COMPUTATIONAL INTELLIGENCE

September 2016 – November 2016

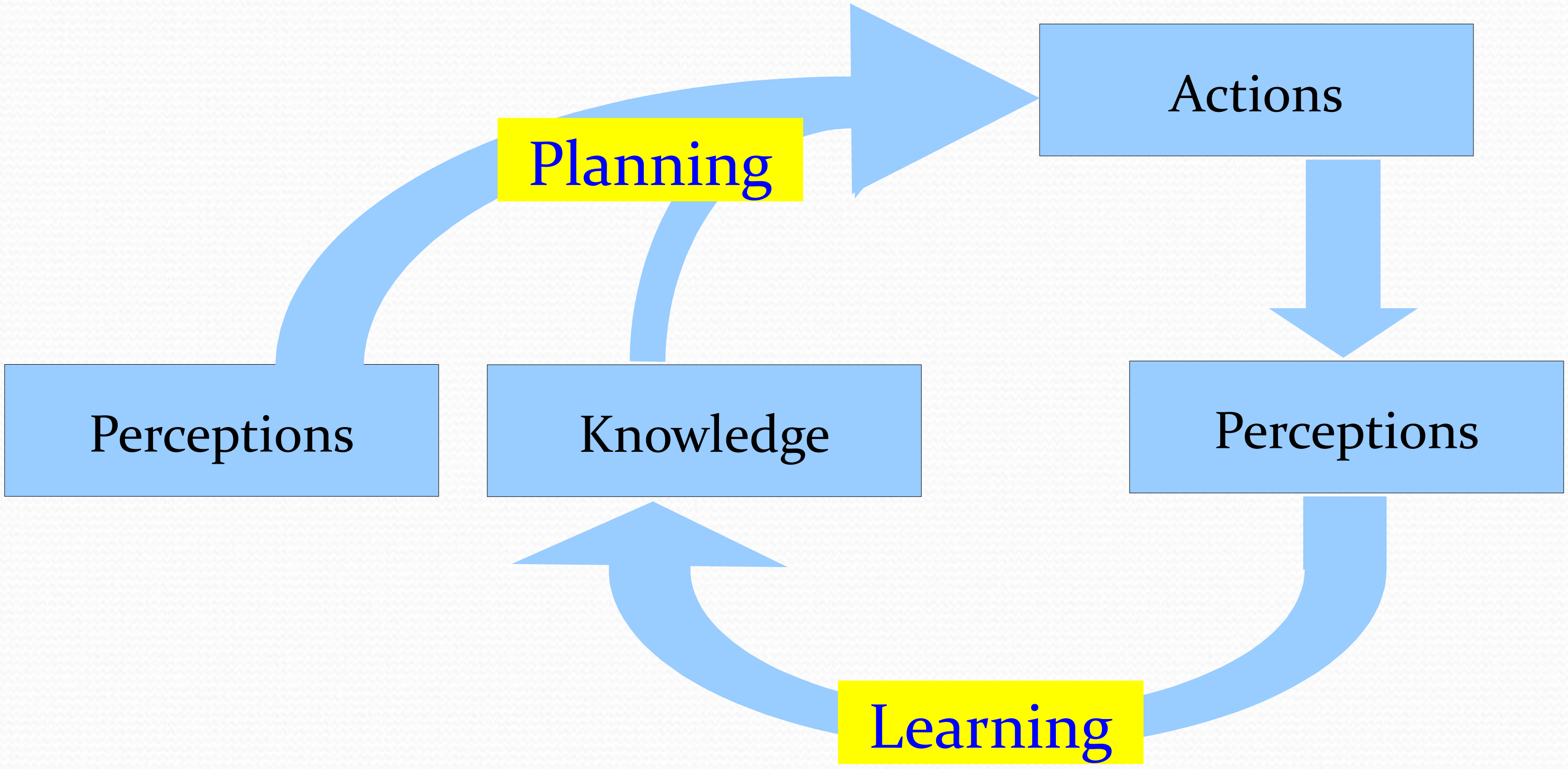
Siegfried Nijssen

Leiden Institute of Advanced Computer Science
e-mail: s.nijssen@liacs.leidenuniv.nl

Artificial Intelligence

- Aims to develop intelligent agents that perceive their environment and take actions that maximize their chances of success
- Requires solving several challenges:
  - Knowledge representation:
    how does an agent represent its knowledge and perceptions?
  - Reasoning, planning:
    how does an agent deduce an action based on its perceptions and its knowledge?
  - Learning:
    how does an agent update its knowledge based on its perceptions?

Artificial Intelligence



Computational Intelligence

- Computational intelligence traditionally studies a subset of three AI techniques:
  - Knowledge representation:
    fuzzy logic & fuzzy set theory
  - Reasoning, planning:
    Evolutionary (genetic) algorithms
  - Learning:
    Neural networks

Knowledge representation: Fuzzy logic

- **Goal:**
  represent "fuzzy" knowledge of an agent

- Traditional logic can be used to represent crisp rules:

  if *A is true* then do *B*

  Boolean in → Boolean out

- Fuzzy logic represents fuzzy rules:

  if *A is true to a high degree* / *A is likely* then try to make *B true to a high degree* / make *B likely*

  Number in → Number out

Fuzzy logic is less sensitive to errors / noise

Knowledge representation: Fuzzy logic

- Used to build control systems

if *A is warm to a high degree* then *B should be turned down to a high degree*

- Used to calculate the overall quality (fitness) of a (hypothetical) situation

if *A is high* then *customer* is *likely good*
if *B is high* then *customer* is *likely good*
if *C is high* and *B is not high* then customer is *likely good*

how good would the situation be in which *A* and *C* are high, and *B* is low?

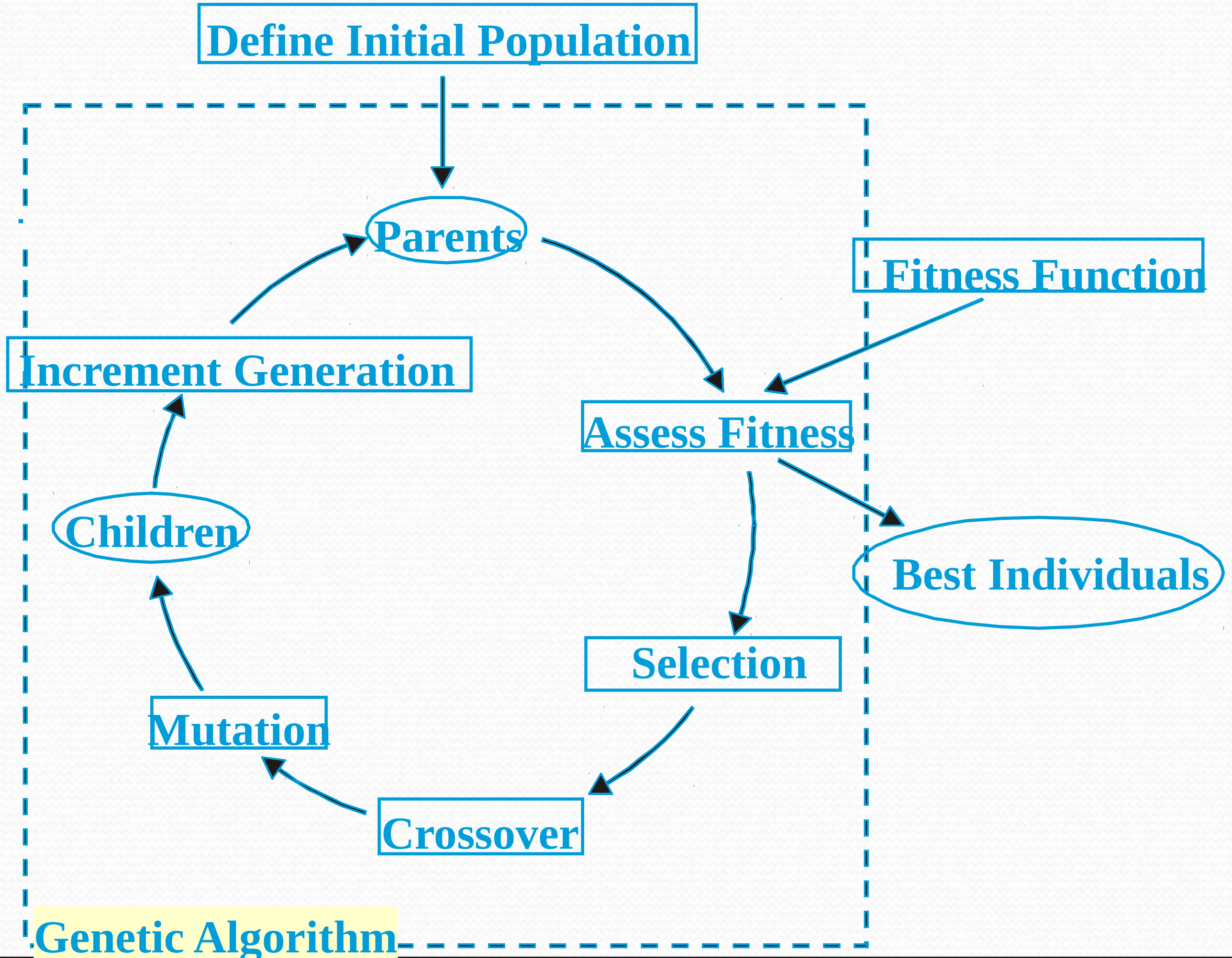
- ***Research challenges: how to interpret fuzzy rules? What are sensible strategies for calculating an output, given inputs? How to make the intuition formal?***

Planning / optimization: Evolutionary Algorithms

- **Goal of an evolutionary algorithm:**
  to find a plan that optimizes a given fitness function
  - the fitness could be defined by means of fuzzy logic, but does not have to be
- **Example:**
  the traveling salesman problem
  - **Given** a number of cities, distances between the cities
  - **Find** an order in which to visit the cities such that the total distance traveled is minimized

Evolutionary Algorithms

- **Method:** evolve populations of solutions by mimicking evolution in nature



Nature-inspired optimization

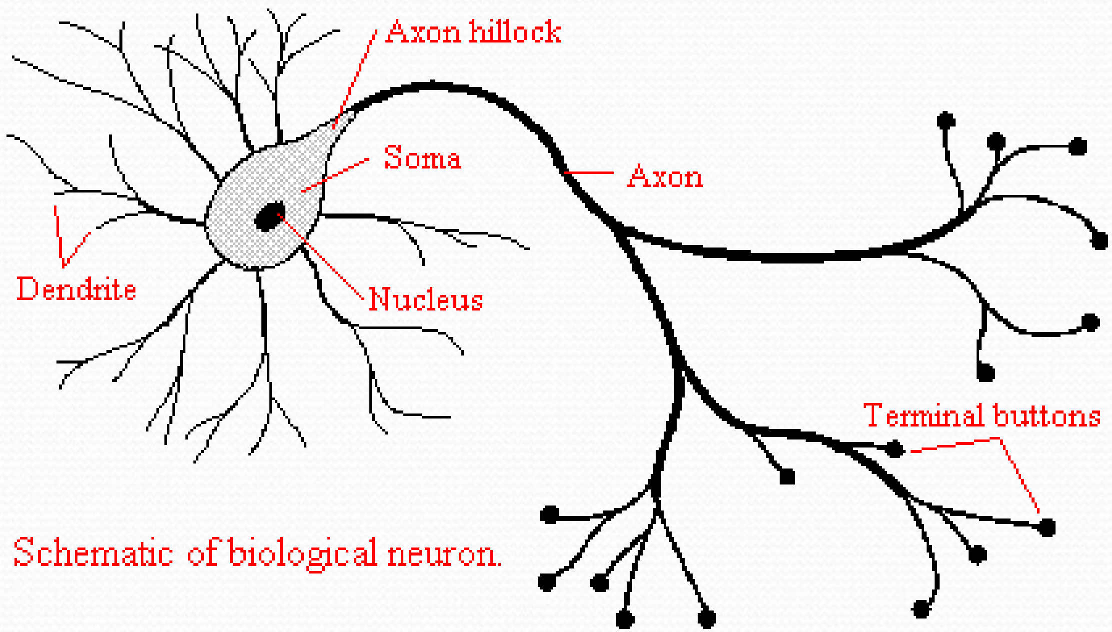
- Evolutionary algorithms
- Particle swarm optimization
- Artificial ants

All are
robust optimization algorithms:
if the fitness function changes, solutions usually adapt relatively easily

- **Research challenge: which algorithm finds a good solution as quickly as possible?**

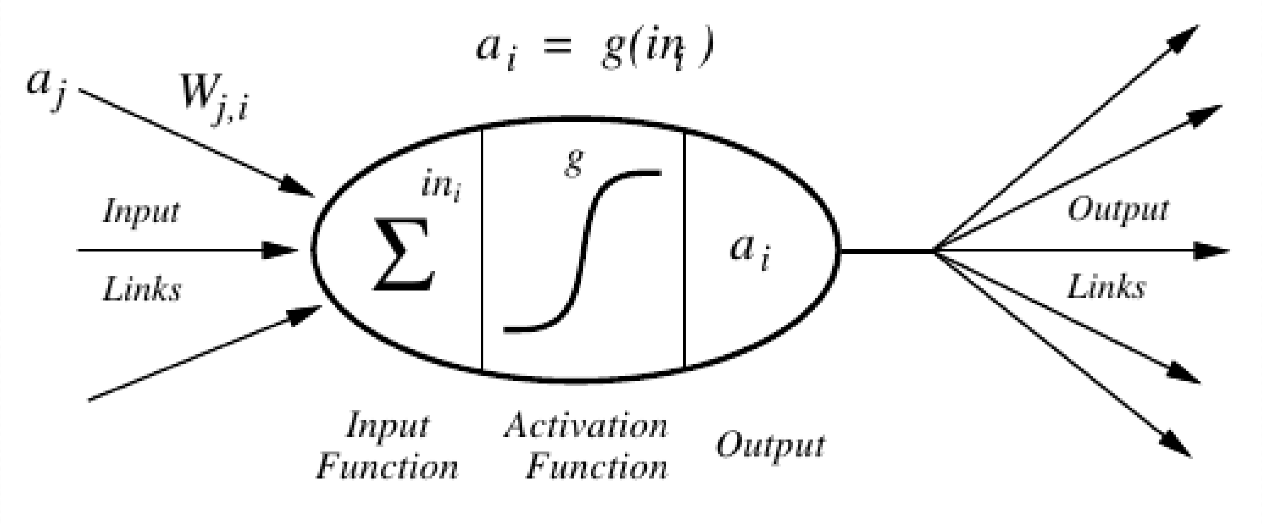
Learning: Neural Networks

- Inspired by biological nervous systems



Learning: Neural Networks

- Artificial neuron



(Neuron/Unit)

- Also a neural network represents knowledge, and is often used used to transform input to output

Learning: Neural Networks

- Different types of neural networks:
  - feed-forward neural networks
  - self-organizing maps
  - recurrent networks
  - radial basis function networks
  - fuzzy-neural networks

Research challenge: how to learn a neural network?
What is a good architecture for a neural network?

Computational Intelligence

- Knowledge representation:
  fuzzy logic & fuzzy set theory → You haven't followed a basic course on logic

- Reasoning, planning:
  Evolutionary (genetic) algorithms → Basis already discussed in course artificial intelligence

- Learning:
  Neural networks → Basis already discussed in course artificial intelligence
  - Also in course on data mining
  - Advanced topics require strong mathematics

Computational Intelligence

- Knowledge representation & planning:
  traditional logic, SAT solvers, constraint programming
- Knowledge representation:
  fuzzy logic & fuzzy set theory
- Reasoning, planning:
  Evolutionary (genetic) algorithms
- ~~Learning:~~
  ~~Neural networks~~

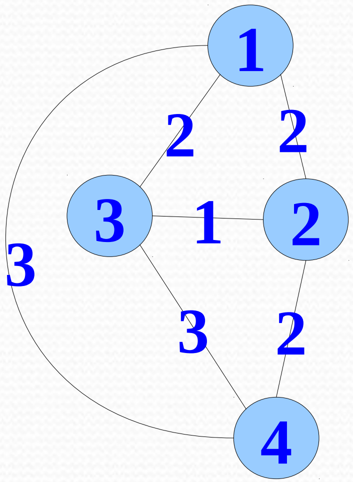
Central Theme

- Artificial intelligence inspired methods for
  - Knowledge representation:
    - Logic
    - Fuzzy logic
  - Optimization & planning:
    - SAT solving
    - Constraint programming
    - Local search
    - Evolutionary algorithms
- Problems related to *operations research*

Template of a Constraint Optimization Problem

- **Given:**
  - …
- **Find:**
  - …
- **Such that:**
  - … is **minimal/maximal**
  - … is satisfied

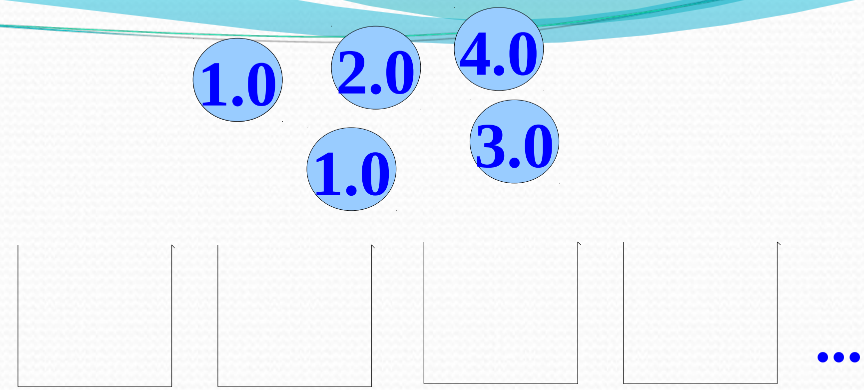
Example 1: Traveling Salesmen



- **Given:**
  - *N* cities
  - *D[i,j]* distances between cities
- **Find:**
  - an assignment *p[i]* for *i=1..N* with *p[i]* in *1..N*, indicating that at step *i* city *p[i]* is visited
- **Such that:**
  - all cities are visited exactly once
  - *D[p[1],p[2]]+D[p[2],p[3]]+…+D[p[n-1],p[n]]+D[p[n],p[1]]* is **minimal**

Optimization

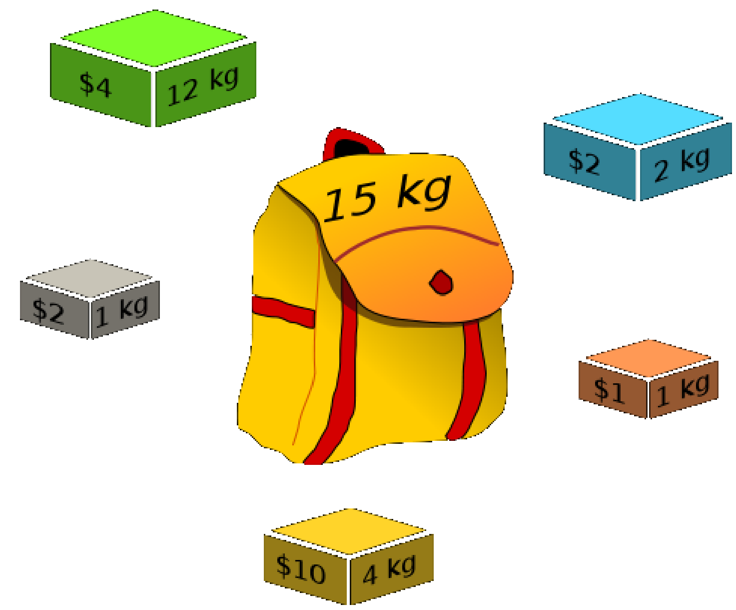
Example 2: Binpacking



Each bin: 4.0

- **Given:**
  - N items with sizes $a_1, \ldots, a_N$
  - A bin size V
- **Find:**
  - an assignment $p[i]$ for $i=1..N$ to positive integers, indicating that item i is put in bin $p[i]$
- **Such that:**
  - $\max_i\ p[i]$ is **minimal** (number of bins is small)
  - $\sum_{p[i]=j} a_i \leq V$ for all bins j (no more than weight V in each bin)

Example 3: Knapsack



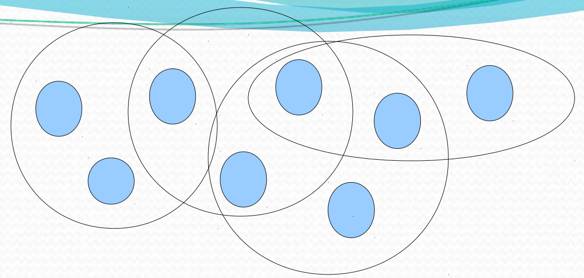
- **Given:**
  - N items with sizes $a_1,\ldots,a_N$, prices $p_1,\ldots,p_N$
  - A maximum weight W
- **Find:**
  - a subset of items I
- **Such that:**
  - $\sum_{i \in I} p_i$ is **maximal** (very valuable knapsack)
  - $\sum_{i \in I} a_i \leq W$ (knapsack with low weight)

Example 3b: Unbounded Knapsack

- **Given:**
  - N possible items with weights $a_1,\ldots,a_N$, prices $p_1,\ldots,p_N$
  - A weight threshold W
- **Find:**
  - an integer *w[i]* for each item *i*
- **Such that:**
  - $\sum_{i=1}^{N} w[i]p_i$ is **maximal** (very valuable knapsack)
  - $\sum_{i=1}^{N} w[i]a_i \le W$ (knapsack with low weight)

Portfolio Optimization

Example 4: Set Cover



- **Given:**
  - N sets, each a subset of the universe $U=\{1,2,\ldots,m\}$
- **Find:**
  - A subset $\boldsymbol{S}$ of the N given sets, i.e. each set in $\boldsymbol{S}$ equals one of the given sets, but not all given sets need to be selected.
- **Such that:**
  - $|\boldsymbol{S}|$ is **minimal** (small subset)
  - $\bigcup_{S \in \mathbf{S}} S = U$ (each element is covered)

Decision vs Optimization Problems

- Optimization problem:

  Find …
  Such that:
  - *f(…)* is minimal
  - constraints are satisfied
- Decision problem:

  Find …
  Such that:
  - *f(…)* < *threshold*
  - constraints are satisfied
- Optimization problems over finite domains can be turned into repeated decision problems: iterate over possible thresholds

How to solve these problems?

- Many such problems are hard
  - "NP hard" $\rightarrow$ no polynomial algorithm is known
- Two solutions:
  - Exact: require exponential time in the worst case
  - Inexact: polynomial, but may not find the best solutions
- Both types of solutions have been studied in artificial intelligence, algorithms, and operations research

High-Level, Declarative Problem Solving in AI

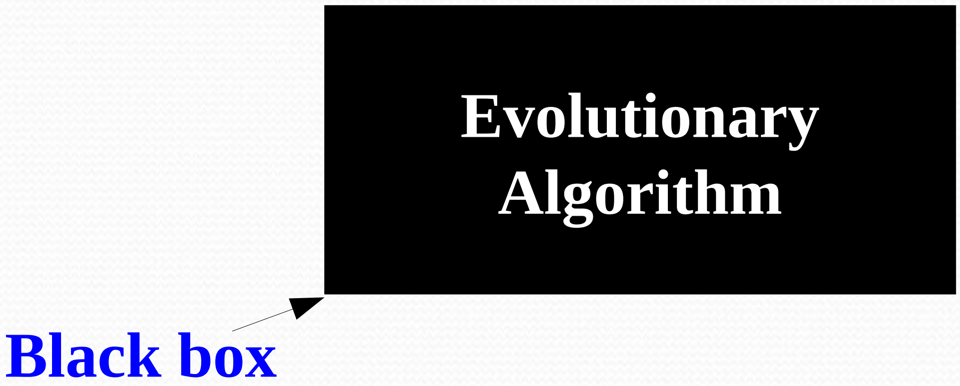
- Distinguishing feature of AI approaches: they aim to be “intelligent” and generic by solving problems (semi-)automatically

- Idea: solve a problem in two stages:
  - 1. Describe the problem in a concise way in a computer language.
  - 2. Run a general algorithm (a “solver” or an “inference engine”) on this description to solve the problem.

i.e., the programmer does **not** write an imperative algorithm.

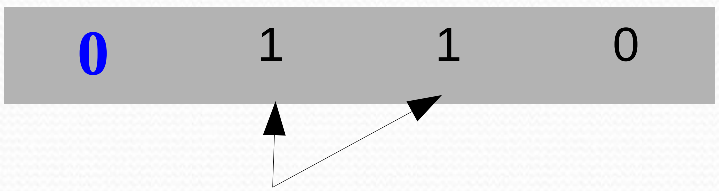
High-Level Declarative Problem Solving in AI

- Example search: evolutionary algorithm
  - Step 1:
    - Specify what the individuals in a population look like
    - Specify the quality of an individual (fitness)
  - Step 2: (Ideal situation)
    - Run an existing evolutionary algorithm without modification



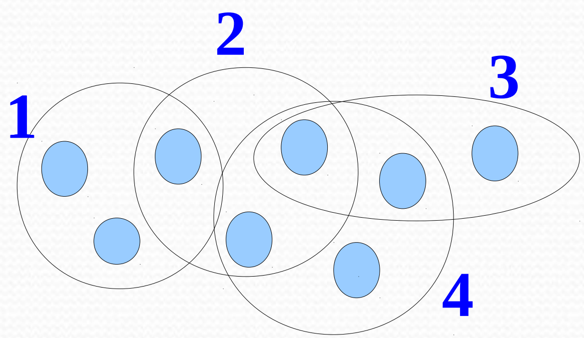
High-Level Declarative Problem Solving in AI

- Example problem: set cover
- Representation of an individual in a bitstring:



2nd and 3rd set are selected

- Fitness: (assuming small=very fit)
  - Number of sets selected?
  - Number of sets selected + (number of uncovered elements) x *w*



Very large weight

High-Level Declarative Problem Solving in AI

- What about optimal solutions?
- Alternative *general* systems that take a *declarative* input specification and find optimal solutions:
  - Constraint programming
  - SAT solvers
  - ILP solver

High-Level Declarative Problem Solving in AI

- Which programming language to use?
  - C++ ?
  - Java ?
  - Prolog ?
  - Python

Why Python?

- Scripting language with a high level of abstraction
  - Implements features also seen in functional and logic programming
- Well-supported language with many libraries available
- Quickly gaining popularity in the scientific community (Coursera)

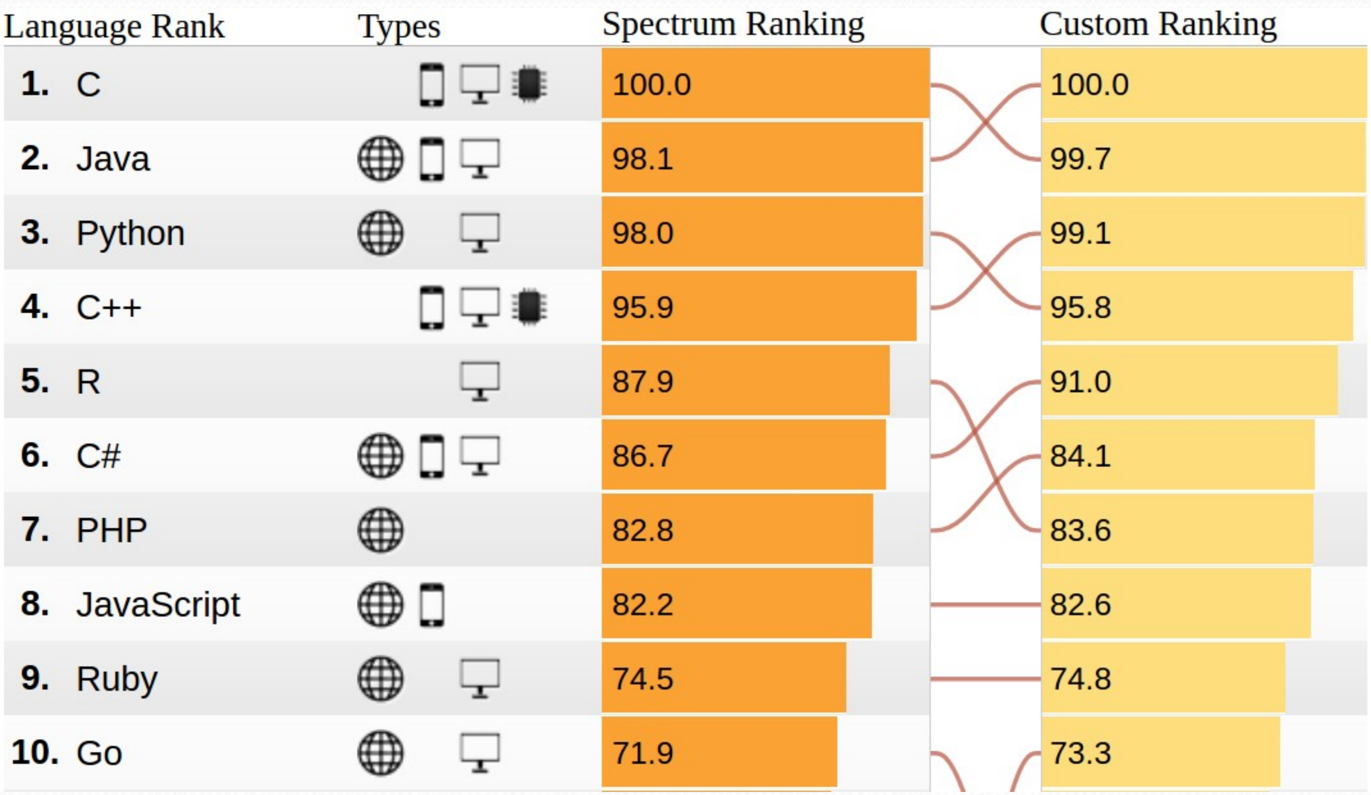
Why Python?

Language Rank	Types	2015 Spectrum Ranking	2014 Spectrum Ranking
1. Java		100.0	100.0
2. C		99.9	99.3
3. C++		99.4	95.5
4. Python		96.5	93.5
5. C#		91.3	92.4
6. R		84.8	84.8
7. PHP		84.5	84.5
8. JavaScript		83.0	78.9
9. Ruby		76.2	74.3
10. Matlab		72.4	72.8

(IEEE Spectrum)

Why Python?

2016 2015



Computational Intelligence

- Basic course in Python
- Knowledge representation & planning:
  traditional logic, SAT solvers, constraint programming
- Knowledge representation:
  fuzzy logic & fuzzy set theory
- Reasoning, planning:
  Evolutionary (genetic) algorithms
- Learning:
  Neural networks

Course overview

- Lectures & practicums will often be combined

@Snellius

Informatica & Economie Derde jaar najaar 2016-2017, BIS

wk	Datum	Maandag								Dinsdag								Woensdag								Donderdag								Vrijdag							
nr	Ma	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
36	5 sep			HCI*										SBIS								CI		pCI				HCI		CI		pCI									
37	12 sep					DaMi								SBIS								CI		pCI				HCI										CI		pCI	
38	19 sep			HCI		DaMi								SBIS														HCI										CI		pCI	
39	26 sep			HCI		DaMi								SBIS								CI		pCI				HCI										CI		pCI	
40	3 okt	Leidens Ontzet												SBIS								CI		pCI				HCI													
41	10 okt			HCI		DaMi								SBIS								CI		pCI				HCI													
42	17 okt			HCI		DaMi								SBIS								CI		pCI				HCI													
43	24 okt			HCI		DaMi						Fin								Fin								HCI						Fin pr		Fin					
44	31 okt			HCI		DaMi			t Fin			Fin								Fin			CI/pr CI					HCI						Fin pr		Fin					
45	7 nov			HCI		DaMi			t Fin			Fin								Fin			CI/pr CI					HCI						Fin pr		Fin					
46	14 nov			HCI		DaMi			t Fin			Fin								Fin			CI/pr CI					HCI						Fin pr		Fin					
47	21 nov			HCI		DaMi			t Fin			Fin			Bach					Fin			CI/pr CI					HCI						Fin pr		Fin					
48	28 nov			HCI		DaMi			t Fin			Fin								Fin								HCI						Fin pr		Fin					
49	5 dec			HCI		DaMi			t Fin			Fin			Bach					Fin								HCI						Fin pr		Fin					
50	12 dec					DaMi								T Fin																T CI											
51	19 dec																																								
52	26 dec	Gesloten								Gesloten								Gesloten								Gesloten								Gesloten							
1	2 jan					T HCI																																			
2	9 jan					T DaMi																																			
3	16 jan														Bach																										
4	23 jan										H CI																														
4	25 jan														Bach																										

- Final mark obtained 70% from a written exam and 30% from practicum assignments
- Http://www.liacs.leidenuniv.nl/~nijssensgr/CI/