UNIVERSITY OF APPLIED SCIENCES UTRECHT

# Evolutionary Algorithms

TCTI-VKAAI-17: Applied Artificial Intelligence

Huib Aldewereld

## Leerdoelen

- Na deze les kan de student:
  - Concepten van evolutionaire algoritmen, bijv. crossover, mutation, selection, uitleggen en toepassen.
  - Een selectie van machine learning technieken toepassen en beperkingen daarvan uitleggen.

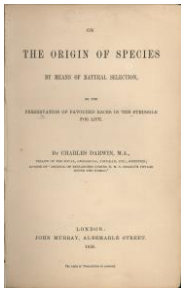
## Inhoudsopgave

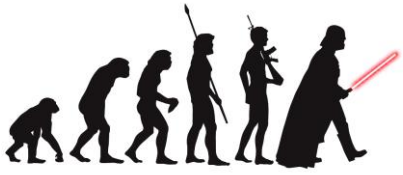


3

## Wat zijn EA's?

- Metaheuristiek van Operations Research
  - Optimalisatie
  - Gebaseerd op natuurlijke selectie (Darwin)
- Onderdeel van *Evolutionary Computing*
- Doel
  - Vinden, genereren of selecteren van een heuristiek die voldoende goed blijkt voor een gegeven probleem



4

## Wat zijn EA's?

- Operaties geïnspireerd op biologische evolutie
- Basis hypothese:
  - Eigenschappen verschillen per individu (phenotype variation)
  - Verschillende eigenschappen kunnen verschil maken in overlevings- en reproductiekansen (differential fitness)
  - Eigenschappen kunnen worden doorgegeven naar volgende generaties (heritability of fitness)
- Operaties
  - Reproduction, recombination, selection, fitness

## Evolution cycle

1. Generation of an initial population of individuals (at random);
2. Evaluation of the fitness of each individual in that population;
3. Repetition of the following re-generational steps until termination;
   a) Select the best-fit individuals for reproduction;
   b) Breed new individuals through crossover and mutation to give birth to offspring;
   c) Evaluate the individual fitness of new individuals;
   d) Replace least-fit population with new individuals.

6

## What's in a name

- Soortgelijke technieken:
  - **Genetic algorithms**: zie rest slides
  - *Genetic programming*: evolutie van (structuur) computer programma's
  - *Evolutionary programming*: evolutie van programma parameters
  - *Gene expression programming*: vergelijkbaar met GP en EP, maar maakt gebruik van genotype-phenotype verandering
  - **Evolutionary strategy**: GA, maar met alleen maar mutatie
  - **Neuro-evolutie**: GP, maar dan op een neuraal netwerk (zie volgend college).

7

## Inhoudsopgave



8

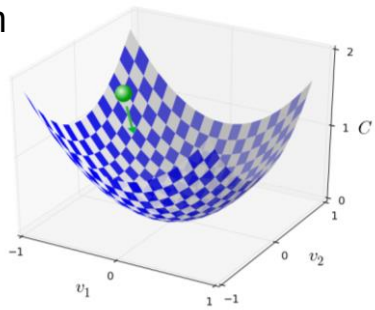
## Optimalisatie

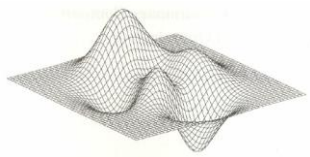
- Specifieke tak van informatica
- Zoeken naar de beste oplossing voor een gegeven probleem
  - Bijv. gate toewijzingen op vliegvelden
  - Of diverse planningsproblemen
  - Maar ook: beste parameters voor je programma
  - Of de optimale vorm van je antenne →
- Zoekruimte vaak erg groot dat doorrekenen **niet** de beste oplossing is

9

## Hill climbing

- Als gradient descend, maar dan andersom
- Op zoek naar (lokaal) maximum
  - Door stapjes te nemen met de hoogste pay-off (sterkst stijgend)
- Nadeel van deze technieken is dat ze vast kunnen komen te zitten in lokale optima/minima

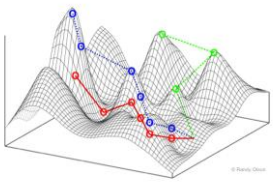


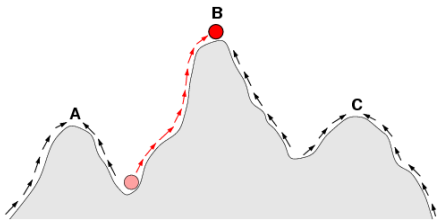



10

## Hill climbing, vervolg

- Algoritme:
  - Genereer (of selecteer) een random start punt $s_0$ initialiseer tijd $t = 0$.
  - Herhaal tot convergentie:
    1. Creëer $s_{new}$ door $s_t$ te veranderen (neem een stap);
    2. Als de fitness van $s_{new}$ groter is dan $s_t$, dan wordt $s_{t+1} = s_{new}$
    3. Anders wordt $s_{t+1} = s_t$ (we blijven op onze plek);
    4. $t = t + 1$






11

## Inhoudsopgave

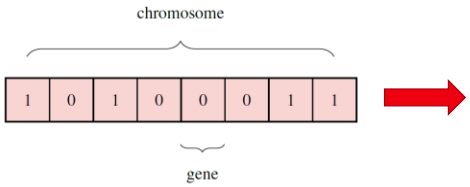
- Wat zijn EA's?
  - What's in a name
- Optimalisatie
- Genetic algorithms
  - Encoding
  - Evaluation
  - Recombination
  - Selection
- Limitations

12

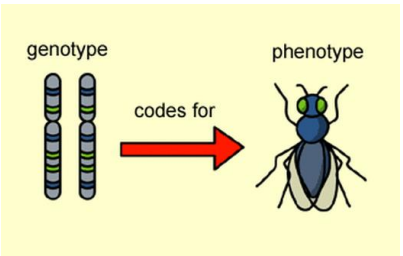
# Encoding

- Voor het oplossen van een probleem is een correcte representatie benodigd
  - Oplossingen zijn individuen of phenotype
  - Representatie individu is genotype (encodering)

- Juiste codering nodig om probleem op te lossen
  - Bijv. binaire string




- Integer
- Real number
- Schedule
- Sequence
- …




13

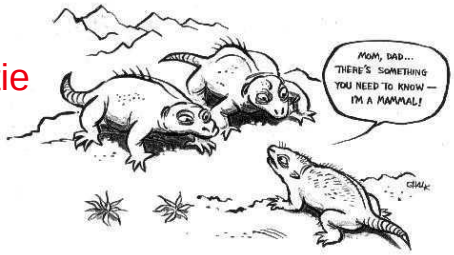
# Example: Encoding

- Doel: creëren van lijst van *N* getallen die samen optellen tot *X*

```
def individual(length, min, max):
    """
    Creates an individual for a population

    :param length: the number of values in the list
    :param min:  the minimum value in the list of values
    :param max:  the maximal value in the list of values
    :return:
    """
    return [randint(min, max) for x in range(length)]


def population(count, length, min, max):
    """
    Create a number of individuals (i.e., a population).

    :param count: the number of individuals in the population
    :param length: the number of values per individual
    :param min: the minimum value in the individual's list of values
    :param max: the maximal value in the individual's list of values
    """
    return [ individual(length, min, max) for x in range(count) ]
```

14

## Evaluation




- Iteratief proces; elke stap is een generatie
- Alleen de besten mogen door,…
  - … maar wat zijn de besten?
- Evaluatie functie (fitness function) om waarde van individuen te kunnen bepalen
- Kan gekoppeld worden aan werkelijke wereld
  - Bijv. voor bepalen fitness van een navigatie algoritme in robots
  - Laat robot met algoritme rijden, en kijk hoe vaak hij botst…

15

## Example: Evaluation

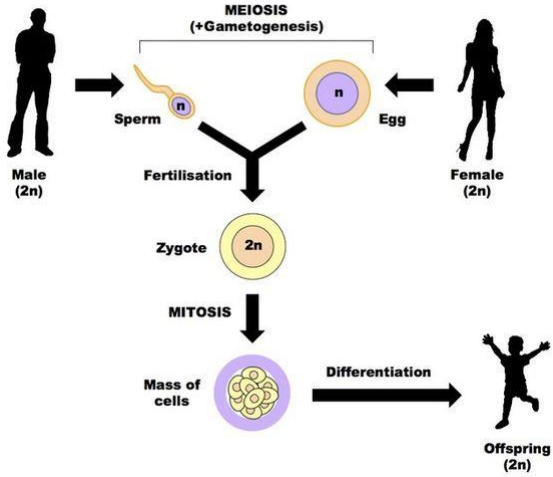
- In ons voorbeeld is de fitness eenvoudig te bepalen als de afstand tussen de som van het individu en het doel *X*

```
def fitness(individual, target):
    """
    Determine the fitness of an individual. Lower is better.

    :param individual: the individual to evaluate
    :param target: the sum of the numbers that we are aiming for (X)
    """
    sum = reduce(add, individual, 0)
    return abs(target-sum)
```

16

## Recombination

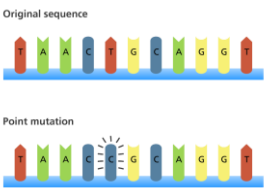
- Behoud de sterke eigenschappen van de populatie door
  - Enkel recombinatie toe te staan op de sterkste individuen
  - Hypothese: kinderen erven eigenschappen van ouders, en dus (hopelijk) de sterkste eigenschappen van vader en moeder




## Recombinatie – Mutatie

- Genetische operator die een genotype lichtjes verandert
- Te sterke verandering leidt tot drastische gevolgen (voor de performance van het GA)

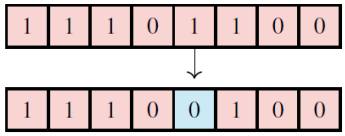
- Mutatie is belangrijk om voldoende spreiding te creëren (exploratie)
  - Helpt je om uit lokale minima te komen
- Denk erom:
  - Pas tenminste 1 mutatie operator toe
  - Mutatie niet te groot
  - Mutatie moet tot valide chromosomen leiden




18

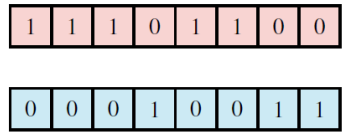
## Recombinatie – Mutatie

- Bitstrings
  - Flip:

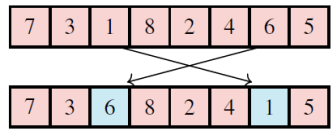



- Integers:
  - Uniform mutation: pas elke gene aan met een kleine verandering
  - Non-uniform mutation: als uniform mutation, maar minder waarschijnlijk in latere generaties

- Invert:



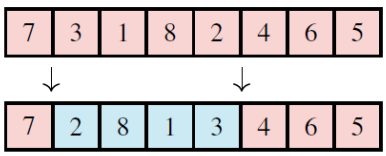

- Gaussian noise: verander waardes met een waarde die waarschijnlijk klein is
- Swap:



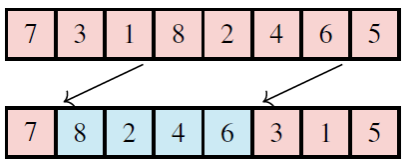

19

## Recombinatie – Mutatie

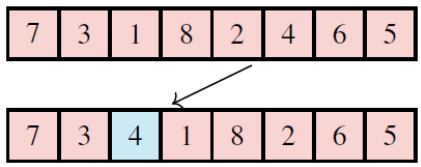
- Inversion:



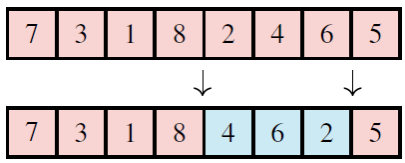

- Displacement:




- Insertion:



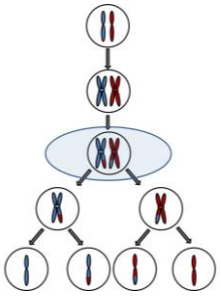

- Scramble:




20

## Recombinatie – Crossover

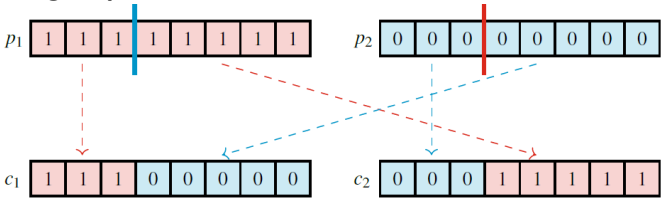
- Mutatie verandert individuen; geen echte recombinatie
- Recombinatie (papa + mama = kindje) door crossover
- Analoog aan biologische crossover
- Er kunnen meerdere kindjes tegelijk geproduceerd worden
- Bedoelt om de sterke eigenschappen van pappa te combineren met de sterke eigenschappen van mamma
  - Of dat werkelijk zo is, wordt pas in de volgende iteratie bepaald!
- Goede crossover is zich bewust van de encoding



21

## Recombinatie – Crossover

- Single-point crossover:

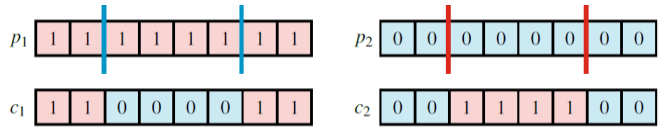



  - Kies 1 enkel punt, en verwissel de genes ervoor van ouder 1 met de genes erachter van ouder 2

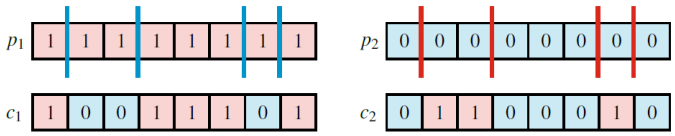
22

## Recombinatie – Crossover

- Varianten
  - Two-point crossover




  - Uniform crossover




  - En diverse positie-gerelateerd varianten (zie reader).

23

## Recombinatie - Crossover

- Hoewel mutatie altijd moet worden uitgevoerd, kan ervoor worden gekozen om geen crossover operatie uit te voeren
- Crossover benut de sterke eigenschappen om sneller te convergeren (exploitation)
- Kan geen nieuwe eigenschappen toevoegen, zoekt alleen naar de beste combinatie van eigenschappen in de populatie
- Genetische algoritmen zonder crossover heten evolutionary strategy

24

## Selection

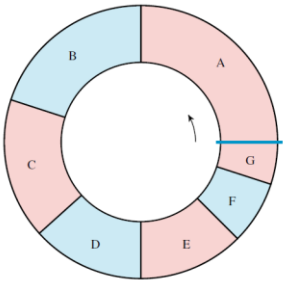
- Bepalen van de sterkste individuen als ouders
- Selectie zet druk op de performance van de individuen

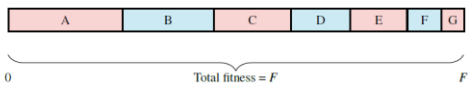
- Wegens genetische diversiteit niet verstandig om alleen de besten te laten paren
  - De besten, op dit moment, zouden immers allemaal rond een lokaal optimum kunnen liggen
  - Zorg voor voldoende spreiding zodat je lokale optima kan ontwijken
- Diverse selectie strategieën

25

## Selection – Fitness proportional

- Fitness proportional selection of roulette-wheel selection
- Kans dat individu ouder mag zijn proportioneel van zijn fitness (t.o.v. de populatie)




26

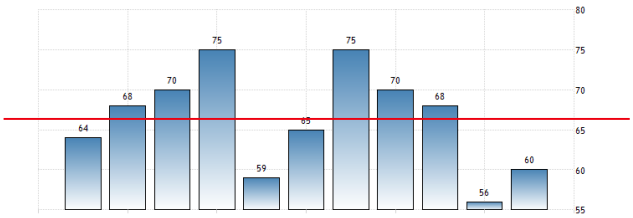
## Selection – Fitness proportional

- Nadelen fitness proportional selection
  - Kleine kans dat beste individuen niet geselecteerd worden
  - Gevaar op pre-mature convergentie; sterke individuen overheersen het roulette wiel
  - Als convergentie te vroeg inzet, gaat de selection pressure omlaag (ongeveer even goed zijn betekent dat je een redelijke kans hebt om te mogen reproduceren)

27

## Selection – Truncate

- Truncate selection of rank-based selection
  - Selecteert top X individuen
  - Risico op convergence als laag presterende individuen niet een kleine kans hebben om te overleven (door naar de volgende generatie)
    - Verkleint te snel de genetische diversiteit




28

## Selection – Tournament

- Tournament selection
  - Speel toernooien met random geselecteerde individuen uit de populatie
  - Winnaar wordt geselecteerd voor reproductie
- Beste genetische diversiteit
  - Laag scorende individuen hebben nog steeds een kans (als ze in een toernooi uit moeten komen tegen andere laag scorende individuen)
- Kan selectie druk opvoeren door toernooien te spelen met meer individuen

29

## Replacement strategieën

- Zijn ouders onderdeel van de volgende generatie?
  - Komt in biologie niet voor (hoewel generaties anders mengen)
  - Individuen in generatie worden geëlimineerd/vervangen door nieuwe individuen (offspring)
- Kan nuttig zijn om de beste individuen niet kwijt te raken
  - Elitist strategy
  - Beste individuen komen sowieso ook in de volgende generatie (ongeacht reproductie en selectie)
  - Gebruik niet te veel elitists, want dan presteert het GA minder optimaal

30

## Example, vervolg

- Selection
  - Elitist strategy
  - Rank-based selection
  - Met random re-insertion van laag scorende individuen

```
graded = [ (fitness(x, target), x) for x in population ]
graded = [ x[1] for x in sorted(graded) ]
retain_length = int(len(graded)*retain)
parents = graded[:retain_length]

# randomly add other individuals to promote genetic diversity
for individual in graded[retain_length:]:
    if random_select > random():
        parents.append(individual)
```

31

## Example, vervolg

- Simpele one-point crossover (halverwege)
- Alleen op geselecteerd parents

```
# crossover parents to create offspring
desired_length = len(population) - len(parents)
children = []
while len(children) < desired_length:
    male = randint(0, len(parents)-1)
    female = randint(0, len(parents)-1)
    if male != female:
        male = parents[male]
        female = parents[female]
        half = int(len(male) / 2)
        child = male[:half] + female[half:]
        children.append(child)
```

32

## Example, vervolg

- Mutatie op toegevoegde kinderen

```
# mutate some individuals
for individual in children:
    if mutate > random():
        pos_to_mutate = randint(0, len(individual)-1)
        # this mutation is not ideal, because it
        # restricts the range of possible values,
        # but the function is unaware of the min/max
        # values used to create the individuals
        individual[pos_to_mutate] = \
            randint(min(individual), max(individual))

parents.extend(children)
```

33

## Inhoudsopgave

- Wat zijn EA's?
  - What's in a name
- Optimalisatie
- Genetic algorithms
  - Encoding
  - Evaluation
  - Recombination
  - Selection
- Limitations

34

## Limitations

- GA's werken goed (beter dan traditionele oplossingen), mits:
  - Zoekruimte groot
  - Zoekruimte bekend en niet unimodaal/glad (vertoont meerdere pieken)
  - Zoekruimte is niet geheel begrepen
  - Fitness functie niet exact (noisy)
  - 'Bijna goed' is goed genoeg, snelheid is belangrijker

35