**Short explainer: The importance of optimization (Voor Nederlands, zie de volgende pagina)**

The word “algorithm” has been in the press a lot recently. But what exactly is an algorithm?

An algorithm is a **list of steps** that a computer follows to complete a certain task. For example, to determine the shortest route from location A to location B, to sort a list of words into alphabetical order, or to compute the most efficient order in which a delivery van should visit a list of customers.

Indeed, behind every real-world problem that a computer solves, is an algorithm and the vast majority of algorithms are designed by humans.

When designing an algorithm, the following questions are crucial.

* Does it really do what it is supposed to do? (Does it really find the **shortest** route from location A to location B? How do we **know** it really is the shortest route?)
* How **fast** is the algorithm? (If you try to solve a problem that is twice as big, does the algorithm take twice as long? Four times as long? Sixteen times as long? If the growth in the time an algorithm takes is too big, it will only be able to solve very small problems in practice!)
* Is there a **better** algorithm out there, or have we actually designed the “best” algorithm possible? This is closely related to the question: **just how difficult** is the problem we are trying to solve? (Is sorting a list of words easier than routing the delivery van? That might seem intuitively reasonable, but how do you turn this intuition into something concrete?)

The academic discipline of **optimization,** which lies at the intersection of computer science, operations research and mathematics,is concerned with all these questions.

In a nutshell: given a challenging problem, that we wish to solve with a computer, how do we design **a fast, efficient** algorithm that **provably** finds an **optimal** solution to the problem, or – if that is too difficult to find – a solution that is **provably “very close” to optimality**?

Optimization is particularly crucial when using computers to support decision-making processes, or to efficiently extract information from data. For example:

* How to sequence tasks on an assembly line to save time and to waste fewer natural resources?
* Which nurses to schedule, and when, such that the quality of care is assured and the nurses have sufficient time to rest between shifts?
* How to draw accurate, verifiable and explainable conclusions from streams of data that are so large that there is only time to look at a fraction of the data?

**Explainer: Het belang van optimalisatie**

Het woord "algoritme" is de laatste tijd veel in het nieuws geweest. Maar, wat is een algoritme precies?

Een algoritme is een **lijst met stappen** die een computer volgt om een bepaalde taak te voltooien. Bijvoorbeeld, om de kortste route van A naar B te bepalen, een lijst met woorden in alfabetische volgorde te sorteren of om de meest efficiënte volgorde te berekenen waarin een koerier een lijst met klanten moet bezoeken.

Achter elk praktisch probleem dat een computer oplost, schuilt een algoritme, en het overgrote deel van de algoritmes zijn ontworpen door mensen.

Bij het ontwerpen van een algoritme zijn de volgende vragen cruciaal.

• Doet het echt wat het zou moeten doen? (Vindt het echt de **kortste** route van A naar B? Hoe weten we **zeker** dat dit de kortste route is?)

• Hoe **snel** is het algoritme? (Als je een probleem probeert op te lossen dat twee keer zo groot is, duurt het dan twee, vier of zelfs zestien keer zo lang? Als een algoritme uiteindelijk te veel tijd nodig heeft, kan het alleen zeer kleine problemen in de praktijk oplossen)

• Is er een **beter** algoritme beschikbaar, of hebben we eigenlijk het "beste" mogelijke algoritme ontworpen? Dit is nauw verbonden met de vraag: **hoe moeilijk is het probleem** dat we proberen op te lossen? (Is het sorteren van een lijst woorden makkelijker dan het plannen van de route van een koerier? Dat lijkt misschien intuïtief redelijk, maar hoe maak je van deze intuïtie iets concreets?)

De academische discipline van **optimalisatie**, die zich richt op het snijvlak van informatica, wiskunde en mathematische besliskunde, houdt zich bezig met al deze vragen.

In het kort: gegeven een uitdagend probleem dat we met een computer willen oplossen, hoe ontwerpen we een **snel, efficiënt** algoritme dat **gegarandeerd** een **optimale** oplossing voor het probleem vindt, of - als dat te moeilijk is - een oplossing die **gegarandeerd zo optimaal mogelijk is**?

Optimalisatie is bijzonder cruciaal bij het gebruik van computers om besluitvormingsprocessen te ondersteunen, of om efficiënt informatie uit gegevens te halen. Bijvoorbeeld:

• Hoe taken op een assemblagelijn te plannen om tijd te besparen en minder natuurlijke hulpbronnen te verspillen?

• Welke verplegers in te roosteren, en wanneer, zodat de kwaliteit van de zorg is gegarandeerd en de verplegers voldoende tijd hebben om uit te rusten tussen diensten?

• Hoe nauwkeurige, controleerbare en begrijpelijke conclusies te trekken uit enorme hoeveelheden data waarvoor alleen tijd is om een klein deel van de data te bekijken?