CO3091 - Computational Intelligence and Software Engineering

Lecture 01

Module Introduction

Leandro L. Minku

CO3091 - Computational Intelligence and Software Engineering

Lecture 01

Module Introduction

Leandro L. Minku

This lecture is being recorded

Content

- Motivation for the use of computational intelligence.
- Brief overview of what the course will cover.
- Course admin.

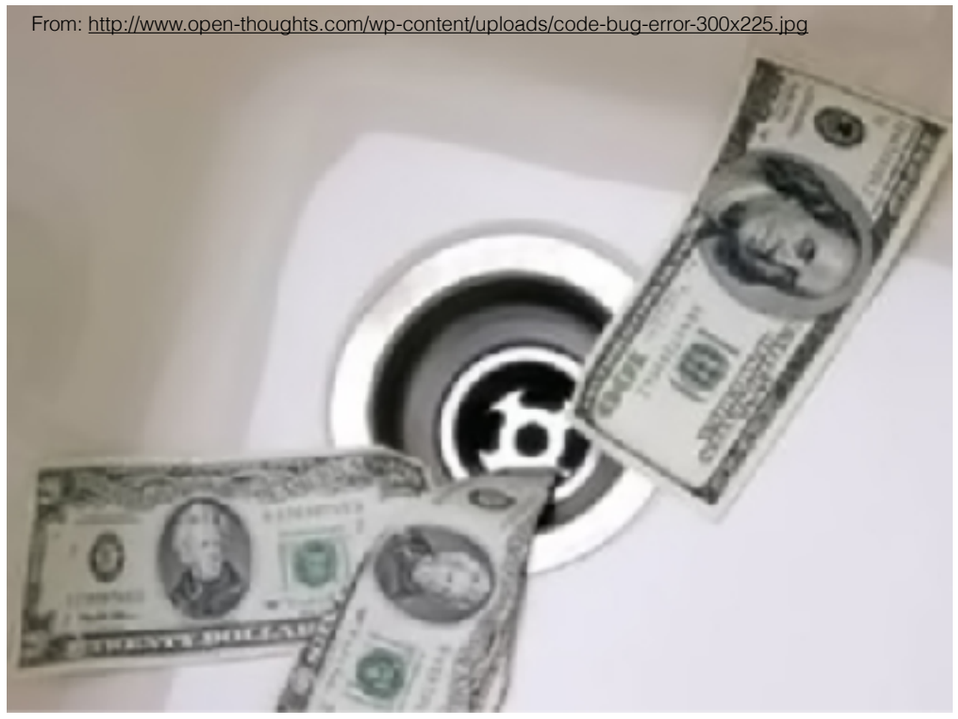
Real World Problems

- Several real world problems are:
  - time consuming and
  - challenging

  to solve manually.

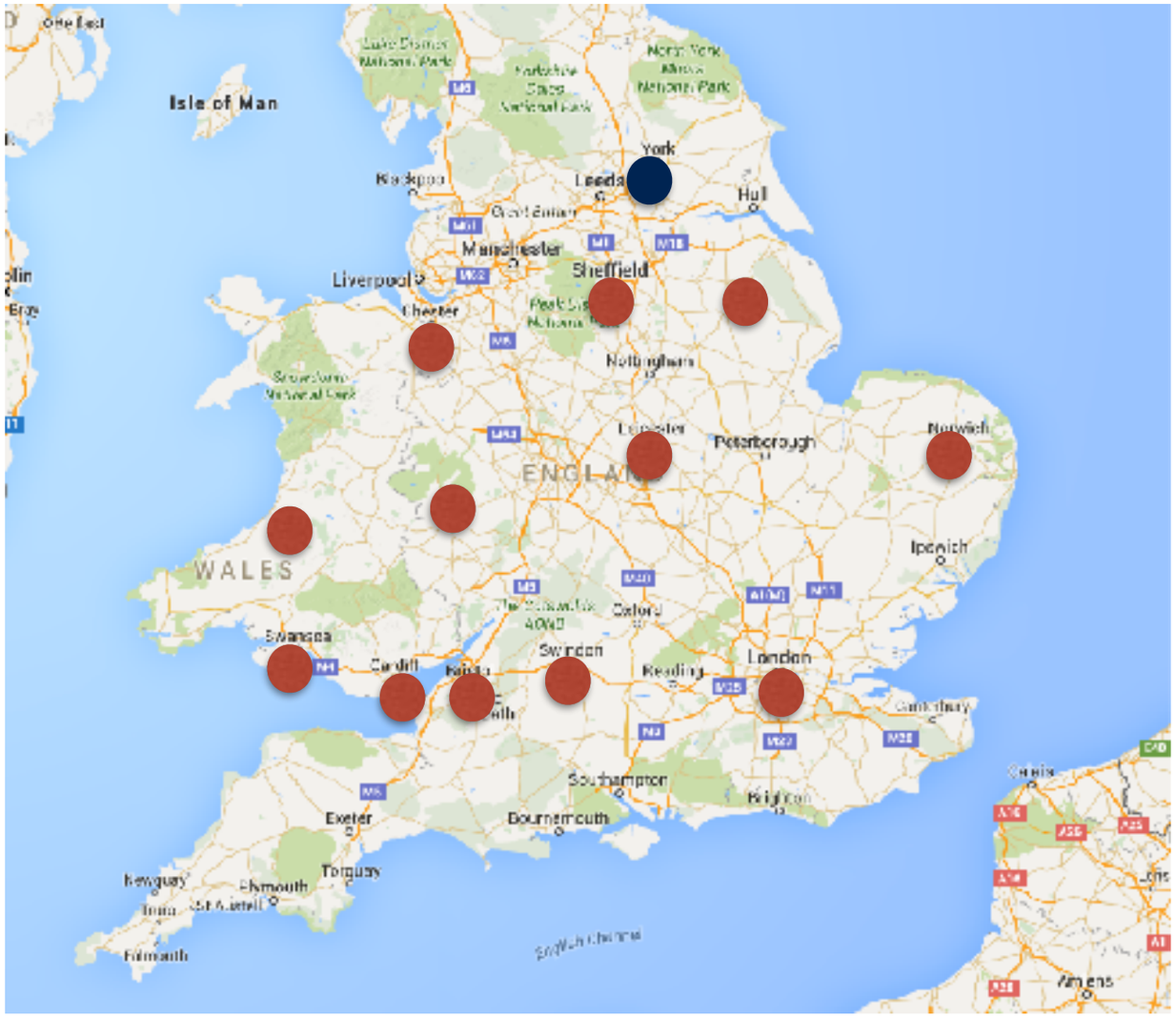
- Result:
  - Takes too much time, and thus a lot of money.
  - Solutions may still be not very good.

From: http://www.open-thoughts.com/wp-content/uploads/code-bug-error-300x225.jpg



Example of Problem

- Traveling Salesman Problem:
  - A salesman must travel passing through *N* cities.
  - Depending on the route the salesman takes, he/she will need to travel longer / shorter distances.



Problem: find a route that minimises traveling distance.

Examples of Other Problems

- routing problems,
- assignment and scheduling problems,
- consumer electronic devices,
- stock market prediction,
- medical, biomedical and bioinformatics problems
- etc.

Software Engineering Problems

- Several software engineering problems are:
  - time consuming and
  - error-prone

  to solve manually.

- Result:
  - Inefficient software development.
  - Low software quality.

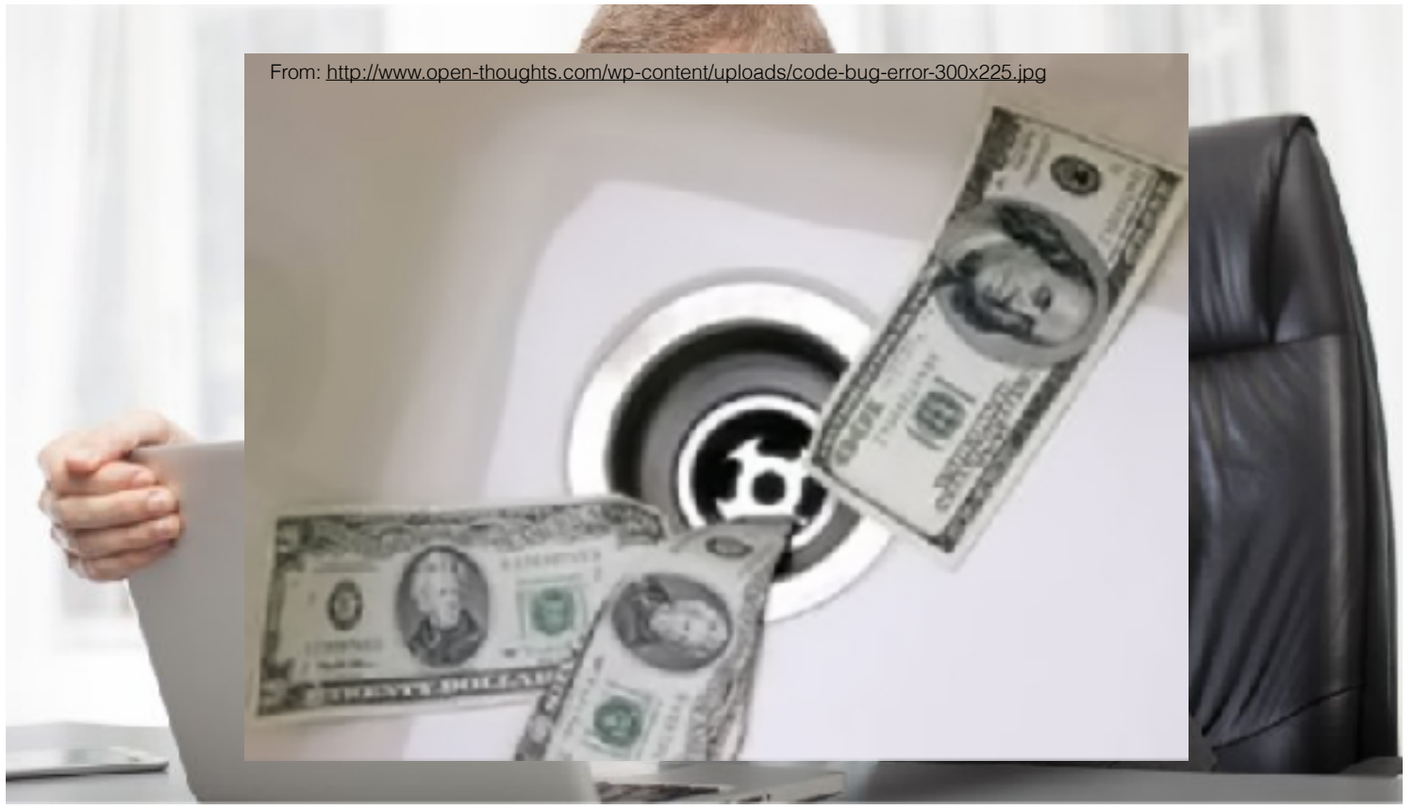


Image from: https://media.licdn.com/mpr/mpr/AAEAAQAAAAAAAAMMAAAAJDY5MjkzNDc4LTAxZmEtNDdlZC1hZTA3LWQ2NzU3OTIyNGQ4Yg.jpg

Software Engineering Problems

- Software energy optimisation
  - Software energy consumption is becoming more and more important, e.g., mobile apps and tablets.
  - How to minimise software energy consumption?

Problem: create a GUI colour scheme that minimises energy consumption.

Image from: https://www.electricchoice.com/wp-content/uploads/2015/09/america-energy-consumption.jpg

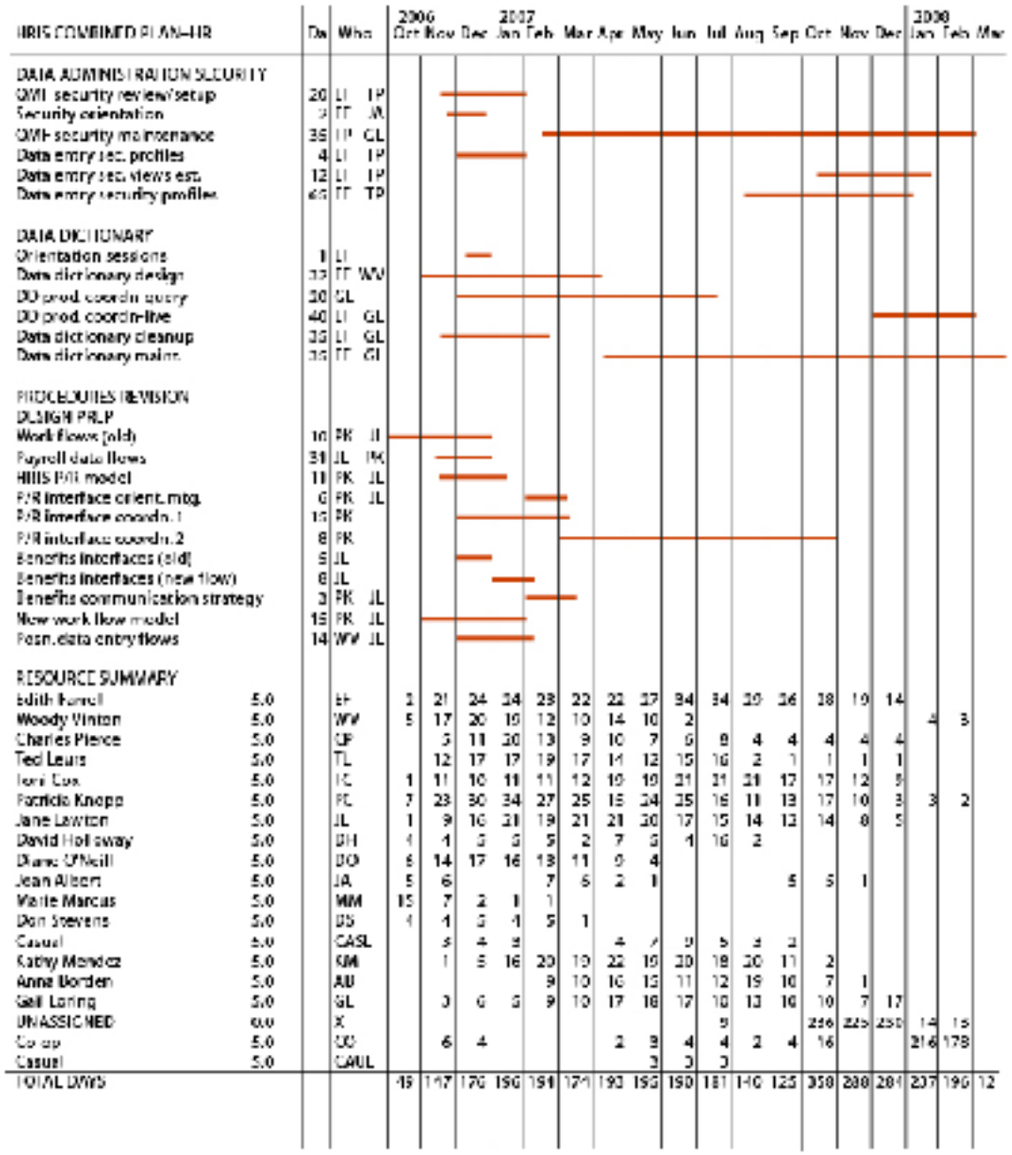
Software Engineering Problems

- Software test suite generation
  - Testing software to reduce the number of crashes is important.
  - Software testing is expensive.

Problem: generate a test suite that maximises coverage and the number of crashes found while minimising size.

Image from https://i.amz.mshcdn.com/M-OBdn8kJi7SxO70zht6eCiR-K0=/950x534/filters:quality(90)/2014%2F08%2F12%2F44%2FFacebookCra.bb9f9.jpg

Software Project Scheduling



Problem: create an allocation of employees to tasks in a software project so as to minimise its cost and duration.

- The space of possible allocations is enormous and there are constraints.

Software Project Scheduling

[Video from VideoVat.com posted by Marc RaZZ at: https://youtu.be/wXaPB7slzNE]

Software Effort Estimation

Problem: estimate how much effort is required to develop a software project.

- *x* person-hours mean that one person would take *x* hours to develop the project.
- Human-made software effort estimations are typically 30%-40% wrong*, and are worse for larger projects.

http://isha.sadhguru.org/blog/wp-content/uploads/2015/05/gut-feeling.jpg

* M. Jorgensen and K. Molokken. A review of software surveys on software effort estimation. International Symposium on Empirical Software Engineering, pp. 223 - 230, 2003.

Software Engineering Problems as Optimisation and Learning Problems

- Several of these problems can be formulated as optimisation or learning problems.
- They can then be solved by computational methods in an automated way.

Computational intelligence approaches can be used as decision-support tools to help performing optimisation and learning software engineering tasks more easily (more efficiently and effectively).

Optimisation and Learning Problems

- Optimisation problems: finding a solution that achieves one or more pre-defined goals.
  - E.g., find the best allocation of employees to tasks so as to minimise cost and duration of the software project.

- Learning problems: creating models able to infer knowledge from data.
  - E.g., based on data describing previous software projects, create a model able to estimate / predict the effort required to develop new projects.

You will learn how to formulate real world problems, including software engineering ones, as optimisation or learning problems.

Computational Intelligence

You will learn computational intelligence algorithms that can be used to solve these real world problems, and how to use them.

- No single definition.
- Heuristic algorithms, which aim to find good solutions to problems in a reasonable amount of time.*
  - Typically not guaranteed to find the optimum, but able to find near-optimal solutions.
  - Good for problems where we cannot afford enumerating all possible solutions to guarantee optimality.
  - Good for problems where no exact algorithm exists to solve the problem in polinomial time.
- Learning algorithms that can build models based on data.

* http://ukci.cs.manchester.ac.uk/intro.html

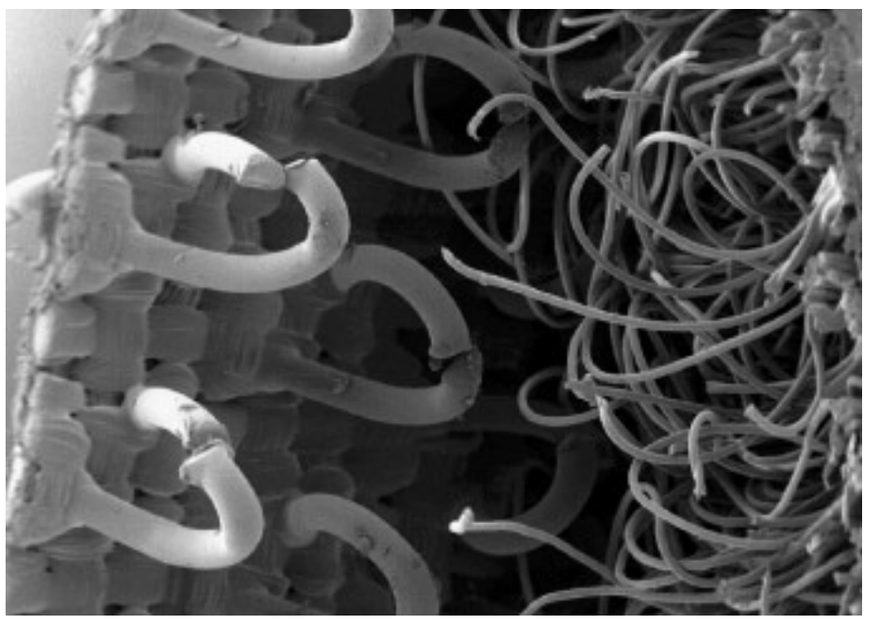
Examples of Computational Intelligence Approaches

- Hill-climbing
- Simulated annealing
- Evolutionary algorithms
- Ant colony optimisation
- K-Nearest Neighbours
- Decision trees
- Naive-Bayes
- …

Inspiration from Nature

From: https://upload.wikimedia.org/wikipedia/commons/thumb/1/13/Bur_Macro_BlackBg.jpg/220px-Bur_Macro_BlackBg.jpg

From: http://i.livescience.com/images/i/000/050/771/i02/velcro-horizontal.jpg?1369176195



From: http://www.todayifoundout.com/wp-content/uploads/2011/09/Velcro-e1315606382449.jpg

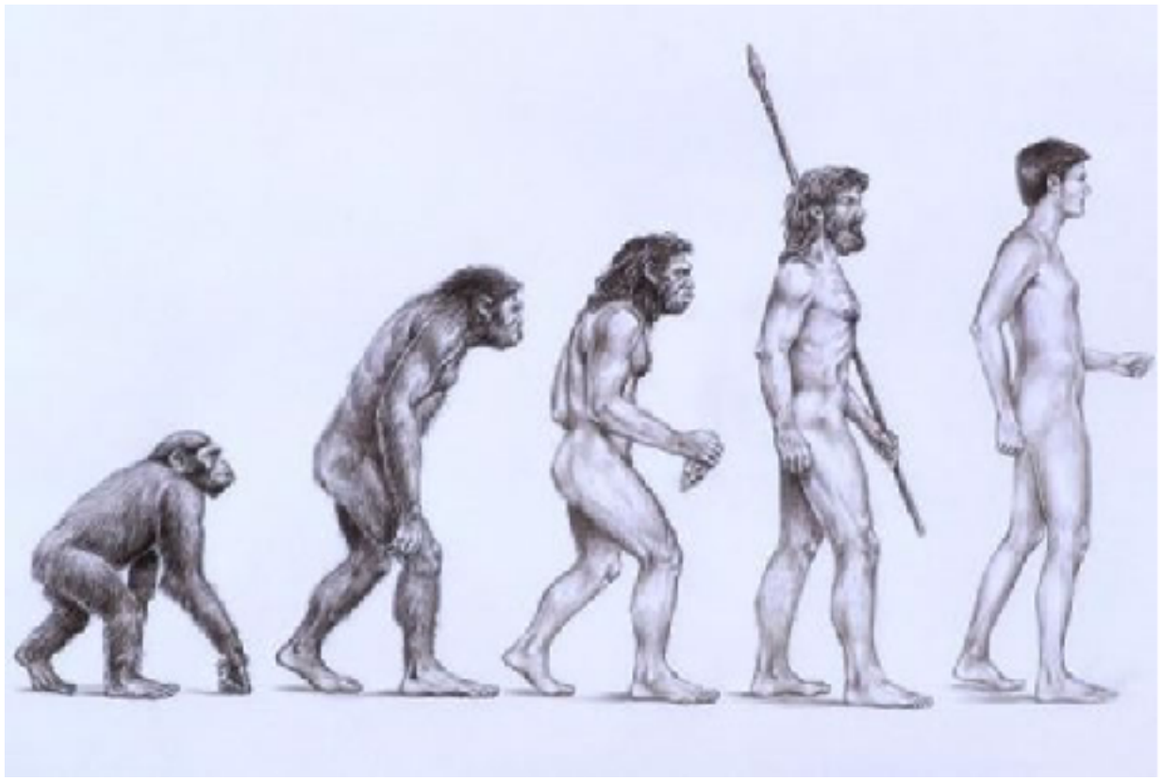
Inspiration from Nature

From: http://www.jornaldaregiaosudeste.com.br/fotos/1296761675-exposicao_Vera_Meo.jpg

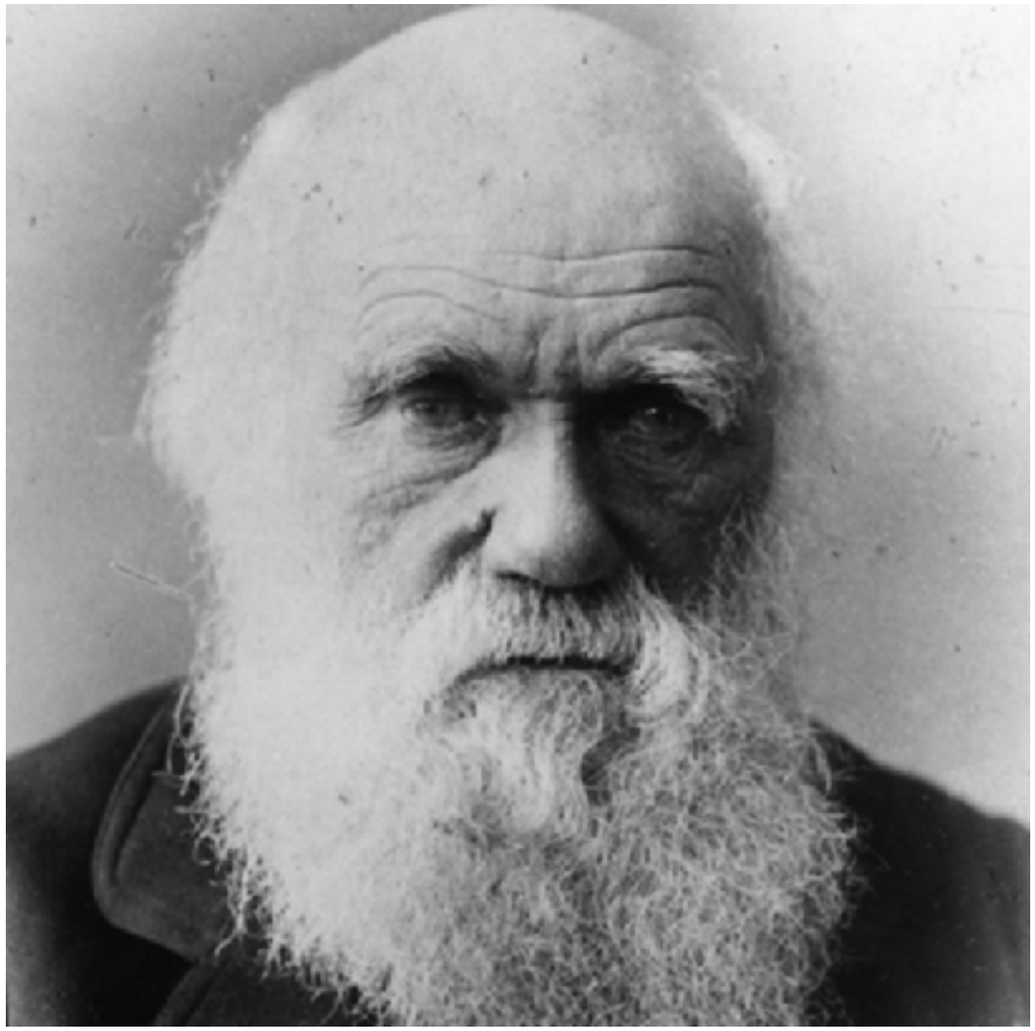
From: http://www.wright-house.com/wright-brothers/wrights/photos/wright-flyer-566x442-10.jpg

From: http://www.planet-science.com/media/12521/jet_499x330.jpg

Evolutionary Algorithms

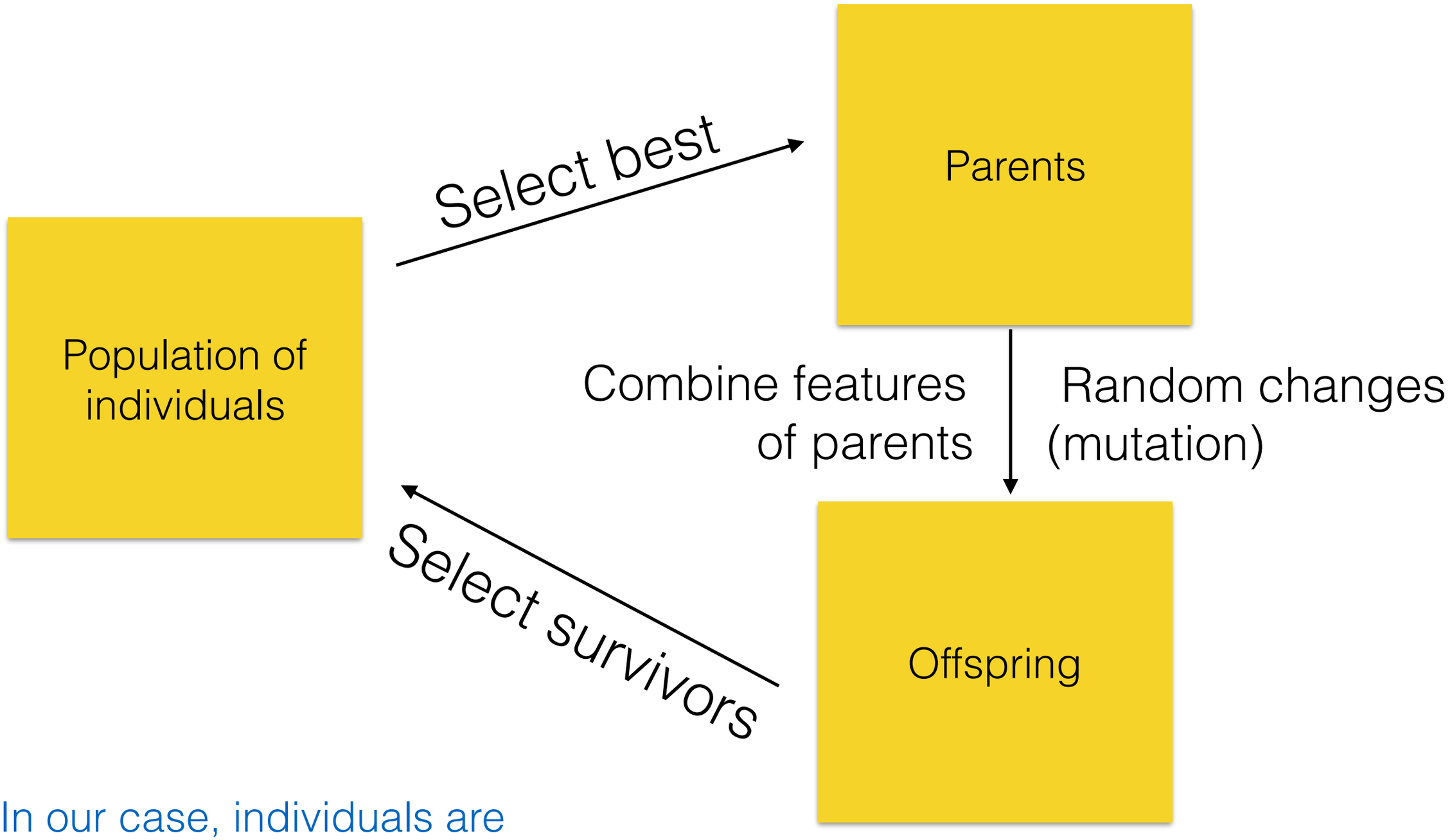


From: http://i0.wp.com/www.knowledgeidea.com/wp-content/uploads/2014/11/Charles-Darwin-Theory-Evolution.jpg?resize=604%2C405



http://a4.files.biography.com/image/upload/c_fit,cs_srgb,dpr_1.0,h_1200,q_80,w_1200/MTE5NDg0MDU0OTM4NjE3MzU5.jpg

Evolutionary Algorithms



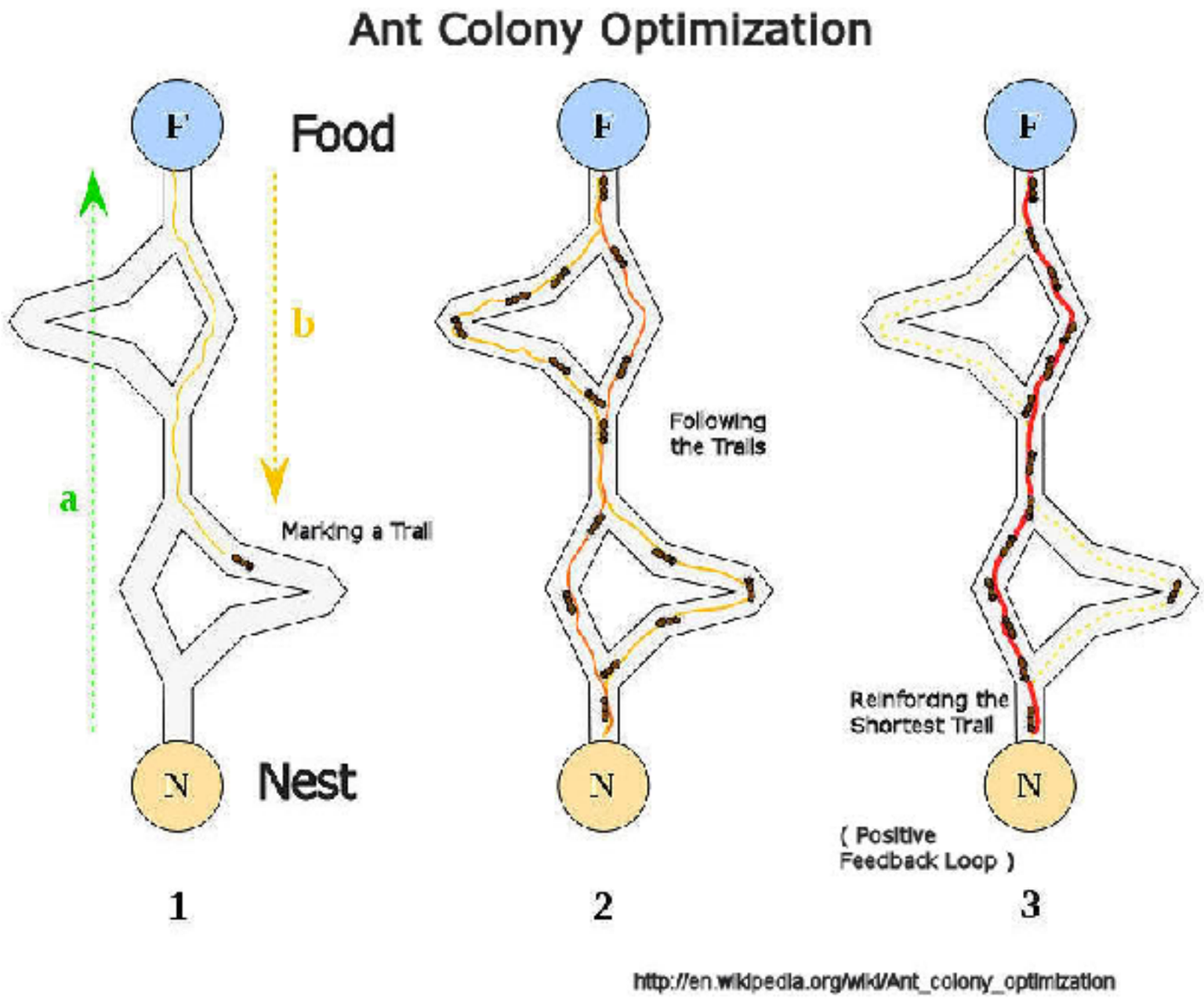
In our case, individuals are candidate solutions to a problem.

Ant Colony Optimisation

From: https://upload.wikimedia.org/wikipedia/commons/3/34/Safari_ants.jpg

Video posted by Sandeep Kumar: https://youtu.be/3oCQ2DWAp4c

Ant Colony Optimisation



http://i158.photobucket.com/albums/t106/OnlyObvious/ants/Aco_branches_svg_800_Notes.jpg

1. The first ant wanders randomly until it finds the food source (F), then it returns to the nest (N), laying a pheromone trail.

2. Other ants follow one of the four possible paths at random, also laying pheromone trails. Since the ants on the shortest path lay pheromone trails faster, this path gets reinforced with more pheromone, making it more appealing to future ants.

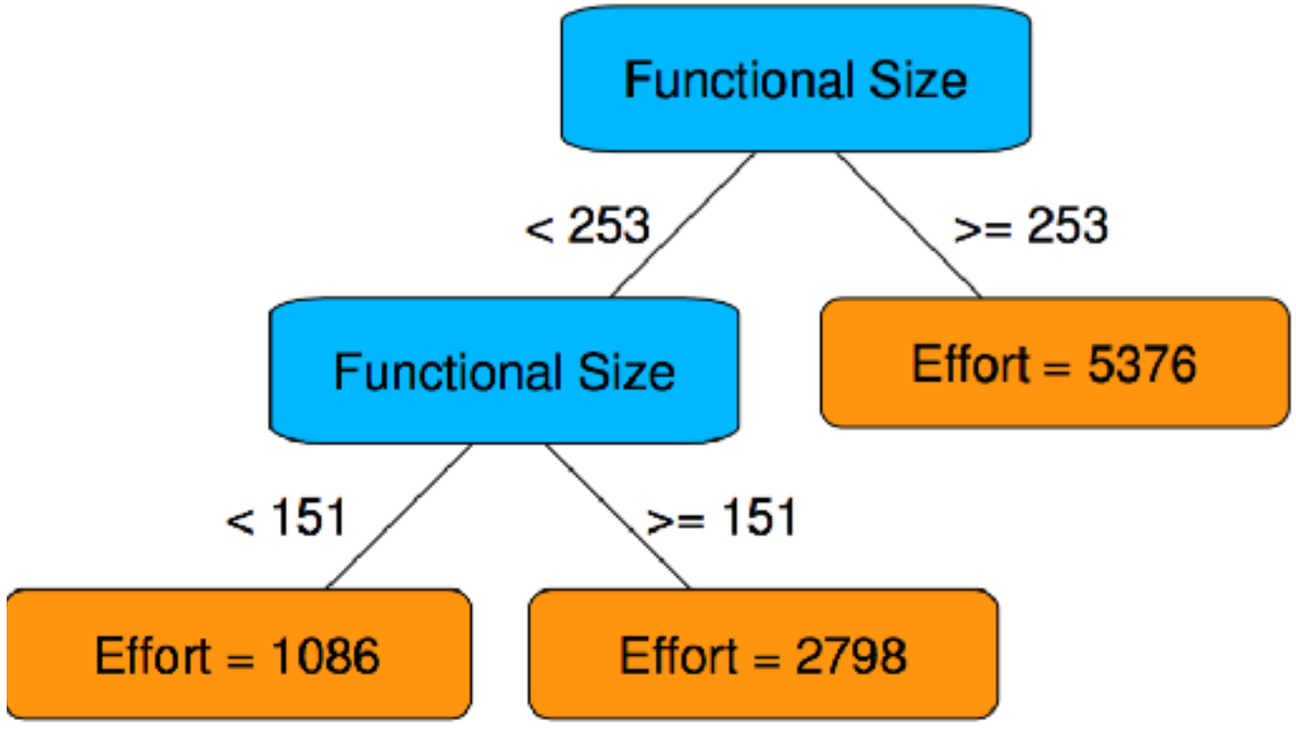
3. The ants become increasingly likely to follow the shortest path since it has the most pheromone. The pheromone trails of the longer paths evaporate.

Why Getting Inspiration from Nature?

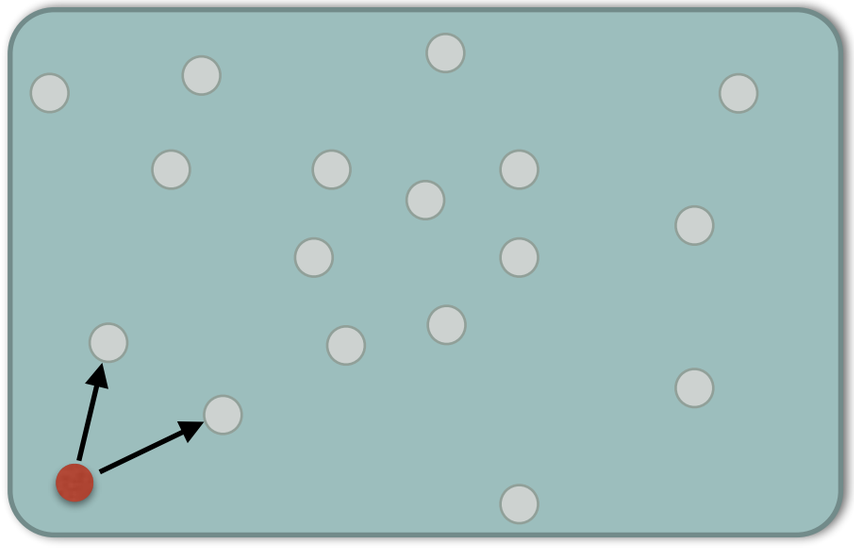
- Flexible: applicable to different problems.
- Robust: can deal with noise and uncertainty.
- Adaptive: can deal with dynamic environments.
- Autonomous: can function without human intervention.

Not All Approaches are Nature-Inspired

Decision Trees



K-Nearest Neighbours



The knowledge you'll gain from this module will help you to identify problems that can be solved with computational intelligence and to design solutions for them.

Technological overview of the next generation Shinkansen high-speed train Series N700: http://uic.org/cdrom/2008/11_wcrr2008/pdf/R.1.3.3.3.pdf

Image from: https://www.japan-guide.com/g17/2018_tokaido_01.jpg

Module Administration

- Website: https://campus.cs.le.ac.uk/teaching/resources/CO3091/

- Class plan and study guide online.

- Lecture notes available online after each lecture.
  - Previous year's students asked not to include each stage of animations.

- Recordings will be available after each lecture.

- Four classes per week (lectures / surgeries / problem classes)

  Mon 16:00 - 17:00 (PHY LTB)
  Tue 12:00 - 13:00 (GP LTC)
  Thu 11:00 - 12:00 (BEN LT3)
  Fri 10:00 - 11:00 (BEN LT10)

Module Administration

- **No classes** from Tue to Fri in the week of 7th November.
- Classes at **9am GP LTA** on
  - Wed 18th October
  - Wed 25th October
  - Wed 1st November

Surgeries and Problem Classes

- Surgeries and problem classes dates are in the class plan, rather than being in a fixed day of the week.
  - Allows for practice and feedback in a timely manner, shortly after the topics that most need it.

- Surgeries:
  - Exercises to help you with learning the content of the lectures and practice for the exam.
  - Exercises to challenge you with improving your knowledge further.
  - Time to solve + time to discuss answers / feedback.

- Problem classes:
  - Discuss answers of surgery exercises that have not been discussed during surgeries.
  - Possibly some extra exercises.

Surgeries and Problem Classes

Important: do attempt to solve the exercises before seeing the answers. This will help you to develop your problem solving skills!

If you miss one of the surgeries / problem classes, do arrange a meeting to see me as soon as you can attempt them, to get feedback.

Lab Sessions

- Lab sessions:
  - Friday 15:00-16:00, CW 301,
    - 13th October
    - 3rd November
    - 17th November
  - Wednesday 10:00-11:00, CW 301,
    - 18th October
  - GTA: Mr Michael Chiu
- Try out:
  - Some tools available for running computational intelligence algorithms.
  - Environment to support comparison between different algorithms.
- Aims:
  - Help out with assessed coursework.
  - Core knowledge for running the tools in the future.

Assessment

- 60% Exam (alternative resit exam)
- 40% Two courseworks (alternative resit coursework)
  - 20% Coursework 1 (less programming)
    - Handed out on 17th October
    - Deadline on 31st October
  - 20% Coursework 2 (more programming)
    - Handed out on 2nd November
    - Deadline 23rd November
- Pass mark: 40%
- Email: leandro.minku@leicester.ac.uk

Further Reading

- None for today!
- We will learn about each of these topics in more detail over the next lectures.