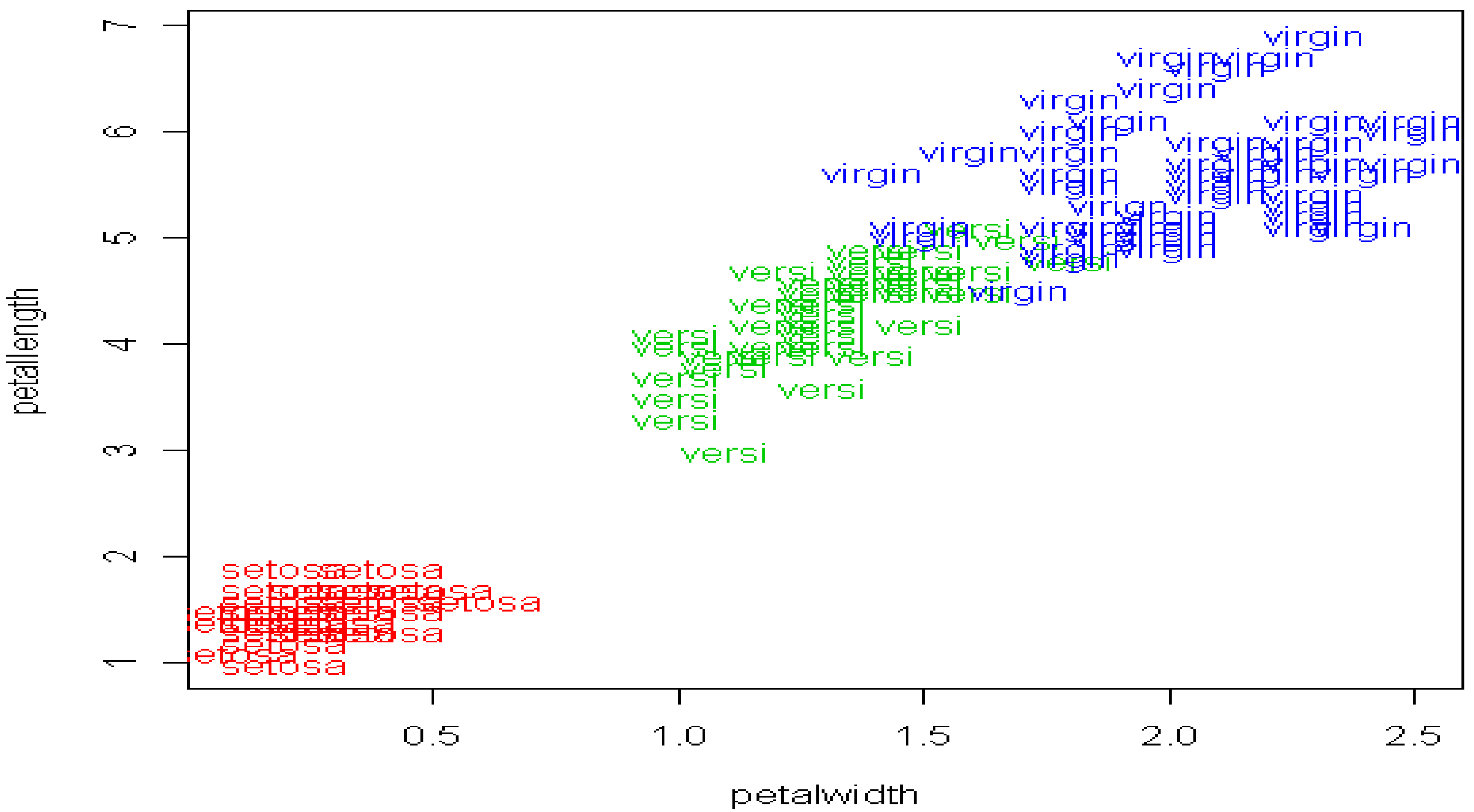
Summary

- Slight increase in accuracy
  - (65 versus 66 mistakes)
- Great reduction in size:
  - approx. 10-fold reduction in R*V product
    - i.e. raw data contains 10 times as many numbers as archetype "spreadsheet"
- Improved insight ?

A bouquet of flowers ?



A tangled thicket ?



Distinctive characteristics of evolutionary-computing traditions

- ES
  - sometimes >2 parents !
  - typically floating-point representation
  - meta-evolution of parameters (e.g. mutation rate)
- GA
  - binary representation
  - generational algorithms
- GP
  - tree-structured representation (Lisp functions)
  - executable genome
- EP
  - no crossover (?)
  - typically finite-state-machine representation

Recommended reading

Eiben, A.E. & Smith, J.D. (2003). Introduction to Evolutionary Computing. Springer-Verlag

Goldberg, D.E. (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley.

Holland, J.H. (1975). Adaptation in Natural and Artificial Systems. University of Michigan Press.

Koza, J.R. (1992). Genetic Programming. MIT Press.

Websites

- http://en.wikipedia.org/wiki/Evolutionary_computation
- http://www.cse.dmu.ac.uk/~rij/gafaq/top.htm
- http://www.genetic-programming.org/
- http://www.ra.cs.uni-tuebingen.de/software/JCell/tutorial/c
- http://bionik.tu-berlin.de/institut/
- http://www.cems.uwe.ac.uk/~jsmith/ecbook/ecbook.html