Onhandelbare problemen

**Concepten:** onhandelbare problemen, algoritme.

**Doel:** Kennis maken met problemen waarbij het bepalen van een optimale oplossing (dus de beste oplossing) niet binnen een redelijke tijd oplosbaar is.

**Leerdoelen:**

Na afloop kun je:

* Beschrijven wat een onhandelbare probleem is.
* Uitleggen wat een brute force algoritme is, en wat daar de voor- en nadelen van zijn.
* Herkennen of er een efficiënte oplossing voor een probleem bestaat, of dat het onhandelbaar is.

**Algoritmes en technieken:** Brute force.

**Praktische context**: Herkennen van problemen die onhandelbaar zijn, en waarvoor dus geen efficiënte algoritme bestaat.

# Inleiding

Er zijn problemen die te moeilijk zijn voor een computer, bijvoorbeeld een gesprek voeren. Dit is te moeilijk omdat wij zelf niet snappen hoe we het doen. Maar er zijn ook dingen die een computer makkelijk kan uitvoeren, maar waar de computer te lang over doet. Dit gebeurt als er bijvoorbeeld heel veel berekeningen uitgevoerd moeten worden. Zo’n probleem noemen we dan ‘onhandelbaar’ of ‘niet efficiënt’. Dan is het niet mogelijk om een oplossingen binnen een redelijke tijd te vinden.

## Opgave 1: Inkopen doen voor een feestje

Met een paar vrienden besluiten jullie een klein feestje te houden. Bij het legen van jullie zakken blijkt dat jullie precies 13 euro hebben. Kan jij een boodschappenlijstje maken waarmee je precies 13 euro opmaakt? Je hoeft geen rekening te houden met statiegeld. Beschrijf je aanpak.

|  |  |
| --- | --- |
| **Artikel** | **Prijs (euro)** |
| Doritos | 1.37 |
| dropveters | 1.25 |
| chips | 1.35 |
| borrelnootjes | 1.95 |
| cola 1L | 1.62 |
| spa rood | 0.85 |

Antwoord: Het is even puzzelen:

|  |  |  |  |
| --- | --- | --- | --- |
| **Artikel** | **Prijs (euro)** | **aantal** |  |
| Doritos | 1.37 | 2 | 2.74 |
| dropveters | 1.25 | 1 | 1.25 |
| chips | 1.35 | 1 | 1.35 |
| borrelnootjes | 1.95 | 1 | 1.95 |
| cola 1L | 1.62 | 3 | 4.86 |
| spa rood | 0.85 | 1 | 0.85 |
|  |  |  |  |
|  | totaal |  | 13 |

|  |
| --- |
| **Slim aanpakken?**  Voor dit soort problemen zijn er helaas (nog) geen slimme algoritmen ontdekt waarmee je dit probleem snel kunt oplossen. De enige bekende manier om dit op te lossen is middels brute force, domweg alle mogelijkheden uitproberen. Het gevolg is dat als het aantal artikelen toeneemt, het steeds moeilijker wordt (en dus langer duurt) om een optimale oplossing te vinden. |

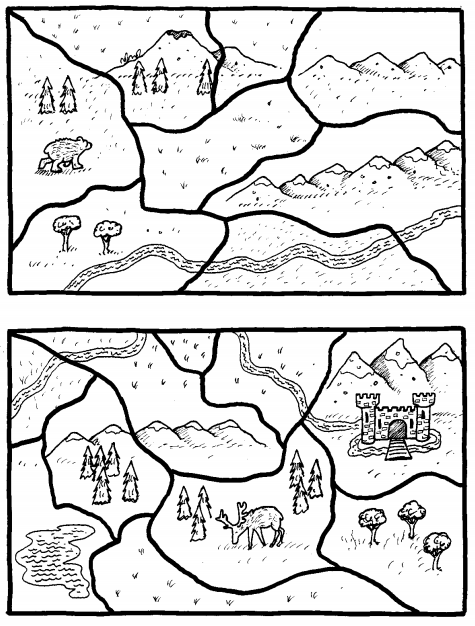
## Opgave 2: Vrachtwagens vullen

Je hebt als opdracht om vrachtwagens vol te laden zodat je zo min mogelijk vrachtwagens nodig hebt. Ga naar <http://csfieldguide.org.nz/en/interactives/bin-packing/index.html> en probeer het eens.

*Beschrijf je algoritme, eventueel als stroomdiagram. Zal dat altijd tot de beste oplossing leiden?*

|  |
| --- |
|  |

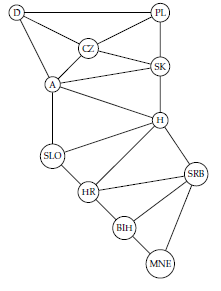
## Opgave 3: Landkaarten inkleuren

In een atlas zijn buurlanden altijd verschillend gekleurd. Kleur de landen op deze kaarten in met zo min mogelijk verschillende kleuren, maar wel zo dat geen twee aangrenzende landen dezelfde kleur hebben.[[1]](#footnote-1)

Antwoord: Je kunt natuurlijk de kleuringen direct gaan uitvoeren. In dit voorbeeld kom je daarmee weg. Maar, het maken van een graaf kan wel helpen om het probleem overzichtelijk weer te geven (zeker als het wat lastiger wordt). Je kunt het probleem als een graaf weergeven. Elk land wordt een knoop. Met een verbindingslijn geef je aan dat landen elkaar grenzen. Je hebt altijd hooguit 4 kleuren nodig om zo’n kaart in te kleuren (dit gegeven is na 120 jaar, in 1976 bewezen). Maar wellicht zou het ook met 2 of 3 kleuren kunnen. Het is niet zo heel erg moeilijk om te bepalen of dit wel of niet kan. Maar het kost wel erg veel tijd om op te lossen.

Hieronder zie je de kaart van een deel van centraal Europa. Kleur de kaart met zo min mogelijk kleuren, zo dat twee aan elkaar grenzende landen verschillende kleuren hebben. Kun je bedenken hoeveel kleuren heb je nodig om dat bij elk willekeurige kaart voor elkaar te krijgen?

## 

Antwoord: Ook deze probleem kun je als een graaf weergeven:  Je moet dan de knopen van de graaf zo kleuren dat twee knopende die met elkaar verbonden zijn verschillende kleuren krijgen. Hiervoor gebruik je natuurlijk zo min mogelijke kleuren.

Er is nog geen efficiënt algoritme bekend om dit voor een willekeurige graaf te doen met het minimale aantal kleuren. Wel zijn er snelle algoritmen die vaak, maar niet altijd, een minimale kleuring vinden, bijvoorbeeld het volgende algoritme:

• Nummer de knopen op volgorde van hoeveel verbindingslijnen ze hebben, de knoop met de hoogste graad eerst;

• Kleur de knopen in de volgorde die je zojuist hebt gemaakt. En doe dat zo zuinig mogelijk. Als knoop met 2 kanten niet verbonden is met knoop met 1 kant, dan krijgt hij dezelfde kleur als knoop nummer 1. Als knoop 3 wel met 2, maar niet met 1 verbonden is, dan krijgt hij dezelfde kleur als 1, behalve als knoop 2 al dezelfde kleur had als 1; dan krijgt 3 een nieuwe kleur. Enzovoort...

## Opgave 4: Rooster met het minst aantal uren

Waarom lukt het de roostermaker nou nooit om het perfecte rooster in elkaar te zetten? Zo moeilijk kan dat toch niet zijn? Wat is het kleinst aantal lesuren dat je nodig hebt voor het inroosteren van de vakken voor deze vier leerlingen?

1. Tom heeft een N&T profiel met wiskunde B, natuurkunde, scheikunde en informatica.
2. Janneke heeft een N&T profiel met wiskunde B, natuurkunde, scheikunde en biologie.
3. Els heeft een N&G profiel met wiskunde A, biologie, scheikunde en natuurkunde.
4. Piet heeft een N&G profiel met wiskunde A, biologie, scheikunde en aardrijkskunde.

Antwoord: Een roosterprobleem kan niet altijd naar een inkleurprobleem van landen worden vertaald, maar wel naar een graafkleurprobleem. Je kunt het dus als graaf weergeven. Ieder vak teken je als een knoop. Een verbindingslijn tussen twee vakken betekend dat er een leerling is die beide vakken volgt, en deze dus niet gelijktijdig geroosterd mogen worden. Het probleem is nu teruggebracht tot het probleem van het inkleuren van de landen. Je hebt hooguit 4 kleuren, oftewel, 4 uren nodig om een compacte rooster te maken, bijvoorbeeld:

1e lesuur: wisA of wisB

2e lesuur: natuurkunde of aardrijkskunde

3e lesuur: scheikunde

4e lesuur: biologie of informatica

**Onhandelbare problemen**

Deze opdrachten waren best klein, die kon je nog met de hand uitwerken, al duurt het misschien wel even voordat je er zeker van bent dat je de **allerbeste** oplossing hebt gevonden. Een rooster maken voor 4 leerlingen is dus misschien wel te doen, maar je had niet te maken met 1000 leerlingen, en hoefde zeker geen rekening gehouden met de werkdagen van de 200 docenten of welke lokalen er wanneer beschikbaar waren. Stel je nu voor dat je het allemaal wel zou uitrekenen, en een docent meld zich ’s morgens vroeg ziek, dan zou je als roostermaker heel snel een nieuw rooster moeten opstellen.

Voor een **onhandelbare probleem** bestaat er geen efficiënte algoritme. Om met een computer de **beste** oplossing te vinden voor dit soort problemen moet je eerst alle mogelijkheden uitrekenen en dan kijken welke de beste is. Dat heet **brute force**. Elke uitbreiding, zoals een leerling met een ander pakket of een extra lading dat mee moet, zorgt er voor dat het aantal mogelijkheden die je moet afgaan steeds veel groter wordt (je moet alle combinaties afgaan). Op een gegeven moment zal de snelste computer er honderden jaren over doen om dit probleem voor maar een relatief klein aantal variabelen op te lossen. Kortom: jouw optimale nieuwe rooster zal pas over heel wat jaartjes klaarliggen. Wat vind je daarvan?

Bekijk dit filmpje over praktisch onoplosbare problemen: <https://www.youtube.com/watch?time_continue=3&v=xWQ7D08eUVw>

1. Bron: www.csunplugged.org [↑](#footnote-ref-1)