1: Introduction to Evolutionary Computing

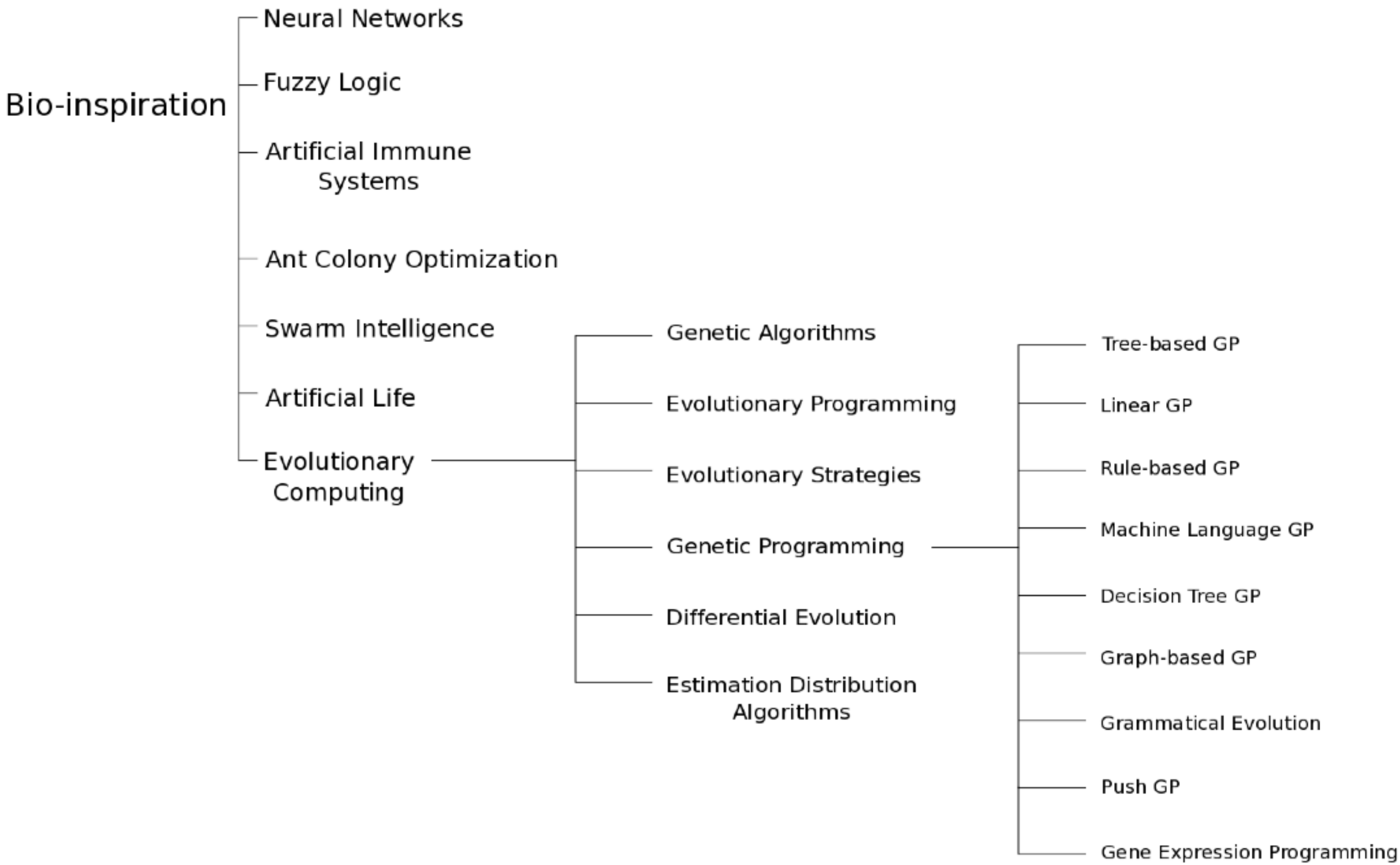
- Positioning of EC
- EC motivation and inspiration
- History of the research field
- Problems categorization
- Textbook Chapter 1.1, 1.2

Prof. Ting Hu, School of Computing, Queen's University

Positioning of EC

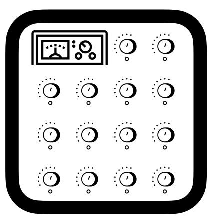
- EC is part of computer science - computational intelligence - bio-inspired computing
- EC is not part of life sciences/biology
- Biology delivered inspiration and terminology
- Trial-and-error (generate-and-test) problem solving
- EC can be applied in biological research

Bio-inspired computing

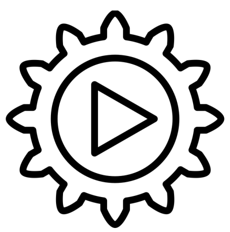


Motivation and Inspiration for EC

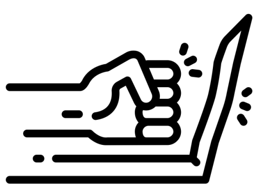
Motivation



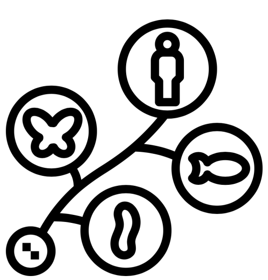
- Solve complex problems (NP-hard, multimodal, noisy, dynamic, multi-objective)



- Develop automatic problem solvers

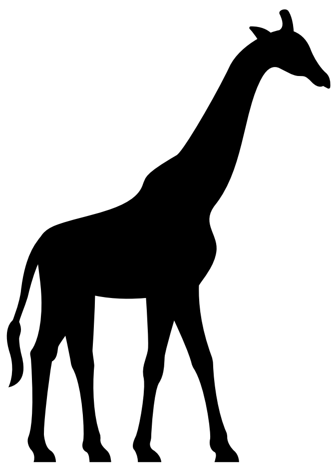


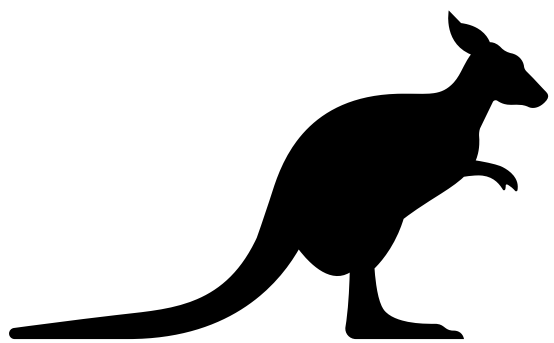
- Need for robust algorithms that are applicable to a wide range of similar problems, without much tailoring for specific problem settings

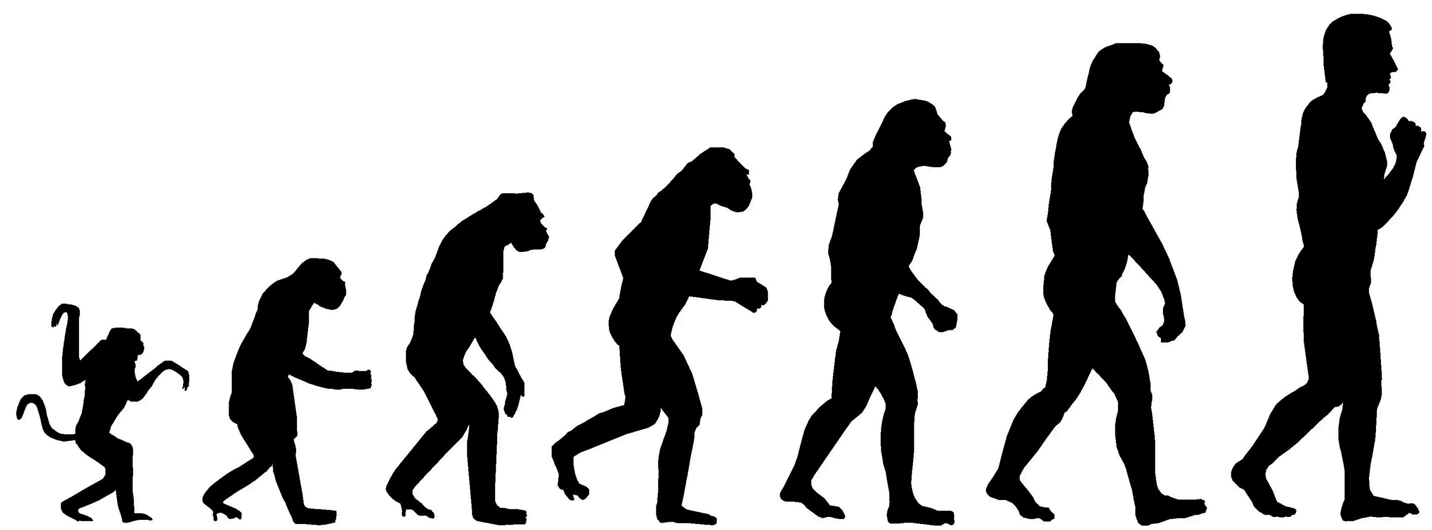


- Simulate natural evolution

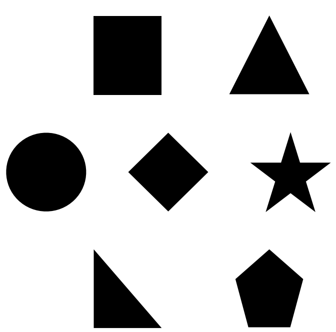
Evolution as a theory







Ingredients of evolution

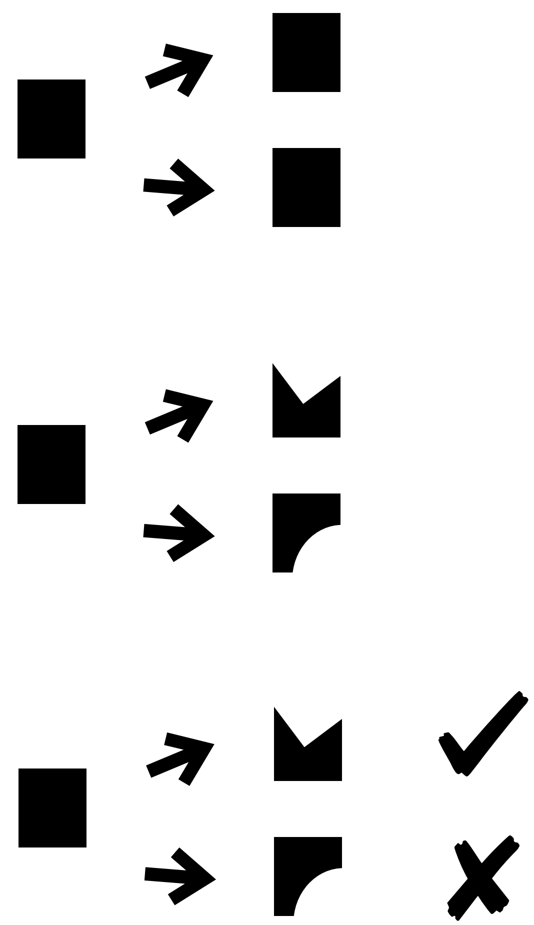


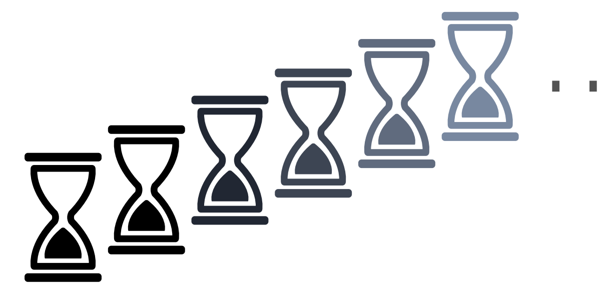
Population

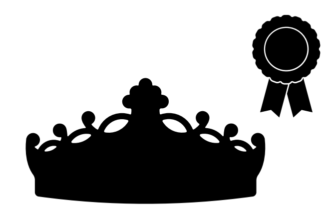
Inheritance

Variation

Selection







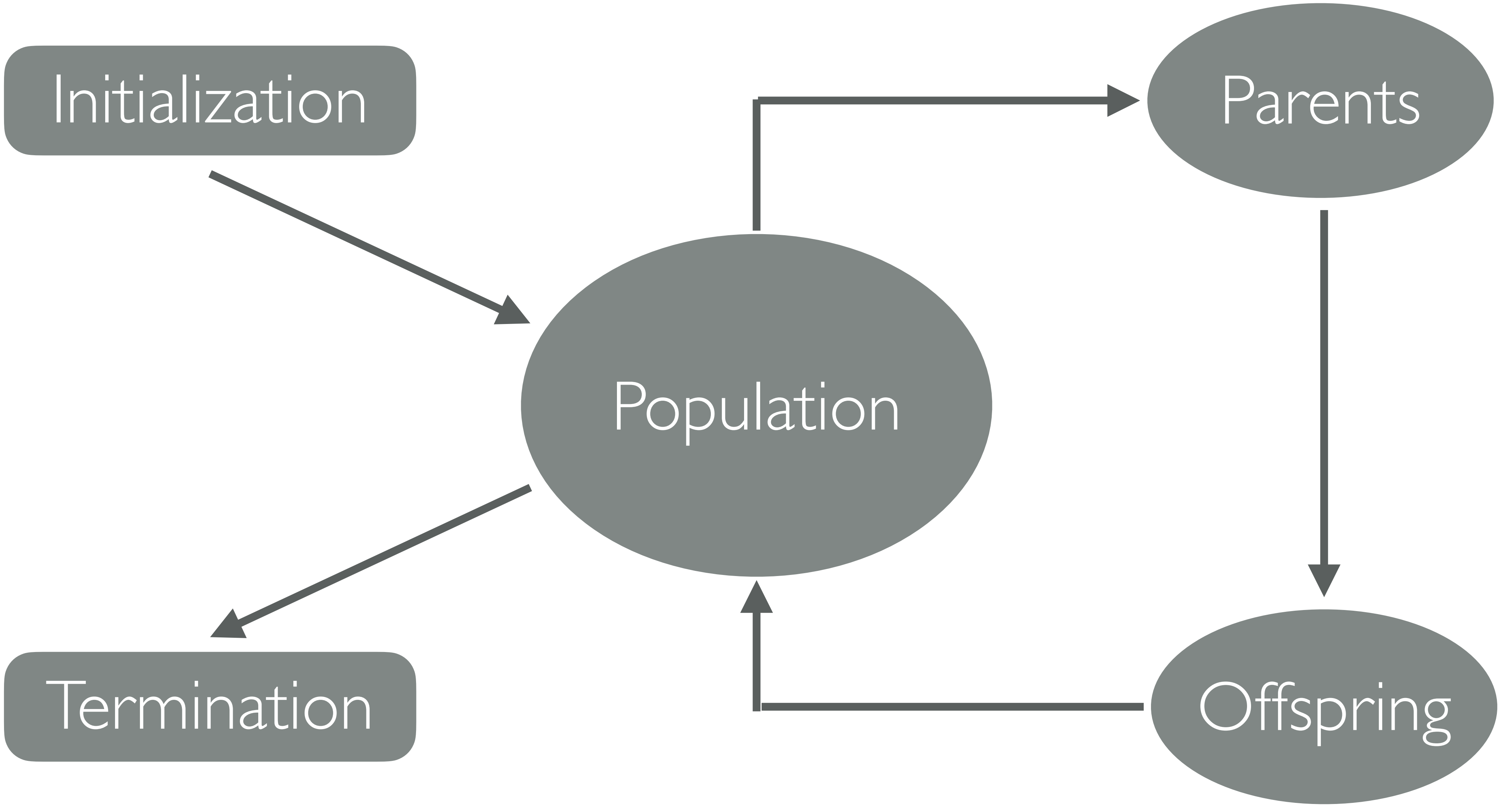
Engineer evolution

EVOLUTION		PROBLEM SOLVING
Environment	⟷	Problem
Individual	⟷	Candidate Solution
Fitness	⟷	Quality

Fitness → chances for survival and reproduction

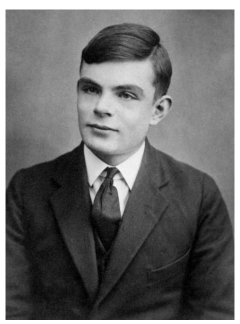
Quality → chance for seeding new solutions

Evolutionary algorithm



History

1948 – Evolutionary search



Alan Turing

1962 – Optimization through evolution

Hans-Joachim Bremermann

1966 – Evolutionary programming

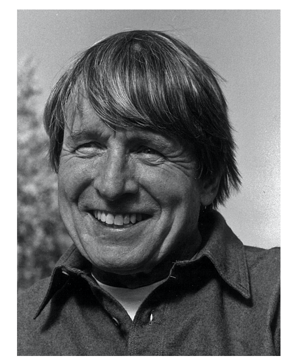
Lawrence Fogel

1972 – Evolution strategies

Ingo Rechenberg

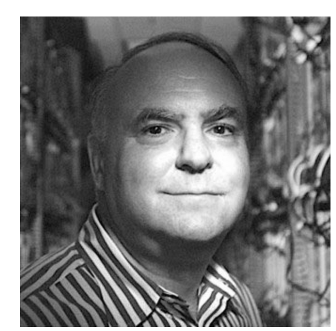
Hans-Paul Schwefel

1975 – Genetic algorithm



John Holland

1988 – Genetic programming



John Koza

1995-2000 – **Evolutionary computing**

Evolutionary AI

- Creative
- Adaptive
- Diversity

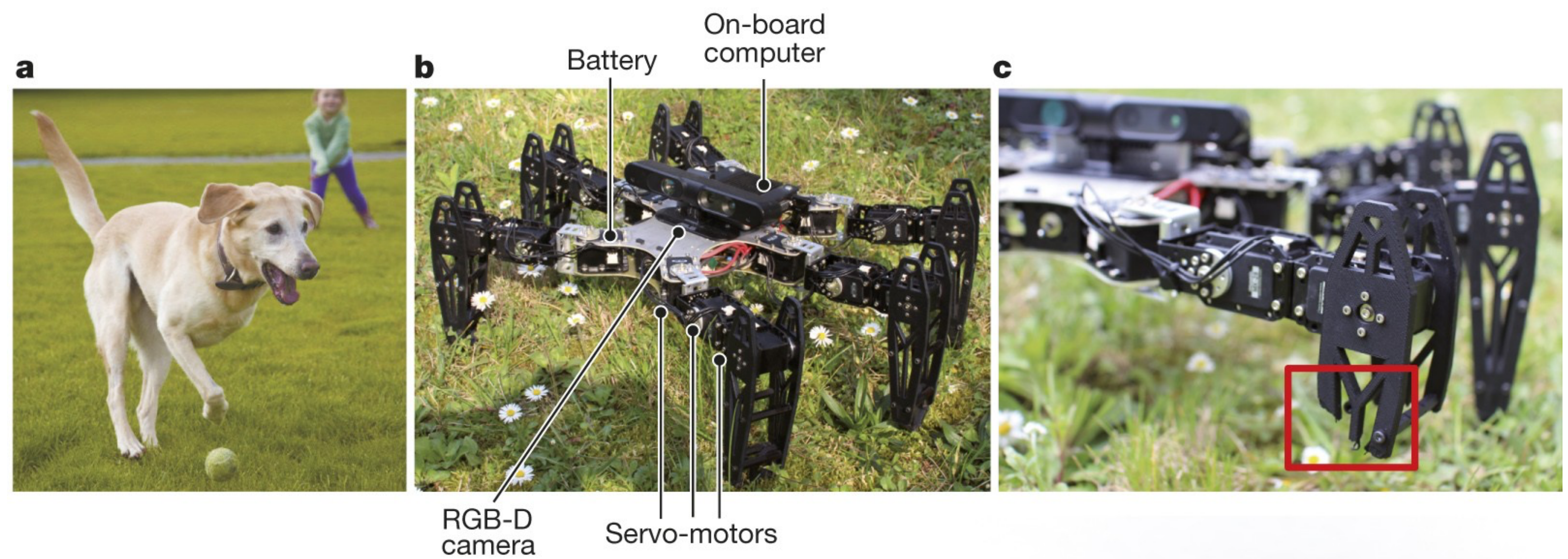
Automated Antenna Design with Evolutionary Algorithms

Gregory S. Hornby* and Al Globus
University of California Santa Cruz, Mailtop 269-3, NASA Ames Research Center, Moffett Field, CA
Derek S. Linden
JEM Engineering, 8683 Cherry Lane, Laurel, Maryland 20707
Jason D. Lohn
NASA Ames Research Center, Mail Stop 269-1, Moffett Field, CA 94035

Whereas the current practice of designing antennas by hand is severely limited because it is both time and labor intensive and requires a significant amount of domain knowledge, evolutionary algorithms can be used to search the design space and automatically find novel antenna designs that are more effective than would otherwise be developed. Here we present automated antenna design and optimization methods based on evolutionary algorithms. We have evolved efficient antennas for a variety of aerospace applications and here we describe one proof-of-concept study and one project that produced fight antennas that flew on NASA's Space Technology 5 (ST5) mission.

Evolutionary AI

- Creative
- Adaptive
- Diversity



LETTER

doi:10.1038/nature14422

Robots that can adapt like animals

Antoine Cully[1,2], Jeff Clune[3], Danesh Tarapore[1,2]† & Jean-Baptiste Mouret[1,2,4,5,6]†

Robots have transformed many industries, most notably manufacturing[1], and have the power to deliver tremendous benefits to society, such as in search and rescue[2], disaster response[3], health care[4] and transportation[5]. They are also invaluable tools for scientific exploration in environments inaccessible to humans, from distant planets[6] to deep oceans[7]. A major obstacle to their widespread adoption in more complex environments outside factories is their fragility[6,8]. Whereas animals can quickly adapt to injuries, current robots cannot 'think outside the box' to find a compensatory behaviour when they are damaged: they are limited to their pre-adapted as creatively and quickly as animals do (for example, in less than 2 min) in larger search spaces and without expensive, self-diagnosing sensors.

Here we show that rapid adaptation can be achieved by guiding an intelligent trial-and-error learning algorithm with an automatically generated, pre-computed behaviour–performance map that predicts the performance of thousands of different behaviours (Supplementary Video 1). Current learning algorithms either start with no knowledge of the search space[12] or with minimal knowledge from a few human demonstrations[12,18]. Our hypothesis is that animals understand the

Evolutionary AI

- Creative
- Adaptive
- Diversity

Creative AI Through Evolutionary Computation: Principles and Examples

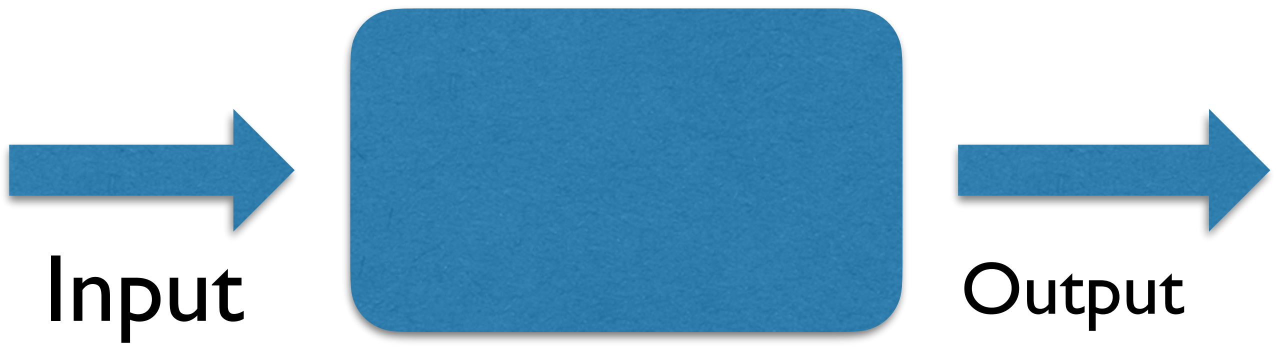
Risto Miikkulainen
The University of Texas at Austin and Cognizant Technology Solutions

Abstract

The main power of artificial intelligence is not in modeling what we already know, but in creating solutions that are new. Such solutions exist in extremely large, high-dimensional, and complex search spaces. Population-based search techniques, i.e. variants of evolutionary computation, are well suited to finding them. These techniques make it possible to find creative solutions to practical problems in the real world, making creative AI through evolutionary computation the likely "next deep learning."

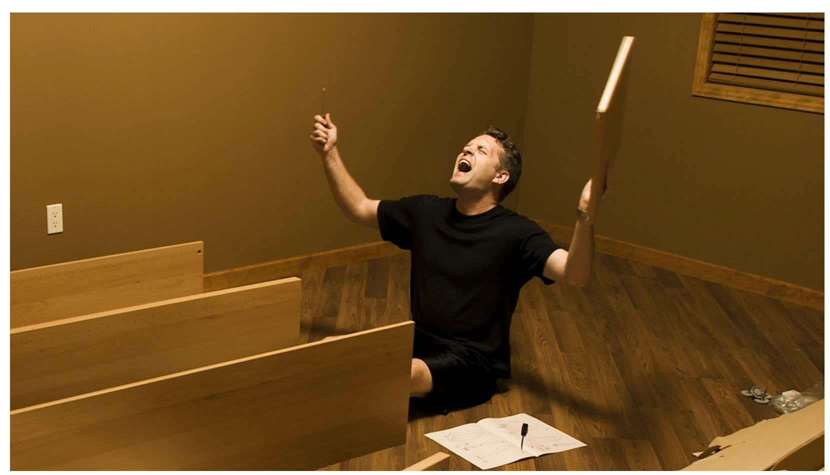
Problems categorization

Model



Input

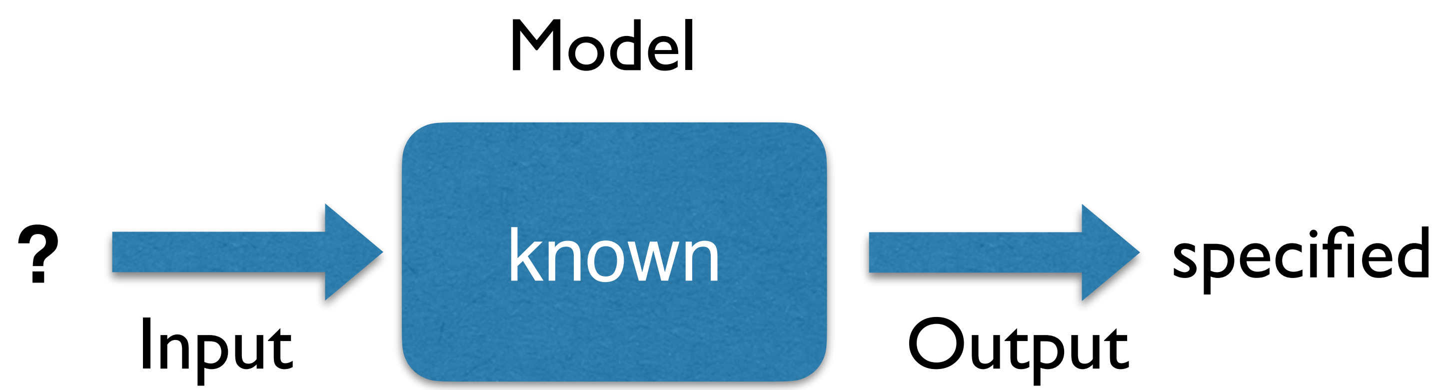
Output



- Consists of three components
- Different problems for different unknown components

Problem type 1: Optimization

- We have a model of our system and seek inputs that give us a specified goal



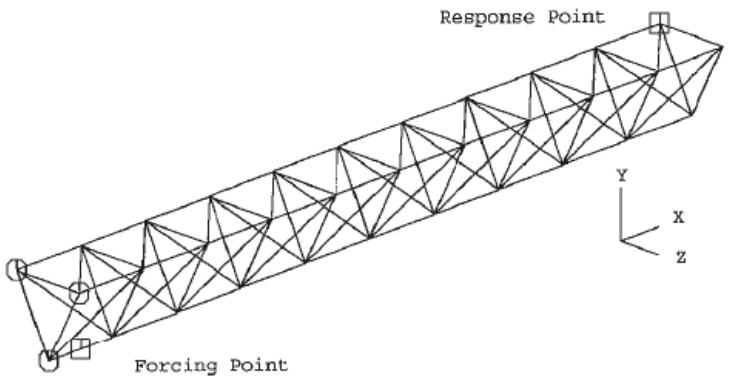
- Examples
  - Time tables for university, call center, or hospital
  - Traveling salesman problem (TSP)
  - Knapsack problem

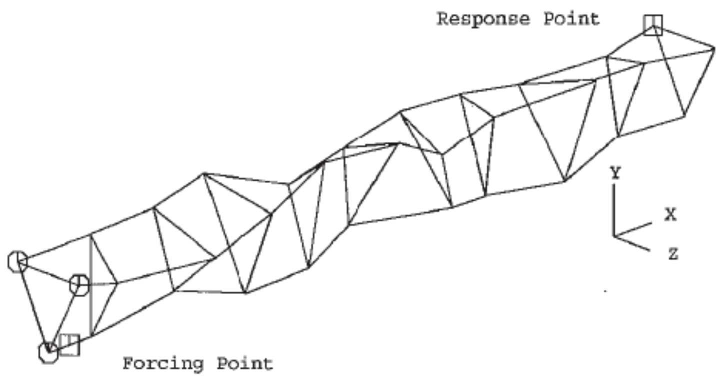
Optimization example 1: University timetabling

- Enormously big search space
- Timetables must be good
- “Good” is defined by a number of competing criteria
- Timetables must be feasible
- Vast majority of search space is infeasible

Optimization example 2: Satellite structure

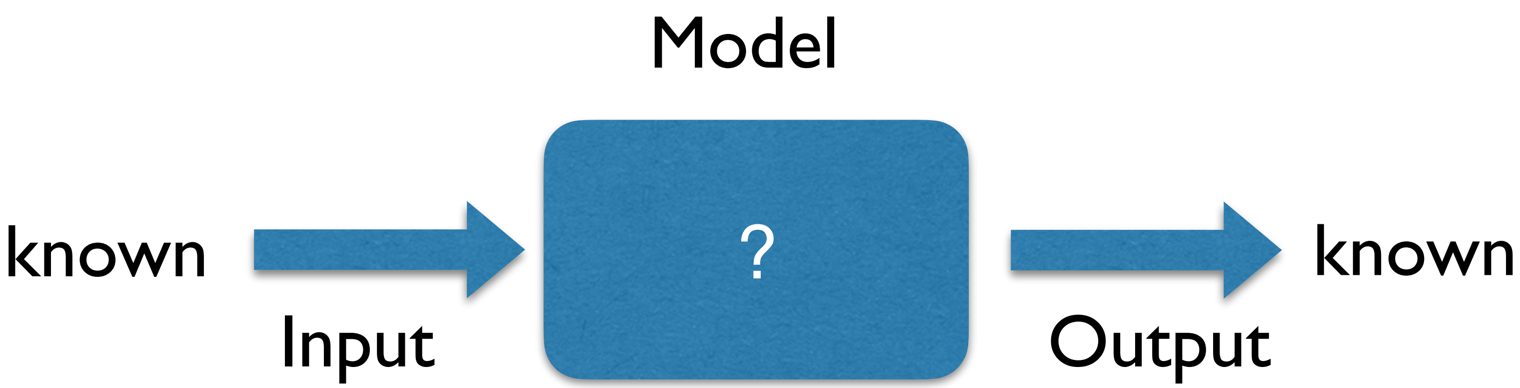
- Optimized satellite designs for NASA to maximize vibration isolation
- Evolving: design structures
- Fitness: vibration resistance
- Evolutionary "creativity"





Problem type 2: Modeling

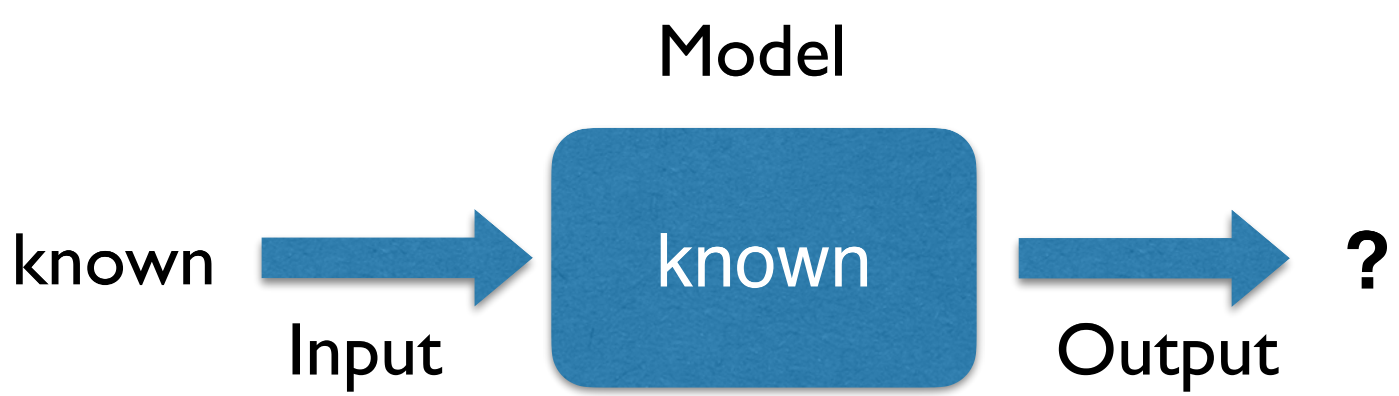
- We have corresponding sets of inputs and outputs and seek model that delivers correct output for every known input



- Examples
  - Classification of data, handwriting recognition
  - Predicting stock exchange price development
  - Voice-control system for smart phone

Problem type 3: Simulation

- We have a given model and wish to know the outputs that arise under different input conditions
- Often used to answer "what-if" questions in evolving dynamic environments



- Examples:
  - Evolutionary economics
  - Artificial life
  - Impact analysis of a new tax system

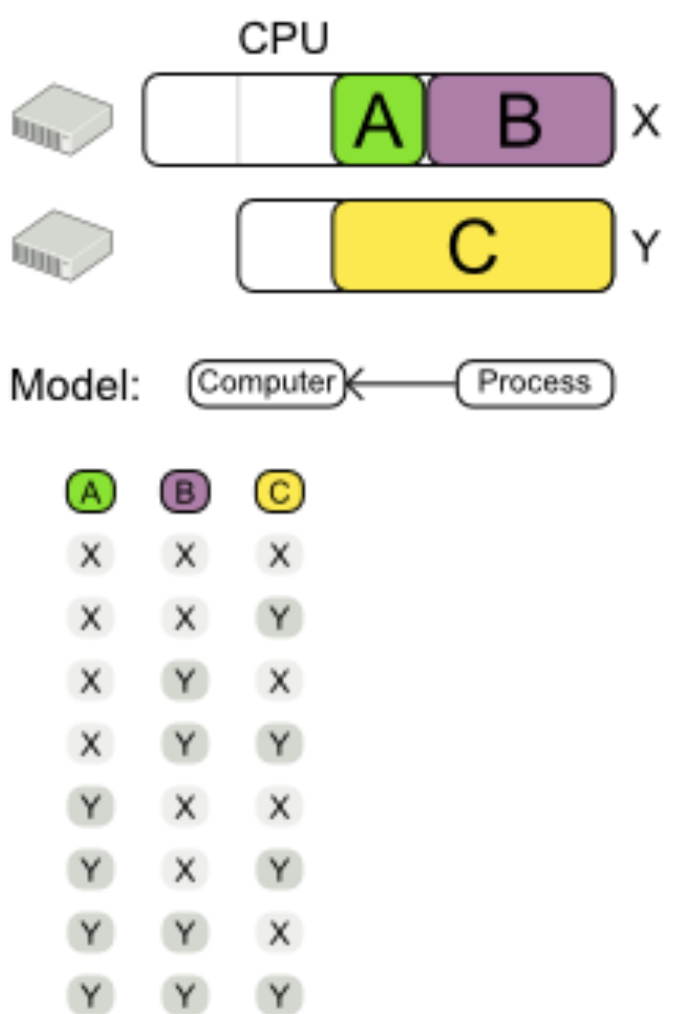
Search

- Solving an optimization or a modeling problem requires the identification of a particular object in a space of possibilities
- This space can be, and usually is, enormous or even infinite
- Problem solving -> search
- Search space: the collection of all objects of interest including the solution we are seeking

Calculate the size of the search space

Given a Solution model, how many different combinations can it represent?

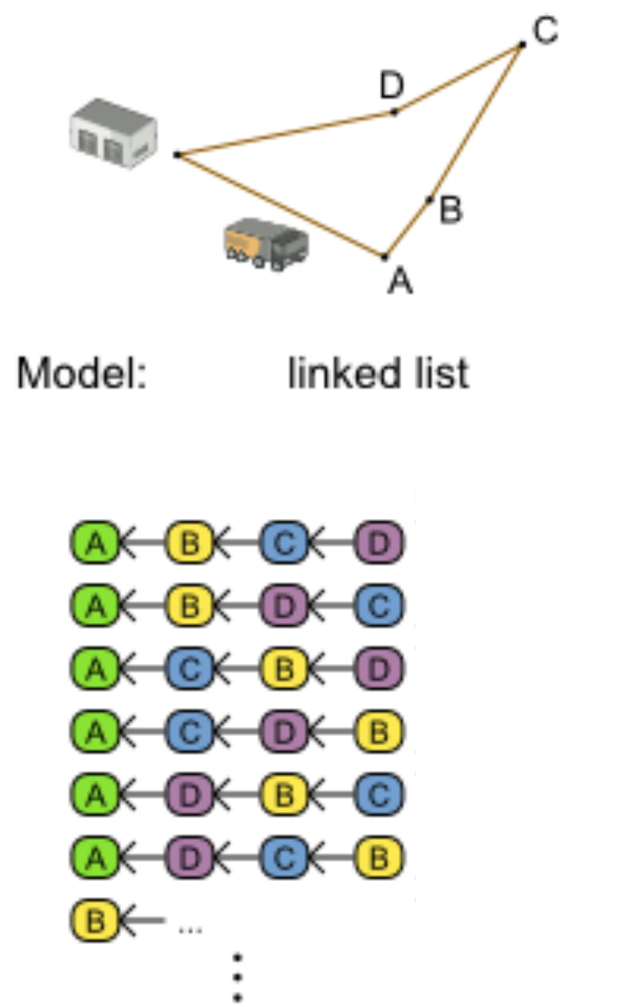
Cloud balancing



Search space: c^{p}

# computers	# processes	search space
2	3	8
100	300	10^{600}
200	600	10^{1380}
400	1200	10^{6967}

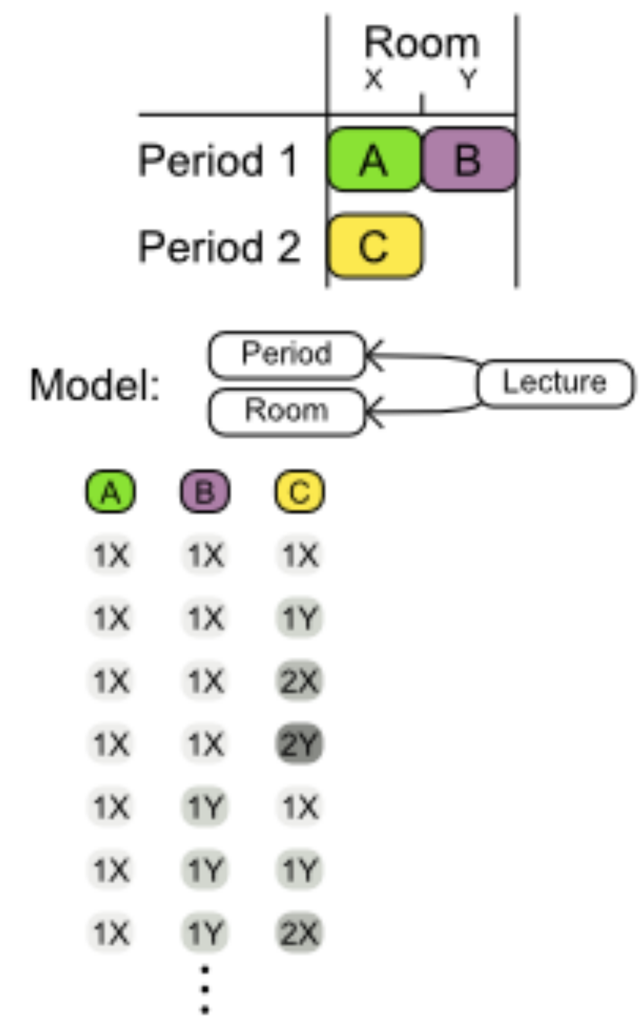
Traveling salesman (TSP)



Search space: $n!$

# customers	search space
4	24
100	10^{157}
1000	10^{2567}
10000	10^{35659}

Course scheduling



Search space: $(p \times r)^{\ell}$

# periods	# rooms	# lectures	space
2	2	3	64
36	6	100	10^{233}
36	18	400	10^{1124}
36	36	800	10^{2490}

https://www.optaplanner.org/blog/2014/03/27/IsTheSearchSpaceOfAnOptimizationProblemReallyThatBig.html