Introduction to Metaheuristic Optimization

Nuno Antunes Ribeiro

Assistant Professor

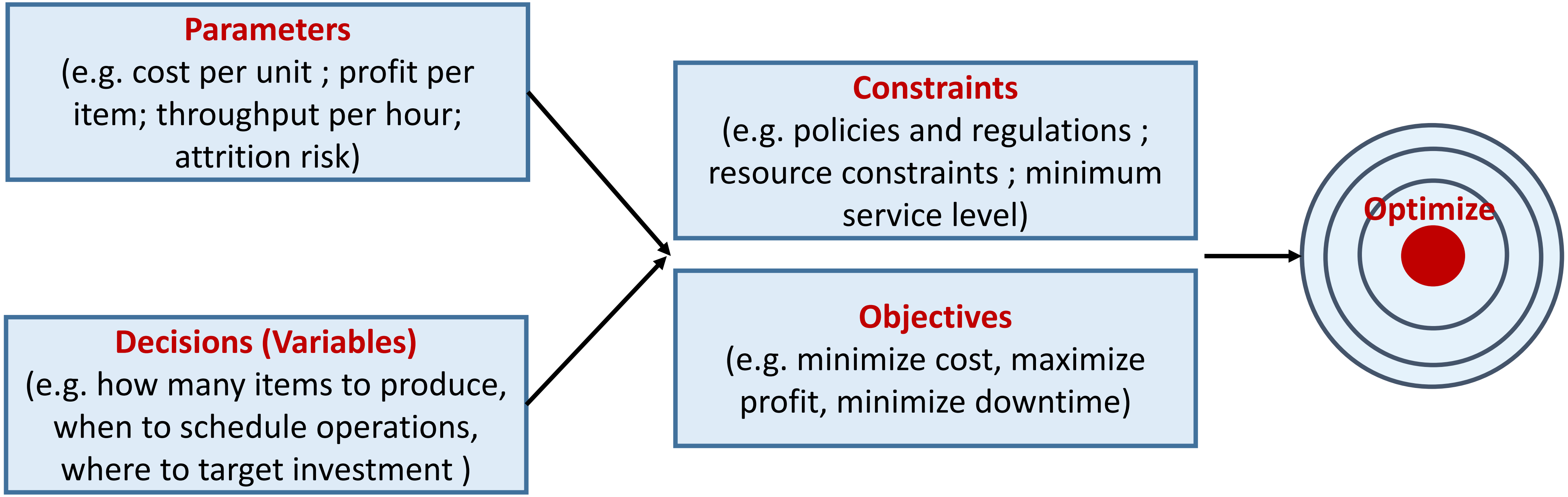
SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Engineering Systems and Design

What is Optimization?

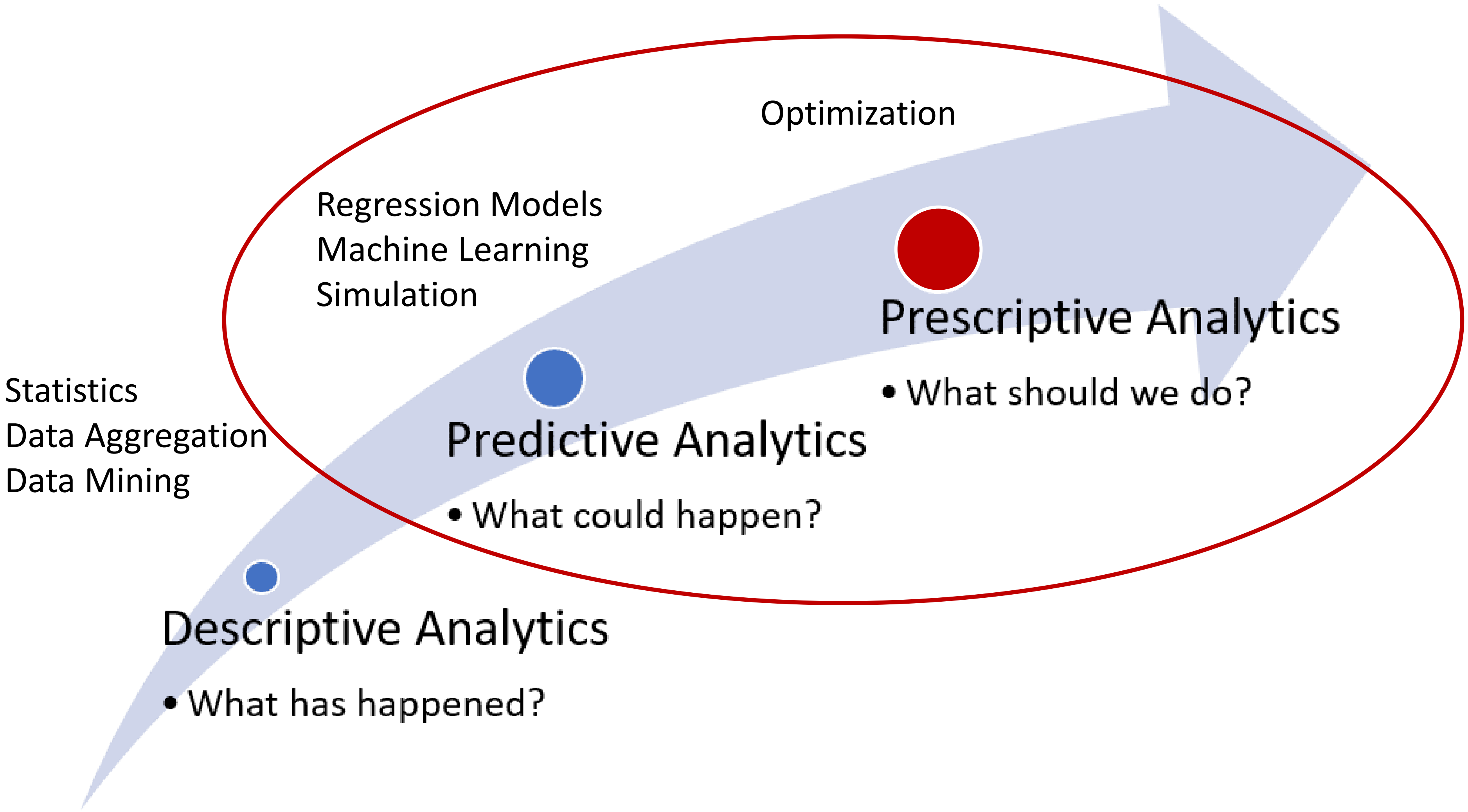
- What is optimization: an act, process, or **methodology** of making something (such as a design, system, or **decision**) as fully **perfect**, functional, or effective as possible

  specifically : the **mathematical procedures** (such as finding the maximum of a function) involved in this (***Merriam Webster Dictionary***)

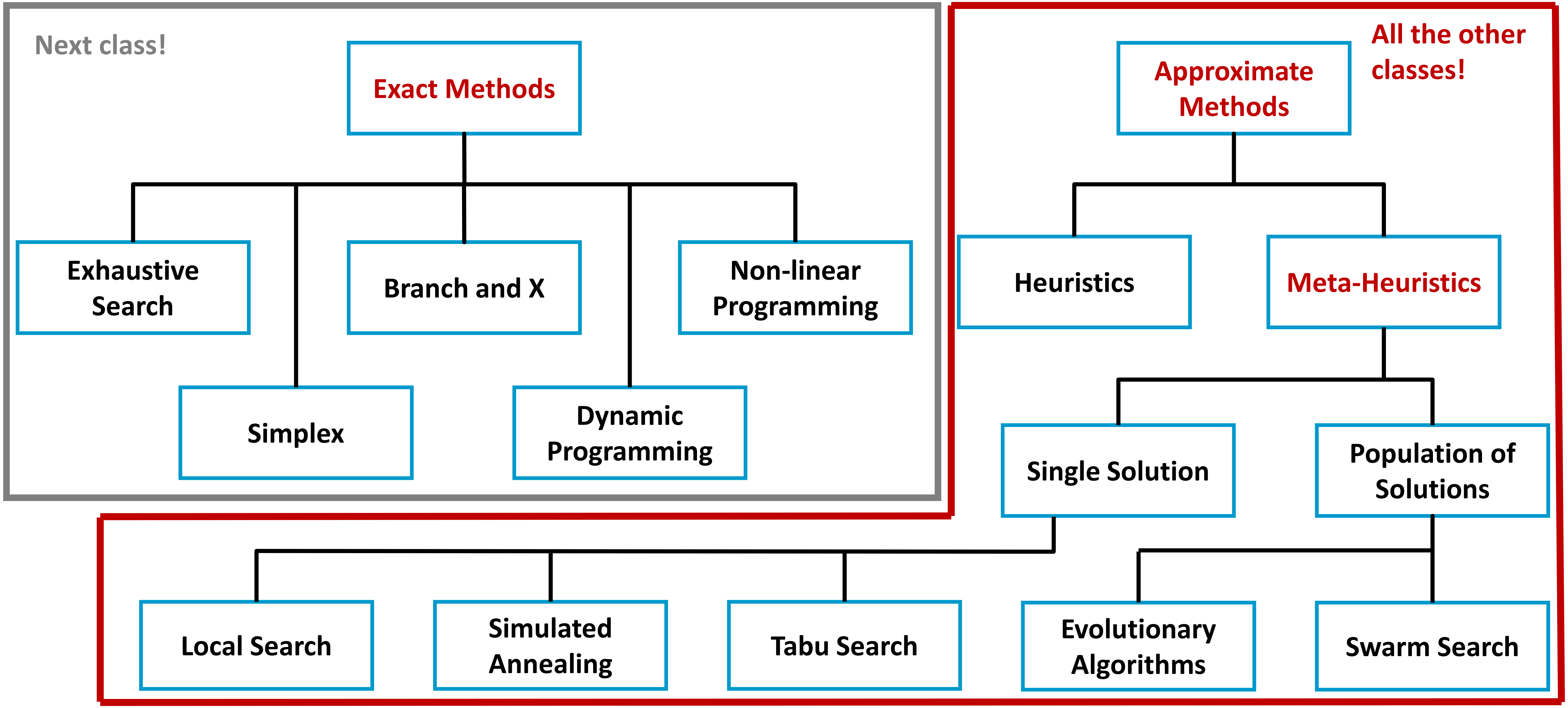


Different Solution Methods

From Predictive to Prescriptive Analytics

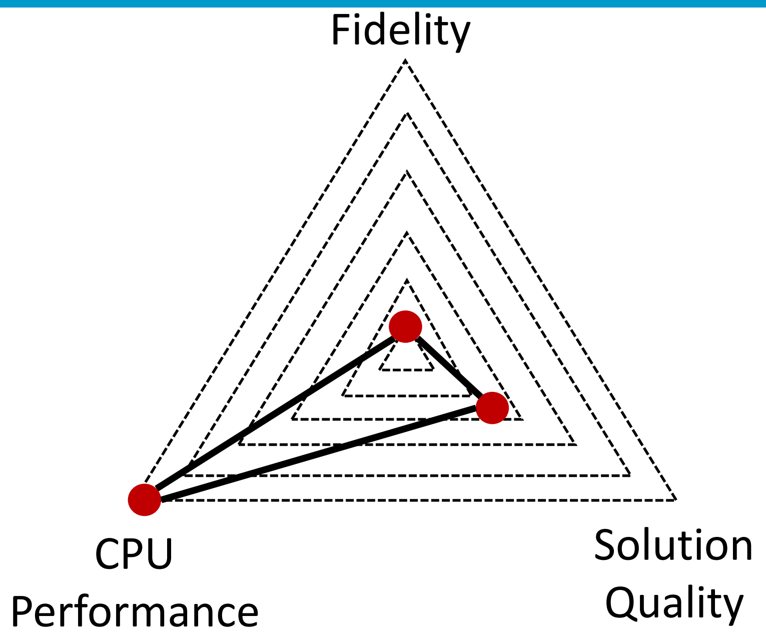


Optimization Solution Methods



Which Solution Method to Select?

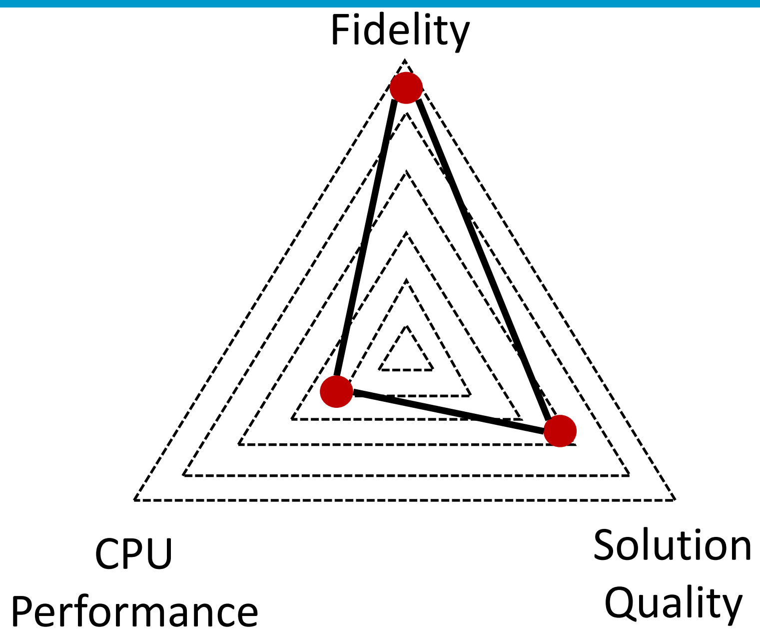
- In order to minimize errors when building your models, be mindful of the following basic steps:
  - **Clarify the problem and intended goal:** What problem is this model designed to solve? Who are the end users? What are users supposed to do with this model?
  - **Keep the model as simple as possible:** What is the minimum number of inputs and outputs required; Remember that the more assumptions a model has, the more complex it becomes.
  - **Identify the solution methods to use:** Plan how the inputs, processing, and outputs will be laid out. Ensure a good trade-off between **model fidelity** (level of detail of the model to replicate the reality); **computation performance** (CPU Time) **and solution quality** (how close is to the real optimal solution).



Model is too simplistic!

Which Solution Method to Select?

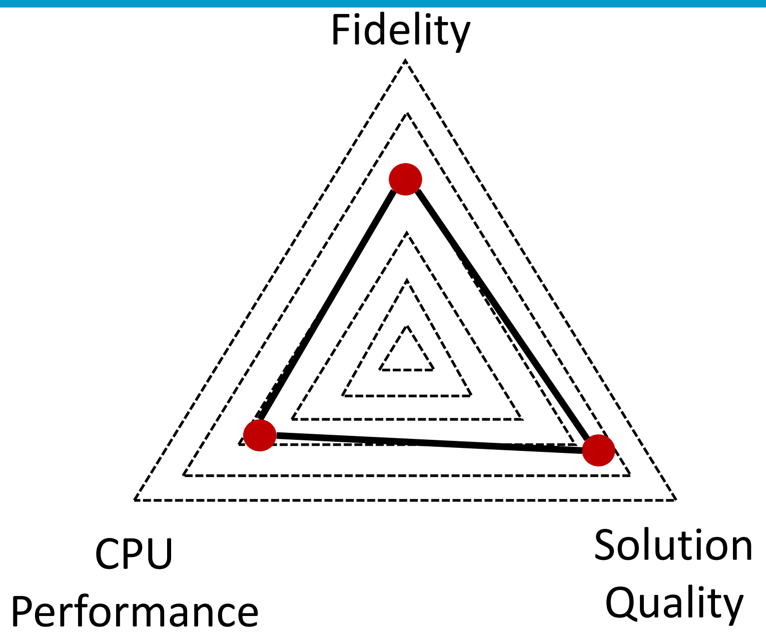
- In order to minimize errors when building your models, be mindful of the following basic steps:
  - **Clarify the problem and intended goal:** What problem is this model designed to solve? Who are the end users? What are users supposed to do with this model?
  - **Keep the model as simple as possible:** What is the minimum number of inputs and outputs required; Remember that the more assumptions a model has, the more complex it becomes.
  - **Identify the solution methods to use:** Plan how the inputs, processing, and outputs will be laid out. Ensure a good trade-off between **model fidelity** (level of detail of the model to replicate the reality)**; computation performance** (CPU Time) **and solution quality** (how close is to the real optimal solution).



Model is too complex!

Which Solution Method to Select?

- In order to minimize errors when building your models, be mindful of the following basic steps:
  - **Clarify the problem and intended goal:** What problem is this model designed to solve? Who are the end users? What are users supposed to do with this model?
  - **Keep the model as simple as possible:** What is the minimum number of inputs and outputs required; Remember that the more assumptions a model has, the more complex it becomes.
  - **Identify the solution methods to use:** Plan how the inputs, processing, and outputs will be laid out. Ensure a good trade-off between **model fidelity** (level of detail of the model to replicate the reality)**; computation performance** (CPU Time) **and solution quality** (how close is to the real optimal solution).



Perhaps a good balance (or not)

Why Metaheuristics?

- A large number of real-life optimization problems in science, engineering, economics, and business are complex and difficult to solve
- These optimization problems are too complicated to find the perfect (optimal) solution in reasonable time
- Therefore, we want to find approximate solutions: solutions which are as good as possible within a feasible time for computation
- This is what metaheuristic optimization algorithms do
- **<u>This is what you will learn in this course</u>**

Real World Optimization Problems?

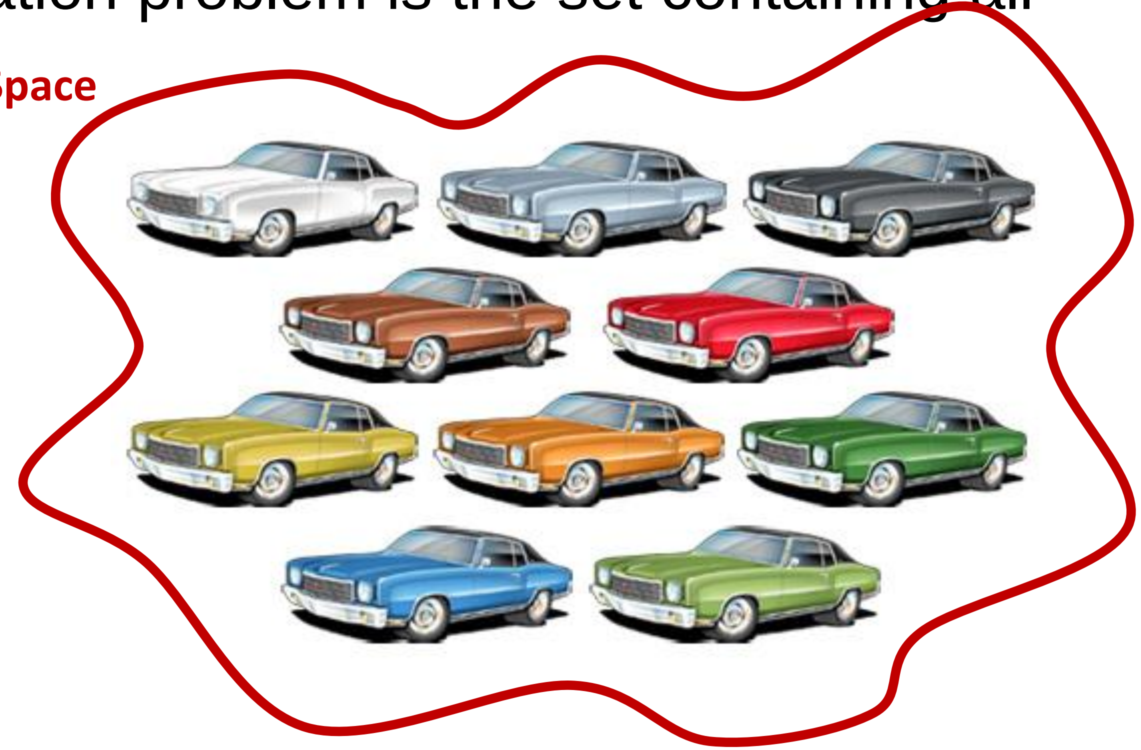
- You are an analyst at a logistic company. You are asked to develop modeling capabilities to support the company making more optimized and informed decisions towards reducing costs, increasing profits, and ensure costumer satisfaction

- Find routes on the map to minimize total distance
- Assignment of orders to containers to minimize the number of containers required
- Containers to trucks to minimize undelivered orders
- Select multiple depots and pickup delivery locations
- Consider that vehicles have capacity limits
- There are time constraints for pickup delivery

- **Time limit for the optimization tool: 5 – 6 hours**

Solution Space

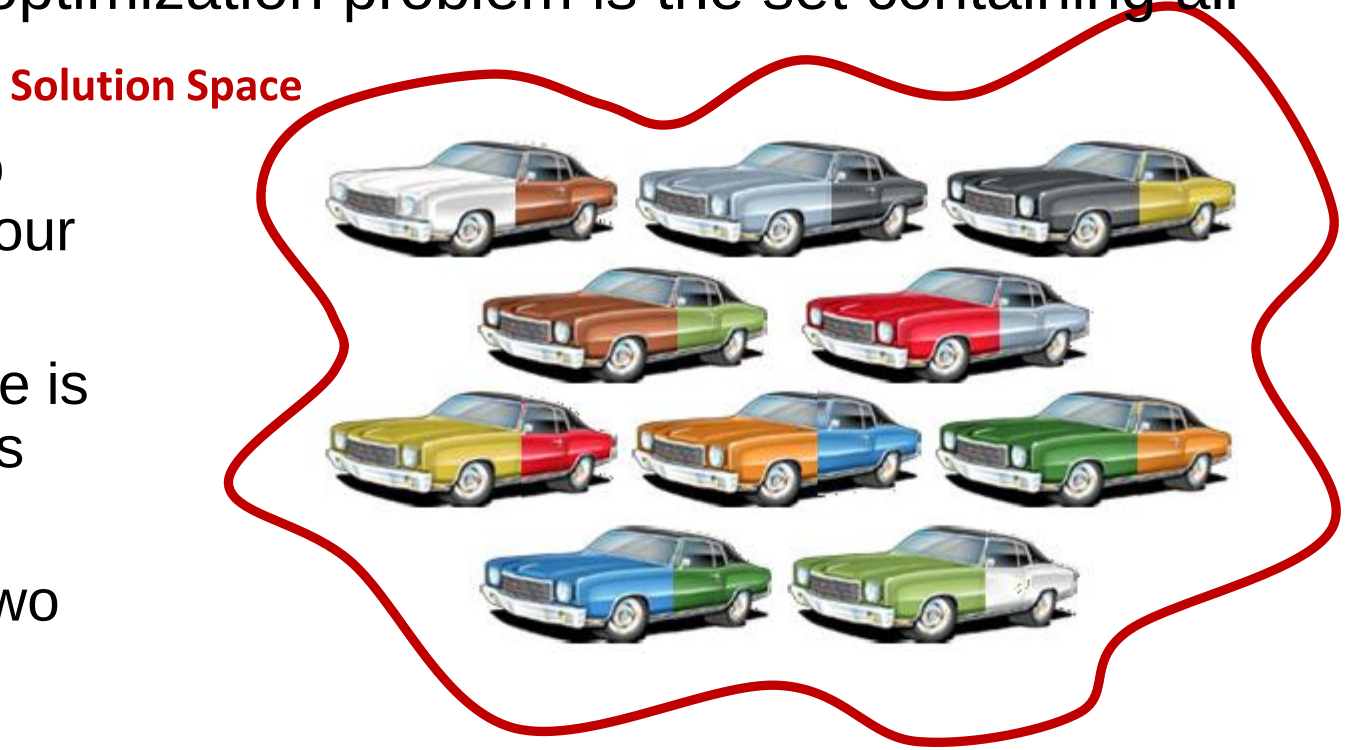
- Before selecting the solution method, first, we need to have an idea of **what we want to find - how many solutions would be generated in the worst-case scenario?**
- The **solution space** of an optimization problem is the set containing all solutions of a problem
- Example, imagine you want to decide which colour to paint your car
- The size of your solution space is given by the number of colours available (in this case $n = 10$)



Number of colours available is n

Solution Space

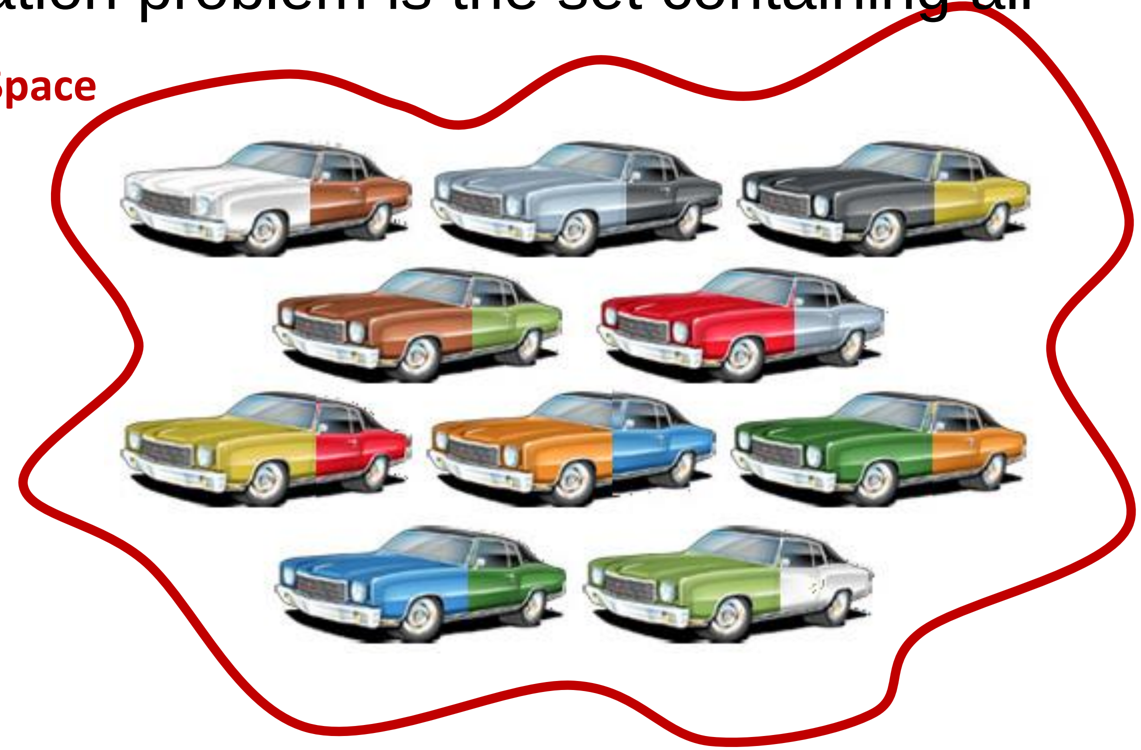
- Before selecting the solution method, first, we need to have an idea of **what we want to find - how many solutions would be generated in the worst-case scenario?**
- The **solution space** of an optimization problem is the set containing all solutions of a problem
- Example, imagine you want to decide which colour to paint your car
- The size of your solution space is given by the number of colours available (in this case 10)
- What if a car can have up to two colours?



Size of the solution space is n^2

Solution Space

- Before selecting the solution method, first, we need to have an idea of **what we want to find - how many solutions would be generated in the worst-case scenario?**
- The **solution space** of an optimization problem is the set containing all solutions of a problem



- Example, imagine you want to decide which colour to paint your car
- The size of your solution space is given by the number of colours available (in this case 10)
- What if a car can have up to two colours?
- or up to *m* colours,

Size of the solution space is n^m

Objective Function

- We also need to define a way to evaluate our solutions.
- Normally, there is not just good and bad, but many different degrees of solution quality
- An **objective function** is a (mathematical) function $f(X)$ which is subject to optimization
- When subject to minimization - if $f(X_1) < f(X_2)$, then X_1 is better than X_2

- Example minimize the cost:

$100

$200

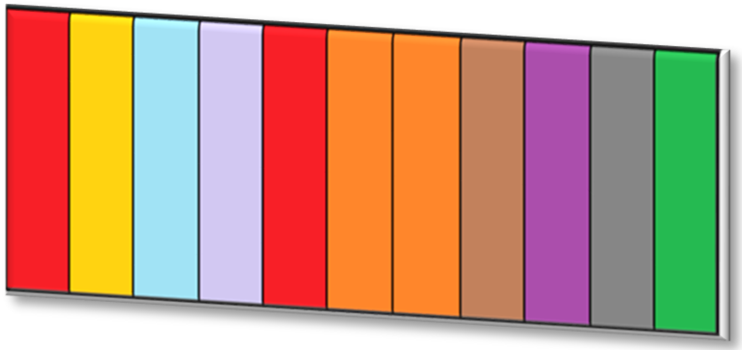
$95

Optimal Solution

Objective Function

- However note that, the objective functions does not necessarily need to be a function as you know it from Maths like $f(x) = x_2 + \ldots$, but may be **arbitrary complex, involve complicated simulations, human reasoning**, etc.
- Example: your personal evaluation of each combination of colours

Number of Colours	Number of Stripes	Size of the Solution Space
10	1	10
10	2	100
10	3	1000
10	4	10000
10	5	100000
10	6	1000000
10	7	10000000
10	8	100000000
10	9	1000000000
10	10	10000000000



Imagine you have a super brain capable to analyse one combination of colours per second

You will take 10000000000 seconds to evaluate all alternatives

That is 316 years and 321 days

Your grand grand …grand son will finally select the car

Optimization Problems

- Many questions in the real world are optimization problems, e.g.
  1. Find the shortest tour for a salesman to visit a certain set of regions in Singapore --- **Traveling Salesman Problem**

Solution space:

The TLS solution can be represented with the following data structure

$$Sol1 = \{1,2,3,\dots,n\}$$
$$Sol2 = \{2,n,3,\dots,1\}$$

etc.

There are $n!$ ways to permute n numbers.

$$e.g.\ if\ n = 4\ then\ |S| = n! = 24$$

But…

$$\{1,2,3,4\} = \{2,3,4,1\} = \{3,4,1,2\} = \{4,1,2,3\}$$

Every tour can be represented in $2n$ different ways

Thus…

$$|S| = \frac{n!}{2n} = \frac{(n-1)!}{2}$$

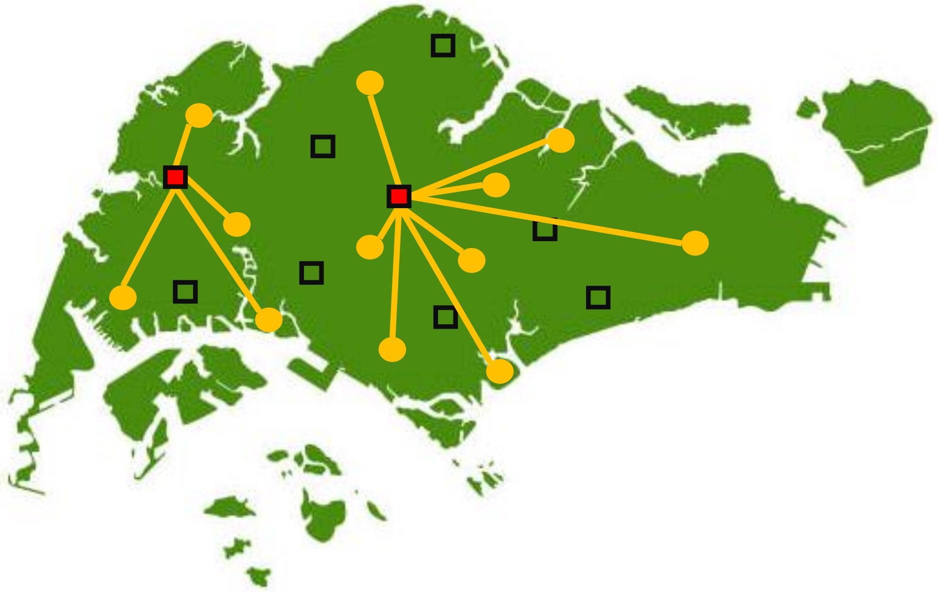
Optimization Problems

- Many questions in the real world are optimization problems, e.g.
    2. Finding the minimum number of facilities and their locations such that all demand points can be "covered" within a maximum distance from the closest facility.--- **Location Set Covering Problem**

Solution space:

Each location can be represented as a binary vector ---
1 if location n is selected, 0 otherwise
The number of candidate solutions is 2^n

The problem becomes much more complex if the locations are capacitated.
In that case, the demand points may not be all assigned to the nearest facility.
This creates additional decisions --- e.g. which facility is assigned to each demand point



Optimization Problems

- Many questions in the real world are optimization problems, e.g.
  3. Given a set of items, each with a weight and a value, determine which items to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible --- **Knapsack Problem**

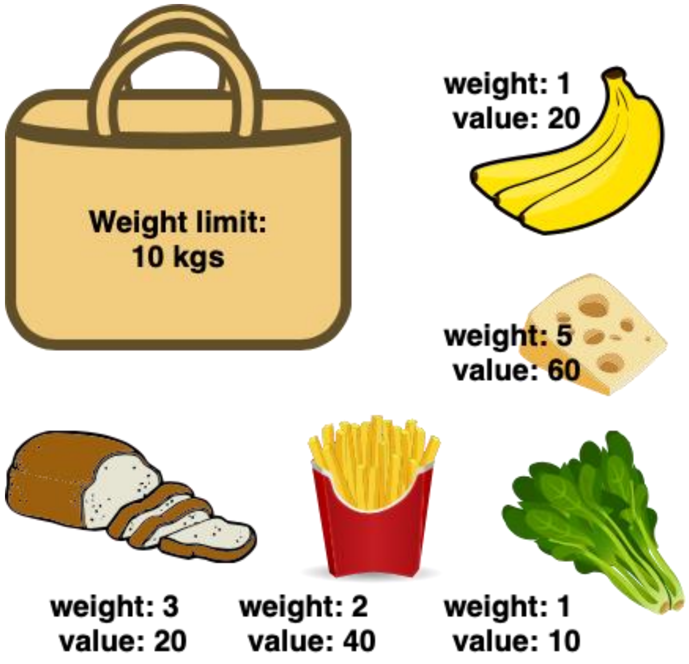
Solution space:

The knapsack solution can be represented with the following **data structure** :

$$Sol1 = \{1,0,0\ \dots, 0\}$$
$$Sol2 = \{1,1,0, \dots, 1\}$$

etc.

Each item can either be included or not
Value 1 means that the item was included, 0 otherwise
The number of possible solutions 2^n
However not all these solutions are feasible



Applications: finding the least wasteful way to cut raw materials; deciding on a selection of investments and portfolios.

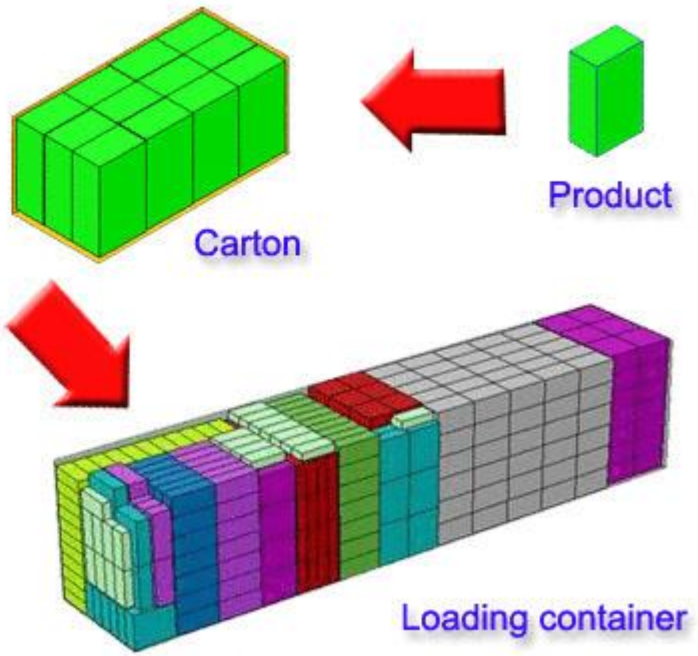
Optimization Problems

- Many questions in the real world are optimization problems, e.g.
  4. I need to transport n items from here to another city but they are too big to transport them all at once. How can I load them best into my car so that I have to travel back and forth the least times? --- **Bin Packing Problem**

Solution space:

<u>**The exact size of the solution space is hard to compute**</u>

<u>Worst scenario:</u>
Each bin can be represented as binary vector --- 1 if item n is included in the bin, 0 otherwise
The number of candidate solutions is 2^n
If we specify a limit number of bins m, then the number of possible solutions is $2^{n^{\mathrm{m}}}$
However, not all these solutions are feasible
We can develop algorithms that explore fewer solutions than $2^{n^{\mathrm{m}}}$



Optimization Problems

- Many questions in the real world are optimization problems, e.g.
  5. Many typical machine learning applications, from customer targeting to medical diagnosis, arise from complex relationships between features. **Feature selection** is the process of finding the most relevant inputs for a model.

Solution space:

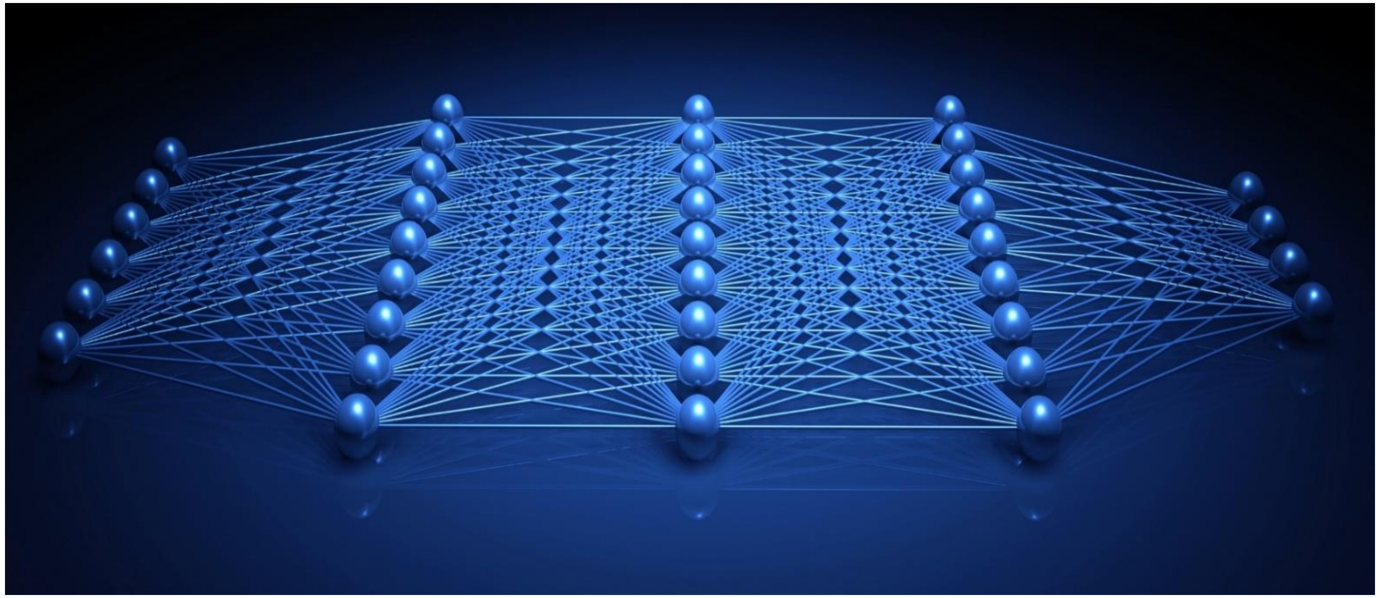
Feature selection can be formulated as an optimization problem. The objective function is the predictive model's generalization performance, represented by the error term on the selection instances of a data set.

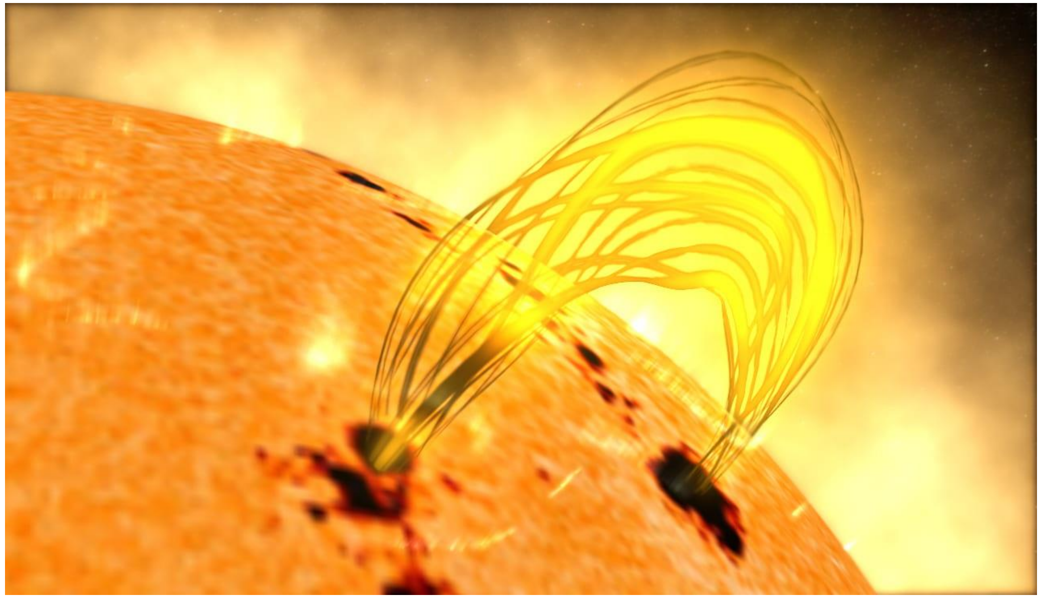
The design variables are the inclusion (1) or the exclusion (0) of the input variables in the ML model.

An exhaustive selection of features would evaluate 2^n different combinations, where n is the number of features

Optimization Problems

- Many questions in the real world are optimization problems, e.g.
  6. Neural Networks have helped us to solve many problems. But there's a huge problem that they still have **Hyperparameter tunning** --- **Training Neural Networks**



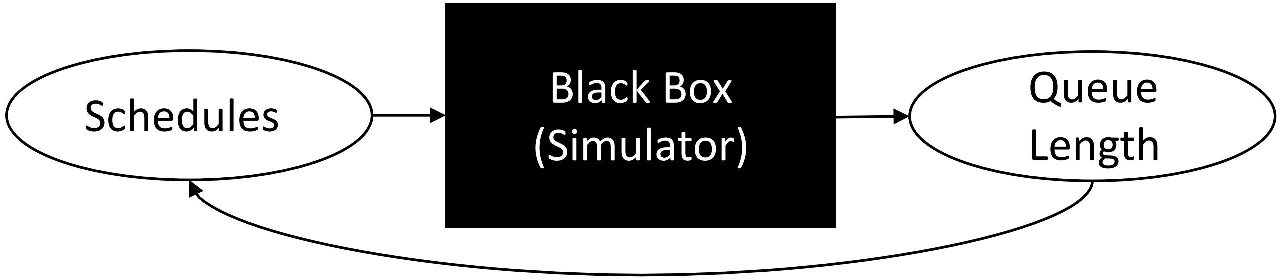


Optimization Problems

- Many questions in the real world are optimization problems, e.g.
  7. Queueing systems are generally modelled using queueing models or simulations. They tend to be used to support decision-makers scheduling “jobs” in a given system. The goal is minimizing queueing time, which according to queueing theory has a non-linear trend – **Scheduling in Queueing Systems**

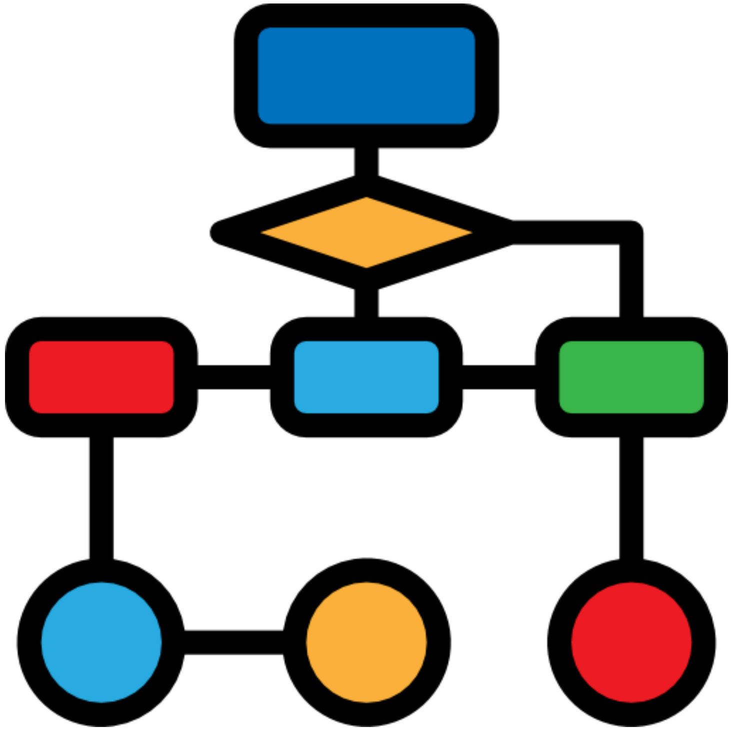
Solution space:

Let us consider a vector of time periods with size s.
Each job n is to be assigned to one period
Therefore each job has n possible solutions
The total number of candidate solutions is n^s



What is an Algorithm?

- An algorithm is a **finite set of well-defined instructions for accomplishing some task**. It starts in some initial state and usually terminates in a final state.
- Algorithms are the very basic of computer science. An algorithm tells us what we can do to solve a given task.
- An optimization algorithm is an algorithm to solve optimization
- Optimization algorithms tell us how to find solutions which are rated best (or at least well) from a set of possible solutions, for a general class of problems.



What is an Heuristic?

- In optimization, there exist **exact** and **approximated** algorithms
- **Heuristics are approximated algorithms** to solve given optimization problems
- However, they are **specialized algorithms** --- Say we have an approximate algorithm for the traveling salesman problem, another one for the set covering problem, etc.
- Should we develop a completely new method for each problem?
- **No! We want general heuristics that can be adapted to different problems. (also to reduce the development time . . . we often want a prototype quickly and add more complex logic later)**

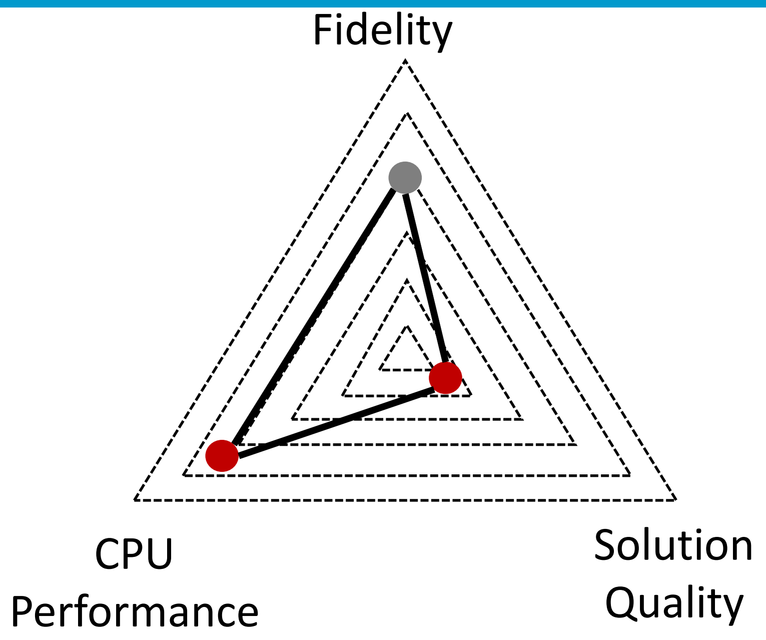
What is a Metaheuristic?

A metaheuristics are approximated algorithms for solving very general classes of optimization problems. They combine objective functions and heuristics in an abstract and hopefully efficient way, usually by treating them as black box-procedures.

- What will you learn in this course?
  - The most common metaheuristic paradigms
  - How to apply different metaheuristic approaches to address real-world challenges.
  - How to code metaheuristics to solve problems of your interest
  - How to evaluate the performance of your metaheuristics
  - How to adapt and improve existing metaheuristics to make them more efficient to tackle specific problems
  - Etc.

Random Sampling

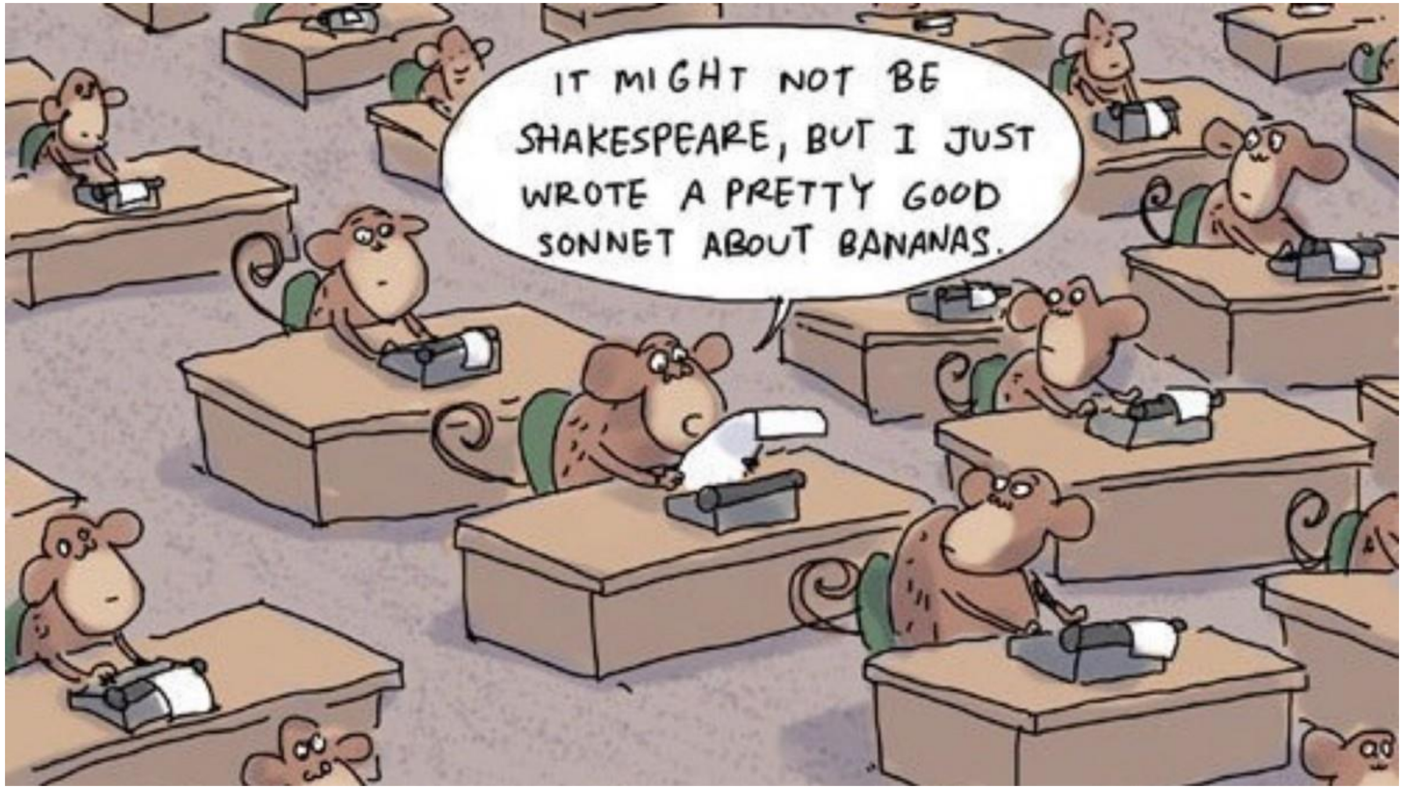
- Random sampling is the most basic "metaheuristic" optimization method
- It relies on the idea that by generating an infinite number of random solutions, eventually, the optimal solution will be found.
- However, random sampling is very ineffective for large problem instances - as the solution space increases the number of random solutions required to "likely" find the optimal solutions increases exponentially
- We need more advanced metaheuristic techniques



We need to improve the solution method

The Shakespeare Monkey

- The infinite monkey theorem states that a monkey hitting keys at random on a typewriter keyboard for an infinite amount of time will almost surely type any given text, such as the complete works of William Shakespeare.



The Shakespeare Monkey

- What is the probability of a monkey typing “to be or not to be that is the question”?
- Number of letter = 26 + space = 27 keys
- Likelihood of typing a “t” randomly = $1/27$
- Likelihood of typing a “to” randomly = $1/27 \times 1/27$
- Likelihood of typing the entire phrase = $(1/27)^\wedge 39$
- 1 in $66{,}555{,}937{,}033{,}867{,}822{,}607{,}895{,}549{,}241{,}096{,}482{,}953{,}017{,}615{,}834{,}735{,}226{,}163$

- We need 66×10^{55} monkeys to get the sentence “to be or not to be that is the question”?

The Shakespeare Monkey

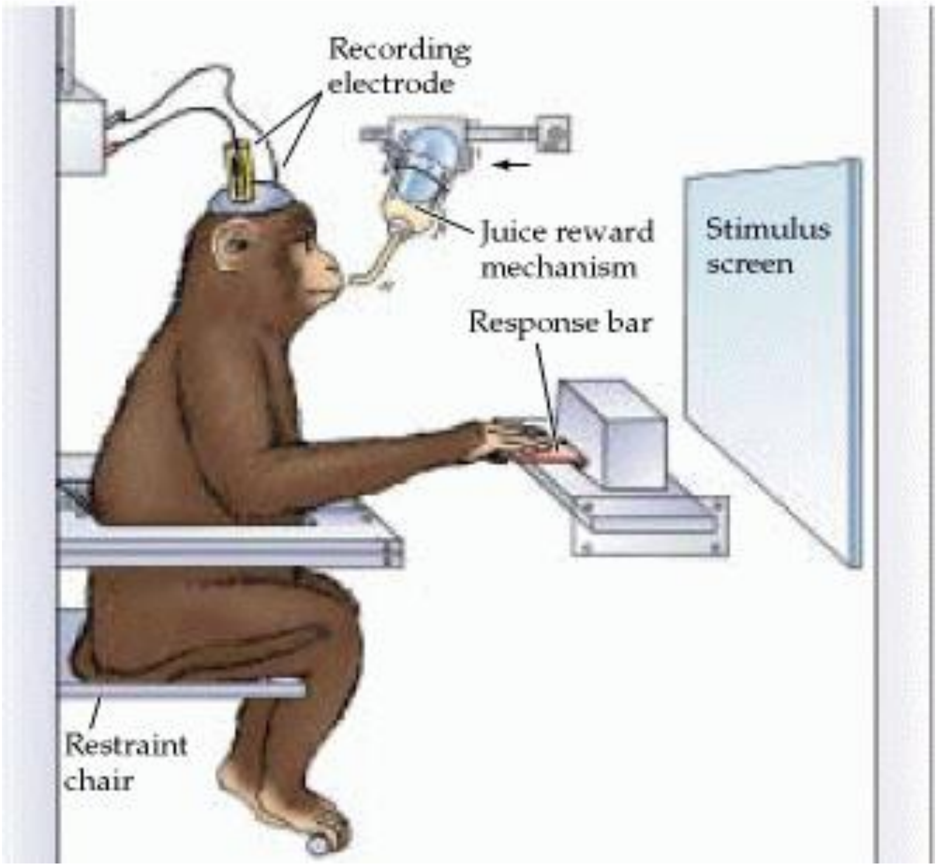
Exhaustive (Brute Force) Search

- A computer simulation that could evaluate 1 million phrases per second would take
- ~9,719,096,182,010,563,073,125,591,133,903,305,625,605,017 years

- Estimated age of the universe:
  13,750,000,000 years

Metaheuristics - Under the hood

- Metaheuristic algorithms attempt to find the near-optimal solutions through random sampling + a **set of well-defined instructions**
- For instance, we may start by generating a initial random solution for the problem. This solution is evaluated and the objective value stored. A new solution is then generated by applying a **small random perturbation** to the solution initially generated. We evaluate the new solution and rank it. The process is **iteratively repeated** by applying random perturbations to the best solutions found so far – *local search, gradient descent, hill climbing*, etc.



Source: https://www.reed.edu/biology/courses/BIO342/2012_syllabus/2012_WEBSITES/CSLP%20Nov%2020%20Monkey%20and%20Addiction/mechanism.html

Course Schedule and Evaluation

Nuno Antunes Ribeiro

Assistant Professor

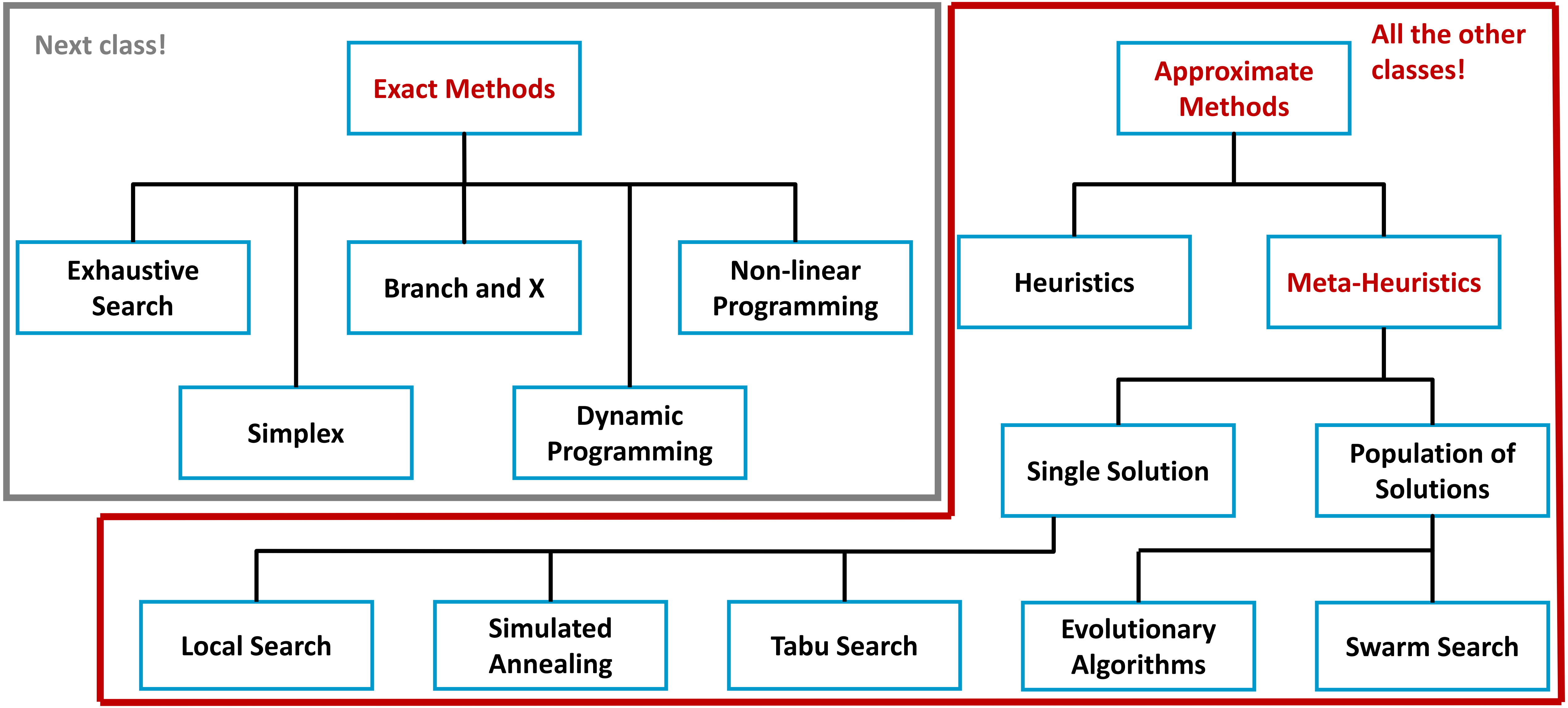
SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Engineering Systems and Design

Course Schedule

Week	Session 1	Session 2	Assignment
1	Introduction to Metaheuristic Optimization; NP-Hard models;	Exhaustive Search Methods and Backtracking; Branch and Bound; Solving Optimization Problems with Pyomo	
2	**Chinese New Year**	Random Sampling ; Local Search	HA1 – LS algorithms
3	Solution Encoding; Move Operators	Escaping Local Optima ; Simulated Annealing	
4	Variable Neighborhood Search; Greedy Constructive Heuristics	Tabu Search	
5	Common Concepts for Metaheuristics; Comparing Optimization Algorithms	Very Large Neighborhood Search (VLNS);	
6	Introduction to Evolutionary Algorithms	Applying Genetic Algorithms No Free Lunch Theorem	HA2 – EA algorithms
7	**BREAK**		
8	Using a Genetic Algorithm to Calibrate Neural Networks	Genetic Programming	
9	Other Types of Evolutionary Algorithms	Multi-Objective Optimization;	
10	Elitist Non-Dominated Sorting GA (NSGA); Meta-models	Particle Swarm Optimization	HA3 – Swarm algorithms
11	Ant Colony Optimization	Real Application Examples	
12	Project Consultation	Project Consultation	
13	Project Presentations	Project Presentations	
14	Final Exam		

Optimization Solution Methods



Assessment Methods and Softwares

Assessment Items	Percentage	Period
Class participation	5%	Throughout the term
Home Assignments	35%	Throughout the term
Project	45%	Throughout the term
Final Exam	15%	Week 14

Software:

Classes: Python

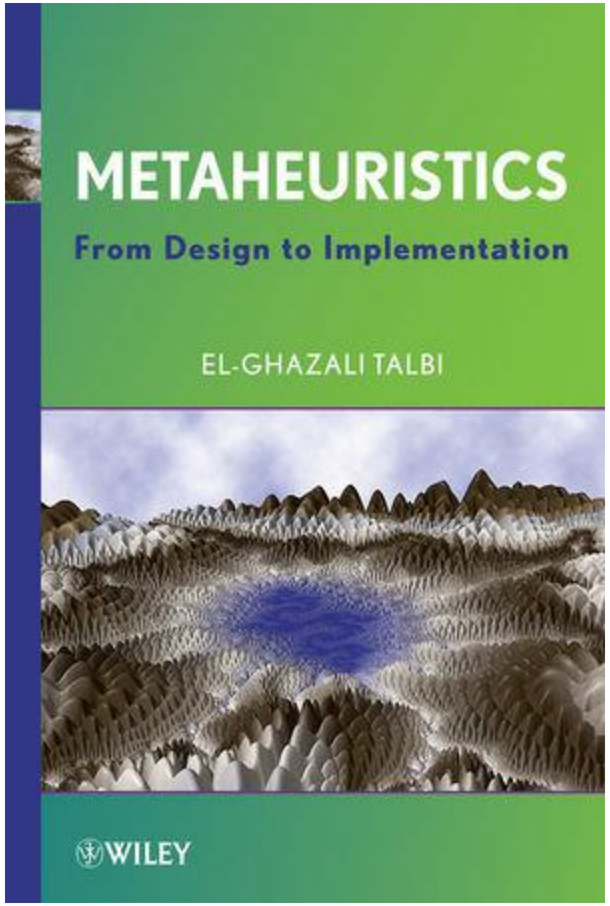
Project and Activities : Any programming language you prefer

Assessment Methods

- **Class Participation:** Includes attendance and active participation in class discussions. 2% reserved for the final **<u>survey</u>**.
- **Home Assignments:** **<u>Three home assignments</u>** are planned. The aim of these assignments is to further enhance students understanding on the various metaheuristic techniques. The home assignments will cover the following topics:
  - HA1 – Local search algorithms (simulated annealing)
  - HA2 – Evolutionary algorithms (genetic algorithm)
  - HA3 – Swarm intelligence algorithms (particle swarm optimization)
- **Project:** The objective of the project is to enable students to design, propose and adapt metaheuristic solutions to solve a real-world challenge. The topic can be selected according to the students' interest. A **<u>maximum of three students</u>** per project team is allowed. Project deliverables include **<u>a research paper (5-10 pages):</u>**
  1. Problem description and computational complexity
  2. Direct implementation of, at least, two algorithms introduced in class.
  3. Improve the solution methods proposed in (2) by developing more suitable solution encoding representations, better move operators or hybridization.
- **Final Exam:** A final written exam will evaluate the student's understanding on the main topics of the course

References

- Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons
- Metaheuristic Optimization Course Institute of Applied Optimization Hefei University http://iao.hfuu.edu.cn/teaching/lectures/metaheuristic-optimization



Big-O Notation

Nuno Antunes Ribeiro

Assistant Professor

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Engineering Systems and Design

Big-O Notation and Time Complexity

- Big O notation is a convenient way to describe how fast a given optimization/algorithm grows as a function of the input size.
- It allow us to compare the complexity of different optimization problems and algorithms without requiring experimentation

The story Behind Big O Notation

- Find all sets of nonnegative integers $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ that sum to integer $\boldsymbol{n}$ such that $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c} \leq \boldsymbol{n}$ and $\boldsymbol{n} \geq 0$; (example $\boldsymbol{n} = 3$)

Nuno's Solution

- **Try all combinations** $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$
- **Accept solution if**$(\boldsymbol{a} + \boldsymbol{b} + \boldsymbol{c} \geq \mathbf{3}$)

$(\mathbf{0}, \mathbf{0}, \mathbf{0})$	$(\mathbf{2}, \mathbf{1}, \mathbf{0})$	
$(\mathbf{1}, \mathbf{0}, \mathbf{0})$	$(\mathbf{2}, \mathbf{1}, \mathbf{1})$	…
$(\mathbf{1}, \mathbf{1}, \mathbf{0})$	$(\mathbf{3}, \mathbf{0}, \mathbf{0})$	
$(\mathbf{1}, \mathbf{1}, \mathbf{1})$	$(\mathbf{3}, \mathbf{1}, \mathbf{0})$	
$(\mathbf{2}, \mathbf{0}, \mathbf{0})$	…	$(\mathbf{3}, \mathbf{3}, \mathbf{3})$

CORE i3 9th Ge

20 secs

Solution Space = $(n + 1)^3 \approx n^3$

Student's Solution

Which Algorithm is More Efficient?

- **For all** $(\boldsymbol{a}, \boldsymbol{b})$**, set** $\boldsymbol{c} = \mathbf{3} - (\boldsymbol{a} + \boldsymbol{b})$
- **Accept solution if** $\boldsymbol{c} \geq \mathbf{0}$

$(\mathbf{0}, \mathbf{0}, \mathbf{3})$	$(\mathbf{3}, \mathbf{1}, \mathbf{0})$
$(\mathbf{1}, \mathbf{0}, \mathbf{2})$	
$(\mathbf{0}, \mathbf{1}, \mathbf{2})$	…
$(\mathbf{2}, \mathbf{1}, \mathbf{0})$	
$(\mathbf{2}, \mathbf{2}, \mathbf{0})$	$(\mathbf{3}, \mathbf{3}, \mathbf{0})$

CORE i7 8th G

25 secs

Solution Space = $(n + 1)^2 \approx n^2$

Big-O Notation and Time Complexity

TABLE 1 Commonly Used Terminology for the Complexity of Algorithms.

Complexity	*Terminology*
$\Theta(1)$	Constant complexity
$\Theta(\log n)$	Logarithmic complexity
$\Theta(n)$	Linear complexity
$\Theta(n \log n)$	Linearithmic complexity
$\Theta(n^b)$	Polynomial complexity
$\Theta(b^n)$, where $b > 1$	Exponential complexity
$\Theta(n!)$	Factorial complexity

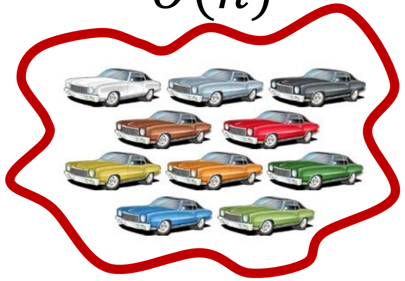
Big-O Notation and Time Complexity

Only 1 colour is available

$O(1)$

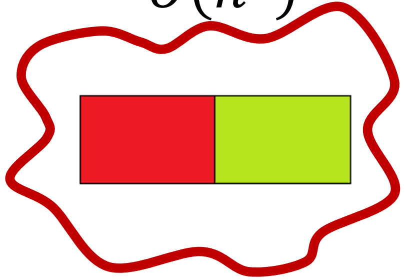
n colours are available, but only one colour can be selected

$O(n)$



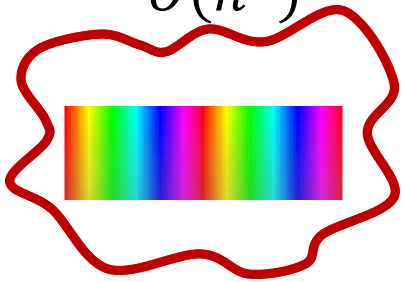
n colours are available. The car can be painted with up to 2 colours

$O(n^2)$



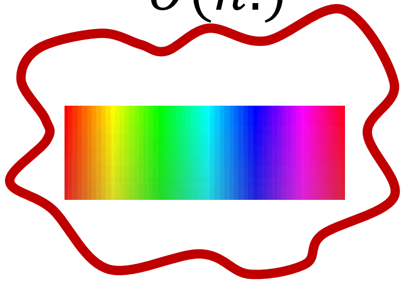
n colours are available. The car can be painted with up to n colours

$O(n^n)$



n colours are available. The car can be painted with up to n colours, but no colours can be repeated

$O(n!)$



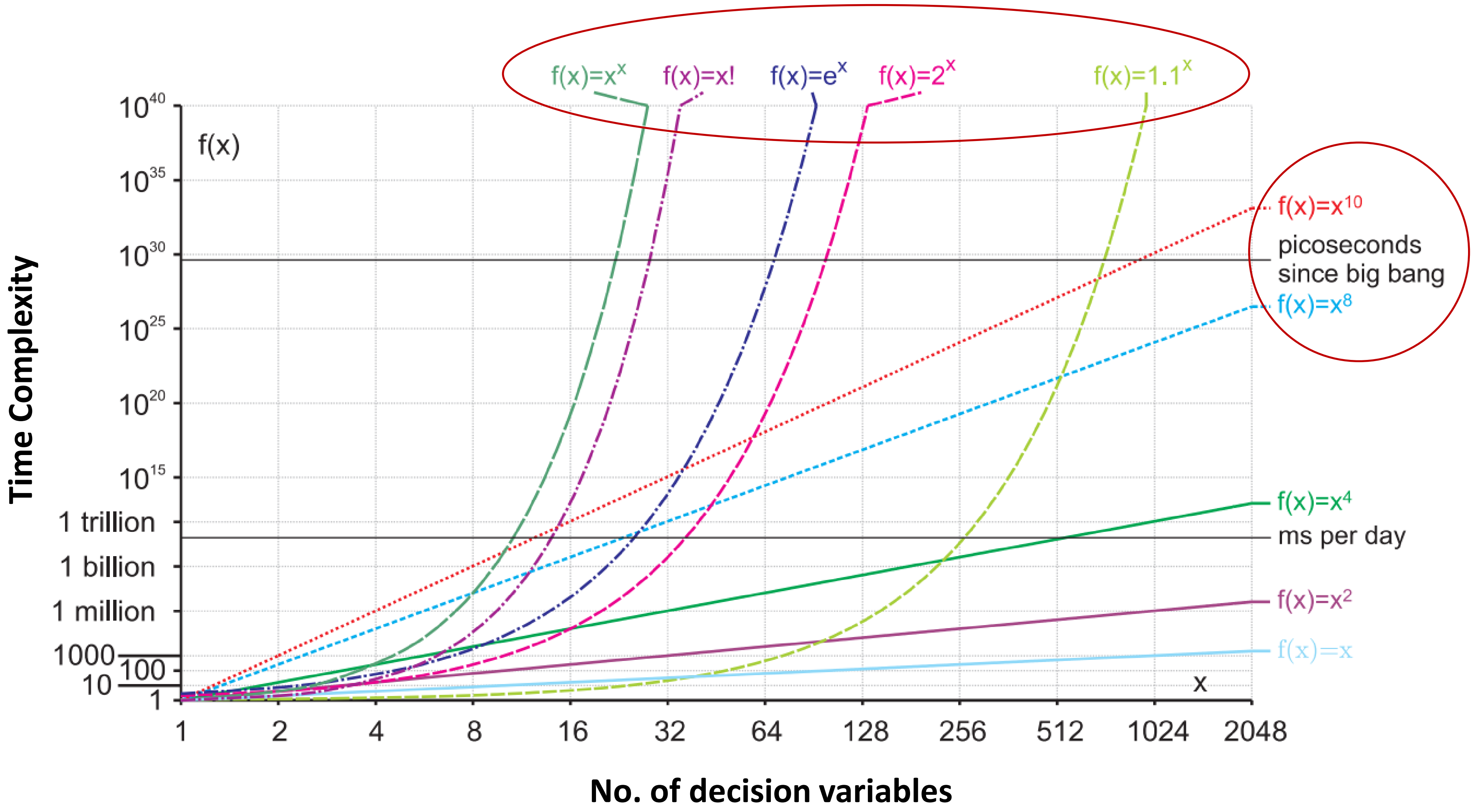
n colours are available. We aim to select up to n different colours. The order of the colours is not important

$O(2^n)$

Big-O Notation and Time Complexity

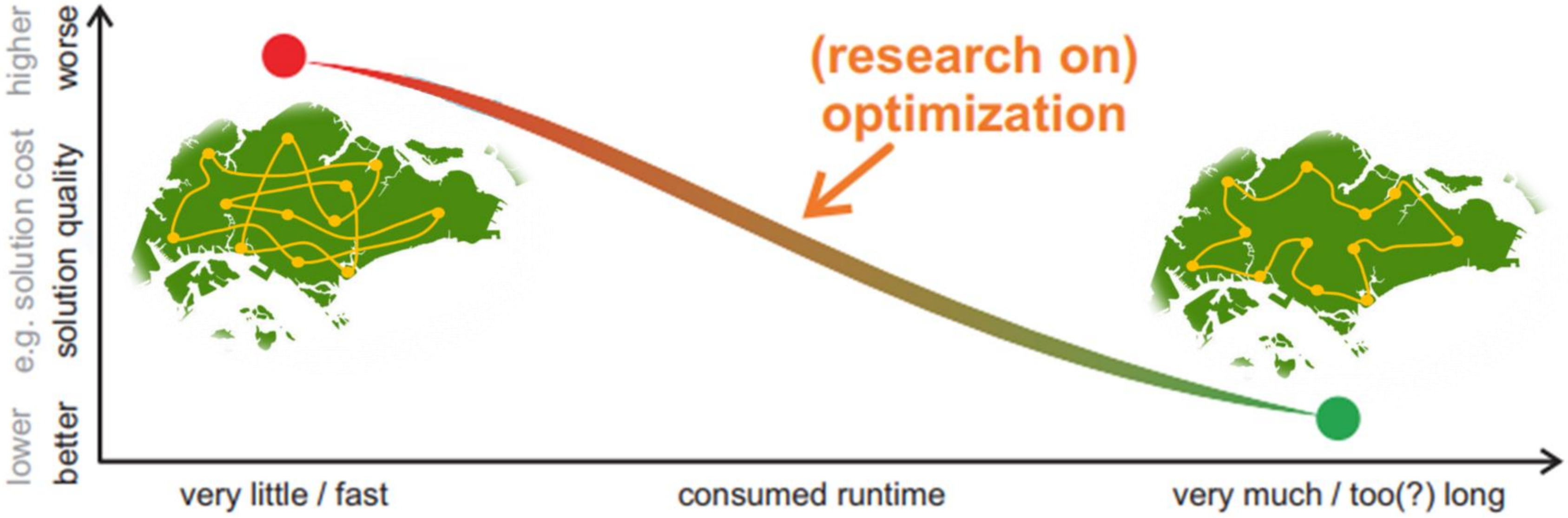
n	O(1)	O(log n)	O(n)	O(n log n)	O(n^2)	O(n^3)	O(n^10)	O(n^1000)	O(2^n)	n!	n^n
1	1	1.00	1	1	1	1	1	1	2	1	1
2	1	1.00	2	2	4	8	1024	1.2677E+30	4	2	4
4	1	2.00	4	8	16	64	1048576	1.6069E+60	16	24	256
8	1	3.00	8	24	64	512	1073741824	2.037E+90	256	40320	16777216
16	1	4.00	16	64	256	4096	1.0995E+12	2.582E+120	65536	2.0923E+13	1.84E+19
32	1	5.00	32	160	1024	32768	1.1259E+15	3.273E+150	4294967296	2.6313E+35	1.46E+48
64	1	6.00	64	384	4096	262144	1.1529E+18	4.15E+180	1.8447E+19	1.2689E+89	3.9E+115
128	1	7.00	128	896	16384	2097152	1.1806E+21	5.26E+210	3.4028E+38	3.856E+215	5.3E+269
256	1	8.00	256	2048	65536	16777216	1.2089E+24	6.668E+240	1.1579E+77	-	-
512	1	9.00	512	4608	262144	134217728	1.2379E+27	8.453E+270	1.341E+154	-	-
1024	1	10.00	1024	10240	1048576	1073741824	1.2677E+30	1.072E+301	-	-	-

Big-O Notation and Time Complexity



Algorithm Performance

- In optimization, there exist exact (brute-force search, branch and bound, etc.) and approximated (heuristic and metaheuristic) algorithms
- Algorithm performance has two dimensions: solution quality and required runtime



Modeling Steps

Nuno Antunes Ribeiro

Assistant Professor

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN | Engineering Systems and Design

Modeling Steps

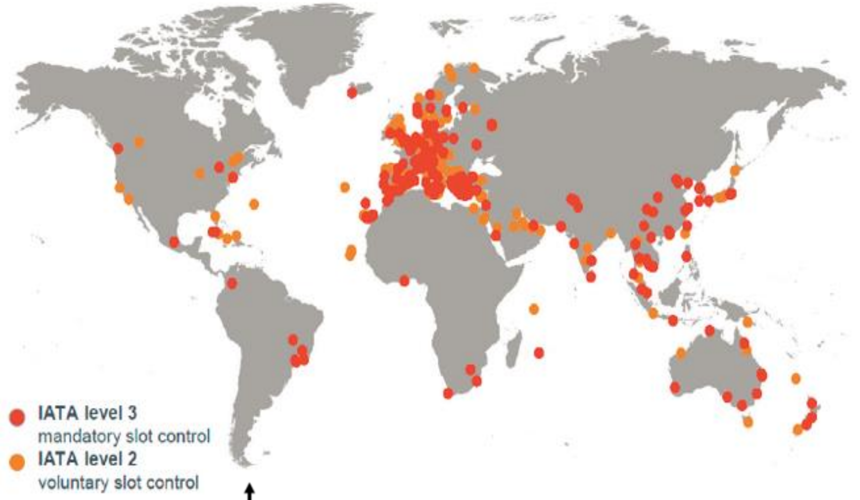
1. Understand the problem and all involved stakeholders, objects, entities, laws, constraints, etc.
2. Define what possible solutions look like, i.e., give a data structure for the solutions
3. Identify the problem/time complexity to solve the proposed problem
4. Define a objective function which rates how good a candidate solution is, how close it comes to what we really want as solution.
5. Given the time complexity, solution data structure, objective function and constraints develop (and select) an appropriate modelling approach to solve the problem

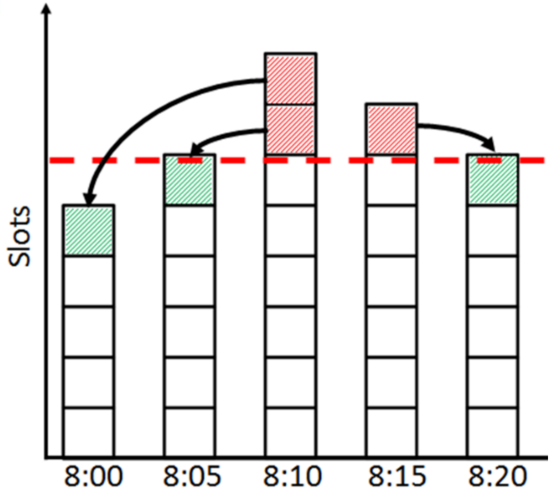
Note: If you develop an optimization software for a client, it is very important to discuss these issues with the client and to formally write them down on paper! The client often does not know exactly what he/she wants AND you may misunderstand him/her. . .

Airport Congestion

1. **Understand the problem and all involved stakeholders, objects, entities, laws, constraints, etc.**

- Capacity has been limited at the busiest airports worldwide
- Airports may rely on airport demand management solutions – i.e. strategic rescheduling of flights to prevent airport overscheduling)
- Slot allocation is the foremost demand management mechanism worldwide
- Airports applying slot allocation serve 55% of all passengers in the world





Metaheuristic Modelling – Example

1. **Understand the problem and all involved stakeholders, objects, entities, laws, constraints, etc.**

- Given a set of flights requested by airlines, we aim to purpose an airport schedule where we ensure that the airport capacities are never exceeded

- Runway capacity: flights comply with arrival/departure/total capacities
- Stand availability: when a flight land a stand needs to be available for parking
- Aircraft turnaround: the turnaround time does not increase/ decrease more than allowable limits
- Passenger connections: flights should allow important itinerary connections to take place
- IATA Guidelines Regulation: the slot allocation needs to be consistent with the regulation

71 pages!

Metaheuristic Modelling – Example

2. **Define what possible solutions look like, i.e., give a data structure for the solutions**

Flights

20	67	56	…	4	234	29	190

Time 1 day = 288 periods of 5 minutes

3. **Identify the problem/time complexity to solve the proposed problem**

n^m

	Madeira	Porto	Lisbon	São Paulo	Singapore	Paris
Nº Flights	13,696	41,547	114,176	161,469	254,129	348,977
Number of Solutions	2.18E+1191	1.38E+1330	3.82E+1456	8.51E+1499	4.52E+1556	2.12E+1596
Number of Solutions*	1.90E+99	7.00E+110	2.41E+121	9.87E+124	5.26E+129	1.06E+133

Metaheuristic Modelling – Example

5. **Given the time complexity, solution data structure, objective function and constraints develop (select) an appropriate modelling approach to solve the problem**

Airport	Exact Methods		Metaheuristics	
	GAP (%)	CPU Time	GAP	CPU Time
Madeira	0%	1 min	-	-
Porto	0%	5 min	-	-
Lisbon	2%	7 days	0.01%	6 hours
São Paulo	-	Memory Error	?	12 hours
Singapore	-	Memory Error	?	12 hours
Paris	-	Memory Error	?	12 hours

Metaheuristic Modelling – Example

4. **Define a objective function which rates how good a candidate solution is, how close it comes to what we really want as solution.**

Main Objective: minimize the difference between the requested schedule of flights and the allocated schedule of flights

However many other objectives have been suggested by airport coordinators

Multi-objective metaheuristics will also be a topic in this course!